Learning to Predict Vehicle Trajectories for Autonomous Driving from Latent Features

Zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften (Dr.-Ing.)

von der KIT-Fakultät für Wirtschaftswissenschaften
des Karlsruher Instituts für Technologie (KIT)

genehmigte

DISSERTATION

von

M.Sc. Faris Janjoš

aus Sarajevo

Tag der mündlichen Prüfung:	12. April 2024
Hauptreferent:	Prof. Dr.-Ing. J. Marius Zöllner, KIT
Korreferent:	Prof. Dr.-Ing. Igor Gilitschenski, University of Toronto

Abstract

In order for automated vehicles to fully share the roads with humans, they should be able to predict how humans drive. The ability to accurately forecast the motion of traffic participants facilitates understanding and solving the problem of human-aware autonomous navigation on public roads. The difficulty of predicting in the long term and the richness of the environment preclude scalable manually engineered solutions. Thus, a new paradigm has arisen: learning prediction from data.

This thesis contributes to the advancement of the nascent field of trajectory prediction in autonomous driving through the application and advancement of machine learning methods. To this end, the overarching trajectory prediction problem is decomposed into its sub-problems, which are addressed individually. These include the ability to

(i) represent the environment of a predicted agent,

(ii) model interaction between traffic participants,

(iii) capture the uncertainty of future motion, and

(iv) provide plausible trajectory candidates at the output.

The proposed solutions to each of these issues are placed into a generic machine learning framework outlined in the thesis, which encompasses

(a) the digestion of context data,

(b) the formation of internal latent representations,

(c) the generation of relevant outputs, and

(d) the application of a learning paradigm designed to extract the most knowledge from the data at hand for (a)-(c).

The proposed approaches address the individual sub-problems within the overall problem while being grounded to specific components within the generic Machine Learning (ML) architecture. They are part of multiple publications; in the first publication, it is shown how translating the prediction problem into the space of actions (i.e. control inputs) guarantees realistic, kinematically feasible trajectories. The action-based perspective further enables reasoning about the influence of actions on the environment. To this end, self-supervised learning

schemes are designed, in which internal latent representations of the environment are predicted prior to trajectories. The second publication presented in this thesis offers new learning paradigms while proposing to predict trajectories in an auto-regressive manner over their segments instead of entirely at once (i.e. one-shot prediction). This allows to ascertain the confidence of the model over evolving segments. In the third work, focus is given to the inputs of the architecture and the problems of representing the environment in an informative manner and modeling the interaction between traffic participants. To this end, an attention-based structure which facilitates joint prediction of multiple agents at once is proposed.

Further, the thesis deals with the problem of modeling future motion uncertainty. The fourth presented publication is a short study dealing with multi-modality in a model's outputs, addressing the questions of how to model distinct modes from the ground-truth distribution with a deterministic model for joint prediction. However, deterministic models are inherently ill-equipped to convey the rich multi-modal distribution of future motion. Thus, focus is given to generative models, where the Variational Autoencoder (VAE) and its Conditional Variational Autoencoder (CVAE) variant are identified as especially promising architectures. The fifth publication reexamines assumptions surrounding the latent space of the VAE and contributes fundamental improvements to this class of generative models. In the sixth publication, these improvements are then translated to the conditional CVAE architecture often used in trajectory prediction. Ultimately, a probabilistic latent space predictor is obtained, capable of modeling a multi-modal distribution in its latent space, sample it effectively, and translate it into trajectories.

The models laid out in this thesis are trained and evaluated on large-scale datasets of real-world driving and exhibit competitive performance against approaches from the literature. Overall, the experimental results show that the proposed ML-based models can accurately predict trajectories of human-driven vehicles in highly-interactive urban driving settings. In addition, the contributed VAE and CVAE architectures show merit on standard image modeling tasks.

Zusammenfassung

Damit autonome Fahrzeuge sich nahtlos in das Verkehrsgeschehen mit Menschen einfügen, sollten sie in der Lage sein, das menschliche Verhalten vorherzusagen und es zu berücksichtigen. Die Fähigkeit, das Verhalten der menschlichen Verkehrsteilnehmer langfristig zu prädizieren und dabei die Fülle an möglichen Situationen und Umgebungen zu berücksichtigen ist ein wichtiger Baustein zur Lösung der autonomen Navigation auf öffentlichen Straßen. Es hat sich gezeigt, dass Verfahren, die auf manueller Modellierung basieren, nicht skalieren. Daher hat sich ein neues Paradigma entwickelt: das Lernen der Prädiktion aus Daten.

Diese Arbeit liefert einen Beitrag zu Ansätzen für die Trajektorienprädiktion für das autonome Fahren durch die Anwendung und Weiterentwicklung von Methoden des maschinellen Lernens. Zu diesem Zweck wird das übergreifende Problem der Trajektorienprädiktion in seine Teilprobleme zerlegt, die einzeln behandelt werden. Dazu gehört die Fähigkeit

(i) die Umgebung eines prädizierten Agenten darzustellen,

(ii) die Interaktion zwischen Verkehrsteilnehmern zu modellieren,

(iii) die Unsicherheit der zukünftigen Bewegung zu erfassen sowie,

(iv) plausible Trajektorienkandidaten am Ausgang bereitzustellen.

Die vorgeschlagenen Lösungen für jedes dieser Probleme werden in folgende Struktur für maschinelles Lernen eingebettet

(i) die Verarbeitung von Kontextdaten,

(ii) die Bildung interner latenter Repräsentationen,

(iii) die Generierung relevanter Prädiktionen,

(iv) das Anwenden eines Lernparadigmas, das darauf ausgelegt ist, für (a)-(c) das meiste Wissen aus den vorliegenden Daten zu extrahieren.

Die vorgeschlagenen Ansätze befassen sich mit den einzelnen Teilproblemen innerhalb des Gesamtproblems der Trajektorienprädiktion für das autonome Fahren und sind gleichzeitig auf spezifische Komponenten innerhalb der generischen Machine Learning (ML)-Architektur ausgerichtet. Sie sind Teil mehrerer Veröffentlichungen; in der ersten Veröffentlichung wird gezeigt, wie die Übertragung des Problems der Trajektorienprädiktion in den Raum der Aktionen (d.h.

Steuereingaben) realistische, kinematisch umsetzbare Trajektorien garantiert. Zudem ermöglicht die Betrachtung des Problems im Aktionsraum, den Einfluss von Aktionen auf die interne Repräsentation der Umgebung zu berücksichtigen. Hierzu werden selbstüberwachte (Englisch: *self-supervised*) Lernverfahren entwickelt, welche zunächst eine interne latente Repräsentation der Umgebung als Basis für die Generierung der Trajektorien prädizieren. Die zweite Veröffentlichung, die in dieser Arbeit vorgestellt wird, entwickelt einen Ansatz, welcher eine auf Segmente basierende autoregressive Prädiktion anwendet, anstatt die gesamte Trajektorie auf einmal zu generieren (d.h. One-Shot-Prädiktion). Dies ermöglicht es, die Zuverlässigkeit des Modells über sich entwickelnde Segmente hinweg zu ermitteln. In der dritten Arbeit liegt der Schwerpunkt auf der Herausforderungen der informativen Darstellung der Umgebung und der Modellierung der Interaktion zwischen Verkehrsteilnehmern. Zu diesem Zweck wird eine Attention-basierte Struktur vorgeschlagen, welche eine gesamtheitliche Prädiktionen für mehrere Agenten ermöglicht.

Außerdem befasst sich die Arbeit mit dem Problem der Modellierung der Unsicherheit von Trajektorienprädiktionen. Die vierte vorgestellte Veröffentlichung ist eine kurze Studie, die sich mit der Multimodalität in den Ausgaben eines Modells befasst und sich mit der Frage auseinandersetzt, wie verschiedene Prädiktionsmodi mit einem deterministischen Modell für gesamtheitliche Prädiktion für mehrere Agenten gegeben einer Ground-Truth-Verteilung modelliert werden können. Deterministische Modelle sind jedoch von Natur aus schlecht geeignet, um die reichhaltige multimodale Verteilung zukünftiger Bewegungen zu vermitteln. Daher wird der Schwerpunkt auf generative Modelle gelegt, wobei das Variational Autoencoder (VAE) und das Conditional Variational Autoencoder (CVAE) als besonders vielversprechende Architekturen identifiziert werden. In der fünften Veröffentlichung werden die Annahmen über den latenten Raum des VAE untersucht und grundlegende Verbesserungen für diese Klasse generativer Modelle vorgenommen. In der sechsten Publikation werden diese Verbesserungen dann auf die CVAE-Architektur übertragen, die häufig in der Trajektorienprädiktion verwendet wird. Das Ergebnis ist ein Prädiktionsverfahren im probabilistischen latenten Raum, welches in der Lage ist, eine multimodale Verteilung in seinem latenten Raum zu modellieren, sie effektiv abzutasten und Abtastpunkte (Englisch: Samples) in Trajektorien zu dekodieren.

Die in dieser Arbeit vorgestellten Modelle werden auf umfangreichen aufgezeichneten Datensätzen für Prädiktion und Planung trainiert und evaluiert und zeigen auf etablierten Bewertungsdatensätzen eine konkurrenzfähige Performanz gegenüber Ansätzen aus der Literatur. Insgesamt zeigen die experimentellen Ergebnisse, dass die vorgeschlagenen ML-basierten Modelle die Trajektorien von von menschlichen Fahrern gesteuerten Fahrzeugen in hochinteraktiven städtischen Fahrsituationen genau prädizieren können. Darüber hinaus zeigen die entwickelten VAE- und CVAE-Architekturen gute Performanz auf etablierten Bewertungsdatensätzen für maschinelle Bildverarbeitung.

Acknowledgments

This thesis is the culmination of the research work I carried out as a doctoral student in the department of Advanced Autonomous Systems (CR/AAS) at Bosch Research in Renningen, Germany. It is not only the product of the three and a half years I spent working at Bosch, but also that of a lifetime of learning from and being supported by colleagues, friends, and family. I would like to thank everyone who has helped me along the way.

First of all, I express my gratitude to Marius for supervising this thesis. I would like to thank him for his helpful feedback, for trusting my vision and letting me be independent, as well as for his support for the completion of this thesis. I would also like to thank Igor for agreeing to co-examine this thesis.

Many thanks go to all my Bosch colleagues for providing the best possible environment to conduct research: focused and without distractions, yet also open, trusting, and collaborative. I want to thank Maxim for his fantastic mentorship, insightful feedback, and the many brainstorming sessions we've had, as well as for the support he gave me in the beginning by providing me with everything I needed to get started and pushing to "close the loop" as soon as possible. I am carrying many of these insights over to my mentorships. I would like to thank Axel, my group and later department manager, for providing the resources to conduct proper research, as well as for being supportive and opening the door which led me to my current role in Bosch.

It is impossible for me not to thank the many people I have been lucky enough to collaborate with: Lars, for the deep discussions about generative modeling, Anthony and Mihai from FiveAI for their prediction and planning expertise, as well as Andrey and Luigi for their robotics perspective on very similar problems. It has also been a pleasure to work with Andi, Jürgen, Luca, Markus, Martin, Max, and Oliver in a common codebase that was the foundation for the implementation of many approaches presented in this thesis. Of course, I am also indebted to my students Yinzhe and Max, who is now my colleague, for their motivation and drive. Additionally, Elizabeth, Lin, and many other CR/AAS colleagues have greatly enriched my doctoral time with dinners, wine walks, sports events, and other activities. I would also like to gratefully give credit to Laura for the extensive administrative support.

I am grateful for my friends and peers from the Bosch PhD network: Marcel, Felicia, and Marvin for the excellent and fruitful collaborations on papers and

student supervision, as well as to Samir, Isabel, Maria, Pascal, Miloš, and Robert, who made my doctoral life much more enjoyable with our many outings and events. On that note, I sincerely thank the many FZI peers for the memorable PhD seminar in Mallorca. Finally, I also would like to thank the people who founded, and currently maintain and contribute to the Bosch PhD program. I never expected that after my unforgettable Bosch-Foundation-sponsored Inter-Rail trip in 2017, I would cross paths with this company in this way again.

Over the course of my professional journey of working in research, I have been fortunate enough to draw inspiration and learn from many professors, teaching assistants and fellow students from Elektrotehnički Fakultet Sarajevo, without whom I would not be here. My desire to conduct research originated during my studies at ETF. Furthermore, I would like to relay my sincere gratitude to all my friends from Munich, who are too numerous to name here. It was not easy to leave everyone and move to Stuttgart during coronavirus times to start a PhD.

Finally, I would like to express my deepest thanks to my family. To Pauline, for her loving support throughout this time, her keen interest in my work, as well as the understanding in the times when I was distracted or occupied with deadlines. I am grateful to my sister Nejra for the constant belief in my abilities, which provided me with the motivation to persevere through challenges. Finally, I am indebted to my parents, Muša and Fuad, for a lifetime of unquestionable love, support, and constant encouragement to grow and learn. Their sacrifices and guidance have shaped me into the person I am today, and I am forever grateful.

Stuttgart, February 2024 Faris Janjoš

Contents

1 Introduction

Autonomous Driving (AD) promises to revolutionize the transportation industry. Among its many potential benefits are the reduction of road accidents [20] – thus saving lives and reducing injury as well as consequently lowering vehicle insurance costs [29], the increase in living spaces through reduced passenger car ownership (i.e. the robo-taxi use-case) [11, 46], improved throughput in the logistics industry (i.e. autonomous trucking) [68], counteracting the labor shortage within commercial driver professions [107], as well as generally having the ability to give back the time humans spend on manually controlling a vehicle in closed loop with environment feedback. These societal and financial rewards have drawn large amounts of investment into the AD space and catalyzed research and development in the field.

The term AD encompasses vehicle systems that actively perform driving tasks on public roads, i.e. vehicles that exhibit a level of autonomy within limited conditions of the environment (e.g. in a highway setting) or universally under any road traffic conditions. Thus, the Autonomous Vehicle (AV) is an autonomous agent that performs a variety of tasks in its environment. It perceives contextual information using a diverse sensor setup, fuses the perceived information in time and space while localizing itself, predicts the motion of surrounding traffic participants, and finally plans and executes a safe trajectory bringing it closer to its goal. A simplified representation of this procedure featuring some details related to each stage is given in Fig. 1.1.

Given the different facets of the AD field, research is distributed across multiple individual tasks. Recent advances in ML, particularly Deep Learning (DL), have shown potential to conquer the expansive landscape of AD research. Large amounts of data have been made available in public datasets supporting research in relevant fields [97, 14, 115, 49]. Thus, boosted by the methodological advances and the availability of data, DL methods have proven themselves capable of processing large amounts of high-dimensional input information, learning useful internal latent representations, and generating high-quality and task-relevant outputs. Therefore, a vast literature of DL approaches for performing perception, prediction, or planning, individually or simultaneously, has emerged in recent years [80, 40, 60, 84, 44].

Within the AD stack comprising perception, prediction, and planning, predicting the motion of traffic participants surrounding an AV requires knowledge of the

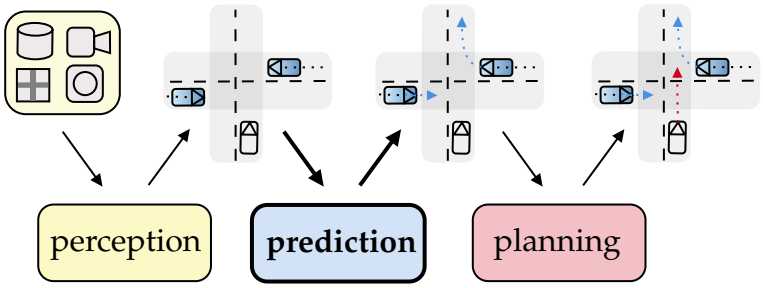


Figure 1.1: A simplified structure of the AD pipeline divided into perception, prediction, and planning modules. The perception module perceives the environment of the AV using its sensor setup (e.g. LiDAR, camera, HD map, radar) and generates a corresponding environment model. This can include for example the observed trajectories of relevant agents (blue) and the geometry of the road. The environment model is then fed as input into a prediction module, which uses this contextual information to predict the motion of relevant agents over a future time horizon. Finally, a planning module generates a trajectory for the AV (red) considering its goal and the motion of others.

road structure (e.g. in the form of a map) and awareness of the agents sharing the road with the AV – such as other vehicles, pedestrians, and bicycles. Then, the AV determines the relevance of the surrounding agents – e.g. whether they are in a parked state, moving but are not relevant to the AV, or moving on a section of the road relevant to the AV and its immediate plans. Given this information, the AV can then predict the motion of relevant agents. Having this information is paramount in highly interactive scenarios such as urban driving, where knowing how the scene can evolve enables the AV to negotiate intersections efficiently, avoid collision, and nudge through congested traffic. Even in highly structured scenarios such as highway driving, predicting the motion of others can be useful for Advanced Driver Assistance Systems (ADAS) functions. For example, an Automated Cruise Control (ACC) system benefits from knowing whether surrounding vehicles are going to perform a lane change in order to adjust its safe distance to vehicles ahead. Overall, accurate prediction of motion can serve as an enabler for safe and performant automated and autonomous vehicles.

1.1 Outline of the Thesis

The present thesis aims to investigate and address the challenges of motion prediction in AD, namely vehicle trajectory prediction. To this end, powerful

ML-based methods are developed and applied. The thesis is structured as follows. First, Sec. 1.1.1 lists the individual publications achieved throughout the doctoral studies. The following Sec. 1.2 further motivates and defines the scope of the proposed approaches. Then, Sec. 1.3 introduces the overall problem addressed in the thesis and divides it into sub-problems that can be addressed individually, presented in Sec. 1.3.1. Then, in Sec. 1.4, a general ML-based model architecture capable of addressing the given sub-problems is proposed; its components are described in Sec. 1.4.1. The aforementioned problem-level and model-level perspectives are used to contextualize the proposed individual approaches later. Chapter 1 is completed with a general description of the experimental setup employed in the proposed approaches.

The following chapters 2-5 group the individual publications listed in Sec. 1.1.1 across four topics. The topics are chosen according to specific sub-problems and architecture components laid out in sections 1.3 and 1.4. Thus, each publication belongs to a single topic and can be thematically mapped to specific sub-problems and model components. The first topic in Sec. 2 comprises two publications dealing with the issue of generating feasible trajectories at the output of the model while at the same time considering mechanisms to train ML-based prediction models. The second topic in Sec. 3 covers one publication and deals with the issue of representing and processing environment data at the input and extracting necessary information from it. The third topic in Sec. 4 also covers a single publication and presents simple approaches for modeling the uncertainty of the future by adapting the outputs of an ML model. The final topic in Sec. 5, comprising two most recent works, deals with the problem of future uncertainty modeling in a comprehensive manner by constructing rich internal representations within an ML model. Within each of the four chapters 2-5, each publication is discussed individually. The publications are prefaced with a short discussion summarizing the research context of the work, such as its motivation and relevant related works, and a summary of its contributions. Finally, Chapter 6 sums up the contributions of the overall thesis, discusses the limitations of the proposed approaches, and offers directions for future work. The actual publications in their original published form can be found in Included Publications.

1.1.1 List of Publications

During the author's doctoral studies, the following publications were realized. They are sorted thematically in the order of presentation in the following chapters 2-5, rather than chronologically.

- Paper I [A1]:
  - Title: *Self-Supervised Action-Space Prediction for Automated Driving*
  - Authors: Faris Janjoš and Maxim Dolgov and J. Marius Zöllner
  - Venue: 2021 IEEE Intelligent Vehicles Symposium (IV)
- Paper II [A2]:
  - Title: *Bridging the Gap Between Multi-Step and One-Shot Trajectory Prediction via Self-Supervision*
  - Authors: Faris Janjoš and Max Keller and Maxim Dolgov and J. Marius Zöllner
  - Venue: 2023 IEEE Intelligent Vehicles Symposium (IV)
- Paper III [A3]:
  - Title: *StarNet: Joint Action-Space Prediction with Star Graphs and Implicit Global-Frame Self-Attention*
  - Authors: Faris Janjoš and Maxim Dolgov and J. Marius Zöllner
  - Venue: 2022 IEEE Intelligent Vehicles Symposium (IV)
  - Best Paper Runner-Up Award
- Paper IV [A4]:
  - Title: *SAN: Scene Anchor Networks for Joint Action-Space Prediction*
  - Authors: Faris Janjoš and Maxim Dolgov and Muhamed Kurić and Yinzhe Shen and J. Marius Zöllner
  - Venue: 2022 IEEE Intelligent Vehicles Symposium (IV) workshop: From Benchmarking Behavior Prediction to Socially Compatible Behavior Generation in Autonomous Driving
- Paper V [A5]:
  - Title: *Unscented Autoencoder*
  - Authors: Faris Janjoš and Lars Rosenbaum and Maxim Dolgov and J. Marius Zöllner
  - Venue: 2023 International Conference on Machine Learning (ICML)
- Paper VI [A6]:
  - Title: *Conditional Unscented Autoencoders for Trajectory Prediction*
  - Authors: Faris Janjoš and Marcel Hallgarten and Anthony Knittel and Maxim Dolgov and Andreas Zell and J. Marius Zöllner

- Venue: submitted to 2024 European Conference on Computer Vision (ECCV) workshop: Event Detection for Situation Awareness in Autonomous Driving

1.2 Motivation and Scope

In the AD context, accurate motion prediction is necessary to serve a planner toward safe yet concurrently proactive and assertive behavior. Without motion prediction of other agents, a motion planner can reason about safe behavior of the overall system w.r.t. others agents by computing criticality metrics such as Time To Collision (TTC). Then, if necessary it can adjust the velocity of the AV in a reactive manner. In contrast, an AV can exhibit proactive behavior and make better progress toward its goal using the information on likely intentions and future motions of other traffic participants. If the system can predict the consequences of its actions w.r.t. interacting agents, it is poised to operate more effectively in this joint environment. However, even in a non-interactive use case, being able to predict the motion of traffic participants is useful. Models able to do so can be used as building blocks for powerful simulators and world models. Such systems can help generate novel traffic scenarios as well as replay existing scenarios in simulation in order to validate the performance of an AV planner before deployment in the real world.

Historically, motion prediction has been a relevant problem in the context of driver assistance systems. They employed classical, kinematics-based methods that extrapolate motion over short-term horizons (e.g. 1-2 seconds). The AD context demands solving the significantly more difficult problem of long-term prediction, which shares challenges with the problem of predicting human motion in general [92]. The difficulty of solving the AD prediction problem is multi-faceted. It demands the ability to capture rich environment contexts brought on by use-cases such as highly interactive urban driving, the ability to represent and capture interaction between traffic participants, as well as most importantly, the ability to model the uncertainty of future motion.

In addressing the complexity and challenges of the problem, ML models have proved indispensable. Enabled by the availability of data through numerous publicly available datasets [97, 14, 115, 49], they have shown the ability to handle challenging prediction scenarios in both AD and robotics [92]. One concept that stands out in the context of ML is the ability to compress this rich high-dimensional space of inputs into useful internal representations, termed latent features. This is a core concept in the subfield of representation learning in ML [9]. The space of latent features can either be modeled in a deterministic manner without uncertainty, or in a probabilistic manner with uncertainty. One

of the main themes of this thesis revolves around the use of latent features for effective prediction.

Within the AD stack comprising perception, prediction, and planning, motion prediction models can be trained and evaluated jointly with other tasks. This enables the propagation of information through learned features from different stages of the pipeline onto prediction, and has the potential to improve the performance of prediction models. Recent approaches such as [16, 51] have shown the benefits of such a setup and made progress toward end-to-end AD models. Nevertheless, many approaches still consider the prediction problem in isolation. This assumes replacing the environment data that would be provided by perception components in an online setting by means of ground-truth historical map and track information, i.e. essentially assuming perfect perception. Based on such data, future motion is predicted and models are evaluated solely on prediction metrics. Studying motion prediction independently in this manner serves to gain domain understanding that can serve as a basis for research in joint, system-level solutions such as end-to-end models. Furthermore, such joint solutions are most useful when system-level metrics such as planning performance are evaluated. These are outside of the scope of this work. Interested readers are referred to the following recent publications [2, 16, 51, 81] that show success in tackling more general problems in the AD space. Nevertheless, an exploration of an approach that opens up the interface between tracking in perception and prediction is featured in the student work [S2] but not elaborated further in this thesis.

The present thesis offers approaches to predict the motion of vehicles sharing a traffic scene with an AV. The presented models are able to consider Vulnerable Road Users (VRU)s for the purposes of vehicle motion prediction but do not predict the motion of VRUs. However, some methods presented in the thesis are well-suited for generalizing toward agent classes such as bicycles. Predicting the motion of pedestrians however is a critical ability if AVs are to operate in close proximity to pedestrians. Usually, it is restricted to distinct scenarios such as crossing behavior. Such a problem heavily relies on subtle cues in body pose and gaze of a human interacting with an AV. In order to understand such signals comprehensively, it is crucial to have high quality perception systems and an abundance of rich perception data to train ML models. Such research is out of the scope of this thesis. Interested readers are referred to [103] for a study on the matter.

Finally, the approaches contributed in this thesis represent predictions as trajectories, i.e. two-dimensional vehicle positions in time. Entire trajectories are predicted in a one-shot manner, i.e. not in a time-series auto-regressive manner where consecutive single-step predictions are chained on top of each other[1].

[1] Nevertheless, concepts from auto-regressive methods can be useful for one-shot prediction as well – they are applied in [A2], see Sec. 2.2.

The one-shot approximation is practical and efficient: it bypasses the necessary repeated calling of a prediction module while concurrently providing accurate predictions. Trajectories offer a generic and precise representation of the high-level intent as well as low-level motion of vehicles. Additionally, trajectories are also the language of motion planners: they are generated for the ego vehicle and ultimately executed by a control module. Finally, trajectories facilitate the representation of multiple proposals as well, so-called *multi-modal* predictions. In this case, a model outputs multiple possible trajectories a vehicle might take. Moreover, the uncertainty associated with a trajectory either can be represented explicitly, by augmenting each proposal with a quantitative representation of uncertainty (e.g. a variance around each point or a score representing the probability of each trajectory), or implicitly by outputting many trajectories (e.g. samples from a distribution) that collectively represent the uncertainty of motion. Nevertheless, alternative representations to trajectories have been introduced recently. For example, works such as [2, 52, 78] predict occupancies in two-dimensional space in time, so-called flows. These are useful to predict whether an area will be occupied by an agent in future time. Occupancies and flows can still be used as intermediate, higher-level representations since one could infer trajectories from them.

1.3 Problem Description

This section describes the overall problem addressed in this thesis. Given any historical context data, the task is to model the future motion of vehicles of interest to the AV. Historical context data is denoted by $\mathbf{X}$ – it includes information such as the map, past motion (such as kinematic and dynamic variables) of vehicles of interest and other agents in the vicinity of the AV, or any other contextual information. Similarly, future motion consisting of trajectories for each vehicle is denoted by $\mathbf{Y}$ and includes future xy positions of any number of vehicles of interest. Since future motion is inherently uncertain, it is described as a probability distribution. Mathematically, the task is to model

$$\mathbf{Y} = \mathcal{P}(\mathbf{Y}|\mathbf{X}) \,, \tag{1.1}$$

where $\mathcal{P}$ is a conditional distribution of future trajectories $\mathbf{Y}$ given the context $\mathbf{X}$. The distribution in Eq. (1.1) is used to obtain multiple future trajectory proposals. If the distribution is defined in an explicit, analytical form, samples can be directly drawn from it. Otherwise, various approximations of the distribution can be used to provide samples (e.g. a deterministic model that provides only the mean of the distribution). Therefore, regardless of the employed methodology, a general solution for trajectory prediction would be able to model this distribution and provide a mechanism to obtain samples.

In modeling the distribution in Eq. (1.1), it is helpful to consider domain knowledge. In many situations, the future motion of vehicles given context information is defined as a set of distinct behaviors. For example, a vehicle stopped at an intersection (such as in Fig. 1.1) can either go straight, turn left, or turn right. Thus, any solution addressing the problem in Eq. (1.1) can be guided by domain-level knowledge. Such domain-level aspects can aid the decomposition of the overall trajectory prediction problem into its sub-problems.

1.3.1 Sub-Problems in Trajectory Prediction

A useful thought experiment to approach solving the vehicle trajectory prediction problem is to "invert" it and assume a solution is already in place. Assume we have a system that does not know the future but is well-capable of modeling it over a fixed time horizon when given limited information about its environment. What qualities would such a system possess? Motivated by domain knowledge, the following is posited. This forecasting model would be able to extract the maximal amount of information from its observations of the environment and relate it to known objects, such as static road infrastructure and relevant dynamic agents. Then, it would be able to model the influence dynamic agents exert on each other, i.e. their interactions. Given this information, it would be able to broadly model the distinct behaviors each relevant vehicle plans to exhibit in the near future (e.g. would a vehicle stopped at an intersection plan to drive straight or take a right or left turn?). In particular, the behaviors of a single vehicle would be consistent with other agents' possible behaviors (e.g. no colliding). The different contingencies represent the overall uncertainty the predictor assigns to the evolution of the scene, i.e they model the inherent uncertainty of the future through a multi-modal distribution in Eq. (1.1). Finally, concrete trajectories would be assigned to each behavior. These trajectories would be kinematically feasible and accurately model the motion of each vehicle. This hypothetical situation reveals that there are distinct sub-problems that a vehicle trajectory predictor should solve over consecutive stages. Solving the sub-problems in each stage relies on successfully tackling the previous stage.

Motivated by the decomposition of the overall problem into stages, the relevant sub-problems of each stage are identified using domain knowledge. These are visualized in Fig. 1.2. The sub-problems serve as formative guidelines for proposed solutions, i.e. the ML-based approaches presented in this thesis are developed with the aim of being capable to address the given sub-problems. The following sections give more detail on each stage.

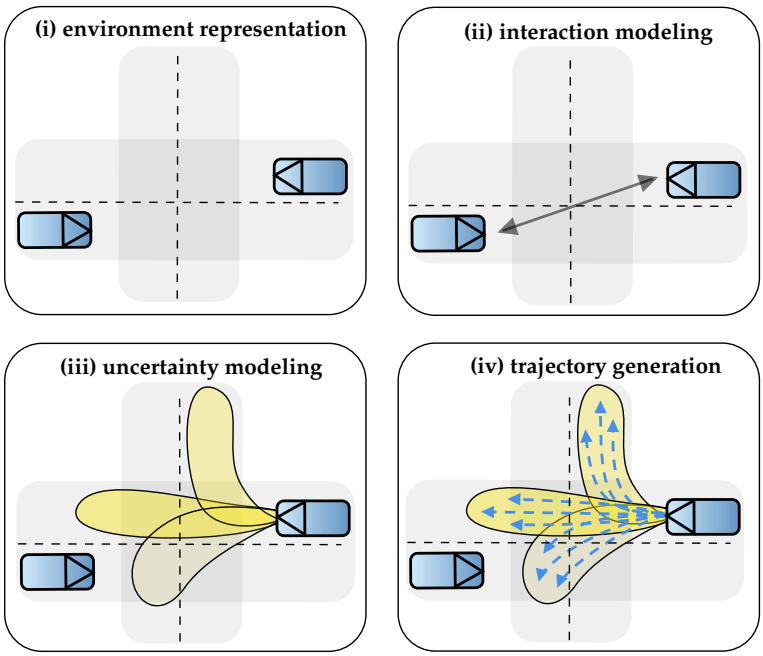


Figure 1.2: Breakdown of the stages of the trajectory prediction problem. A trajectory predictor is first tasked with finding a suitable environment representation from which it can extract useful information. Then, the predictor creates an internal model according to which the vehicles of interest interact with their environment (e.g. the map and other agents). Then, based on the environment representation and the interaction model, the predictor creates a high-level representation of the uncertainty of future motion. Finally, it creates fine-grained trajectory candidates as the output of the model.

Environment Representation (i)

Environment representation focuses on representing and processing rich high-dimensional input information with the goal of forming latent representations in later stages. In order to achieve this, a choice needs to be made on how to first represent the input information before feeding it into the model. For example, simplified Bird's Eye View (BEV) grids can be used, so-called rasters [30], where agents and the map are drawn in a binary multi-channel image [5] or an Red Green Blue (RGB) three-channel image [30]. An advantage is that different information is unified into a single domain that represents distances faithfully. In other words, the Euclidean space the problem naturally "lives in" is accurately quantized by the pixels in the raster image. However, grid-based representations do not scale well with distance – a vehicle moving at a high velocity requires a large image to represent its motion, putting high computational demands on the Convolutional Neural Network (CNN) architectures that process images. Furthermore, seemingly arbitrary choices regarding different entities have to be made: how to differentiate between the types of agents in an image (e.g. by shape, size, or color), how to represent different types of road lines, how to communicate traffic rules, how to represent historical information, etc. If the resulting raster image is rather minimalistic, it puts hard demands on the ML model to discern objects from the static background by relating image features to semantic information. This is exacerbated by fundamental inductive biases present in CNNs, such as translation invariance and locality [7], which fall short when statistics within an image are not stationary (e.g most relevant information in a raster image of a traffic scene is contained within road boundaries). An alternative to images are direct representations such as graphs. These are naturally able to represent discrete entities as well as the topological organization of the map for example through nodes and edges. Furthermore, it is easy to model the relationships within a graph through Graph Neural Networks (GNN) architectures. However, enriching such a representation with distance information is difficult owing to the lack of an underlying manifold. Overall, benefits and drawbacks of different representations have to be considered when designing a trajectory prediction model, see [S1] for a discussion. Representing the environment is usually particularly relevant to the input parts of the overall ML model.

Interaction Modeling (ii)

In this problem, the predictor relates its choice of environment representation to an internal model of the interaction that relevant agents exhibit with regard to their environment. More specifically, historically observed information (captured in a given environment representation) is used to infer the interaction of relevant agents and the static and dynamic entities in their vicinity. Here,

an important element is the attention that each agent places on other agents and road elements given the agent's immediate goals. For example, assume an AV aims to drive straight on a road with a pedestrian crossing ahead. It will pay attention to the crossing as well as any pedestrian present on or about to cross it. Naturally, it will also pay attention to the road boundaries in order not to veer off road. This information is crucial for the purpose of predicting the vehicle's future motion. In the general case, the predictor should capture such information for multiple vehicles at once if these vehicles are relevant to the AV. Thus, the model should choose a subset of vehicles among all agents (if it is performing vehicle motion prediction) and model the interaction the vehicles exhibit among each other as well as with other static entities or dynamic agents. These vehicles will ultimately have their motion predicted jointly in order to ensure consistency. This is especially important when predicting multiple futures, i.e. in multi-modal prediction. While modeling any interaction between entities, an important characteristic of the problem that the ML model should capture is bi-directionality, i.e. it should allow the possibility that two agents will not consider each other equally.

Uncertainty Modeling (iii)

In solving this problem, the prediction model needs to draw from its environment representation and interaction understanding in order to make inferences about the future. Since the intents of others are inherently uncertain and usually involves multiple distinct possibilities (e.g. turn left, right, or drive straight in an intersection), a useful way to approach the problem is to first model the possibilities that can occur in a scene. In this sense, the AV identifies the modes of the true ground-truth distribution of future motion of relevant predicted vehicles, which is often multi-modal. This forms the concept of multi-modal prediction. The modes can be directly modeled in the output space, i.e. as targets or areas on the road where a vehicle can go, or internally in the space of the model's latent representations, which are usually uninterpretable. Many models which have detailed knowledge of the environment (such as topological organization of lanes) use the former approach to facilitate future multi-modality modeling. Additionally, one can assign uncertainty to each mode regardless of the modeling approach, in the form of a parametric continuous distribution (e.g. a Gaussian with a mean and variance) or through approximating a non-parametric probability distribution (e.g. a heat map over discrete grid cells around an area of the road). Naturally, an important issue when modeling individual modes is to predict the probability of each occurring mode. Here, a model can assign scores that approximate individual probabilities to each mode. Finally, when modeling any type of uncertainty, it is important that a model discerns whether the uncertainty is coming from variability in the data (e.g. turning left or right), so-called aleatoric uncertainty, or from the model's lack of knowledge (e.g. by

not having seen a certain scenario), so-called epistemic uncertainty. This is a difficult question that requires considering the data the model is trained on in order to have a comprehensive answer.

Trajectory Generation (iv)

This stage involves using the future uncertainty representation from the previous step to generate trajectory candidates. Conditioned on the modes, the predictor should generate plausible trajectories as final model outputs. This step hinges on the previous stage; if a model outputs discrete modes in the form of targets, the task is to "attach" trajectories to these targets. In contrast, if a multi-modal probability distribution is provided, then the trajectory generation stage consists of sampling from this distribution in a representative and meaningful manner. In this context, the goal is to provide a set of diverse trajectories containing critical samples rather than those that do not affect ego planning. Finally, the obtained trajectories should be feasible in the sense that they respect the kinematic constraints of the vehicle.

It is apparent that the previously presented sub-problems are sequential, i.e. solving each opens the door to solving the following one. Consequently, if a previous sub-problem is not addressed well, it impedes solving the following one. For example, it is hard to have quality final trajectory candidates in case the model is not able to consume environment information and relate it to known objects in the first place.

Finally, an important consideration in all four stages is of a temporal nature. Usually, only past information is used for the first two stages while the last two deal with the future. Thus, the longer the past horizon is, the more information can be extracted to predict the future. Similarly, the longer the prediction horizon is, the harder it is to make accurate inferences on high-level behaviors as well as fine-grained trajectories. Thus, special emphasis is placed on capturing this larger fallibility in the latter sections of the prediction horizon in stages (iii) and (iv). As will be seen in Sec. 2.2, some methods of generating trajectories in stage (iv) are naturally pertinent to combating such errors.

1.4 Method

This section describes the approach toward solving the overall problem whose sub-tasks were outlined in the previous chapter. What kind of an ML model could solve these problems? Guided by the problem decomposition, the ML model first needs to have sufficient representation capacity to digest rich contextual information by building useful internal latent representations. The structure

of the latent representations should facilitate the modeling of interaction between traffic entities as well as future uncertainty. Then, the model needs to be able to map its latent state to multiple proposals of future motion, specifically multiple realistic trajectories for each agent of interest.

The ML model should be trained to extract maximal relevant information from the data at hand. The training approach can be placed within a general learning paradigm, i.e. supervised, self-supervised, or unsupervised learning. In this context, the training task is to parameterize a model that provides a suitable explicit or implicit approximation to the distribution in Eq. (1.1). The key to training such a model is to provide it with sufficient amount of data that exhibits all relevant behavior in a balanced manner. Given ground-truth driving data, which includes trajectories and context information, the model can learn to imitate[2] future trajectories given past context. It does so by employing supervised or self-supervised training objectives. In the purely supervised case, the model would directly receive a loss value for the error that it makes on the task at hand, i.e. a deviation of the predicted trajectory from the ground-truth future. In contrast, the self-supervised case involves learning relevant secondary tasks from the data prior to performing prediction. For example, the model could be trained to predict environment context features in addition to trajectories. This would help the model learn useful latent representations since they are used for meaningful auxiliary tasks.

In terms of modeling latent features, ML-based trajectory predictors are usually guided by two general approaches, *discriminative* and *generative*. In the first approach, a deterministic mapping between input context information $\mathbf{X}$ and latent features $\mathbf{z}$ is learned first, i.e. $\mathbf{z} = f(\mathbf{X})$. Then, another mapping, which is usually also deterministic, maps the latent feature vector $\mathbf{z}$ to a fixed number of future trajectories $\mathbf{Y}$ for multiple vehicles of interest, i.e. $\mathbf{Y} = g(\mathbf{z})$. Some models learn a probabilistic mapping, i.e. $\mathcal{G}(\mathbf{Y}|\mathbf{z})$ or $\mathcal{G}(\mathbf{Y}|\mathbf{z}, \mathbf{X})$. Regardless of the output mapping, these models can loosely be placed under the umbrella of *discriminative* models, where a probabilistic model either learns a conditional probability distribution $\mathcal{P}(\mathbf{Y}|\mathbf{X})$ or a deterministic model directly approximates samples from it. Another class of approaches are *generative*; they usually learn a proxy for the generative process of the data, i.e. the joint distribution $\mathcal{P}(\mathbf{X}, \mathbf{Y})$ between a context $\mathbf{X}$ and the ground-truth future $\mathbf{Y}$. For this class of models, latent features $\mathbf{Z}$ can facilitate the modeling of the joint distribution between the context and the ground-truth. For example, the CVAE [95] framework describes latent features probabilistically as samples $\mathbf{Z} \sim \mathcal{P}(\mathbf{Z}|\mathbf{X}, \mathbf{Y})$ from an analytically describable distribution $\mathcal{P}(\mathbf{Z}|\mathbf{X}, \mathbf{Y})$. The availability of such a distribution of latent features

[2]Some approaches place trajectory prediction in the context of Imitation Learning (IL) without interaction with the environment [90].

enables sampling[3] and transforming samples $\mathbf{Z}$ into output trajectories through a deterministic mapping, $\mathbf{Y} = g(\mathbf{Z}, \mathbf{X})$. Overall, this thesis contains approaches for modeling latent features which pertain to both discriminative and generative probabilistic architectures.

1.4.1 Architecture

This section introduces a general ML architecture capable of handling the problems outlined above. This architecture serves as a template wherein the individual contributions are placed. The architecture is inspired by the autoencoder model [63, 64], which is a seminal approach in unsupervised learning. The autoencoder compresses input data into a latent representation via an encoder network and reconstructs the same type of data from the compressed representation via a decoder network. Note that the present trajectory prediction task differs from the classical autoencoder use-case as the same type of data is not compressed and then recreated; the aim is to generate future motion predictions from historical context data, which is usually of a different nature. Nevertheless, core concepts such as the encoder and the decoder can be borrowed. In this context, these components act as a bottleneck employed to force the model to extract the information that is sufficient for the prediction task and discard irrelevant information. Furthermore, the emphasis is placed on the modeling of the latent space as well as the employed learning paradigm which shapes the training approach, e.g. a purely supervised or a self-supervised approach. See Fig. 1.3 for a visualization.

Encoder (a)

The encoder component is the entry point of the architecture. It is tasked with digesting the potentially high-dimensional context information from the chosen environment representation. Relevant sub-problems for this component are environment representation (i) and interaction modeling (ii). The encoder is constrained by the choice of environment representation – if it receives an image it needs to be able to process such data, thus a CNN is a fitting choice. Similarly, graph-based representations require GNNs or Transformers [101]. By distilling the input data into the network parameters under given training conditions, the encoder captures the information within the data and translates it into uninterpretable features that govern the dynamics of the problem, such as the relationship between vehicles of interest and other agents or road elements.

[3] In inference, $\mathcal{P}(\mathbf{Z}|\mathbf{X}, \mathbf{Y})$ is not used since $\mathbf{Y}$ is not available. Thus, CVAEs employ another latent distribution $\mathcal{P}(\mathbf{Z}|\mathbf{X})$ in inference that is matched to the training latent distribution in training.

Latent Model (b)

This component in the general ML model establishes assumptions about the structure of internal representations within the model. This internal structure can facilitate the analysis of interactions and the modeling of future uncertainty. Practically however, the latent model defines an internal structure for the output of the encoder. The latent model is therefore closely connected to the encoder since the output of the latter must conform to the chosen structure of the former. For example, the latent model can be realized in a fully deterministic manner, where it is simply a tensor of latent features that is then later directly propagated to the decoder. In contrast, a probabilistic latent model asserts a distributional assumption on the latent space and thus propagates a distribution through the decoder, usually by means of sampling and transforming individual samples. Such a distribution can take the form of a Gaussian distribution or a Gaussian mixture. A probabilistic latent space is an important concept in variational inference, where latent variables are often used to find a lower-dimensional structure in the form of a distribution within higher-dimensional data. This structure is in general not observable in the data itself. In the case of a probabilistic latent model, the latent model stage would impose a structure for the posterior distribution of latent variables given the input data and the ground-truth future. This structure would be defined in the form of a specific analytically-defined prior distribution. Ultimately, it can serve as a basis toward addressing the issues in the future uncertainty modeling stage (iii).

Decoder (c)

The decoder provides the output of the model. It is the component which transforms the internal latent model into a useful trajectory-level output representation. The task of the decoder is to generate realistic outputs given the latent structure, thus addressing the trajectory generation stage (iv). With a probabilistic latent structure, the decoder should have a mechanism to translate the distribution of uninterpretable variables into the output space, usually via sampling. In contrast, a deterministic latent space necessitates a mere (nonlinear) transformation of the latent features into the output space. Regardless of the choice of latent model, the decoder can draw on hybrid structures in its architecture and incorporate fully-differentiable kinematic models in addition to learned layers. This ensures realistic, kinematically-feasible trajectories.

Learning Paradigm (d)

The described structure comprising an encoder, a latent model, and a decoder is jointly trained under the assumptions of a certain learning paradigm employed

in order to achieve a set of training objectives. The main objective when training a prediction model is to ensure the accuracy and realism of generated predictions. This objective can be characterized as pure imitation of ground-truth trajectories observed in the training dataset. Thus, the model aims to reproduce the modes of the ground truth distribution represented by the training dataset, without necessarily extensively covering its support[4]. Toward this purpose, the model is trained in a supervised-learning manner by penalizing the deviations from ground-truth future data. Each prediction architecture proposed in this thesis includes loss functions promoting such objectives in training. Additional training criteria can promote a certain latent space structure. For example, variational models such as VAEs [19, 58] and CVAEs [95] applied in this thesis employ objectives that impose a certain structure on the probabilistic latent space, i.e. a prior distribution. This is done in addition to a term promoting the realism of the decoder output (i.e. the ground-truth imitation objective).

Another core topic of this thesis is the usage of self-supervision schemes for training prediction models. These are introduced in order to improve the components of the entire model, i.e. the parameterizations of the encoder, decoder, and the latent model. Great practical benefits can be found by using additional objectives that teach the model to use its latent features in predicting other quantities than trajectories. This is reasonable since in reality the scene dynamics are governed by a joint distribution of trajectories and contextual environment information. In the context of additional objectives, future embeddings of the environment are used, i.e. the model's internal representations of future context data, in addition to historical context data. Latent feature imitation criteria can then be used in the loss functions alongside ground-truth trajectory imitation. These criteria can be placed into an auto-regressive scheme, where multiple self-supervised loss functions over evolving time segments reap additional benefits in training. Overall, the choice of the learning paradigm, be it purely supervised or including self-supervised terms, can greatly improve overall performance and offer better solutions to individual prediction sub-problems (i)-(iv).

1.5 Experiments

This section shortly describes the general experimental setup used across the proposed trajectory prediction models. Since the aim is to advance motion prediction research for the purpose of enabling general autonomy, evaluation use-cases are not restricted to specific types of driving scenarios. Thus, in training and evaluating the proposed ML models, publicly-available large-scale datasets of naturalistic driving are used. They comprise scenarios such as

[4]Methods that aim to achieve extensive coverage should include a mechanism for detection of out-of-distribution scenarios. See [31] for an example.

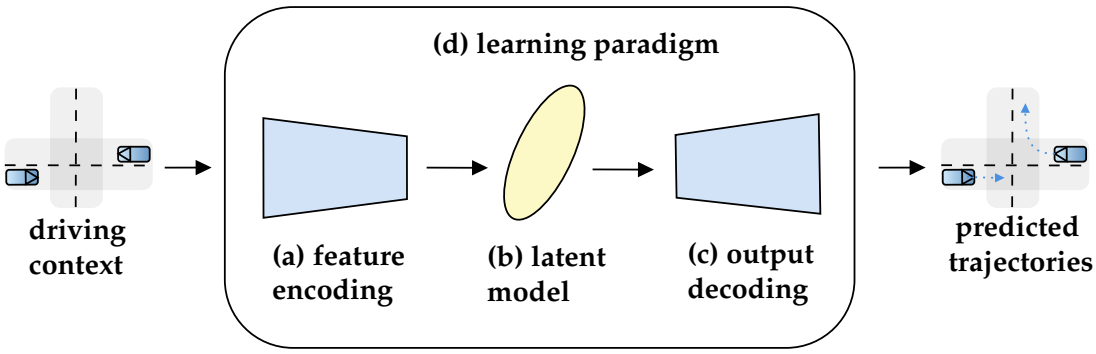


Figure 1.3: Breakdown of the general model structure: generic context data is fed into an encoder component that generates an internal latent representation. The latent space is modeled under certain assumptions on its structure, e.g. deterministic or probabilistic. The latent space is then transformed via the decoder into realistic predicted future trajectories. The encoder, latent model, and the decoder are trained under the assumptions of a learning paradigm designed to extract the most knowledge from the data at hand.

intersections, roundabouts, and highway merges exhibiting rich interaction among its participants. These include traffic agents such as passenger cars, trucks, bicycles, pedestrians, etc. For example, the INTERACTION dataset [115] is used, which contains scenes recorded in the United States, Germany, China, and Bulgaria. This dataset contains critical behaviors such as aggressive driving, negotiations in tight spaces, as well as near-collisions.

Using established, large-scale datasets for the proposed models allows for a comparison with other approaches from the state of the art. Usually, the training and validation splits are explicitly defined. Some datasets also provide online leaderboards with scenarios not available otherwise. Generally, the accuracy of the predicted trajectories is evaluated through generally accepted metrics such as Minimum Average Displacement Error (minADE) and Minimum Final Displacement Error (minFDE), widely adopted throughout the AD prediction community. The minADE represents the average Euclidean distance of the predicted xy positions against the ground-truth trajectory for the closest predicted trajectory among a predetermined number. Similarly, the minFDE considers the final point instead of the entire trajectory. See Fig. 1.4 for an illustration. In the presented probabilistic approaches, metrics such as the Negative Log Likelihood (NLL) are used, which consider the predicted probability distribution (in terms of its mean and covariance) by evaluating the likelihood of the ground-truth trajectory. In such cases, uni-modal and multi-modal probability distributions (i.e. mixture distributions) are evaluated.

In addition to trajectory prediction approaches, this thesis contributes fundamental advances to generative model architectures, more specifically the VAE model.

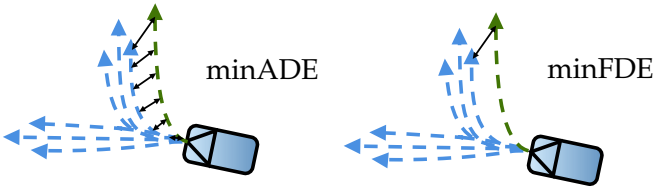


Figure 1.4: Visualization of the minADE and minFDE metrics: the minADE represents the average Euclidean distance between the ground-truth future trajectory (green) and the closest trajectory among the predicted trajectories (blue) for all points. Similarly, minFDE only considers the average distance between the final points.

The primary motivation of the proposed VAE improvements is to use them as a stepping stone toward developing more effective CVAE models in trajectory prediction. However, the VAE model is used throughout a wide variety of domains. Thus, in order to evaluate such approaches, standard image generation tasks are employed. Example datasets in this domain are CIFAR10 [65], CelebA [73], and Fashion-MNIST [109], where the quality of reconstructed images after compressing and decompressing them from the model's latent space is gauged, as well as the realism of samples from the latent space. For this purpose, the well-established FID score between datasets of images [47] is used. By ensuring the proposed models perform well in such tasks, wide applicability across the ML landscape is enabled.

2 Feasible Trajectory Generation and Self-Supervision in Prediction

This chapter approaches the overall trajectory prediction problem from the perspective of outputs. In this context, the issues of trajectory generation (iv) within the overall problem definition are discussed (see Sec. 1.3.1), while other relevant sub-problems assume simplistic solutions. The research questions tackled in this chapter pertain to the kinematic feasibility of generated trajectories and the process of constructing outputs, i.e. whether trajectories are constructed in an autoregressive or a one-shot manner. The corresponding model component relevant for this stage is the output decoder (c), see Sec. 1.4.1. An additional related topic of this chapter is the learning paradigm (d) employed in order to extract the maximal knowledge from the data at hand. Specifically, the usage of self-supervision in training is discussed, which is enabled by the employed mechanism of generating trajectories. The topics of this chapter cover the following publications.

- Paper I [A1]:
  - Title: *Self-Supervised Action-Space Prediction for Automated Driving*
  - Authors: Faris Janjoš and Maxim Dolgov and J. Marius Zöllner
  - Venue: 2021 IEEE Intelligent Vehicles Symposium (IV)
  - Full version: Included Publications: Paper I
- Paper II [A2]:
  - Title: *Bridging the Gap Between Multi-Step and One-Shot Trajectory Prediction via Self-Supervision*
  - Authors: Faris Janjoš and Max Keller and Maxim Dolgov and J. Marius Zöllner
  - Venue: 2023 IEEE Intelligent Vehicles Symposium (IV)
  - Full version: Included Publications: Paper II

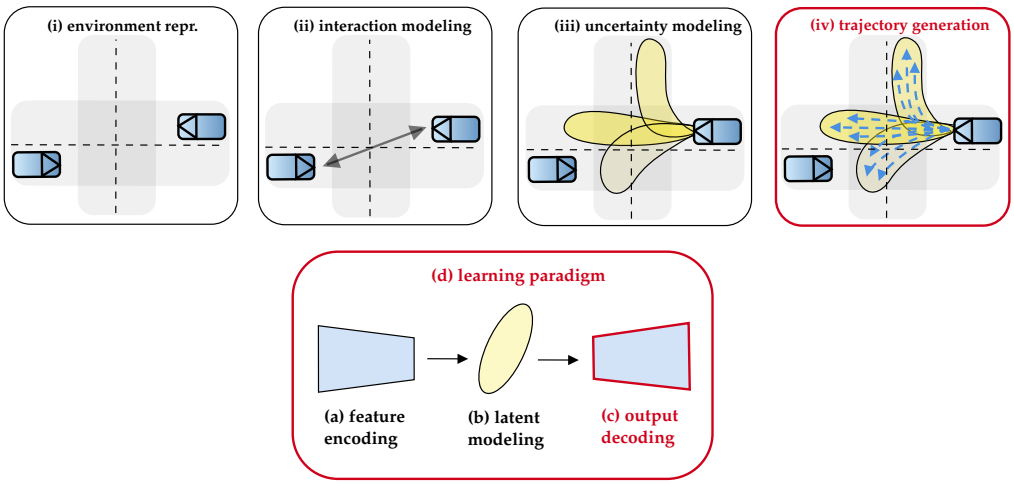


Figure 2.1: Problem-wise, Paper I [A1] addresses the trajectory generation subproblem (iv) (top). Model-wise, it focuses on the output decoding (c) and the learning paradigm (d) (bottom).

2.1 Self-Supervised Action-Space Prediction for Automated Driving

This section concerns Paper I [A1], which offers solutions for generating predictions that exhibit high-quality motion characteristics, e.g. kinematic feasibility. Thus, problem-wise, its contributions can be placed into the trajectory generation context in stage (iv) and the decoder component (c) of the overall model. Furthermore, the solutions employed open the door toward novel self-supervised training procedures. Thus, another model-wise focus of the paper is the employed learning paradigm (d). Fig. 2.1 shows the placement of Paper I [A1] within the problem-wise and model-wise divisions.

2.1.1 Research Context

When framing vehicle trajectory prediction as a learning problem, it is important to consider that aspects of the problem are analytically modellable, e.g. vehicle kinematic constraints. Many works neglect such knowledge by not incorporating well-known heuristics such as kinematic motion models into their architectures. For example, a neural network receives a rich combination of context features as inputs and directly learns the relationship between these features and future positions in time. Foregoing motion heuristics impedes the learning of smooth and realistic trajectories since the networks are tasked with learning a physical motion model implicitly from data instead of explicitly using an analytical definition.

In the publication context of Paper I [A1], approaches that address aforementioned issues are works such as [23] and [88]. The former introduces a differentiable kinematic bicycle model at the network output, which ensures kinematically feasible outputs. However, the approach it employs is limited; the networks are tasked with learning an inverse motion model from data by mapping past positions to future actions (e.g. acceleration and steering angle). The approach in [88] entirely avoids the need to regress future motion by performing classification from a large set of physically feasible trajectories instead. Such set-based approaches have not been common in the literature ever since; they put hard demands on context understanding and lack the flexibility of regression-based approaches which profit from strong priors of past motion.

After publication of Paper I [A1], other approaches addressing the problem of incorporating motion constraints emerged. For example, [39] applies a tracking controller along a reference path to stabilize neural-network-predicted trajectories. In contrast, [100] investigates regression of polynomial trajectory parameters as well as actions, albeit with a simpler model than the bicycle model in [23]. Furthermore, [96] applies a set-based approach, however, it circumvents the issues of [88] by using a model-based planner to generate a set of strong candidate trajectories that are reachable, feasible, and map-aware, and only subsequently selects trajectories from this set. Finally, works such as [106, 105, 55] demonstrate the utility of learnable neural Ordinary Differential Equation (ODE)s in trajectory prediction – they provide another path toward augmenting learning-based solutions with model-based knowledge.

In terms of the learning paradigm, which is another focus of Paper I [A1], a large majority of prediction approaches strictly follow a supervised learning paradigm. In accordance with the depiction in Fig. 1.3, such models embed past context in their encoder and directly predict future trajectories via the decoder (usually assuming a deterministic latent model), training the entire model only by penalizing output trajectory deviation to the ground truth. In this context, only historical information is used to predict the future and the only predicted quantities are trajectories. Naturally, such constraints are reasonable in inference, however in training, one can augment the setup by exploiting meaningful auxiliary tasks that utilize both historical and future ground-truth data. Such approaches are sparsely explored in the trajectory prediction literature; notable are [112], which promotes temporal and spatial consistency of trajectories through self-supervised tasks, [33], which introduces additional map completion tasks, [77], which proposes contrastive losses with perturbed inputs, and [34], which introduces additional online fine-tuning based on scenarios observed in inference. However, in the field of IL and (model-based) Reinforcement Learning (RL), learning environment models (so-called world models) as an auxiliary task facilitating planning is a major area of focus [42, 108, 61]. A notable application of such ideas into the field of AD can be found in [50], where prior to generating actions, the AV predicts rich observations of the environ-

ment. Finally, a different flavor of self-supervision can be found in concurrently learned forward and inverse models (e.g. reconstruction of past inputs given future actions), first developed for robot motion planning in [1] and applied for AD planning in [110].

2.1.2 Contributions

The contributions of Paper I [A1] relate to incorporating motion constraints into the trajectory generation problem in stage (iv) (via decoder component (c) adaptations) as well as introducing a novel learning paradigm (d) for trajectory prediction. In terms of motion constraints, Paper I [A1] introduces the action-space prediction framework, which fully casts the learning problem into the space of actions (i.e. control signals). Specifically, the decoder network is tasked with mapping past actions to future actions, which decouples modellable aspects of motion from learning. Thus, only weak correlations between actions are required to be captured by the network, as opposed to a full forward or inverse motion model in the case when positions or states are used as inputs or outputs. Nevertheless, future positions are obtained by unrolling actions via a differentiable kinematic bicycle model. The models are trained and evaluated on inD [10] and rounD [62] datasets and experimentally demonstrate the superiority of this approach over the motion modeling approach in [23].

Learning actions encourages a novel interpretation of the prediction problem. If a vehicle aims to perform a certain action, how will its environment react? The proposed self-supervised action-space prediction model in Paper I [A1] aims to answer this question by first predicting the evolution of the environment, in the form of latent features, and only subsequently predicting future trajectories given the future latent features. Training is enabled in a self-supervised manner, where the model utilizes observations from the future (available in the training data) to train the feature prediction. Furthermore, by having access to future features, the model can "invert" the problem and introduce a complementary task of predicting past trajectories given future features, inspired by [110]. The two self-supervised tasks of feature prediction and past trajectory prediction offer strong additional training signals and extend the learning paradigm (d) beyond classical supervision. Their superior utility to the straightforward procedure of encoding the past and decoding the future, ubiquitous in trajectory prediction, is empirically validated.

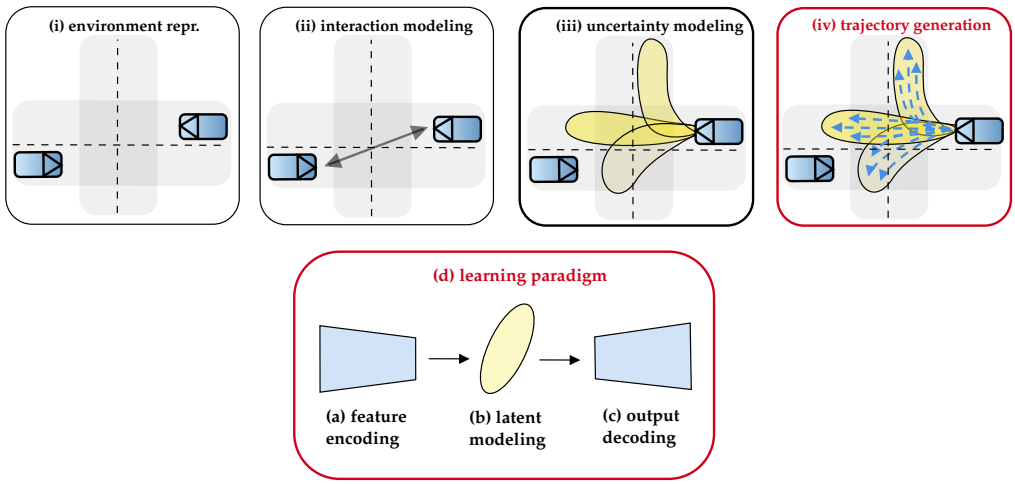


Figure 2.2: Problem-wise, Paper II [A2] addresses the trajectory generation sub-problem (iv) (top). Model-wise, it focuses on the learning paradigm (d) (bottom).

2.2 Bridging the Gap Between Multi-Step and One-Shot Trajectory Prediction via Self-Supervision

This section concerns Paper II [A2], whose contributions directly build on the contributions of Paper I [A1]. Instead of focusing on kinematic properties of motion, the approach considers temporal aspects of trajectory generation in stage (iv) of the problem-wise decomposition. More specifically, questions related to employing one-shot or autoregressive modeling procedures for trajectories are answered. This unlocks novel perspectives within the previously presented self-supervised learning paradigm. See Fig. 2.2 for the placement of Paper II [A2] within the problem-wise and model-wise divisions.

2.2.1 Research Context

The state of the art in trajectory prediction literature is dominated by one-shot approaches, where ML-based models regress entire trajectories at once. This is in contrast to autoregressive approaches, grounded in the field of time-series modeling, which build predictions sequentially over single time steps. Mathematically, a one-shot predictor would model the marginal distribution of a future trajectory $\mathbf{Y}$, $\mathbf{Y} = [\mathbf{Y}_1, ..., \mathbf{Y}_T]$, given a context $\mathbf{X}$ (i.e. Eq. (1.1)) over T individual time steps, $\prod_{i=1}^{T} \mathcal{P}(\mathbf{Y}_i|\mathbf{X})$. In contrast, an autoregressive model would construct $\mathcal{P}(\mathbf{Y}_1|\mathbf{X}) \prod_{i=1}^{i=T-1} \mathcal{P}(\mathbf{Y}_{i+1}|\mathbf{Y}_i, \mathbf{X})$. Even though autoregressive

modeling is a theoretically more sensible approach to the problem (since it captures causality), one-shot predictors have been more successful in trajectory prediction due to their simplicity, ease-of-use and good performance in general. However, autoregressive models have the favorable properties of shorter time horizon requirements as well as the ability to include more information at each predicted time step (e.g future observations in the world model of [50]). To the best of the author's knowledge, approaches that combine ideas of one-shot and autoregressive predictions did not exist in the context of the publication of Paper II [A2]. In the field of ML-based trajectory planning for AD however, tree-based approaches are an example of successfully using ideas from autoregressive modeling. Approaches such as [18, 53] generate one-shot plans over short time segments and combine them into a tree structure that enables reasoning over long horizons.

Paper II [A2] focuses on designing a novel middle ground between one-shot and autoregressive approaches for trajectory generation, which expands the capabilities of the self-supervised learning procedures introduced in Paper I [A1]. Its relevant literature can therefore be found beyond trajectory prediction in the autoregressive world models of [42, 108, 61, 50] and inverse models of [1, 109]. Furthermore, the approaches developed in Paper II [A2] are methodologically inspired by the autoregressive models of [102, 43, 76, 69] employed in IL and RL. For example, [69] shows that rolling out multiple parallel autoregressive predictions starting from incremental time steps in the future enhances performance. Similarly, [76] empirically shows that multi-step losses increase the reward in a model-based RL setting. In the context of trajectory prediction approaches, Paper II [A2] shares its relevant literature with Paper I [A1] in terms of self-supervision, e.g. approaches such as [112, 33, 77, 34]. Finally, for general examples of one-shot regression approaches in trajectory prediction, see [21, 28, 100]. Similarly, representative examples of autoregressive approaches can be found in [90, 74, 94].

2.2.2 Contributions

The contributions of Paper II [A2] include a first-of-its-kind middle ground between autoregressive and one-shot trajectory prediction. The approach splits the trajectory generation process into segments – for instance, a three-second prediction is performed by chaining three consecutive one-second prediction segments. The powerful self-supervision schemes introduced in Paper I [A1], consisting of future feature prediction and past trajectory prediction, are carried over into the proposed segment-wise prediction setting and perform particularly well. Furthermore, the segment-wise formulation helps to combat the accumulating error over the latter parts of a prediction horizon. In this context, multiple parallel prediction paths are realized simultaneously, inspired by [69], e.g. a

three-second prediction starting at the present, a two-second prediction starting one second in the future, and a one-second prediction shifted two seconds into the future. The model architecture is kept deliberately simplistic, with simple CNNs that show limited capability of context understanding and interaction modeling used throughout the approach. This is done for the purpose of isolating the effects of segment-wise prediction and self-supervision. Despite the simplistic components, the proposed approach shows strong performance over competitors and achieves 3rd place w.r.t. minADE and minFDE metrics on the INTERACTION dataset leaderboard at the time of publication in February 2023. Finally, another important contribution of Paper II [A2] pertains to procedures for model confidence estimation, i.e. epistemic uncertainty. This question is addressed in the context of the segment-wise prediction formulation – since the model can chain predictions iteratively, how confident is it that they are accurate, especially in the latter segments? The proposed methodology is thus extended with an approach to gauge the evolving uncertainty over consecutive segments. The uncertainty is evaluated with two proposed metrics, a past-prediction-based and a dropout-Bayesian metric, based on the insights in [32]. A contribution toward the uncertainty estimation was first developed in [S3].

2.3 Summary

This chapter offered solutions for generating trajectories that incorporate physical constraints and are constructed in a manner that combines the best of the worlds of autoregressive and one-shot output modeling. The employed action-space paradigm allowed to reason about the influence of actions on internal latent states, which enhanced the learning mechanisms beyond pure imitation of future ground-truth trajectories. The segment-based temporal framing modeled causal dependencies within a balanced process of generating the output trajectories. Other aspects of the overall trajectory prediction problem assumed simplistic solutions – thus, important challenges remain in terms of reasoning about the environment and the interactions within it. Solving these issues is a basis for addressing the uncertainty of the future, which the generated trajectories must consider. The following chapters address these issues.

3 Environment Representation and Interaction Modeling via Attention

This chapter assumes the perspective of representing and analyzing input data and extracting information from it. The topics discussed are the issues of environment representation (i) and interaction modeling (ii) within the overall problem definition, see Sec. 1.3.1. The addressed research questions pertain to using object-level representations that aid the understanding and reasoning about the environment of the predicted vehicles and the relationships present among relevant entities, either dynamic agents or static elements. The corresponding model component relevant for these problems is the encoder (a), see Sec. 1.4.1, which digests contextual inputs into an internal structure capable of capturing relevant information. This chapter concerns the following publication.

- Paper III [A3]:
  - Title: *StarNet: Joint Action-Space Prediction with Star Graphs and Implicit Global-Frame Self-Attention*
  - Authors: Faris Janjoš and Maxim Dolgov and J. Marius Zöllner
  - Venue: 2022 IEEE Intelligent Vehicles Symposium (IV)
  - Best Paper Runner-Up Award
  - Full version: Included Publications: Paper III

3.1 StarNet: Joint Prediction via Star Graphs and Implicit Global Frames

This section concerns Paper III [A3], which proposes graph-based environment representations (i) that facilitate the understanding and reasoning about the given context. In terms of interaction modeling (ii), the approach in Paper III [A3] considers the problem in the context of joint prediction of multiple

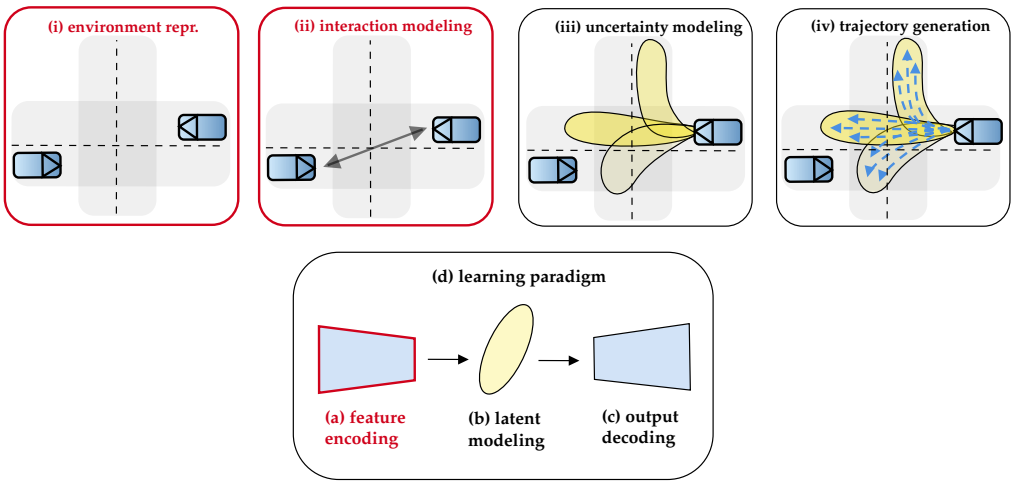


Figure 3.1: Problem-wise, Paper III [A3] addresses the environment representation (i) and interaction modeling (ii) sub-problems (top). Model-wise, it focuses on the feature encoding (a) (bottom).

vehicles simultaneously, which is a more holistic alternative to marginal, single-vehicle predictions. Model-wise, the onus is placed on the encoder component (a), which employs graph- and attention-based architectures that are capable of naturally modeling relationships between relevant entities (e.g. agents or map elements). See Fig. 3.1 for the placement of Paper III [A3] within the problem-wise and model-wise divisions.

3.1.1 Research Context

The question of environment representation in trajectory prediction is essential as it facilitates understanding a given driving context, which is crucial to make informed inferences about the future. Furthermore, it guides practical model choices since the employed models should be capable of processing a given input. Historically, many learned prediction approaches opted for a raster image representation, where different static and dynamic entities (e.g. map, agents) are combined into a fixed-size grid that can easily be fed into a CNN encoder. The simple image construction and ease-of-use of CNNs has made such prediction approaches popular, however, it quickly became apparent that their ability to extract knowledge is limited, owing to the issues of overly implicit representation which were exacerbated by fundamental properties of CNNs (see Environment Representation (i) in Sec. 1.3.1 for a discussion). In contrast, explicit object-level representations used in graphs and GNNs or within attention mechanisms [101] have meanwhile become a de facto standard for learning-based trajectory prediction models. Importantly, the object-level representation

of entities aids reasoning about the underlying relationships within an assumed interaction model.

Relationships between entities are modeled for the purpose of generating accurate predictions for a number of vehicles of interest interacting among each other and with other entities on the road. If these predictions are generated in the form of marginal distributions where each vehicle is predicted separately (e.g. the $\mathbf{Y}$ in Eq. (1.1) refers to the trajectory of a single vehicle), other predicted vehicles are considered only implicitly via their future in expectation. Furthermore, this approach is computationally disadvantageous since predictions have to be redone for every vehicle of interest. Thus, it is clear that joint prediction is a more principled and feasible approach. Importantly, joint prediction enables the explicit consideration of the bidirectional interaction between participants, i.e. the possibility of non-symmetric relationships, while marginal prediction only implicitly considers relationships. However, a difficult practical consideration in joint prediction pertains to the coordinate frames that features are represented in. One option is to represent the positional information on the road (e.g. the positions of map elements and traffic agents) in the reference frame of a single, pivot agent (e.g. using the AV as the origin) or a pre-determined global origin. However, this introduces a strong dependence on an arbitrary pose and impedes the models to be pose-invariant while learning. Consequently, generalization in interaction understanding and modeling is hampered.

The publication context of Paper III [A3] can be divided along two themes, graph-based map representation and joint interaction modeling. Regarding the former, relevant approaches are [33, 57, 87, 71, 114]. The VectorNet model in [33] is a seminal approach that aggregates map entities such as border lines and centerlines, represented as polylines (sequences of connected vectors), into fully-connected local graphs. Similarly, it represents the past motion of agents as polylines. Then, all local graphs are first aggregated into individual vectors, which are then aggregated again within a global, scene-level graph. The simplicity of the approach has made it very popular. However, questions surround the necessity of the fully-connected aggregation of potentially distal vectors within a single polyline, in turn making the relationship between an agent track and a single polyline rather indirect (via the aggregation of their local graphs). Alternatives to VectorNet are [71, 114], which account for topological semantic information by redefining graph convolution operations for longitudinal and lateral lanes. Further, [57, 87] construct a reference polyline and model the relationship of the predicted agent to lanes and sections along this polyline. Overall, in constructing graph-based map representations using geometric and topological information, approaches balance between simplicity (e.g. [33]) and intricacy (e.g. [57, 87, 71, 114]). In terms of joint interaction modeling however, key questions surround the reference frame used to represent positional information. Here, many approaches such as [93, 15, 21] use AV-centered reference frames and thus sacrifice pose invariance. In contrast, [57, 82] use local frames

but design the prediction pipeline to encode the context and decode trajectories repeatedly for each predicted vehicle, scaling linearly with the number of vehicles. Finally, approaches such as [71, 86] attempt to strike a balance by combining AV-centric context information with prediction in a local frame.

After the publication of Paper III [A3], a plethora of additional approaches utilizing object-level environment representations emerged. Considering the aforementioned qualities of intricacy and simplicity, approaches such as [28, 72] emphasize the former; they attempt to build a detailed topological lane graph and sample path proposals along it. A common factor among such approaches is a strong requirement on rich and accurate a priori map information. Thus, investigations of robustness to perturbations in this context have emerged in [3, 45]. Related line of research favoring intricacy are approaches that attempt to build a detailed knowledge graph capturing semantic information [83, 4]. A middle-ground in terms of the complexity is [100], which collects detailed positional information about the road network relative to an agent of interest but disregards connectivity. An approach emphasizing simplicity is [85], which uses similar map information to [100] but combines it with other contextual information in an early fusion manner[1] through a relatively simple attention-based encoder. In contrast to other approaches, [45] argues that center lines are a sufficient map representation; it shifts the prediction problem into the Frenet frame where lateral and longitudinal deviations to the center line are predicted. Similar insights on the importance of center lines are shown in [26] for the use case of open-loop ego planning in AD. Finally, considering joint interaction modeling and the importance of coordinate frames, a notable approach published after Paper III [A3] is [22]. It avoids the trade-off between pose-dependent AV-centric representations and the linear scaling of local-frame models by offering a convenient pair-wise relative representation between the agents and the lanes – said representation is reference frame agnostic and enables sampling goals along the lanes.

3.1.2 Contributions

Paper III [A3] proposes the StarNet approach for graph-based environment representation and joint interaction modeling. Building on the action-space prediction framework introduced in Paper I [A1], StarNet introduces attention-based encoders that model the relationship between predicted agents and their static environment as well as between dynamic agents. In the first step, agent-centric star graphs are constructed and processed via Graph Attention Network (GAT)s, offering a mechanism to relate the motion of a single predicted agent to fine-grained map elements (e.g. individual vectors within a polyline) via graph

[1] Various sources of input features are minimally processed; they are only projected to a common dimensionality before being tokenized in accordance with [101].

attention. Thus, it avoids the pitfalls of VectorNet [33], which aggregates entire polylines and in turn only indirectly models such relationships. Overall, it favors simplicity by requiring minimal map information such as road boundaries and lane markings but not connectivity. In the second step, a map-dependent interaction model is constructed by relating the star graph embeddings to the features representing the motion of other agents using self-attention [101]. In this way, the attention mechanism directly models the importance a single agent places on other agents. In the third step involving the generation of joint predictions, the StarNet model offers an approach to combine the local single-agent-centric representations from the previous step. Here, an additional attention layer is used, which builds an implicit global frame by combining the local-frame information. Importantly, the model ensures efficiency by combining graphs and attention layers of different scenes into single graphs and attention layers; non-interference is ensured by masking and block-diagonal graph adjacency matrices. Overall, StarNet achieves superior performance on the INTERACTION validation dataset to its competitors, achieving state-of-the-art minADE and minFDE metrics in its publication context.

3.2 Summary

The approaches presented in this chapter contribute solutions toward reasoning about the contextual information of a prediction scenario. The employed object-level representations enable to directly consider relevant entities, while attention-based structures help to model the relationships between them. These are used as a basis for considering the joint distribution of future motion – a more principled approach than marginalizing over single agents – in an unbiased manner of locally-centered predictions for each predicted vehicle. The following chapters build on the solutions presented here (and in the previous chapter) in order to address a major remaining issue: capturing the uncertainty of future motion.

4 Deterministic Multi-Modality Modeling via Multiple Decoders

This chapter shifts the focus toward reasoning about the future. In this context, simplistic solutions for addressing the issue of uncertainty modeling (iii) within the overall problem definition (see Sec. 1.3.1) are offered. The addressed research questions pertain to modeling multiple distinct futures that an entire scene can evolve into, with the constraint of using deterministic models with limited capabilities of capturing the full distribution in Eq. 1.1. Model-wise, the employed approaches are contained to the decoder component (c), see Sec. 1.4.1. This chapter concerns the following publication.

- Paper IV [A4]:
  - Title: *SAN: Scene Anchor Networks for Joint Action-Space Prediction*
  - Authors: Faris Janjoš and Maxim Dolgov and Muhamed Kurić and Yinzhe Shen and J. Marius Zöllner
  - Venue: 2022 IEEE Intelligent Vehicles Symposium (IV) workshop: From Benchmarking Behavior Prediction to Socially Compatible Behavior Generation in Autonomous Driving
  - Full version: Included Publications: Paper IV

4.1 SAN: Scene Anchor Networks for Joint Action-Space Prediction

This section concerns Paper IV [A4], which is a short study that addresses the challenges of modeling future motion uncertainty (iii) with deterministic models. It focuses on the multi-modality property of the ground-truth future distribution, where future motion is usually characterized by a set of distinct behaviors (e.g turning left, right, going straight). To this purpose, it applies simple modifications in the decoder stage (c) to promote the multi-modality of predictions generated jointly for multiple predicted vehicles, without relying on strong contextual heuristics such as map information. See Fig. 4.1 for the placement of Paper III [A3] within the problem-wise and model-wise divisions.

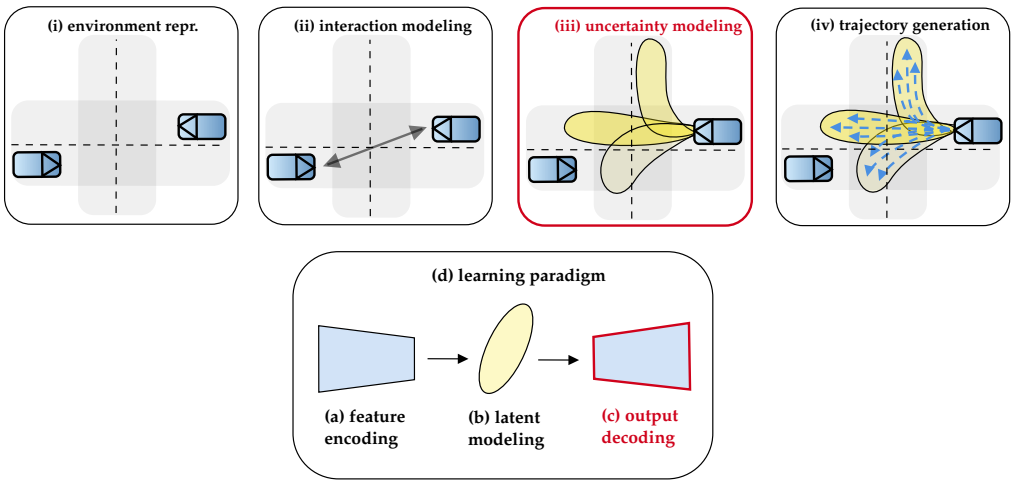


Figure 4.1: Problem-wise, Paper IV [A4] addresses the uncertainty modeling stage (iii) (top). Model-wise, it focuses on the output decoding step (c) (bottom).

4.1.1 Research Context

Probabilistic solutions for trajectory prediction tackle the inherent uncertainty of the future in a principled manner, i.e. by learning a probability distribution in Eq. (1.1). However, many approaches are fully deterministic; instead of learning Eq. (1.1), they directly predict a (usually fixed) number of samples from the distribution. By doing this, they forego the ability to sample a distribution, to directly obtain likelihoods from it, as well as exhibit a generally lesser ability to model the inherent uncertainty of the problem compared to probabilistic models. Nevertheless, in the process of approximating Eq. (1.1) via a set of trajectories, many facilitating heuristics can be utilized. For example, the concept of intermediate goals (often called targets or anchors) that are predicted prior to trajectories can be useful. Here, the intuition is that the goals, usually grounded to a map element such as an area of the road or a lane, refer to intents and thus capture most of the uncertainty of the future, especially the multi-modality. Albeit a reasonable approach, in the absence of detailed map information (e.g. the drivable space or the road topology informing the target selection on specific lanes) it is often difficult to construct such goals. Furthermore, in the joint prediction use case where multiple scene-level outputs are predicted, it is non-trivial to ensure scene consistency for multiple proposals using only map heuristics.

In the publication context of Paper IV [A4], a large majority of approaches addressing future uncertainty with deterministic solutions focus on intermediate goal construction. The approaches in [79, 116, 41] share characteristics by predicting map-conditioned endpoints of a trajectory prior to generating the entire trajectory itself. In the case of [116], multiple potential endpoints are constructed using map prior information (e.g. by discretizing lane centerlines surrounding a

predicted vehicle and predicting offsets to the points) and then used to condition full trajectories. The approach in [41] builds on [116] by providing a denser set of goals that are refined into goal sets with the help of a set predictor trained with offline optimization techniques. In contrast, [36] uses convolutional networks to predict a heatmap probability distribution, which is then sampled in a deterministic manner to obtain trajectory candidates. A notable approach that does neither uses strong map heuristics nor predicts auxiliary representations is [27], which designs a multi-generator Generative Adversarial Network (GAN) for pedestrian prediction, where distinct generators contribute to a multi-modal output distribution with separable modes.

After the publication of Paper IV [A4], many extensions and improvements of the aforementioned approaches emerged. For example, [37] builds on the heat map representation in [36] by constraining it to the topological lane graph via a GNN encoder, while [38] extends the heat map approach to the joint prediction use case by combining the trajectories sampled from the heat map in a scene-consistent manner. Utilizing discrete goals instead of areas, the approaches [28, 72] mentioned in the context of Paper III [A3] build an intricate topological lane graph and constrain the generated trajectories along paths in the graph. Similarly, [74] extends the notion of goals along entire trajectories, introducing keyframes that trace out points along a trajectory and offer a stronger conditioning than a single endpoint. Finally, [45] can also be considered a goal-based approach. Here, the goals are centerlines, and longitudinal and lateral offsets in the Frenet frame are predicted.

4.1.2 Contributions

The contributions of Paper IV [A4] consist of a relatively simple approach to promote multi-modality in deterministic models in the absence of strong map conditioning or auxiliary outputs. Focusing on joint predictions, the approach extends the setup from Paper III [A3] with multiple decoder heads, where different scene-level realizations are provided by individual decoder heads. Importantly, only a single head is trained at a time (the one whose outputs match the ground-truth the best), helping each head to specialize in a specific scene-level future given the common environment context. This procedure helps against mode collapse[1] by ensuring that for a specific sample, only the weights of a single head are updated. Furthermore, the model allows for additional, fine-grained modeling of future uncertainty given a scene-level realization by outputting multiple trajectories for each vehicle given a scene mode. Through so-called motion modes, minor motion deviation within a scene realization can

[1] A well-known phenomenon in multi-modal predictions where multiple outputs of a network collapse to a single mode as a result of the model minimizing the incurred loss value by averaging out the outputs.

be modeled. An additional contribution of Paper IV [A4] relates to the usage of a novel, outlier-robust adaptive loss function carried over from [6]. It is proposed as an alternative to the L_1, Huber L_1, and L_2 loss functions, which are ubiquitous in trajectory prediction but do not exhibit consistent behavior w.r.t. outliers (i.e. they either ignore or harshly penalize samples that incur very large loss values). The adaptive loss function in contrast automatically learns the best outlier treatment to achieve the highest performance. Overall, even though the model proposed in Paper IV [A4] exhibits good results on the INTERACTION dataset (outperforming the approach in Paper III [A3] in terms of minADE and minFDE), its ability to model future uncertainty is limited. This stems from its deterministic formulation, resulting in inherent limitations in modeling stochasticity, as well as the lack of any intricate contextual heuristics.

4.2 Summary

This chapter offered a simple approach for modeling the uncertainty of the future in a deterministic manner. The emphasis was placed on joint prediction models that output scene-consistent modes for multiple predicted vehicles at once. The proposed solution involved a multi-headed model where different heads specialize to different scene realizations. Despite a reasonable performance, such deterministic approaches are limited in their ability to model the problem described by Eq. (1.1). Capturing the uncertainty of the future in a comprehensive manner requires probabilistic solutions that reason about the joint distribution of the context and the ground-truth future induced by it; the following chapter presents such approaches.

5 Latent Variable Models in Probabilistic Prediction and Beyond

This chapter focuses on internal representations that enable reasoning about the future in a comprehensive manner. Thus, it deals with the topic of uncertainty modeling (iii) within the overall problem definition, see Sec. 1.3.1. To this end, latent variable models are employed for prediction; they are a class of probabilistic generative models that compress the joint distribution of the context and future trajectories into a tractable latent space. The research contribution presented in this chapter deals with fundamental properties of VAEs, a class of latent variable models, and comprises methods applicable to the base VAE architecture (and the derived CVAE) and translatable toward trajectory prediction applications. Model-wise, the employed approaches observe the latent modeling (b) component of the overall architecture, see Sec. 1.4.1. This chapter concerns the following publications.

- Paper V [A5]:
  - Title: *Unscented Autoencoder*
  - Authors: Faris Janjoš and Lars Rosenbaum and Maxim Dolgov and J. Marius Zöllner
  - Venue: 2023 International Conference on Machine Learning (ICML)
  - Full version: Included Publications: Paper V
- Paper VI [A6]:
  - Title: *Conditional Unscented Autoencoders for Trajectory Prediction*
  - Authors: Faris Janjoš and Marcel Hallgarten and Anthony Knittel and Maxim Dolgov and Andreas Zell and J. Marius Zöllner
  - Venue: submitted to 2024 European Conference on Computer Vision (ECCV) workshop: Event Detection for Situation Awareness in Autonomous Driving
  - Full version: Included Publications: Paper VI

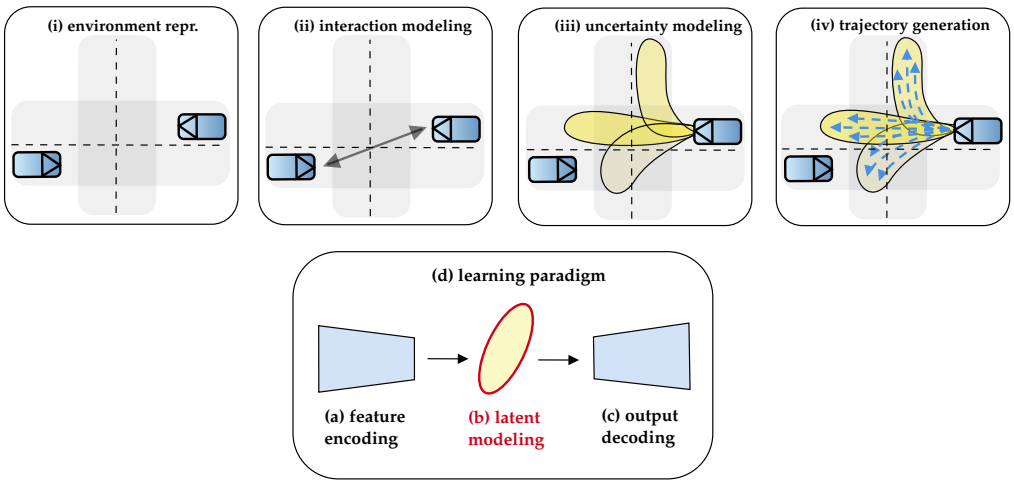


Figure 5.1: Paper V [A5] contains fundamental contributions and does not address specific challenges within trajectory prediction (top). Model-wise however, it follows the general architecture from Sec. 1.4.1 and focuses on the latent modeling component (b) (bottom).

5.1 Unscented Autoencoder

This section concerns Paper V [A5], in which the overall focus of the story is shifted beyond the use case of trajectory prediction and toward generative latent variable models, namely VAEs. In this publication, fundamental properties of the VAE architecture are analyzed, which is commonly used as a basis for prediction models. Paper V [A5] addresses well-known issues of VAE models by incorporating concepts from the field of filtering and control. The fundamental contributions made in Paper V [A5] ultimately enable a more successful utilization of a whole class of algorithms for trajectory prediction. Thus, Paper V [A5] does not directly target a specific sub-problem within the trajectory prediction problem as shown in Fig. 1.2. However, since the VAE follows the general architecture in Fig. 1.3, the contributions of Paper V [A5] can be placed in the latent modeling (b) component. See Fig. 5.1 for an overall placement.

5.1.1 Research Context

The VAE is one of the most widely used generative model architectures. VAEs compress high-dimensional input information into a lower-dimensional latent space under assumptions of a prior distribution. The latent space represents the underlying structure within the high-dimensional inputs in a probabilistic manner. It is learned by training a conditional latent posterior distribution to match an easy-to-sample standard normal prior over an entire dataset of inputs (i.e. the average posterior over all data points should resemble the standard normal).

The training is formulated through the objective of maximizing the data likelihood via a reparameterized version of the well-known Evidence Lower Bound (ELBO) [58]. In addition to pushing the learned posterior to match the standard normal prior, the VAE decoder is trained to generate realistic outputs by approximating an expectation of decoded latent space samples. Thus, sampling the posterior distribution of a single input enables the reconstruction of the inputs at hand from the latent space (in training), while sampling a standard normal prior provides new examples that resemble the original input data distribution (in inference). The compression capability and the simplicity of training (compared to GANs for example) have made VAEs a method of choice in image [99], language [66, 12], and dynamics [56] generative modeling applications. However, VAEs still suffer from important drawbacks. They tend to exhibit a trade-off between reconstruction quality and prior adherence, i.e. prior sample quality, which can be attributed to various factors. These are the simplistic standard normal prior [8], variance in the model components [25], weighting between the loss function terms [98, 48], as well as the mismatch between the aggregated posterior and the prior [35]. Furthermore, in certain contexts the posterior of a single data point collapses into an uninformative noise distribution, rendering the latent representation of the input useless, a phenomenon known as posterior collapse [104] or polarized regime [91]. Another issue is high training variance due to random sampling of the posterior, performed in order to approximate expectations. While in most cases it merely makes training sensitive to hyperparameters, random sampling can be detrimental in applications such as AD where reproducibility of outputs is paramount.

The approaches proposed in Paper V [A5] have the potential to simultaneously deal with the aforementioned issues of high training variance, reconstruction-sampling trade-off, and posterior collapse. Thus, the relevant literature includes an abundance of approaches that study fundamental properties of the VAE. In the following, several approaches are highlighted. The approach in [8] attributes the reconstruction-sampling trade-off to overly simplistic prior distributions. Similarly, [25] studies the Gaussian distribution assumptions in the encoder/decoder. Further, approaches in [48, 98] analyze the VAE loss function component weighting w.r.t. the criteria of output quality and posterior/prior distribution mismatch. On this note, [35] proposes a deterministic architecture and alternatives to sampling from the prior in inference, in turn circumventing the need to train a posterior toward an assumed prior. In the context of posterior collapse, [24, 75, 17] find undesired local maxima within the VAE objective. Finally, [13] combats the increased gradient variance brought on by random sampling with importance weighting of samples, in turn obtaining a tighter log likelihood bound of the ELBO objective. However, it has been shown that such methods induce a diminishing gradient signal for an increased number of samples [89].

5.1.2 Contributions

In Paper V [A5], multiple modifications to core components of the VAE are offered, resulting in the proposed Unscented Autoencoder (UAE) model. The key contribution pertains to the stage of generating the decoded outputs; instead of approximating the latent space expectation with random sampling (the well-known reparameterization trick [59]), the Unscented Transform from the field of filtering and control is used (e.g. in the popular Unscented Kalman Filter (UKF) approach). The key insight leading to this improvement is the fact that the VAE decoder nonlinearly transforms the latent space distribution in the process of generating outputs. Thus, instead of randomly sampling the latent space and transforming individual samples, analytically obtainable sigma points[1] are transformed and used to obtain a distribution at the output. This process therefore imitates the decoding of the entire latent distribution instead of individual samples. This simple tweak in the sampling and decoding in training simultaneously brings improved reconstruction and sample quality, as well as a reduced gradient variance while avoiding a diminishing gradient signal the more sigma points are used. Another contribution of Paper V [A5] frames the posterior-prior mismatch loss differently to the vanilla VAE, replacing the theoretically-motivated Kullback-Leibler Divergence (KL) divergence with the Wasserstein regularization of the latent embedding of a single data point. This simultaneously addresses the issues of the reconstruction-sampling quality trade-off as well as the posterior collapse. Sharper distributions are obtained in training (in turn bringing higher quality outputs), while in inference the alternative ex-post sampling is used, carried over from [35]. An important limitation of this methodology is the lack of a proper ELBO formulation. However, in light of the significantly improved image quality, trading off theoretical guarantees can be worth considering in certain use cases. Furthermore, in Paper V [A5] it shown that the proposed Wasserstein regularization is a generalization of the fully deterministic formulation in [35], as well as that there exist theoretical connections to the Wasserstein model in [98]. The effectiveness of the improvements is demonstrated on CIFAR10 [65], CelebA [73], and Fashion-MNIST [109] standard image datasets.

5.2 Conditional Unscented Autoencoders for Trajectory Prediction

This section concerns Paper VI [A6], which transfers the fundamental improvements in the context of VAEs to the context of the conditional VAE variant,

[1] The number of available sigma points depends on the latent space dimensionality. In practice, a small subset of all available sigmas is used.

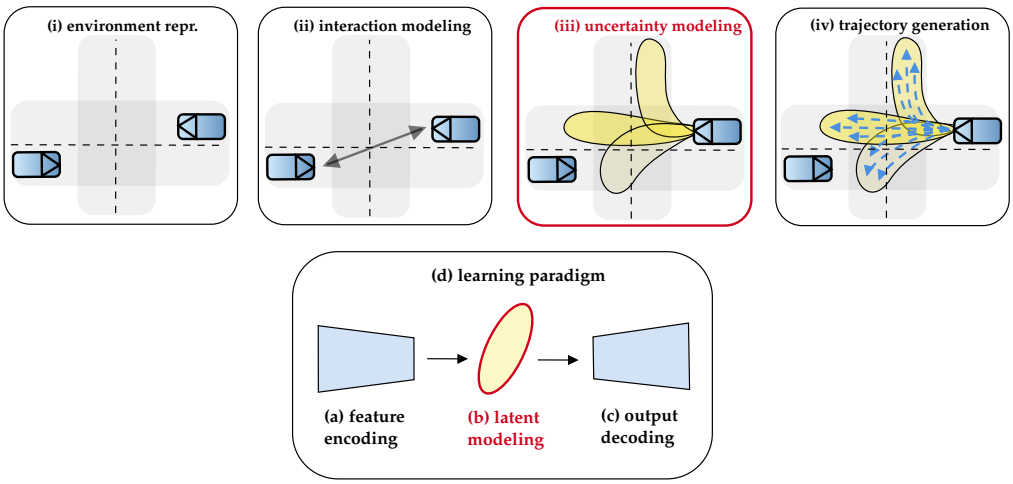


Figure 5.2: Problem-wise, Paper VI [A6] addresses the uncertainty modeling sub-problem (iii) (top). Model-wise, it focuses on the latent modeling (c) (bottom).

the CVAE [95]. The primary demonstrated application is trajectory prediction, however, the proposed approaches are validated on the image modeling use case as well. Furthermore, Paper VI [A6] offers additional domain-motivated architectural improvements in CVAEs for trajectory prediction, which promote multi-modality of the generated outputs. Thus, Paper VI [A6] tackles the uncertainty modeling aspect in stage (iii) of the overall problem by proposing novel latent space modeling approaches (c). See Fig. 5.2 for the placement of Paper VI [A6] within the problem-wise and model-wise divisions.

5.2.1 Research Context

The inherent uncertainty of the future necessitates probabilistic solutions for trajectory prediction. Among a plethora of potential architectures, generative models are promising since they capture the generative process of the data, i.e. the joint distribution between a ground-truth future and the context that induces it, see Sec. 1.4. An especially powerful class of models are CVAEs, which capture this relationship into a rich but relatively low-dimensional latent space, amenable to sampling in order to generate new predictions. However, pitfalls exist, some of which can be traced back to properties of the base VAE. For example, a CVAE employs random sampling in training and inference while generating predictions. This inherent stochasticity can produce unlikely or unsafe trajectories, in addition to being potentially detrimental for downstream application (e.g. a planner receiving significantly different trajectories over consecutive prediction iterations). Another issue of CVAEs relates to their simplistic latent space modeling. More specifically, it is difficult to ensure that

the output distribution exhibits distinctly separate behaviors (e.g. turning left or right), since the trajectories are recovered from a smooth Gaussian latent space. Similar issues of output distribution separability have been discussed in [27] in the context of GANs. Finally, CVAEs do not offer a direct mechanism to ascertain the likelihood of different trajectories. To this end, many approaches augment CVAE architectures with additional learned classifier components.

State-of-the-art CVAE approaches such as [54, 15, 93, 113, 21, 67] employ various strategies to mitigate the aforementioned issues. For example, [93] addresses the modeling of distinct output modes by imposing a discrete latent space over a fixed number of latent vectors, where each vector maps to a specific trajectory. However, such discrete CVAE setups give up the expressiveness of a continuous latent distribution. Similarly, [15] preserves the continuous latent space but focuses on scene consistency in multi-modal joint predictions, employing GNN encoders and decoders whose every node within a graph is grounded to an actor. In contrast, [21, 113] promote multi-modality in an implicit manner by using auxiliary loss terms encouraging diversity. This enables [21] to use only the latent mean in inference, abandoning the learned rich latent space. Finally, [67] focuses on more domain-relevant aspects and designs a hierarchical model where the CVAE is only tasked with predicting long-term goals via heat maps.

5.2.2 Contributions

Paper VI [A6] broadly investigates the role of CVAEs in trajectory prediction by applying the fundamental improvements proposed in Paper V [A5] to the prediction problem, as well as offering novel procedures to address the aforementioned issues of simplistic latent modeling. The first contribution relates to replacing the potentially dangerous random sampling with a more structured selection of samples in the latent space, based on the unscented sampling procedure applied to the VAE in Paper V [A5]. In this context, it is shown that sigma points generate more diverse samples than random samples, due to better coverage of the learned latent space. Additionally, directly applying the Unscented Transform procedure in training brings an improved performance in trajectory prediction over the vanilla CVAE. Thus, the Conditional Unscented Autoencoder (CUAE) model is introduced in Paper VI [A6], a conditional variant of the UAE from Paper V [A5]. The CUAE trajectory predictor incorporates the action-space prediction concepts from Paper I [A1] and uses the attention-based environment and interaction modeling from Paper III [A3].

In terms of addressing the important latent space modeling questions surrounding the trajectory prediction use case, Paper VI [A6] offers two approaches for a more principled latent modeling. First, a Gaussian Mixture Model (GMM) latent space is proposed, which promotes multi-modality in the output space

by mapping each latent mixture component to a separable output component. Thus, each component in the latent space can correspond to a specific behavior in the output space. The training procedure is adapted toward a winner-takes-all setting, where only the closest component to the ground-truth trajectory is used in the reconstruction loss. The second approach is purely an inference procedure, which assumes an already trained CVAE model in place. It is an extension of the ex-post sampling introduced in [35] to a conditional setting, where the training set latent embeddings are collected and used to fit a highly expressive mixture distribution (e.g. a Gaussian mixture with a large number of components). This mixture can be regarded as a non-parametric model of latent space relationships between a ground-truth future and the driving context that induces it. In inference, given an embedding of the encountered driving context, this distribution is then conditioned online in order to obtain another mixture to sample from. The act of conditioning provides a lower-dimensional mixture that can be sampled, and in turn contains relevant ground-truth future trajectories from the training set (via their latent embeddings). Overall, the merit of both latent modeling approaches over the standard Gaussian latent space CVAE and CUAE is experimentally demonstrated, thus obtaining new state-of-the-art performance on the INTERACTION validation set. Furthermore, since the proposed CVAE modifications are applicable beyond trajectory prediction, Paper VI [A6] offers additional image modeling experiments, where the CUAE and the conditional ex-post variant are shown to generate high quality images on the CelebA dataset.

5.3 Summary

This chapter presented an advancement of fundamental concepts in generative latent modeling and consequently successfully applying them to the problem of uncertainty modeling in trajectory prediction. The focus of the contributed approaches was the seminal VAE architecture, used throughout the ML landscape and especially prominent in trajectory prediction. By employing variants of deterministic sampling in the VAE, latent distributions are described in a more structured and representative manner while avoiding unlikely outcomes, ultimately contributing to higher quality image outputs in the basic VAE use cases and more accurate predictions in trajectory prediction application. Furthermore, the proposed mixture latent spaces used as prior distributions in training as well as constructed post-training contribute to richer and more expressive internal representations. Ultimately, a comprehensive probabilistic generative model is proposed, which also incorporates the action-space trajectory generation introduced in Chapter 2 as well as the powerful attention-based environment processing from Chapter 3, thus making it well-capable of addressing the challenges of trajectory prediction.

6 Conclusion

In this chapter, the contributions of the present thesis are summarized. First, the findings of the achieved publications are discussed and their importance is highlighted, both individually as well as in the context of the overall thesis. Then, the limitations of the proposed approaches as well as relevant issues that were out of scope of this thesis are reflected on. Throughout the discussion in this section, potential directions for future work are pointed out.

6.1 Summary of the Thesis

This thesis aimed to answer relevant research questions in order to advance the development of general AD systems that share public roads with humans. AD is a growing and multi-faceted field; in this thesis, the problem of predicting the motion of vehicles was analyzed and solutions in the form of versatile and powerful ML models were offered. To facilitate understanding and solving the problem, trajectory prediction was decomposed into multiple sub-problems that can be addressed individually. These include the issues of

(i) environment representation, where contextual information is represented in a suitable form,

(ii) interaction modeling, where relationships between distinct relevant entities are modeled,

(iii) uncertainty modeling, where multiple potential futures are broadly defined,

(iv) trajectory generation, where final outputs in the form of realistic trajectories are provided.

Over several publications, ML-based solutions for solving such sub-problems were presented. Additionally in this thesis, the proposed ML models were integrated into a single overarching generic structure comprising multiple components. These include

(a) feature encoding, where input information from the environment is processed,

(b) latent model, where internal representations of the model are constructed,

(c) output decoding, where the latent information is propagated into a suitable trajectory representation,

(d) learning paradigm, which involves methods to train the overall model in the most effective manner.

Thus, each presented publication solving specific sub-problems also focused on specific components of the overall architecture.

The first two works presented in this thesis, Paper I [A1] and Paper II [A2] deal with the sub-problem of generating feasible trajectories with favorable kinematic properties. In this respect, action-space decoders are useful for modeling motion; by learning to first predict accelerations and steering angles and then to transform them into positions, kinematic feasibility of trajectories is easily ensured. The action-space framework proposed in Paper I [A1] allows to predict latent environment features as auxiliary information conditioned by the predicted actions. Thus, a self-supervised training approach has emerged, which facilitates learning the main task of future trajectory prediction by training on additional tasks. These allow for a more resourceful usage of the training data compared to the standard supervised-learning case. Furthermore, by dividing the prediction horizon in an autoregressive manner over time segments, additional self-supervision is unlocked and more performant segment-based self-supervised approaches are introduced in Paper II [A2].

The aforementioned works use simplistic environment representations and encoders with limited capabilities. In response, a powerful graph-based representation and a corresponding attention-based encoder are developed in Paper III [A3]. The encoder directly models the attention that a predicted vehicle exhibits to fine-grained map elements and other agents on the road. In terms of interaction modeling, the importance of joint prediction, performed for multiple vehicles at once, over marginal prediction is emphasized. Thus, the local attention-based approach is extended for the joint prediction use case with an additional, global-attention stage that combines the outputs of individual encoders in an efficient manner. The final approach enables the modeling of non-symmetric bidirectional relationships between jointly-predicted vehicles while concurrently preserving valuable local contextual information pertaining to each predicted vehicle.

Moving on to modeling of prediction uncertainty, first a simple, fully-deterministic approach for the joint prediction use case was proposed in Paper IV [A4]. It involves decomposing the future motion uncertainty along scene modes and motion modes, where scene modes model a joint realization of the future, promoting consistency for multiple predicted vehicles, while motion modes model minor deviation for each predicted vehicle given a scene mode. This structure was realized by training multiple decoders where each decoder

focuses on a single scene mode. However, deterministic models are inherently limited in modeling future motion uncertainty, which can be naturally described by a probability distribution.

Recognizing the limitations of deterministic approaches, the last two works, Paper V [A5] and Paper VI [A6] present a shift toward probabilistic generative models. They capture the generative process of the data by reasoning about the joint distribution between a ground-truth future trajectory and the context information that precedes it, thus offering the most general approach to the trajectory prediction problem. To this end, CVAEs were identified as a promising architecture due to their expressive and easy-to-use latent space. However, disadvantages such as the inherent randomness and an overly simplistic Gaussian latent distribution exist and can be traced back to the base VAE framework. Fundamental contributions for the VAE are offered in Paper V [A5] by replacing the random sampling with deterministic sampling, ubiquitous in the field of filtering and control, and thus bridging them together with generative modeling. These improvements translated to the trajectory prediction use case in Paper VI [A6] and yield more effective CVAE predictors. Ultimately, a powerful and versatile generative model architecture was provided, which shows success in both classical image modeling as well as in trajectory prediction, and is particularly adept at uncertainty modeling via the rich latent space.

The individual contributions presented in this thesis are guided by the problem- and model-level decomposition: they address specific sub-problems while being grounded to specific components of the overall architecture. Nevertheless, the presented approaches mostly build upon contributions introduced in previous works. The action-space framework from Paper I [A1], for example, is used throughout the later prediction works – Paper II [A2], Paper III [A3], Paper IV [A4], and Paper VI [A6]. Similarly, the StarNet encoder introduced in Paper III [A3] is used in the CUAE model proposed in Paper VI [A6]. However, the self-supervised approach in Paper I [A1] and the follow-up in Paper II [A2] were evaluated with simplistic encoders instead of the encoder proposed in Paper III [A3] with the goal of delineating the improvements brought on solely by applying resourceful training procedures as opposed to addressing problem-relevant aspects (e.g. interaction modeling). These approaches exploring self-supervision and autoregressive trajectory construction are investigated independently from the probabilistic latent space analysis in Paper VI [A6] and vice-versa. There are nevertheless no inherent restrictions in the proposed models precluding the combination of these individual approaches. Thus, a future work could simply combine the strengths of multiple approaches – e.g. by using graph-based encoders from Paper III [A3], rich probabilistic latent spaces from Paper VI [A6], and self-supervised and autoregressive training architectures from Paper I [A1] and Paper II [A2], ultimately yielding a well-rounded approach concurrently addressing multiple sub-problems in trajectory prediction in a principled manner.

6.2 Limitations and Future Work

Throughout the presented works, the problem of trajectory prediction has been observed in isolation from the other components of the AD stack which either provide inputs to prediction, such as perception, or use prediction outputs, such as planning, see Fig. 1.1. Therein lies the main limitation of the presented approaches pertaining to their use in real-world AD systems: they are trained on idealized input data as opposed to real-life noisy perception data and they do not assess the practical applicability of predicted trajectories in an ego planning context. As discussed in Sec. 1, these problems are out of scope of this work. However, they present important research questions to be answered before the deployment of prediction systems in real-world AD settings. In terms of upstream perception integration for example, key questions surround the handling of noisy or incomplete track data as well as the propagation of detection uncertainty[1] onto future motion prediction. For this purpose, a closer integration of perception and prediction systems is needed. The student work of [S2] proposes novel approaches to integrate the task of tracking with prediction and thus make predicted trajectories more robust to errors in tracks. In contrast, the integration of prediction and planning is bidirectional. Important research questions relate to the most effective usage of predicted trajectories within a planner system, especially considering the hierarchical structure[2] as well as the uncertainty present in predicted trajectories, e.g. a multi-modal prediction where each trajectory has an associated probability. However, predictions can be conditioned on planner goals as well. Knowing the intended route or the trajectory of the AV informs the potential future motion of other traffic participants since they are influenced by the AV's cues, particularly if the AV behaves in an assertive manner. Considering the importance of integrating prediction with planning as well as perception, future research will be tasked with answering crucial questions pertaining to more holistic AD systems.

Considering solely the prediction component of the AD stack, the limitations of the approaches proposed in this thesis pertain to the lack of intricate goal heuristics which can condition predicted trajectories. As discussed in Sec. 4.1, many approaches in literature (especially deterministic models) employ heuristics in the form of goals (e.g. target areas, positions on the road, or lane centerlines) to

[1] Object detection deals with the awareness of the existence of certain static or dynamic objects in the vicinity of an AV. Determining the class of an object (e.g. pedestrian, vehicle) and its state information (e.g. position) is subject to uncertainty that can contain valuable information in itself.

[2] The task of AV planning is usually decomposed along multiple levels. For example, a strategic planner determines the high-level route considering the navigation goal, a motion planner computes a path to perform a certain maneuver (e.g. a lane change), whereas a trajectory planner uses the path to create a time-dependent and collision-aware trajectory that is ultimately executed by a control module.

reduce the future uncertainty. Usually, incorporating such heuristics requires detailed knowledge of map geometry and topology but brings the benefit of "grounding" predicted trajectories to specific behaviors. This can be useful in situations in which intentions of agents surrounding the AV are known (e.g. a blinker light turned on, indicating that a vehicle intends to move to a specific lane) and a mechanism to feed this information into the prediction system exists. The approaches proposed in this thesis have not yet been evaluated in such scenarios; they are more general and thus have reduced requirements for intricate map information as a prerequisite for describing concrete goals. Nevertheless, various mechanisms to include goal information can be incorporated. They range from simple solutions, such as feeding additional goal inputs to the prediction decoder, to more involved ones, such as designing additional loss functions that penalize the noncompliance of predicted trajectories with given goals. Additionally, extensions of existing approaches can be designed. For example, the self-supervised models in Paper I [A1] and Paper II [A2] perform a prediction of the environment features prior to the trajectory prediction and condition the predicted trajectories on predicted environment features. Instead of using predicted features, one can use future environment features that already contain assumed goals[3]. Similarly, the CUAE model in Paper VI [A6] trains a joint posterior distribution that models the interplay between a ground-truth future trajectory and the environment context that precedes it. This distribution is not used in inference, rather a prior distribution conditioned only on the environment context is sampled to decode future trajectories. Hence, the posterior could be used in inference by replacing the ground-truth trajectory with any trajectory passing through an assumed goal.

Finally, the large majority of learned state-of-the-art prediction approaches, including the approaches presented in this thesis, lack the ability to draw from knowledge beyond the available training data. The performance of many DL models in AD depends solely on the quantity and quality of training data and architectural design decisions – they lack the ability to learn in a similar fashion as humans do, i.e. aided by experience and skills from other domains. In prediction, such abilities would be especially helpful to learn traffic rules, which are hard to engineer into learned systems, as well as handle rare or out-of-distribution scenarios. In this context, a class of models that holds promise to augment and transcend existing prediction approaches includes Foundation Model (FM)s and Large Language Model (LLM)s. Extraordinary progress has recently been made in these areas and first applications in AD exist; the reader can refer to the following surveys [70, 111, 117] for an overview. In prediction, such models could boost performance and interpretability by incorporating novel input and output representations such as text. For example, a prediction

[3]This can be implemented via the future feature encoder component that encodes the ground-truth observations, which is concurrently trained with the feature predictor in Paper I [A1] and Paper II [A2].

encoder could use a textual description of a scene in addition to information such as map geometry and past tracks of relevant agents, leading to a richer understanding of the environmental context within the internal latent model. Similarly, textual outputs can help explain the reasoning behind a model's decision to predict a specific motion for an agent. In the future, it is likely that trajectory prediction models will be imbued with "common sense".

Acronyms

ACC Automated Cruise Control.

AD Autonomous Driving.

ADAS Advanced Driver Assistance Systems.

AV Autonomous Vehicle.

BEV Bird's Eye View.

CNN Convolutional Neural Network.

CUAE Conditional Unscented Autoencoder.

CVAE Conditional Variational Autoencoder.

DL Deep Learning.

ELBO Evidence Lower Bound.

FID Fréchet Inception Distance.

FM Foundation Model.

GAN Generative Adversarial Network.

GAT Graph Attention Network.

GMM Gaussian Mixture Model.

GNN Graph Neural Networks.

IL Imitation Learning.

KL Kullback-Leibler Divergence.

LLM Large Language Model.

minADE Minimum Average Displacement Error.

minFDE Minimum Final Displacement Error.

ML Machine Learning.

NLL Negative Log Likelihood.

NMS Non-Maximum Suppression.

ODE Ordinary Differential Equation.

RGB Red Green Blue.

RL Reinforcement Learning.

TTC Time To Collision.

UAE Unscented Autoencoder.

UKF Unscented Kalman Filter.

VAE Variational Autoencoder.

VRU Vulnerable Road Users.

Bibliography

The following lists the references used in the present thesis.

[1] Pulkit Agrawal, Ashvin V Nair, Pieter Abbeel, Jitendra Malik, and Sergey Levine. Learning To Poke By Poking: Experiential Learning Of Intuitive Physics. *Advances in neural information processing systems*, 29, 2016.

[2] Ben Agro, Quinlan Sykora, Sergio Casas, and Raquel Urtasun. Implicit Occupancy Flow Fields For Perception And Prediction In Self-Driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1379–1388, 2023.

[3] Mohammadhossein Bahari, Saeed Saadatnejad, Ahmad Rahimi, Mohammad Shaverdikondori, Amir Hossein Shahidzadeh, Seyed-Mohsen Moosavi-Dezfooli, and Alexandre Alahi. Vehicle Trajectory Prediction Works, But Not Everywhere. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17123–17133, 2022.

[4] Yutong Ban, Xiao Li, Guy Rosman, Igor Gilitschenski, Ozanan Meireles, Sertac Karaman, and Daniela Rus. A Deep Concept Graph Network For Interaction-aware Trajectory Prediction. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 8992–8998. IEEE, 2022.

[5] Mayank Bansal, Alex Krizhevsky, and Abhijit Ogale. ChauffeurNet: Learning To Drive By Imitating The Best And Synthesizing The Worst. *arXiv preprint arXiv:1812.03079*, 2018.

[6] Jonathan T Barron. A General And Adaptive Robust Loss Function. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4331–4339, 2019.

[7] Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational Inductive Biases, Deep Learning, And Graph Networks. *arXiv preprint arXiv:1806.01261*, 2018.

[8] Matthias Bauer and Andriy Mnih. Resampled Priors For Variational Autoencoders. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 66–75. PMLR, 2019.

[9] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation Learning: A Review And New Perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[10] Julian Bock, Robert Krajewski, Tobias Moers, Steffen Runde, Lennart Vater, and Lutz Eckstein. The InD Dataset: A Drone Dataset Of Naturalistic Road User Trajectories At German Intersections. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 1929–1934. IEEE, 2020.

[11] Shannon Bouton, Eric Hannon, Stefan Knupfer, and Surya Ramkumar. The Future (s) Of Mobility: How Cities Can Benefit. Technical report, McKinsey Global Institute, 2017.

[12] Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. Generating Sentences From A Continuous Space. *arXiv preprint arXiv:1511.06349*, 2015.

[13] Yuri Burda, Roger B. Grosse, and Ruslan Salakhutdinov. Importance Weighted Autoencoders. In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[14] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. NuScenes: A Multimodal Dataset For Autonomous Driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020.

[15] Sergio Casas, Cole Gulino, Simon Suo, Katie Luo, Renjie Liao, and Raquel Urtasun. Implicit Latent Variable Model For Scene-consistent Motion Forecasting. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIII 16*. Springer, 2020.

[16] Sergio Casas, Abbas Sadat, and Raquel Urtasun. MP3: A Unified Model To Map, Perceive, Predict And Plan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14403–14412, 2021.

[17] Xi Chen, Diederik P. Kingma, Tim Salimans, Yan Duan, Prafulla Dhariwal, John Schulman, Ilya Sutskever, and Pieter Abbeel. Variational Lossy Autoencoder. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[18] Yuxiao Chen, Peter Karkus, Boris Ivanovic, Xinshuo Weng, and Marco Pavone. Tree-structured Policy Planning With Learned Behavior Models. *arXiv preprint arXiv:2301.11902*, 2023.

[19] Lucas Pinheiro Cinelli, Matheus Araújo Marins, Eduardo Antonio Barros Da Silva, and Sérgio Lima Netto. *Variational Methods For Machine Learning With Applications To Deep Networks*. Springer, 2021.

[20] Travis J Crayton and Benjamin Mason Meier. Autonomous Vehicles: Developing A Public Health Research Agenda To Frame The Future Of Transportation Policy. *Journal of Transport & Health*, 6:245–252, 2017.

[21] Alexander Cui, Sergio Casas, Abbas Sadat, Renjie Liao, and Raquel Urtasun. LookOut: Diverse Multi-future Prediction And Planning For Self-driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16107–16116, 2021.

[22] Alexander Cui, Sergio Casas, Kelvin Wong, Simon Suo, and Raquel Urtasun. GoReLa: Go Relative For Viewpoint-invariant Motion Forecasting. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7801–7807. IEEE, 2023.

[23] Henggang Cui, Thi Nguyen, Fang-Chieh Chou, Tsung-Han Lin, Jeff Schneider, David Bradley, and Nemanja Djuric. Deep Kinematic Models For Kinematically Feasible Vehicle Trajectory Predictions. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10563–10569. IEEE, 2020.

[24] Bin Dai, Ziyu Wang, and David Wipf. The Usual Suspects? Reassessing Blame For VAE Posterior Collapse. In *International Conference on Machine Learning*, pages 2313–2322. PMLR, 2020.

[25] Bin Dai and David Wipf. Diagnosing And Enhancing VAE Models. In *International Conference on Learning Representations*, 2019.

[26] Daniel Dauner, Marcel Hallgarten, Andreas Geiger, and Kashyap Chitta. Parting With Misconceptions About Learning-based Vehicle Motion Planning. *arXiv preprint arXiv:2306.07962*, 2023.

[27] Patrick Dendorfer, Sven Elflein, and Laura Leal-Taixé. MG-GAN: A Multi-generator Model Preventing Out-of-distribution Samples In Pedestrian Trajectory Prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13158–13167, 2021.

[28] Nachiket Deo, Eric Wolff, and Oscar Beijbom. Multimodal Trajectory Prediction Conditioned On Lane-graph Traversals. In *Conference on Robot Learning*, pages 203–212. PMLR, 2022.

[29] Luigi Di Lillo, Tilia Gode, Xilin Zhou, Margherita Atzei, Ruoshu Chen, and Trent Victor. Comparative Safety Performance Of Autonomous And Human Drivers: A Real-World Case Study Of The Waymo One Service. *arXiv preprint arXiv:2309.01206*, 2023.

[30] Nemanja Djuric, Vladan Radosavljevic, Henggang Cui, Thi Nguyen, Fang-Chieh Chou, Tsung-Han Lin, Nitin Singh, and Jeff Schneider. Uncertainty-aware Short-term Motion Prediction Of Traffic Actors For Autonomous Driving. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2095–2104, 2020.

[31] Angelos Filos, Panagiotis Tigkas, Rowan McAllister, Nicholas Rhinehart, Sergey Levine, and Yarin Gal. Can autonomous vehicles identify, recover from, and adapt to distribution shifts? In *International Conference on Machine Learning*, pages 3145–3153. PMLR, 2020.

[32] Yarin Gal and Zoubin Ghahramani. Dropout As A Bayesian Approximation: Representing Model Uncertainty In Deep Learning. In *international conference on machine learning*, pages 1050–1059. PMLR, 2016.

[33] Jiyang Gao, Chen Sun, Hang Zhao, Yi Shen, Dragomir Anguelov, Congcong Li, and Cordelia Schmid. VectorNet: Encoding HD Maps And Agent Dynamics From Vectorized Representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[34] Maximilian Geisslinger, Phillip Karle, Johannes Betz, and Markus Lienkamp. Watch-and-learn-net: Self-supervised Online Learning For Probabilistic Vehicle Trajectory Prediction. In *2021 IEEE international conference on systems, man, and cybernetics (SMC)*, pages 869–875. IEEE, 2021.

[35] Partha Ghosh, Mehdi SM Sajjadi, Antonio Vergari, Michael Black, and Bernhard Schölkopf. From Variational To Deterministic Autoencoders. *arXiv preprint arXiv:1903.12436*, 2019.

[36] Thomas Gilles, Stefano Sabatini, Dzmitry Tsishkou, Bogdan Stanciulescu, and Fabien Moutarde. HOME: Heatmap Output For Future Motion Estimation. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 500–507. IEEE, 2021.

[37] Thomas Gilles, Stefano Sabatini, Dzmitry Tsishkou, Bogdan Stanciulescu, and Fabien Moutarde. GOHOME: Graph-oriented Heatmap Output For Future Motion Estimation. In *2022 international conference on robotics and automation (ICRA)*, pages 9107–9114. IEEE, 2022.

[38] Thomas Gilles, Stefano Sabatini, Dzmitry Tsishkou, Bogdan Stanciulescu, and Fabien Moutarde. THOMAS: Trajectory Heatmap Output With Learned Multi-Agent Sampling. In *International Conference on Learning Representations*, 2022.

[39] Harshayu Girase, Jerrick Hoang, Sai Yalamanchi, and Micol Marchetti-Bowick. Physically Feasible Vehicle Trajectory Prediction. *arXiv preprint arXiv:2104.14679*, 2021.

[40] Sorin Grigorescu, Bogdan Trasnea, Tiberiu Cocias, and Gigel Macesanu. A Survey Of Deep Learning Techniques For Autonomous Driving. *Journal of Field Robotics*, 37(3):362–386, 2020.

[41] Junru Gu, Chen Sun, and Hang Zhao. DenseTNT: End-to-end Trajectory Prediction From Dense Goal Sets. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15303–15312, 2021.

[42] Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream To Control: Learning Behaviors By Latent Imagination. *arXiv:1912.01603*, 2019.

[43] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning Latent Dynamics For Planning From Pixels. In *International conference on machine learning*, pages 2555–2565. PMLR, 2019.

[44] Steffen Hagedorn, Marcel Hallgarten, Martin Stoll, and Alexandru Condurache. Rethinking Integration Of Prediction And Planning In Deep Learning-based Automated Driving Systems: A Review. *arXiv preprint arXiv:2308.05731*, 2023.

[45] Marcel Hallgarten, Ismail Kisa, Martin Stoll, and Andreas Zell. Stay On Track: A Frenet Wrapper To Overcome Off-road Trajectories In Vehicle Motion Prediction. *arXiv preprint arXiv:2306.00605*, 2023.

[46] Kersten Heineke, Nicholas Laverty, Felix Ziegler, and Timo Möller. The Future Of Mobility. 2023.

[47] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs Trained By A Two Time-scale Update Rule Converge To A Local Nash Equilibrium. *Advances in neural information processing systems*, 30, 2017.

[48] Irina Higgins, Loïc Matthey, Arka Pal, Christopher P. Burgess, Xavier Glorot, Matthew M. Botvinick, Shakir Mohamed, and Alexander Lerchner. Beta-VAE: Learning Basic Visual Concepts With A Constrained Variational Framework. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[49] John Houston, Guido Zuidhof, Luca Bergamini, Yawei Ye, Long Chen, Ashesh Jain, Sammy Omari, Vladimir Iglovikov, and Peter Ondruska. One Thousand And One Hours: Self-driving Motion Prediction Dataset. In *Conference on Robot Learning*, pages 409–418. PMLR, 2021.

[50] Anthony Hu, Gianluca Corrado, Nicolas Griffiths, Zak Murez, Corina Gurau, Hudson Yeo, Alex Kendall, Roberto Cipolla, and Jamie Shotton. Model-based Imitation Learning For Urban Driving. *arXiv preprint arXiv:2210.07729*, 2022.

[51] Yihan Hu, Jiazhi Yang, Li Chen, Keyu Li, Chonghao Sima, Xizhou Zhu, Siqi Chai, Senyao Du, Tianwei Lin, Wenhai Wang, Lewei Lu, Xiaosong Jia, Qiang Liu, Jifeng Dai, Yu Qiao, and Hongyang Li. Planning-Oriented Autonomous Driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17853–17862, June 2023.

[52] Xin Huang, Xiaoyu Tian, Junru Gu, Qiao Sun, and Hang Zhao. VectorFlow: Combining Images And Vectors For Traffic Occupancy And Flow Prediction. *arXiv preprint arXiv:2208.04530*, 2022.

[53] Zhiyu Huang, Peter Karkus, Boris Ivanovic, Yuxiao Chen, Marco Pavone, and Chen Lv. DTPP: Differentiable Joint Conditional Prediction And Cost Evaluation For Tree Policy Planning In Autonomous Driving. *arXiv preprint arXiv:2310.05885*, 2023.

[54] Boris Ivanovic, Karen Leung, Edward Schmerling, and Marco Pavone. Multimodal Deep Generative Models For Trajectory Prediction: A Conditional Variational Autoencoder Approach. *IEEE Robotics and Automation Letters*, 2020.

[55] Ruochen Jiao, Yixuan Wang, Xiangguo Liu, Chao Huang, and Qi Zhu. Kinematics-aware Trajectory Generation And Prediction With Latent Stochastic Differential Modeling. *arXiv preprint arXiv:2309.09317*, 2023.

[56] Maximilian Karl, Maximilian Soelch, Justin Bayer, and Patrick Van der Smagt. Deep Variational Bayes Filters: Unsupervised Learning Of State Space Models From Raw Data. *arXiv preprint arXiv:1605.06432*, 2016.

[57] Siddhesh Khandelwal, William Qi, Jagjeet Singh, Andrew Hartnett, and Deva Ramanan. What-if Motion Prediction For Autonomous Driving. *arXiv preprint arXiv:2008.10587*, 2020.

[58] Diederik P Kingma and Max Welling. Auto-encoding Variational Bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[59] Durk P Kingma, Tim Salimans, and Max Welling. Variational Dropout And The Local Reparameterization Trick. *Advances in neural information processing systems*, 28, 2015.

[60] B Ravi Kiran, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A Al Sallab, Senthil Yogamani, and Patrick Pérez. Deep Reinforcement Learning For Autonomous Driving: A Survey. *IEEE Transactions on Intelligent Transportation Systems*, 23(6):4909–4926, 2021.

[61] Jing Yu Koh, Honglak Lee, Yinfei Yang, Jason Baldridge, and Peter Anderson. Path: A World Model For Indoor Navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

[62] Robert Krajewski, Tobias Moers, Julian Bock, Lennart Vater, and Lutz Eckstein. The RounD Dataset: A Drone Dataset Of Road User Trajectories At Roundabouts In Germany. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–6. IEEE, 2020.

[63] Mark A Kramer. Nonlinear Principal Component Analysis Using Autoassociative Neural Networks. *AIChE journal*, 37(2):233–243, 1991.

[64] Mark A Kramer. Autoassociative Neural Networks. *Computers & chemical engineering*, 16(4):313–328, 1992.

[65] Alex Krizhevsky, Geoffrey Hinton, et al. Learning Multiple Layers Of Features From Tiny Images. 2009.

[66] Matt J. Kusner, Brooks Paige, and José Miguel Hernández-Lobato. Grammar Variational Autoencoder. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1945–1954. PMLR, 06–11 Aug 2017.

[67] Mihee Lee, Samuel S Sohn, Seonghyeon Moon, Sejong Yoon, Mubbasir Kapadia, and Vladimir Pavlovic. Muse-VAE: Multi-scale VAE For Environment-aware Long Term Trajectory Prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

[68] John J Leonard, David A Mindell, and Erik L Stayton. Autonomous Vehicles, Mobility, And Employment Policy: The Roads Ahead. *Massachusetts Inst. Technol., Cambridge, MA, Rep. RB02-2020*, 2020.

[69] Albert H Li, Philipp Wu, and Monroe Kennedy. Replay Overshooting: Learning Stochastic Latent Dynamics With The Extended Kalman Filter. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 852–858. IEEE, 2021.

[70] Xin Li, Yeqi Bai, Pinlong Cai, Licheng Wen, Daocheng Fu, Bo Zhang, Xuemeng Yang, Xinyu Cai, Tao Ma, Jianfei Guo, et al. Towards Knowledge-driven Autonomous Driving. *arXiv preprint arXiv:2312.04316*, 2023.

[71] Ming Liang, Bin Yang, Rui Hu, Yun Chen, Renjie Liao, Song Feng, and Raquel Urtasun. Learning Lane Graph Representations For Motion Forecasting. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 541–556. Springer, 2020.

[72] Mengmeng Liu, Hao Cheng, Lin Chen, Hellward Broszio, Jiangtao Li, Runjiang Zhao, Monika Sester, and Michael Ying Yang. LAformer: Trajectory Prediction For Autonomous Driving With Lane-Aware Scene Constraints. *arXiv preprint arXiv:2302.13933*, 2023.

[73] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep Learning Face Attributes In The Wild. In *Proceedings of the IEEE international conference on computer vision*, pages 3730–3738, 2015.

[74] Qiujing Lu, Weiqiao Han, Jeffrey Ling, Minfa Wang, Haoyu Chen, Balakrishnan Varadarajan, and Paul Covington. KEMP: Keyframe-based Hierarchical End-to-end Deep Model For Long-term Trajectory Prediction. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 646–652. IEEE, 2022.

[75] James Lucas, George Tucker, Roger B. Grosse, and Mohammad Norouzi. Understanding Posterior Collapse In Generative Latent Variable Models. In *Deep Generative Models for Highly Structured Data, ICLR 2019 Workshop, New Orleans, Louisiana, United States, May 6, 2019*. OpenReview.net, 2019.

[76] Michael Lutter, Leonard Hasenclever, Arunkumar Byravan, Gabriel Dulac-Arnold, Piotr Trochim, Nicolas Heess, Josh Merel, and Yuval Tassa. Learning Dynamics Models For Model Predictive Agents. *arXiv preprint arXiv:2109.14311*, 2021.

[77] Hengbo Ma, Yaofeng Sun, Jiachen Li, and Masayoshi Tomizuka. Multi-agent Driving Behavior Prediction Across Different Scenarios With Self-supervised Domain Knowledge. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 3122–3129. IEEE, 2021.

[78] Reza Mahjourian, Jinkyu Kim, Yuning Chai, Mingxing Tan, Ben Sapp, and Dragomir Anguelov. Occupancy Flow Fields For Motion Forecasting In Autonomous Driving. *IEEE Robotics and Automation Letters*, 7(2):5639–5646, 2022.

[79] Karttikeya Mangalam, Harshayu Girase, Shreyas Agarwal, Kuan-Hui Lee, Ehsan Adeli, Jitendra Malik, and Adrien Gaidon. It Is Not The Journey But The Destination: Endpoint Conditioned Trajectory Prediction. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 759–776. Springer, 2020.

[80] Jiageng Mao, Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. 3D Object Detection For Autonomous Driving: A Review And New Outlooks. *arXiv preprint arXiv:2206.09474*, 2022.

[81] Jiageng Mao, Junjie Ye, Yuxi Qian, Marco Pavone, and Yue Wang. A Language Agent For Autonomous Driving. *arXiv preprint arXiv:2311.10813*, 2023.

[82] Jean Mercat, Thomas Gilles, Nicole El Zoghby, Guillaume Sandou, Dominique Beauvois, and Guillermo Pita Gil. Multi-head Attention For Multi-modal Joint Vehicle Motion Forecasting. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9638–9644. IEEE, 2020.

[83] Leon Mlodzian, Zhigang Sun, Hendrik Berkemeyer, Sebastian Monka, Zixu Wang, Stefan Dietze, Lavdim Halilaj, and Juergen Luettin. NuScenes Knowledge Graph-A Comprehensive Semantic Representation Of Traffic Scenes For Trajectory Prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 42–52, 2023.

[84] Khan Muhammad, Amin Ullah, Jaime Lloret, Javier Del Ser, and Victor Hugo C de Albuquerque. Deep Learning For Safe Autonomous Driving: Current Challenges And Future Directions. *IEEE Transactions on Intelligent Transportation Systems*, 22(7):4316–4336, 2020.

[85] Nigamaa Nayakanti, Rami Al-Rfou, Aurick Zhou, Kratarth Goel, Khaled S Refaat, and Benjamin Sapp. Wayformer: Motion Forecasting Via Simple & Efficient Attention Networks. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2980–2987. IEEE, 2023.

[86] Jiquan Ngiam, Benjamin Caine, Vijay Vasudevan, Zhengdong Zhang, Hao-Tien Lewis Chiang, Jeffrey Ling, Rebecca Roelofs, Alex Bewley, Chenxi Liu, Ashish Venugopal, et al. Scene Transformer: A Unified Architecture For Predicting Multiple Agent Trajectories. *arXiv preprint arXiv:2106.08417*, 2021.

[87] Jiacheng Pan, Hongyi Sun, Kecheng Xu, Yifei Jiang, Xiangquan Xiao, Jiangtao Hu, and Jinghao Miao. Lane-attention: Predicting Vehicles' Moving Trajectories By Learning Their Attention Over Lanes. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7949–7956. IEEE, 2020.

[88] Tung Phan-Minh, Elena Corina Grigore, Freddy A Boulton, Oscar Beijbom, and Eric M Wolff. CoverNet: Multimodal Behavior Prediction Using Trajectory Sets. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14074–14083, 2020.

[89] Tom Rainforth, Adam Kosiorek, Tuan Anh Le, Chris Maddison, Maximilian Igl, Frank Wood, and Yee Whye Teh. Tighter Variational Bounds Are Not Necessarily Better. In *International Conference on Machine Learning*, pages 4277–4285. PMLR, 2018.

[90] Nicholas Rhinehart, Rowan McAllister, Kris Kitani, and Sergey Levine. PRECOG: Prediction Conditioned On Goals In Visual Multi-agent Settings. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2821–2830, 2019.

[91] Michal Rolinek, Dominik Zietlow, and Georg Martius. Variational Autoencoders Pursue Pca Directions (by Accident). In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12406–12415, 2019.

[92] Andrey Rudenko, Luigi Palmieri, Michael Herman, Kris M Kitani, Dariu M Gavrila, and Kai O Arras. Human Motion Trajectory Prediction: A Survey. *The International Journal of Robotics Research*, 39(8):895–935, 2020.

[93] Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. Trajectron++: Dynamically-feasible Trajectory Forecasting With Heterogeneous Data. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVIII 16*. Springer, 2020.

[94] Ari Seff, Brian Cera, Dian Chen, Mason Ng, Aurick Zhou, Nigamaa Nayakanti, Khaled S Refaat, Rami Al-Rfou, and Benjamin Sapp. MotionLM: Multi-agent Motion Forecasting As Language Modeling. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8579–8590, 2023.

[95] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning Structured Output Representation Using Deep Conditional Generative Models. *Advances in neural information processing systems*, 28, 2015.

[96] Haoran Song, Di Luan, Wenchao Ding, Michael Y Wang, and Qifeng Chen. Learning To Predict Vehicle Trajectories With Model-based Planning. In *Conference on Robot Learning*, pages 1035–1045. PMLR, 2022.

[97] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability In Perception For Autonomous Driving: Waymo Open Dataset. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2446–2454, 2020.

[98] Ilya O. Tolstikhin, Olivier Bousquet, Sylvain Gelly, and Bernhard Schölkopf. Wasserstein Auto-Encoders. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[99] Arash Vahdat and Jan Kautz. NVAE: A Deep Hierarchical Variational Autoencoder. In *Neural Information Processing Systems (NeurIPS)*, 2020.

[100] Balakrishnan Varadarajan, Ahmed Hefny, Avikalp Srivastava, Khaled S Refaat, Nigamaa Nayakanti, Andre Cornman, Kan Chen, Bertrand Douillard, Chi Pang Lam, Dragomir Anguelov, et al. MultiPath++: Efficient Information Fusion And Trajectory Aggregation For Behavior Prediction. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 7814–7821. IEEE, 2022.

[101] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. *Advances in neural information processing systems*, 30, 2017.

[102] Arun Venkatraman, Martial Hebert, and J Bagnell. Improving Multi-step Prediction Of Learned Time Series Models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29, 2015.

[103] Benjamin Völz. *Learning To Predict Pedestrians For Urban Automated Driving*. PhD thesis, ETH Zurich, 2020.

[104] Yixin Wang, David Blei, and John P Cunningham. Posterior Collapse And Latent Variable Non-identifiability. *Advances in Neural Information Processing Systems*, 34:5443–5455, 2021.

[105] Theodor Westny, Joel Oskarsson, Björn Olofsson, and Erik Frisk. Evaluation Of Differentially Constrained Motion Models For Graph-based Trajectory Prediction. *arXiv preprint arXiv:2304.05116*, 2023.

[106] Theodor Westny, Joel Oskarsson, Björn Olofsson, and Erik Frisk. MTP-GO: Graph-Based Probabilistic Multi-Agent Trajectory Prediction With Neural ODEs. *IEEE Transactions on Intelligent Vehicles*, 2023.

[107] Carol A Wrenn. Can Autonomous Technology Reduce The Driver Shortage In The Commercial Trucking Industry. *Doctoral Dissertation*, 2017.

[108] Philipp Wu, Alejandro Escontrela, Danijar Hafner, Ken Goldberg, and Pieter Abbeel. Daydreamer: World Models For Physical Robot Learning. *arXiv preprint arXiv:2206.14176*, 2022.

[109] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: A Novel Image Dataset For Benchmarking Machine Learning Algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[110] Yi Xiao, Felipe Codevilla, Christopher Pal, and Antonio Lopez. Action-based Representation Learning For Autonomous Driving. In *Conference on Robot Learning*, pages 232–246. PMLR, 2021.

[111] Zhenjie Yang, Xiaosong Jia, Hongyang Li, and Junchi Yan. LLM4Drive: A Survey Of Large Language Models For Autonomous Driving. *arXiv e-prints*, pages arXiv–2311, 2023.

[112] Maosheng Ye, Jiamiao Xu, Xunnong Xu, Tongyi Cao, and Qifeng Chen. DCMS: Motion Forecasting With Dual Consistency And Multi-pseudo-target Supervision. *arXiv preprint arXiv:2204.05859*, 2022.

[113] Ye Yuan, Xinshuo Weng, Yanglan Ou, and Kris M Kitani. Agentformer: Agent-aware Transformers For Socio-temporal Multi-agent Forecasting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

[114] Wenyuan Zeng, Ming Liang, Renjie Liao, and Raquel Urtasun. LaneRCNN: Distributed Representations For Graph-centric Motion Forecasting. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 532–539. IEEE, 2021.

[115] Wei Zhan, Liting Sun, Di Wang, Haojie Shi, Aubrey Clausse, Maximilian Naumann, Julius Kummerle, Hendrik Konigshof, Christoph Stiller, Arnaud de La Fortelle, et al. INTERACTION Dataset: An International, Adversarial And Cooperative Motion Dataset In Interactive Driving Scenarios With Semantic Maps. *arXiv preprint arXiv:1910.03088*, 2019.

[116] Hang Zhao, Jiyang Gao, Tian Lan, Chen Sun, Ben Sapp, Balakrishnan Varadarajan, Yue Shen, Yi Shen, Yuning Chai, Cordelia Schmid, et al. TNT: Target-driven Trajectory Prediction. In *Conference on Robot Learning*, pages 895–904. PMLR, 2021.

[117] Xingcheng Zhou, Mingyu Liu, Bare Luka Zagar, Ekim Yurtsever, and Alois C Knoll. Vision Language Models In Autonomous Driving And Intelligent Transportation Systems. *arXiv preprint arXiv:2310.14414*, 2023.

Author's Publications

The following lists the publications achieved by the author during the author's doctoral studies.

[A1] Faris Janjoš, Maxim Dolgov, and J Marius Zöllner. Self-Supervised Action-Space Prediction for Automated Driving. In *2021 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2021. In *2021 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2021. 10.1109/IV48863.2021.9575822.

[A2] Faris Janjoš, Max Keller, Maxim Dolgov, and J Marius Zöllner. Bridging the Gap Between Multi-step and One-Shot Trajectory Prediction via Self-Supervision. In *2023 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2023. 10.1109/IV55152.2023.10186621.

[A3] Faris Janjoš, Maxim Dolgov, and J Marius Zöllner. StarNet: Joint Action-Space Prediction with Star Graphs and Implicit Global-Frame Self-Attention. In *2022 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2022. 10.1109/IV51971.2022.9827091.

[A4] Faris Janjoš, Maxim Dolgov, Muhamed Kurić, Yinzhe Shen, and J Marius Zöllner. SAN: Scene Anchor Networks for Joint Action-Space Prediction. In *2022 IEEE Intelligent Vehicles Symposium (IV) workshops*. IEEE, 2022. 10.1109/IV51971.2022.9827239.

[A5] Faris Janjoš, Lars Rosenbaum, Maxim Dolgov, and J Marius Zöllner. Unscented Autoencoder. In *International Conference on Machine Learning (ICML)*. PMLR, 2023. PMLR 202:14758-14779, 2023.

[A6] Faris Janjoš, Marcel Hallgarten, Anthony Knittel, Maxim Dolgov, Andreas Zell, and J Marius Zöllner. Conditional Unscented Autoencoders for Trajectory Prediction. Submitted to *2024 European Conference on Computer Vision (ECCV) workshops*.

Student Theses

The following lists the student theses supervised by the author during the author's doctoral studies.

[S1] Yinzhe Shen. Graph-Based Map Representations in Trajectory Prediction for Automated Driving, 2022. Research Thesis, University of Stuttgart, 2022.

[S2] Yinzhe Shen. End-to-End Perception and Prediction in Autonomous Driving. Master's Thesis, University of Stuttgart, 2023.

[S3] Max Keller. Self-Supervised Long-Term Trajectory Prediction. Master's Thesis, University of Stuttgart, 2022.

Patent Applications

The following lists the patent applications related to the author's doctoral studies.

[P1] Faris Janjoš, Maxim Dolgov. Bewegungsvorhersage für Verkehrsteilnehmer. DE 102021206014.5 14.06.2021. US 18/570376 01.06.2022.

[P2] Maxim Dolgov, Faris Janjoš. Computer-implementiertes System und Verfahren zur Prädiktion von zukünftigen Entwicklungen einer Verkehrsszene. CN 202211507342.3 29.11.2022. DE 102021213481.5 30.11.2021 US 17/988552 16.11.2022

[P3] Faris Janjoš, Maxim Dolgov. Verfahren zum Ermitteln von Agenten-Trajektorien in einem Multi-Agenten-Szenario. CN 202211488892.5 25.11.2022. DE 102021213344.4 26.11.2021. US 18/058416 23.11.2022.

[P4] Maxim Dolgov, Faris Janjoš. Verfahren, System und Programmprodukt zum Trainieren eines Computerimplementierten Systems zur Prädiktion von zukünftigen Entwicklungen einer Verkehrsszene. CN 202211507338.7 29.11.2022. DE 102021213482.3 30.11.2021. US 17/989079 17.11.2022.

[P5] Maxim Dolgov, Faris Janjoš. Computer-implementiertes Verfahren und System zur Prädiktion von zukünftigen Entwicklungen einer Verkehrsszene. CN 202310121290.4 15.02.2023. DE 102022201770.6 21.02.2022. US 18/171080 17.02.2023.

[P6] Maxim Dolgov, Faris Janjoš, Yinzhe Shen. Computer-implementiertes Verfahren und System zur Verhaltensplanung eines Teilnehmers einer Verkehrsszene. DE 102023205056.0 31.05.2023.

[P7] Faris Janjoš, Maxim Dolgov, Lars Rosenbaum. Device and method for training a variational autoencoder. CN 202311222429.0 21.09.2023. EP 22196963.7 21.09.2022. US 18/465627 12.09.2023.

[P8] Max Keller, Faris Janjoš, Maxim Dolgov. Vorhersage der Fortentwicklung einer Szenerie mit Aggregation latenter Repräsentationen. CN 202311728659.4 14.12.2023. DE 102022213710.8 15.12.2022. US 18/527630 04.12.2023.

[P9] Max Keller, Faris Janjoš, Maxim Dolgov. Bewegungsvorhersage für verkehrsrelevante Objekte. DE 102023203110.8 04.04.2023.

[P10] Yinzhe Shen, Felicia Ruppel, Faris Janjoš, Maxim Dolgov. Computer-implementiertes Verfahren und System zur Analyse des Verhaltens eines Teilnehmers einer Verkehrsszene. DE 102023201197.2 14.02.2023.

[P11] Max Keller, Faris Janjoš, Maxim Dolgov. Computer-implementiertes Verfahren zur Verhaltensplanung für einen Teilnehmer einer Verkehrsszene. DE 102023204069.7 03.05.2023.

[P12] Max Keller, Faris Janjoš, Maxim Dolgov. Computer-implementiertes Verfahren zur Prädiktion des Verhaltens eines Teilnehmers einer Verkehrsszene. DE 102023203666.5 20.04.2023.

[P13] Faris Janjoš, Maxim Dolgov, Marcel Hallgarten. Computerimplementiertes Verfahren zum Trainieren eines Modells sowie Verfahren und System zur Prädiktion des Verhaltens eines Verkehrsteilnehmers. DE 102023210638.8 27.10.2023.

[P14] Marcel Hallgarten, Faris Janjoš, Anthony Knittel. Computerimplementiertes Verfahren zum Trainieren eines Modells sowie Verfahren und System zur Prädiktion des Verhaltens eines Verkehrsteilnehmers. DE 102023210641.8 27.10.2023.

Included Publications

Paper I

- Title: *Self-Supervised Action-Space Prediction for Automated Driving*
- Authors: Faris Janjoš and Maxim Dolgov and J. Marius Zöllner
- Venue: 2021 IEEE Intelligent Vehicles Symposium (IV)

2021 IEEE Intelligent Vehicles Symposium (IV)
July 11-17, 2021. Nagoya, Japan

Self-Supervised Action-Space Prediction for Automated Driving

Faris Janjoš[1], Maxim Dolgov[1], and J. Marius Zöllner[2]

Abstract— **Making informed driving decisions requires reliable prediction of other vehicles' trajectories. In this paper, we present a novel learned multi-modal trajectory prediction architecture for automated driving. It achieves kinematically feasible predictions by casting the learning problem into the space of accelerations and steering angles – by performing action-space prediction, we can leverage valuable model knowledge. Additionally, the dimensionality of the action manifold is lower than that of the state manifold, whose intrinsically correlated states are more difficult to capture in a learned manner. For the purpose of action-space prediction, we present the simple Feed-Forward Action-Space Prediction (FFW-ASP) architecture. Then, we build on this notion and introduce the novel Self-Supervised Action-Space Prediction (SSP-ASP) architecture that outputs future environment context features in addition to trajectories. A key element in the self-supervised architecture is that, based on an observed action history and past context features, future context features are predicted prior to future trajectories. The proposed methods are evaluated on real-world datasets containing urban intersections and roundabouts, and show accurate predictions, outperforming state-of-the-art for kinematically feasible predictions in several prediction metrics.**

I. INTRODUCTION

Trajectory prediction is a crucial component of automated driving stacks. Its task is to digest traffic context information given in the form of raw sensor data or intermediate representations, in order to infer goals and intended motions of other traffic participants. Accurately predicting trajectories of surrounding agents is a prerequisite for downstream planning components, whose job is to navigate the autonomous vehicle reasonably and safely by executing high-level behavior plans and performing lower-level trajectory planning and vehicle control [1].

Predicting trajectories of human-driven vehicles shares complexities of understanding and predicting human motion in general [2]. Historically, most of the interest in human driving behavior and vehicle trajectory prediction has been shown in the context of driver assistance systems that employed classical robotics methods such as Kalman Filters (KF) [3], which perform well for short-term predictions but fail to capture intent-motivated long-term behavior. More information collected aboard vehicles, availability of large datasets, and the computing power of Graphics Processing Units (GPU) have driven the use of Deep Neural Networks (DNN) to achieve longer prediction horizons, as well as addressing full self-driving [4].

[1] Robert Bosch GmbH, Corporate Research, Advanced Autonomous Systems, 71272 Renningen, Germany. {faris.janjos, maxim.dolgov}@de.bosch.com
[2] Research Center for Information Technology (FZI), 76131 Karlsruhe, Germany. zoellner@fzi.de

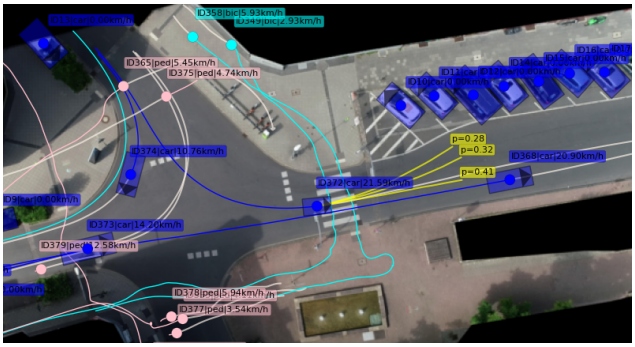


Fig. 1. Multi-modal trajectory prediction (yellow) trained on the inD dataset [6], using the SSP-ASP approach

Despite the large body of research, addressing vehicle trajectory prediction still remains a difficult problem to be solved especially for reasons such as complexity of understanding subtle social cues in multi-agent interactions and necessity of predicting multi-modal trajectories (see Fig. 1). Furthermore, many works neglect kinematic vehicle constraints in the framing of the learning problem [5]. Instead, the networks are tasked with capturing kinematic motion models in addition to the social and topological context by feeding inputs containing full vehicle states (position, heading, velocity, acceleration, etc.) and predicting future positions or waypoints as outputs.

In this work, we avoid these issues by constraining the learning problem to the action space and design architectures that benefit from such formulation. Our contributions are:

- We present a novel action-space learning paradigm, where the learning of motions is constrained purely to the action space of controlled accelerations and steering angles, both from the input and output sides. Position values or state values are then inferred via explicit kinematic motion models.
- A self-supervised action-based prediction architecture with several novel concepts:
  - Prediction of context features that are used for trajectory prediction
  - Reconstruction of action inputs using context features, helping the network learn a manifold sufficient for prediction of both features and trajectories
  - Long-term prediction (longer than 3s) by chaining two predicted segments, which can be regarded as a conceptual middle ground between autoregressive/single-step approaches and one-shot/multi-step approaches.

2021 IEEE Intelligent Vehicles Symposium (IV) | 978-1-7281-5394-0/21/$31.00 ©2021 IEEE | DOI: 10.1109/IV48863.2021.9575822

II. RELATED WORK

Learned models such as DNNs trained on large datasets of potentially high-dimensional data (real or simulated) have shown capability at tackling vehicle trajectory prediction. They perform combined reasoning about static environment information (e.g. geometric road structure) and dynamic multi-agent interaction to extract sufficient latent statistics necessary to infer reasonable future trajectories. Among these models, different deep architectures as well as their combinations are:

- *Convolutional Neural Networks (CNN)* [7] capture planar or spatial information from multi-channel images to construct locational visual features relevant to a scene.
- *Recurrent Neural Networks (RNN)* [7] naturally model the sequential nature of the prediction problem, since consecutive points of a trajectory are propagated in a recurrent manner.
- *Graph Neural Networks (GNN)* [8] encode geometric structure and interactions among different agents into an attention-based graph. By having lower-dimensional inputs than CNNs, large reductions in the number of model weights can be made. A special class are Graph Convolutional Networks (GCN), which employ so-called graph convolutions to extract richer topological representations than vanilla GNNs [9].

In relation to this distinction, a categorization by different input modalities to *raster-*, *polyline-*, or *sensor-based* approaches can be made. *Raster-based* approaches [10], [5], [11], [12], [13], [14] take in Bird's Eye View (BEV) images of an agent's environment with elements such as road lanes and bounding boxes of other perceived agents. By doing so, the CNN network can extract local spatial information from a single representational domain. Even though notions of distance are represented adequately, a large computational effort in training is needed to extract relevant features. In contrast, *polyline-based* approaches [15], [16], [9], [17] use GNNs that accept polylines or vectors representing surrounding roads and nearby agents. Another approach is predicting directly from sensor data such as LiDAR point clouds [18], [19], [20], [21], [22]. Such *sensor-based* approaches can perform object detection in addition to prediction, and propagate perception uncertainty throughout the pipeline. However, they require a very large computational effort in training and usually are not robust to sensor set changes.

More fundamentally, prediction approaches can be categorized by different learning paradigms that answer the question whether the prediction yields identical outputs given the same deterministic input data. Many *supervised-learning* approaches have consistent outputs given deterministic data – they either directly regress a trajectory [10], [5], or model uncertainty by predicting trajectories as sequences of means and covariances of positions, thus modeling the trajectories as uncorrelated distributions [12], [23], [24]. Similarly, some approaches improve prediction by performing classification over a predefined trajectory set [12], [14] or a target position set [16]. In contrast, *latent variable* approaches output random values w.r.t. the same deterministic data, by sampling a latent distribution while inferring trajectory candidates [20], [21], [25], [26], [19]. For example [21] defines prediction in this setting as trajectory forecasting, a class of imitation learning with non-interaction. Such approaches usually incorporate probabilistic latent variable priors that capture uncertainty as a set of non-interpretable random variables, or leverage Generative Adversarial Networks (GAN) [27]. However, their major drawback is the reliance on sequential sampling, which accumulates errors in each step. A notable exception is [19], offering a high-dimensional continuous latent model employing GNN encoder and decoder networks.

Furthermore, approaches can be distinguished into two classes based on how a trajectory is generated: (i) *one-shot* prediction approaches [12], [10], [19] that output an entire trajectory and (ii) *single-step*, autoregressive approaches that build a trajectory sequentially [20], [21], [26]. According to how they model multi-agent interaction, most existing works perform prediction for each agent individually. Examples of these *single-agent* approaches are [10], [5], [12], [11]. Exceptions exist that consider the set of agents *jointly*, such as [26], [19], [21]. For example, [26] offers a discrete latent variable model with recurrent encoder and decoder networks, enabling joint prediction of all actors' trajectories in a scene.

Because the agents' true intentions are not known, trajectory predictions are inherently multi-modal. When it comes to modeling multi-modality, many *deterministic-mode* approaches directly predict multiple candidate trajectories along with their probabilities [10], [5]. Similarly, *Gaussian-mode* approaches represent a trajectory as a weighted mixture (each element is a candidate trajectory) of multi-variate Gaussians time-wise and predict means and variances [12], [28]. Alternatively, previously mentioned *sampling-based* approaches draw many samples from their decoder networks to obtain a notion of how likely a certain mode is. A notable class are *heuristics-based* approaches, which either train an additional network to predict an anchor trajectory from a set of heuristically chosen trajectories [12], [14], or use the road structure to infer viable target positions or regions [17], [16].

Many of the presented approaches can be abstracted to a duo of generic encoder/decoder networks (potentially with an additional prior network) with deterministic or random outputs, where backbone encoders compress the scene into features from which decoders infer trajectories. This assumes a complete reliance on learning to model physically feasible trajectory generation for the agent(s) of interest, which can be accurately described by a kinematic motion model instead. Such approaches opt to learn this model by reasoning about correlations among individual state variables, which in turn produces strong requirements on diversity in training data. A notable exception is [5], where a network learns action variables that are propagated through a bicycle model to infer trajectories. However, it achieves this by mapping states to actions, which assumes learning an inverse motion model instead. Similar approach that learns actions for prediction is [29], which employs a large set of manually designed situational features as inputs.

In our approach, we argue that combining learned and physical models, in addition to yielding kinematically feasible trajectories, improves robustness by uncoupling modelable aspects from the learning task. To this end, we move the learning into the action space, by transforming action inputs to action outputs, and then use kinematic models to reconstruct trajectories. More fundamentally, action inputs in data are direct results of driver commands, as apposed to vehicle positions that are results of applied actions, and thus have the potential to model driving behavior more closely.

Learning action outputs is prevalent in end-to-end approaches such as [4], [30], [31]. An advantage of using action outputs for the prediction problem is that they allow to reason more closely about causal effects of actions on internal network representations. Practically, this can be achieved by principles of self-supervision, which is used in the context of representation learning for applications such as automatic label generation [32], [33]. However, it finds applications in imitation learning as well [34], [31]. Here, self-supervision usually entails learning forward and inverse transformations concurrently [34]. For example, if a certain action applied to a forward model results in an output described by generic features, it should be possible to reconstruct this applied action by feeding two consecutive features to an inverse model, whose job is to output actions. Inspired by these principles, we design a self-supervised trajectory prediction architecture, presented in Sec. IV-C.

III. PROBLEM DESCRIPTION

In general, trajectory prediction of human-driven vehicles can be framed in an imitation learning setting of modeling a distribution of trajectories $\mathbf{Y}$ given data $\mathcal{D}$

$$\mathbf{Y} \sim P(\mathbf{Y}|\mathcal{D}) \ . \tag{1}$$

Data $\mathcal{D}$ can include prior trajectory information of the vehicle for which prediction is done (prediction-ego), sensor data acquired from the environment, or any kind of stored knowledge such as map information. It is an established approach to introduce a categorical hidden variable – mode, and predict trajectories deterministically or as a probability distribution, conditioned on the mode. Then, in view of the *deterministic-mode* approaches mentioned in Sec. II, the problem in (1) can be simplified by predicting m modes of the distribution, i.e. deterministic trajectory Y_m and probability p_m pairs .

Trajectories Y can be defined in several ways – a vehicle's position trajectory is a T-step sequence of planar coordinates $X_{1:T}$, where each X_t is represented by $[x, y]_t$. Similarly, we choose to define a full-state trajectory by the orientation and velocity in addition to positions; it is a sequence $S_{1:T}$, where S_t is $[x, y, \theta, v]_t$, and $X_{1:T} \subset S_{1:T}$. Additionally, we define an action trajectory $A_{1:T}$ as a sequence of controlled accelerations a_t and steering angles δ_t, where A_t is $[a, \delta]_t$. These action values fully capture the behavior of the driver along the time horizon $1:T$ in a given driving situation.

More precisely, we are interested in inferring the future position trajectory $X_{1:T}$ for the prediction ego vehicle, given data $\mathcal{D}$ that includes either past $X_{-T:0}$, $S_{-T:0}$, or $A_{-T:0}$, as well as generic environment observations $O_{-T:0}$. Each O_t can contain observations in the form of LiDAR or BEV rasterized images of the vehicle's surroundings, including road structure and other agents (vehicles, cyclists, pedestrians). In our implementation, we opt for rasterized BEV representations, as displayed in Fig. 5.

A. Action-space prediction

In action prediction, the task is to predict future driving actions $A_{1:T}$. Therefore, learned action-space prediction can be defined as a generic learned mapping of past actions (without past positions and states) and generic observations to future actions

$$\{A_{-T:0}, O_{-T:0}\} \mapsto A_{1:T} \ . \tag{2}$$

For simplicity, (2) only shows uni-modal action prediction, however a multi-modal extension to m action modes $A_{1:T,m}, p_m$ is straightforward.

Future position values $X_{1:T}$ are included in $S_{1:T}$, which is obtained by iteratively propagating actions $A_{1:T}$ through a kinematic motion model f

$$S_{t+1} = f(S_t, A_t) \ . \tag{3}$$

Kinematic models of reasonable complexity sufficiently approximate true vehicle motion. They are virtually always valid and the cases where they are wrong (e.g. skidding) are rare and easy to detect. We opt for the bicycle model [35], and perform the full state update in the following manner

$$\begin{aligned} x_{t+1} &= x_t + v_t \cos(\theta_t + \beta_t)\Delta T \ , \\ y_{t+1} &= y_t + v_t \sin(\theta_t + \beta_t)\Delta T \ , \\ \theta_{t+1} &= \theta_t + \frac{v_t}{l_r} \sin(\beta_t)\Delta T \ , \\ v_{t+1} &= v_t + a_t \Delta T \ , \\ \beta_t &= \tan^{-1}\left(\frac{l_r}{l_r + l_r} \tan\delta_t\right) \ . \end{aligned} \tag{4}$$

Here, ΔT is the sampling time, β is the angle between the center-of-mass velocity and the longitudinal axis of the car, while l_f and l_r are distances from the center of mass to the front and rear wheels. Similarly, we define an inverse bicycle model as a state-to-action mapping. Due to overdetermined (4), we choose to obtain the actions via v and θ

$$\begin{aligned} a_t &= \frac{v_{t+1} - v_t}{\Delta T} \ , \\ \delta_t &= \operatorname{sgn}\left(\frac{\theta_{t+1} - \theta_t}{\bar{v}_t}\right) \arctan\left(\frac{l_f + l_r}{\sqrt{(\frac{\bar{v}_t}{\theta_{t+1} - \theta_t})^2 - l_r^2}}\right) , \end{aligned} \tag{5}$$

where $\bar{v}_t = \frac{v_t + v_{t+1}}{2}$. Identical inverse model is used in [29].

B. Learned mappings

In terms of modeling kinematic characteristics and while ignoring generic observations, the previously presented *action-to-action* mapping in (2) brings several benefits to the learning problem, contrary to learning *state-to-position* [10], *state-to-state* [11], or *state-to-action* [5] mappings:

- In a *state-to-position* or *state-to-state* mapping, the network has to implicitly capture a kinematic motion model (e.g. (4)) while reasoning about the correlation among individual states. To capture a valid model, a large number of diverse motions need to be present in the data (and not be underrepresented), such as driving straight, slight/sharp turns, U-turns, hard braking, etc.
- A *state-to-action* mapping, used in Deep Kinematic Models (DKM) approach [5], incorporates an explicit forward motion model (e.g. (4)) to obtain states from actions. However, the network still has to implicitly capture an inverse model (e.g. (5)) from sufficiently diverse data.
- An *action-to-action* mapping fully captures kinematic characteristics and reduces requirements on the motion representation in the data. Importantly, it is of lower dimensionality than *state-to-state* for instance, and makes capturing correlations between action variables easier.

Action-to-action prediction requires acceleration and steering angle values in tracked data, otherwise they can be inferred from states via inverse models such as (5). Examples of datasets where accelerations are provided are [6], [36].

IV. ACTION-SPACE PREDICTION METHODS

In this section, we present architectures that employ action-space prediction. We introduce the following notation:

- τ_0 is the past T-step time interval $[-T:0]$
- τ_1 is the future T-step time interval $[1:T+1]$
- τ_2 is the future T-step time interval $[T+2:2T+2]$
- $\hat{}$ denotes predicted values
- $*$ denotes ground truth values

A. Feed-forward action-space prediction: FFW-ASP

The FFW-ASP architecture, shown in Fig. 2, realizes the mapping of past actions and observations to future actions (given in (2)) via the standard encoder/decoder architecture. It can be described by the following sequence of mappings

$$o_{\tau_0} \xrightarrow{\phi} z_{\tau_0} \ , \tag{6}$$

$$z_{\tau_0}, a_{\tau_0} \xrightarrow{\gamma} \hat{a}_{\tau_1} \ , \tag{7}$$

$$\hat{x}_{\tau_1} = f(\hat{a}_{\tau_1}, x_0) \ , \tag{8}$$

where ϕ parameterizes an encoder network that maps past observations to features, and γ parameterizes a decoder that maps features and past actions to future actions. Future positions are obtained from (4) via the kinematic model $f(\cdot)$ that requires the present position x_0 in addition to future actions $\hat{a}_{\tau_1}$, in order to reconstruct future positions $\hat{x}_{\tau_1}$.

The loss function penalizes the difference between predicted positions and ground truth positions. In our implementation, we opt for the Huber loss with a cut-off at $h = 1.0$

$$\Delta x_{\tau_1} = \hat{x}_{\tau_1} - x^*_{\tau_1} \ ,$$

$$\mathcal{L} = \|\Delta x_{\tau_1}\| = \begin{cases} \frac{1}{2}\Delta x^2_{\tau_1}, & |\Delta x_{\tau_1}| < h \\ h(|\Delta x_{\tau_1}| - \frac{1}{2}h), & \text{otherwise} \ . \end{cases} \tag{9}$$

This loss function incorporates predicted positions obtained via the fully-differentiable kinematic model, making back-propagation possible.

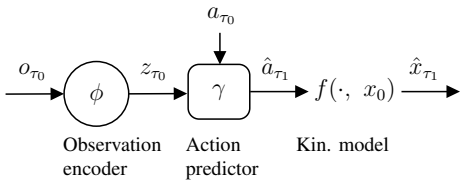


Fig. 2. FFW-ASP architecture: encoder with parameters ϕ accepts a history of observations o_{τ_0} and generates features z_{τ_0}. Then, a decoder with parameters γ combines features z_{τ_0} with action history a_{τ_0} to generate future actions $\hat{a}_{\tau_1}$. Kinematic model $f(\cdot, x_0)$ reconstructs a position trajectory

It is straightforward to extend this architecture to multi-modal predictions. Inspired by [10], the action decoder can be adapted to predict M action trajectories $\hat{a}_{\tau_1,m}$ along with their probabilities p_m, where $m \in \{1, ..., M\}$. In this case, the loss function is extended with a binary indicator function I and a cross-entropy loss term

$$\mathcal{L}_{FFW} = \sum_{m=1}^{M} I_{m=m^\dagger}(\|\Delta x_{\tau_1,m}\|) - \frac{\exp(p_{m^\dagger})}{\sum_m \exp(p_m)} \ , \tag{10}$$

where the indicator selects the mode closest to the ground-truth (denoted by $\dagger$), while the cross-entropy term rewards confidence in the chosen mode. The mode-to-ground-truth distance uses a combination of angle difference between the last points of the two trajectories and L_2 distance between each point [10].

This simple feed-forward architecture capitalizes on the benefits of *action-to-action* mappings from Sec. III-A. In the next section, we describe the methodology of self-supervision in an action-based context, before presenting a self-supervised trajectory prediction architecture.

B. Self-supervision background

The forward and inverse transformations in the context of self-supervision, previously mentioned in Sec. II, have been used in robotic manipulation tasks [34] to jointly learn mappings from visual features generated from images of objects to physical robot actions on the objects and vice-versa. This architecture has been adapted for end-to-end driving in [31], where a pre-trained self-supervised forward/inverse model outperforms pure end-to-end models that map images to driving actions. Furthermore, a categorization of different learned transformations is offered in [31], which we summarize here with one-step predictions.

1) **Behavioral cloning** models map an observation to latent features with a ϕ-encoder, and map features to action output with a γ-decoder

$$o_t \xrightarrow{\phi} z_t \xrightarrow{\gamma} \hat{a}_t \ . \tag{11}$$

The loss is a difference between the label and the output

$$\mathcal{L}_{BC} = \|a^*_t - \hat{a}_t\| \ . \tag{12}$$

2) **Inverse** models encode observations into features at two successive time-steps. Then, they learn a ξ-parameterized mapping from two consecutive features

to an action, to reconstruct the applied action that resulted in a new observation

$$\left.\begin{array}{ll} o_t & \xrightarrow{\phi} z_t \\ o_{t+1} & \xrightarrow{\phi} z_{t+1} \end{array}\right\} \xrightarrow{\xi} \hat{a}_t \ , \tag{13}$$

with a reconstruction loss

$$\mathcal{L}_{INV} = \|a_t^* - \hat{a}_t\| \ . \tag{14}$$

3) **Forward** models, parameterized by ψ, predict the new latent features given past features and action

$$\left.\begin{array}{ll} o_t \xrightarrow{\phi} & z_t \\ & a_t \end{array}\right\} \xrightarrow{\psi} \hat{z}_{t+1} \ , \tag{15}$$

$$o_{t+1} \xrightarrow{\phi} z_{t+1} \ . \tag{16}$$

Optimizing only the feature mismatch between the predicted future features $\hat{z}_{t+1}$ and the encoded features z_{t+1} can lead to networks outputting the trivial solution of zero features. Therefore, a regularization term is added in the form of the inverse model loss (14)

$$\mathcal{L}_{FW} = \|z_{t+1} - \hat{z}_{t+1}\| + \|a_t^* - \hat{a}_t\| \ . \tag{17}$$

The regularized forward model is preferable to pure behavioral cloning, since the networks gain a richer understanding of the interplay between actions and features. The model can reconstruct its inputs, as well as predict its internal representation of the environment (in the form of features), as opposed to predicting observations directly, which can be infeasible in the case of camera images for example.

C. Self-supervised action-space prediction: SSP-ASP

The previously presented forward model (with an inverse model regularizer) predicts only future latent features. In order to use it for trajectory prediction, we add an action predictor component parameterized by γ

$$a_t, z_t, \{z_{t+1}, \hat{z}_{t+1}\} \xrightarrow{\gamma} \hat{a}_{t+1} \ . \tag{18}$$

This mapping predicts the future action $\hat{a}_{t+1}$ by combining the known past action a_t, encoded past latent feature z_t, and one of the future features $\{z_{t+1}, \hat{z}_{t+1}\}$. The difference is that z_{t+1} is used at training time and obtained via the encoding ϕ (16), while $\hat{z}_{t+1}$ is used at inference time and obtained via the prediction ψ (15). With the additional action predictor, the total self-supervised action prediction loss extends the regularized forward model loss (17) according to

$$\mathcal{L} = \|z_{t+1} - \hat{z}_{t+1}\| + \|a_t - \hat{a}_t\| + \|a_{t+1}^* - \hat{a}_{t+1}\| \ . \tag{19}$$

In summary, it incorporates the forward model that predicts future features (feature predictor), the inverse model regularizer that 'predicts' the past actions (action reconstructor), and the action predictor, predicting future actions.

The complete procedure to obtain the training loss (19) is visualized in Fig. 3 for T-step intervals. First, we encode observations from past and future intervals o_{τ_0} and o_{τ_1} separately into features z_{τ_0} and z_{τ_1}. Then, we reconstruct the past action by generating $\hat{a}_{\tau_0}$ while simultaneously predicting future features $\hat{z}_{\tau_1}$. Finally, we predict future actions $\hat{a}_{\tau_1}$ using a_{τ_0}, z_{τ_0}, and z_{τ_1} (in training) or $\hat{z}_{\tau_1}$ (during inference).

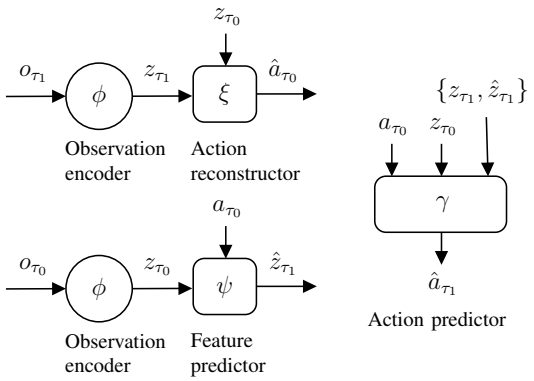


Fig. 3. SSP-ASP *training* architecture: encoder with parameters ϕ accepts observations at two consecutive intervals o_{τ_0} and o_{τ_1} separately, generates features z_{τ_0} and z_{τ_1}, and then the action reconstructor ξ and feature predictor ψ reconstruct past actions $\hat{a}_{\tau_0}$ and predict future features $\hat{z}_{\tau_1}$, respectively. The action predictor uses past actions a_{τ_0} and consecutive features z_{τ_0} and z_{τ_1} (or $\hat{z}_{\tau_1}$) to predict future actions $\hat{a}_{\tau_1}$. Kinematic transformation is omitted for clarity.

In practice, we run the action predictor with both encoded and predicted features in training – we use both outputs in the loss, in order to avoid a distribution mismatch during inference. For inference, the architecture in Fig. 3 reduces to encoding only past observation o_{τ_0} (since o_{τ_1} is not available), predicting features $\hat{z}_{\tau_1}$, and then predicting future actions $\hat{a}_{\tau_1}$ using a_{τ_0}, z_{τ_0}, and $\hat{z}_{\tau_1}$. Finally, since we are interested in multi-modal position prediction over a T-step interval as opposed to a single time-step, (19) becomes

$$\begin{aligned} \mathcal{L}_{SSP} = w_1\|a_{\tau_0} - \hat{a}_{\tau_0}\| + w_2\|z_{\tau_1} - \hat{z}_{\tau_1}\| + \\ w_3 \sum_{m=1}^{M} I_{m=m^\dagger}(\|\Delta x_{\tau_1,m}\|) - w_4 \frac{\exp(p_{m^\dagger})}{\sum_m \exp(p_m)} \ . \end{aligned} \tag{20}$$

Here, the weights $\{w_i\}_{i=1}^4$ penalize action reconstruction, feature mismatch, multi-modal regression, and classification.

An important aspect of the self-supervised action prediction problem is its time-interval formulation. It enables us to introduce a distinction between *single-segment* prediction, and a long-term, *multi-segment* prediction.

1) Single-segment prediction: We have so far considered the same T-step time interval for all the past and future values. In this setup, we predict a single segment of values of interest, exemplified in Fig. 3, with the constraint of same time interval length for the past and future. The duration can be regarded as a design choice. We opt for 3s segment length since we assume that [-3:0]s history is long enough to capture relevant scene context, and [0:3]s future is a meaningful, albeit not long-term, prediction interval.

2) Multi-segment prediction: The self-supervised architecture makes it possible to chain several predicted segments and achieve long-term prediction. This can be done both in training and inference by additionally generating features and actions along the τ_2 interval. The inference architecture is visualized in Fig. 4. Here, the feature predictor ψ is run once more to obtain $\hat{z}_{\tau_2}$, and action predictor γ to obtain $\hat{a}_{\tau_2}$. For the $3s$ segment length it would constitute a prediction on

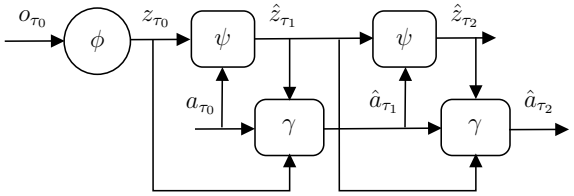


Fig. 4. SSP-ASP two-segment *inference* architecture: encoder with parameters ϕ accepts observations o_{τ_0} and then alternately, future features $\hat{z}_{\tau_1}$ and $\hat{z}_{\tau_2}$, and actions $\hat{a}_{\tau_1}$ and $\hat{a}_{\tau_2}$ are generated. First, $\hat{z}_{\tau_1}$ and $\hat{a}_{\tau_1}$ are generated with the help of encoded features z_{τ_0}, and then, $\hat{z}_{\tau_2}$ and $\hat{a}_{\tau_2}$ with the help of predicted features $\hat{z}_{\tau_1}$. Note how the action reconstructor ξ from Fig. 3 is not needed in inference.

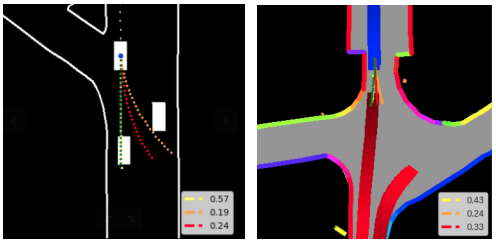


Fig. 5. Example predicted trajectories with ChauffeurNet (all channels superimposed) and MTP rasters on inD [6]; green trajectory indicates ground truth, while red, orange, and yellow trajectories are predicted modes

the [3:6]s interval. The training case is similar to Fig. 3, with o_{τ_1} and o_{τ_2} as inputs. The same procedure is carried out, obtaining encoded features z_{τ_1} and z_{τ_2}, and predicting $\hat{z}_{\tau_2}$. Finally, $\hat{a}_{\tau_2}$ can be obtained either with the encoded z_{τ_2} or the predicted $\hat{z}_{\tau_2}$. In general, the multi-segment chaining can be performed with the same procedure for even more segments further into the future. In our implementation, we focus on two segments.

Multi-segment prediction requires non-trivial chaining of multi-modal trajectories. For example, the predicted action output along τ_1 is multi-modal with m modes, and it needs to be fed again into the action predictor γ that accepts uni-modal inputs, and outputs multi-modal actions along τ_2. Naturally, it is possible to feed m modes individually and obtain m^2 chained trajectories at the output, requiring a large computational effort both in training and inference. A less costly alternative is feeding the highest probability trajectory within the multi-modal actions. Similarly, *label matching* can be done – the action mode closest to the label position trajectory can be chosen (after a kinematic transform of the actions), with the distance metric the same as in the regression component of the losses (10) and (20). In this context, label matching can be regarded as a variant of guided teacher forcing [37] – a method to accelerate training of RNNs by replacing the outputs of earlier stages of the network with ground truth samples.

In the context of prediction, the self-supervised architectures in Fig. 3 and Fig. 4 improve on classic encoder-decoder setups exemplified by Fig. 2. They achieve this by incorporating future observations in training, as opposed to only past observations, while at the same time predicting the future context and reconstructing the past actions. This joint learning helps all the networks form richer representations.

V. RESULTS

In this section, we describe the implementation details of the approach proposed in Sec. IV. We specify the training procedure, the datasets that were used for training, and present the obtained results.

A. Implementation

In Sec. IV, we have assumed generic observations as inputs to generic encoder networks that can reason with different input representations (as categorized in Sec. II). In our implementation, we opt for *raster-based* images and work with two variants visualized in Fig. 5. The first variant is a sparse black and white 12-channel 360x240 representation, similar to the ChauffeurNet representation [13]. Here, individual channels contain the road boundary polylines, prediction-ego bounding box, prediction-ego trajectory history, and multiple snapshots (sampled at regular time-intervals) of bounding boxes of all traffic agents. The second variant is a 3-channel 360x360 RGB-image similar to the Multi-Modal Trajectory Prediction (MTP) representation [10]. Here, the road boundary polylines, drivable areas, and prediction-ego and traffic bounding boxes are overlaid on a single image, where different semantics have specific color values. We implemented this variant with the help of nuScenes devkit [38].

The rasterized inputs are fed into ResNet18 [39] encoder backbone networks returning a 512-sized feature vector, both for the case of feed-forward and self-supervised architectures. For the former, the γ action predictor (Fig. 2) is realized as a 2-layer Gated Rectifier Unit (GRU) with 512 hidden units. For the self-supervised architecture (Fig. 3), we use the following networks: feature predictor ψ and the action predictor γ are also 2-layer GRUs with 512 hidden units, whereas the action reconstructor ξ is a Multilayer Perceptron (MLP) with layer sizes $\{256, 128, 64\}$. The γ and ξ networks output acceleration and steering angle values in the range $[-1, 1]$, which are then scaled to the range observed in the dataset.

In both feed-forward and self-supervised architectures we consider a 3s history and observe two use-cases: predicting a 3s and a 6s trajectory. For the 6s prediction, the trajectory is generated via two different prediction paradigms: one-shot prediction for the feed-forward architecture and chaining two 3s predictions for the self-supervised architecture (with label matching to account for multi-modality).

B. Datasets

We evaluated our approach on two real-world datasets recorded on German roads: inD (urban intersections) [6] and rounD (roundabouts) [36]. In total, they contain highly accurate drone-recorded trajectories of over 25000 agents of different classes, such as cars, trucks, buses, motorcycles, cyclists, and pedestrians. We performed prediction for all agent classes except cyclists and pedestrians. A drawback of the datasets is the lack of signalized intersections, which

were not considered in our approach but can be easily integrated into the input representation. For example, the ChauffeurNet [13] rasterization considers traffic light status as an additional channel.

Depending on the prediction horizon, we extracted all 3+3s (3s past and 3s prediction) or 3+6s (3s past and 6s prediction) trajectory snippets from the datasets, with a 0.6s spacing. We split each dataset into training, validation, and testing subsets according to an 8:1:1 ratio, with each subset containing recordings taken at a different time of the day, thus ensuring no overlap. To generate the road boundaries polylines, we used the provided lanelet maps [40].

C. Training setup

The feed-forward and self-supervised architectures were trained on the accompanying losses (10) and (20). For the self-supervised case, we used equal weighting for the loss components. Similarly to (10), each loss component is represented by the Huber loss function (9).

The self-supervised architecture offers great flexibility in different ways to pre-train the models. We experimented with training on only the action reconstruction and feature mismatch loss components from (20) prior to the full loss, essentially implementing (17). We observed that this *forward-model pre-training* improves performance since it enables the features to be learned to a certain degree before the actual prediction is performed. Additionally, in the 6s prediction horizon case, we observed performance gains by loading and retraining a model used for 3s prediction.

All models were implemented in PyTorch [41] and trained on two NVIDIA GeForce GTX 1080 GPUs using the Adam optimizer [42], a batch size of 32, and 10 epochs. The learning rate was set to 10^{-4} and reduced by a factor of five if the validation loss did not improve over three consecutive epochs. In self-supervised prediction, we performed additional forward-model pre-training over 3 epochs.

D. Performance

We evaluated the performance of our FFW-ASP (Sec. IV-A) and SSP-ASP (Sec. IV-B) approaches on inD [6] and rounD [36] datasets. We benchmarked them against DKM [5], another method that achieves kinematically feasible predictions, albeit by using a *state-to-action* mapping introduced in Sec. III-B compared to our proposed *action-to-action* mapping. This method improves on the authors' prior work [10], whose multi-modality concept and rasterization we leverage. All methods were compared in both ChauffeurNet [13] and MTP [10] rasterization setups. For a fair comparison, we used a ResNet18 encoder in all setups and performed 3s and 6s prediction based on a 3s history (with 3 modes). As evaluation metrics, we calculated the Mean Absolute Error (MAE)[1], Mean Squared Error (MSE)[2], and Final Displacement Error (FDE)[3] to the ground-truth on the testing datasets.

[1]mean $L1$ distance across all time-steps
[2]mean $L2$ distance across all time-steps
[3]$L2$ distance at final time-step

inD		3s			6s		
		MAE	MSE	FDE	MAE	MSE	FDE
Chauffeur Net	DKM	0.48	0.86	2.01	1.27	7.44	5.91
	FFW-ASP	0.27	0.33	1.23	**1.13**	**6.42**	5.45
	SSP-ASP	**0.21**	**0.24**	**0.98**	1.23	6.52	**5.15**
MTP	DKM	0.58	1.19	2.36	1.34	7.89	6.29
	FFW-ASP	0.30	0.39	1.41	1.19	6.78	5.72
	SSP-ASP	**0.22**	**0.25**	**1.04**	**1.15**	**6.08**	**5.02**

rounD		3s			6s		
		MAE	MSE	FDE	MAE	MSE	FDE
Chauffeur Net	DKM	0.29	0.29	1.12	1.35	7.22	5.55
	FFW-ASP	0.22	0.18	0.93	**1.12**	**6.17**	5.08
	SSP-ASP	**0.17**	**0.12**	**0.73**	1.34	6.44	**4.69**
MTP	DKM	0.29	0.28	1.22	1.25	6.13	5.51
	FFW-ASP	0.22	0.18	0.94	**1.08**	**5.84**	4.76
	SSP-ASP	**0.17**	**0.12**	**0.74**	1.25	7.08	**4.61**

TABLE I
RESULTS OBTAINED ON THE IND [6] AND ROUND [36] DATASETS

The results are provided in Tab. I, showing that FFW-ASP already brings improvements over DKM on both datasets. Since the two approaches differ in the choice of the prediction head network (DKM has a fully connected layer, while FFW-ASP has a GRU), we tested FFW-ASP using a DKM prediction head and still achieved improvements (omitted in the table). Furthermore, SSP-ASP offers clear improvements to FFW-ASP on both datasets for 3s prediction, while for 6s prediction the feed-forward architecture achieves similar results overall. This indicates that the method of chaining multi-modal trajectories, specific to the multi-segment variant, could potentially be improved. Nevertheless, SSP-ASP achieves the lowest FDE across all settings.

VI. CONCLUSION

In this paper, we offered an action-based perspective into the problem of vehicle trajectory prediction. Results indicate that uncoupling modelable aspects from the learning problem, such as learning forward and inverse motion models, improves prediction performance while ensuring kinematic feasibility. As a main result, we proposed a novel self-supervised action-based architecture for prediction. By predicting future latent features and reconstructing past actions simultaneously, we capture the interplay between actions and their effects on the networks' internal representations. We extract more information by leveraging future observations in training, which we use to predict future latent features – a feasible alternative to predicting observations. Finally, we achieve long-term prediction via a new paradigm: in contrast to one-shot or single-step prediction approaches, we obtain a trajectory by chaining individual trajectory segments.

In future work, we plan to replace the computationally intensive CNN encoder with a lightweight GNN, as this is currently the bottleneck of the self-supervised approach. We expect more robust feature representation and thus a performance improvement, since the network would not have to extract objects from pixels. Furthermore, we would like to analyze the effects of time segment length on prediction quality. It remains to be seen whether shorter or longer segments can capture more latent information in a scene and whether chaining more than two segments could potentially achieve prediction horizons longer than 6s.

REFERENCES

[1] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, "A Survey of Motion Planning and Control Techniques for Self-Driving Urban Vehicles," *IEEE Transactions on Intelligent Vehicles*, vol. 1, no. 1, 2016.

[2] A. Rudenko, L. Palmieri, M. Herman, K. M. Kitani, D. M. Gavrila, and K. O. Arras, "Human Motion Trajectory Prediction: A Survey," *The International Journal of Robotics Research*, vol. 39, no. 8, 2020.

[3] S. Lefèvre, D. Vasquez, and C. Laugier, "A Survey on Motion Prediction and Risk Assessment for Intelligent Vehicles," *ROBOMECH Journal*, vol. 1, no. 1, 2014.

[4] A. Tampuu, M. Semikin, N. Muhammad, D. Fishman, and T. Matiisen, "A Survey of End-to-End Driving: Architectures and Training Methods," *arXiv preprint arXiv:2003.06404*, 2020.

[5] H. Cui, T. Nguyen, F.-C. Chou, T.-H. Lin, J. Schneider, D. Bradley, and N. Djuric, "Deep Kinematic Models for Physically Realistic Prediction of Vehicle Trajectories," *arXiv preprint arXiv:1908.00219*, 2019.

[6] J. Bock, R. Krajewski, T. Moers, S. Runde, L. Vater, and L. Eckstein, "The inD Dataset: A Drone Dataset of Naturalistic Road User Trajectories at German Intersections," *arXiv preprint arXiv:1911.07602*, 2019.

[7] Y. LeCun, Y. Bengio, and G. Hinton, "Deep Learning," *Nature*, vol. 521, no. 7553, 2015.

[8] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The Graph Neural Network Model," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, 2008.

[9] M. Liang, B. Yang, R. Hu, Y. Chen, R. Liao, S. Feng, and R. Urtasun, "Learning Lane Graph Representations for Motion Forecasting," in *European Conference on Computer Vision*. Springer, 2020.

[10] H. Cui, V. Radosavljevic, F.-C. Chou, T.-H. Lin, T. Nguyen, T.-K. Huang, J. Schneider, and N. Djuric, "Multimodal Trajectory Predictions for Autonomous Driving Using Deep Convolutional Networks," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019.

[11] J. Strohbeck, V. Belagiannis, J. Müller, M. Schreiber, M. Herrmann, D. Wolf, and M. Buchholz, "Multiple Trajectory Prediction with Deep Temporal and Spatial Convolutional Neural Networks," in *2020 International Conference on Intelligent Robots and Systems (IROS)*. IEEE/RSJ, 2020.

[12] Y. Chai, B. Sapp, M. Bansal, and D. Anguelov, "Multipath: Multiple Probabilistic Anchor Trajectory Hypotheses for Behavior Prediction," *arXiv preprint arXiv:1910.05449*, 2019.

[13] M. Bansal, A. Krizhevsky, and A. Ogale, "ChauffeurNet: Learning to Drive by Imitating the Best and Synthesizing the Worst," *arXiv preprint arXiv:1812.03079*, 2018.

[14] T. Phan-Minh, E. C. Grigore, F. A. Boulton, O. Beijbom, and E. M. Wolff, "Covernet: Multimodal Behavior Prediction Using Trajectory Sets," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[15] J. Gao, C. Sun, H. Zhao, Y. Shen, D. Anguelov, C. Li, and C. Schmid, "VectorNet: Encoding HD Maps and Agent Dynamics from Vectorized Representation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[16] H. Zhao, J. Gao, T. Lan, C. Sun, B. Sapp, B. Varadarajan, Y. Shen, Y. Shen, Y. Chai, C. Schmid, *et al.*, "TNT: Target-driveN Trajectory Prediction," *arXiv preprint arXiv:2008.08294*, 2020.

[17] S. Khandelwal, W. Qi, J. Singh, A. Hartnett, and D. Ramanan, "What-If Motion Prediction for Autonomous Driving," *arXiv preprint arXiv:2008.10587*, 2020.

[18] S. Casas, W. Luo, and R. Urtasun, "IntentNet: Learning to Predict Intention from Raw Sensor Data," in *Conference on Robot Learning*, 2018.

[19] S. Casas, C. Gulino, S. Suo, K. Luo, R. Liao, and R. Urtasun, "Implicit Latent Variable Model for Scene-Consistent Motion Forecasting," *arXiv preprint arXiv:2007.12036*, 2020.

[20] N. Rhinehart, K. M. Kitani, and P. Vernaza, "R2P2: A Reparameterized Pushforward Policy for Diverse, Precise Generative Path Forecasting," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[21] N. Rhinehart, R. McAllister, K. Kitani, and S. Levine, "PRECOG: Prediction Conditioned on Goals in Visual Multi-Agent Settings," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019.

[22] M. Liang, B. Yang, W. Zeng, Y. Chen, R. Hu, S. Casas, and R. Urtasun, "PnPNet: End-to-End Perception and Prediction with Tracking in the Loop," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[23] N. Djuric, V. Radosavljevic, H. Cui, T. Nguyen, F.-C. Chou, T.-H. Lin, N. Singh, and J. Schneider, "Uncertainty-Aware Short-Term Motion Prediction of Traffic Actors for Autonomous Driving," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2020.

[24] N. Djuric, H. Cui, Z. Su, S. Wu, H. Wang, F.-C. Chou, L. S. Martin, S. Feng, R. Hu, Y. Xu, *et al.*, "MultiNet: Multiclass Multistage Multimodal Motion Prediction," *arXiv preprint arXiv:2006.02000*, 2020.

[25] N. Rhinehart, R. McAllister, and S. Levine, "Deep Imitative Models for Flexible Inference, Planning, and Control," *arXiv preprint arXiv:1810.06544*, 2018.

[26] C. Tang and R. R. Salakhutdinov, "Multiple Futures Prediction," in *Advances in Neural Information Processing Systems*, 2019.

[27] X. Huang, S. G. McGill, J. A. DeCastro, B. C. Williams, L. Fletcher, J. J. Leonard, and G. Rosman, "Diversity-Aware Vehicle Motion Prediction via Latent Semantic Sampling," *arXiv preprint arXiv:1911.12736*, 2019.

[28] J. Hong, B. Sapp, and J. Philbin, "Rules of the Road: Predicting Driving Behavior with a Convolutional Model of Semantic Interactions," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

[29] J. Schulz, C. Hubmann, N. Morin, J. Löchner, and D. Burschka, "Learning Interaction-Aware Probabilistic Driver Behavior Models from Urban Scenarios," in *2019 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2019.

[30] M. Bojarsi, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, *et al.*, "End-to-end Learning for Self-Driving Cars," *arXiv preprint arXiv:1604.07316*, 2016.

[31] Y. Xiao, F. Codevilla, C. Pal, and A. M. Lopez, "Action-Based Representation Learning for Autonomous Driving," *arXiv preprint arXiv:2008.09417*, 2020.

[32] A. Kolesnikov, X. Zhai, and L. Beyer, "Revisiting Self-Supervised Visual Representation Learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[33] P. Goyal, D. Mahajan, A. Gupta, and I. Misra, "Scaling and Benchmarking Self-Supervised Visual Representation Learning," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019.

[34] P. Agrawal, A. V. Nair, P. Abbeel, J. Malik, and S. Levine, "Learning to Poke by Poking: Experiential Learning of Intuitive Physics," in *Advances in Neural Information Processing Systems*, 2016.

[35] J. Kong, M. Pfeiffer, G. Schildbach, and F. Borrelli, "Kinematic and Dynamic Vehicle Models for Autonomous Driving Control Design," in *2015 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2015.

[36] R. Krajewski, T. Moers, J. Bock, L. Vater, and L. Eckstein, "The rounD Dataset: A Drone Dataset of Road User Trajectories at Roundabouts in Germany," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2020, pp. 1–6.

[37] R. J. Williams and D. Zipser, "A Learning Algorithm for Continually Running Fully Recurrent Neural Networks," *Neural Computation*, vol. 1, no. 2, 1989.

[38] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuScenes: A Multimodal Dataset for Autonomous Driving," *arXiv preprint arXiv:1903.11027*, 2019.

[39] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[40] F. Poggenhans, J.-H. Pauls, J. Janosovits, S. Orf, M. Naumann, F. Kuhnt, and M. Mayr, "Lanelet2: A High-Definition Map Framework for the Future of Automated Driving," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018.

[41] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, *et al.*, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," *arXiv preprint arXiv:1912.01703*, 2019.

[42] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv preprint arXiv:1412.6980*, 2014.

Paper II

- Title: *Bridging the Gap Between Multi-Step and One-Shot Trajectory Prediction via Self-Supervision*
- Authors: Faris Janjoš and Max Keller and Maxim Dolgov and J. Marius Zöllner
- Venue: 2023 IEEE Intelligent Vehicles Symposium (IV)

Bridging the Gap Between Multi-Step and One-Shot Trajectory Prediction via Self-Supervision

Faris Janjoš
Corporate Research
Robert Bosch GmbH
71272 Renningen, Germany
faris.janjos@de.bosch.com

Max Keller
Corporate Research
Robert Bosch GmbH
71272 Renningen, Germany
max.keller@de.bosch.com

Maxim Dolgov
Corporate Research
Robert Bosch GmbH
71272 Renningen, Germany
maxim.dolgov@de.bosch.com

J. Marius Zöllner
Research Center for
Information Technology (FZI)
76131 Karlsruhe, Germany
zoellner@fzi.de

2023 IEEE Intelligent Vehicles Symposium (IV) | 979-8-3503-4691-6/23/$31.00 ©2023 IEEE | DOI: 10.1109/IV55152.2023.10186621

***Abstract*—Accurate vehicle trajectory prediction is an unsolved problem in autonomous driving with various open research questions. State-of-the-art approaches regress trajectories either in a one-shot or step-wise manner. Although one-shot approaches are usually preferred for their simplicity, they relinquish powerful self-supervision schemes that can be constructed by chaining multiple time-steps. We address this issue by proposing a middle-ground where multiple trajectory segments are chained together. Our proposed Multi-Branch Self-Supervised Predictor receives additional training on new predictions starting at intermediate future segments. In addition, the model 'imagines' the latent context and 'predicts the past' while combining multi-modal trajectories in a tree-like manner. We deliberately keep aspects such as interaction and environment modeling simplistic and nevertheless achieve competitive results on the INTERACTION dataset. Furthermore, we investigate the sparsely explored uncertainty estimation of deterministic predictors. We find positive correlations between the prediction error and two proposed metrics, which might pave way for determining prediction confidence.**

I. INTRODUCTION

The common separation of the Autonomous Driving (AD) stack into perception, prediction, and planning components drives a need for accurate forecasts of the future as a planner input. In prediction, different challenges exist such as but not limited to, representing the environment [1], modeling multi-agent interactions [2], capturing the multi-modality of the future motion distribution [3], adhering to kinematic constraints [4], as well as modeling a long prediction horizon [5]. In solving these tasks, various Deep Learning (DL) models usually generate predicted trajectories of the agents sharing a road situation with an autonomous vehicle (AV).

In terms of the approach to construct trajectories, prediction models can be categorized into one-shot approaches [4], [6]–[8], where a full trajectory is directly regressed, and step-wise, autoregressive approaches that generate a trajectory sequentially for each time step based on the previously predicted time steps [5], [9]–[13]. The main drawback of one-shot approaches is that long prediction horizons make it difficult to reason about comprehensive changes in scene dynamics. Since the models do not condition on future observations, the potential for errors and high uncertainty is much larger toward the end of a long prediction horizon. As opposed to one-shot models, step-wise, autoregressive approaches build each prediction based on the previous inference step. Similarly to one-shot approaches, they tend to accumulate errors for longer horizons, especially in interactive settings under distribution shift [14]. Furthermore, the autoregressive context makes it non-trivial to combine multi-modal predictions between successive time steps. In our work, we aim to bridge the two approaches and offer potential answers to the drawbacks at hand.

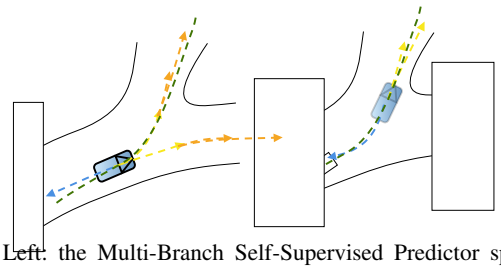


Fig. 1: Left: the Multi-Branch Self-Supervised Predictor splits the trajectory prediction problem into three equal time-segments, where the ground-truth history and future are shown in green. It builds a tree of multi-modal trajectories over two future segments (yellow and orange), with a multi-modal trajectory (two modes) chained at the output of each previous segment mode. Additionally, it 'predicts' a uni-modal trajectory in the past (blue). Right: knowing the ground-truth in training, the model builds another tree shifted by one segment into the future. By constructing multiple trees, reconstructing the past, as well as 'imagining' how the context will evolve in each segment (not depicted), the model receives additional (self-)supervision.

Our starting point in addressing the question of combining the different paradigms is the Self-Supervised Action-Space Predictor (SS-ASP) model in [4]. Its main characteristic is the ability to predict a latent representation of the driving context (e.g. map and motion of non-predicted agents) prior to predicting future trajectories for the vehicle of interest. In achieving this, it is trained with future context data in addition to future trajectories of the predicted agents, which separates it from the existing state-of-the-art models. Furthermore, it can 'go back in time' and check whether its reconstructions of the past are consistent with the observed ground-truth history through an auxiliary, inverse model task. These two aspects can be regarded as a form of self-supervision and they enable the model to interpret the trajectory prediction problem in a segment-wise manner, where each segment consists of multi-

This work was financially supported by the Federal Ministry of Economic Affairs and Energy of Germany, grant number 19A20026H, based on a decision of the German Bundestag.

ple time steps (for example, one second length) and multiple segments are chained together. With each new segment, future trajectories and context are predicted, and past trajectories of the previous segment are reconstructed. This is in contrast to pure one-shot or step-wise approaches, which do not employ such strategies. However, these abilities are sparsely explored in [4] and the proposed chaining of segments actually regresses the results compared to a simple one-shot-like setup.

In this work, we aim to incorporate novel autoregressive schemes from [15]–[17] to enable a multi-segment formulation of a one-shot trajectory prediction model. With this, we aim to answer the *research question (i): can the prediction performance of a one-shot model be improved by an autoregressive formulation that creates room for additional self-supervision?* Furthermore, in light of the multi-segment structure, we aim to investigate how the model uncertainty evolves with each segment, given a deterministic problem formulation. In this sense, we aim to answer the *research question (ii): can the confidence of the deterministic multi-segment model be accurately rated, in order to define the limitations of the model?* This might pave the way toward the prediction of a variable number of segments and thus a variable time-horizon.[1] The main contributions of our work can be summarized along:

- Segment-wise prediction: a novel paradigm where the resulting trajectory is obtained by predicting a number of 1-second time segments and chaining them together.
- Multi-Branch SS-ASP: a multi-branch and multi-segment extension of the approach from [4], trained by generating additional prediction paths (termed branches). The proposed model 'imagines' future context, reconstructs past trajectories, and combines segment-wise multi-modal predictions with various tree-search strategies. The model is deliberately simplistic in terms of interaction and multi-modality modeling but still competitive on the INTERACTION dataset [18].
- Uncertainty of deterministic predictors: we assess the given model's confidence along evolving time-segments with two novel approaches that exhibit a significant positive correlation with the measured prediction error.

II. RELATED WORK

The focus of our work is self-supervised autoregressive trajectory prediction and uncertainty estimation in the context of our proposed model. Therefore, we are interested in **autoregressive models** (both in trajectory prediction and other contexts), **self-supervision**, and **uncertainty estimation** in trajectory prediction; we outline the section accordingly.

A. Autoregressive prediction

Autoregressive trajectory prediction models [5], [9]–[13] incrementally model changes in scene dynamics compared to one-shot models [4], [6]–[8], which have larger requirements on model expressiveness within a single prediction. Among such models, [11] predicts the waypoints of the next step based on an agent's own state and the waypoints of the surrounding agents at the previous step. It handles multi-modality by predicting a fixed amount of latent intents, which condition the step-wise rolled-out trajectories. Similarly, [9] extends the problem by inferring time-varying discrete intents of the surrounding agents, which are incorporated into the step-wise discrete-continuous hybrid model. It uses a learned proposal function to 'traverse' the system and obtain multiple modes. In [19], step-wise environment prediction is done, similar to the latent context prediction in [4]. The environment prediction benefits from the autoregressive formulation; it is simpler to predict observations in a single time step than a long horizon.

Outside of vehicle trajectory prediction, autoregressive models are present in robotics and Imitation Learning (IL) [15]–[17], [20]. In [16], the so-called latent overshooting method is introduced that rolls-out new autoregressive predictions at each intermediate future step, in parallel to the first prediction sequence. This allows to increase the learning signal without additional data. An autoregressive formulation is useful in Model-Based Reinforcement Learning (RL) as well; [17] showed that a multi-step loss, based on the autoregressively predicted steps, increases the reward for deterministic planning modules compared to a single-step objective. Similarly, the Dreamer model in [21] learns behavior directly from autoregressive latent predictions ('imagination') instead of exploration. This significantly reduces the training time compared to an explorative RL agent. In summary, the potential for performance improvement as well as the larger design space compared to one-shot models motivate the usage of an autoregressive formulation in this work.

B. Self-supervision in trajectory prediction

Existing trajectory prediction models incorporate self-supervision either through (i) a separate training stage or (ii) through additional tasks. There are fewer approaches performing (ii) in the literature; [22] proposes a contrastive pre-training in which rasterized representations of intersecting trajectories are rotated or semantics are exchanged in order to learn an internal interaction representation. In contrast, [23] fine-tunes a pre-trained predictor in an online-setting to adapt to behaviors observed in inference. Among (ii), [24] enforces additional temporal and spatial consistency tasks for trajectory refinement that robustify the outputs in terms of pertubations. Additionally, [25] takes a graph-based approach where certain map and agent node features are masked out and presented as a completion task for the model.

Learning environment models through self-supervision and using the internal representations of scene dynamics for planning has received strong attention in RL [19], [26], [27]. It has shown to be a promising direction in IL-based AD as well; in [21] the model predicts an evolution of the scene in the latent space and reconstructs camera images, semantic maps, and actions taken by the AV. Similarly, the SS-ASP model [4] performs prediction in the latent space but

[1] A use-case is stopping the prediction after a certain segment in case the uncertainty is too high. Providing only the 'confident' predictions to a planner might improve its performance since the output is not based on highly uncertain predictions (and thus potentially overconfident).

without full observation reconstruction. Instead, latent context predictions are compared to encoded future observations and an inverse model is learned as well [28], [29], which introduces another transition prior on the environment. Compared to [21], this is more efficient (due to lower dimensionality), however, reconstructing rich observations from the latent space induces stronger requirements on the expressiveness of the latent space.

C. Prediction uncertainty estimation

Despite the large number of deterministic trajectory predictors in literature and the importance of communicating the model uncertainty to a planner, the task has received limited attention in literature. In general, estimating the epistemic uncertainty of a prediction model is a challenging problem in DL. Several approaches in trajectory prediction use the computationally cumbersome deep ensembles [3], [30]. A more general approach is Bayesian inference, where uncertainty is directly estimated in conjunction with the prediction. However, significantly more effort is needed to design and train Bayesian networks compared to standard neural networks. In [31], a theoretical framework is described that "casts dropout training in deep neural networks as approximate Bayesian inference in deep Gaussian processes". In practice, the parameters of a Gaussian distribution are approximated by the mean and variance of multiple inference runs, each with different deactivated (dropped-out) neurons. Along these results, [32] provide a study of dropout-based Bayesian approximation in pedestrian trajectory prediction and find improved accuracy through in inference. In our work, we apply dropout-based techniques to estimate the prediction uncertainty of the developed model, due to the theoretical grounding and ease-of-use.

III. METHOD

In this section, we describe our method. In Sec. III-A we define the addressed problem of trajectory prediction and introduce notation. Sec. III-B describes the SS-ASP model from [4]. Sec. III-C extends the SS-ASP into an autoregressive formulation and offers strategies how to combine segment-wise multi-modal predictions. The proposed Multi-Branch SS-ASP model is given in Sec. III-D. Strategies how to determine the prediction uncertainty of the Multi-Branch SS-ASP are introduced in Sec. III-E.

A. Problem definition and notation

Vehicle trajectory prediction can be framed as non-interactive imitation learning [10], where given the observed information $\mathcal{D}$ and ground-truth future trajectories Y^*, we learn the conditional distribution $P(\hat{Y}|\mathcal{D})$ of future trajectories $\hat{Y}$. In practice, deterministic models represent the distribution by predicting K likely samples (modes) $\{Y_j\}_{j=1}^K$ as well as their associated pseudo-probabilities $\{p_j\}_{j=1}^K$. Furthermore, the prediction can be performed for a single vehicle or jointly for multiple vehicles in a scene. Even though joint prediction is a more sound approach to the problem [6], we limit the analysis to a single-agent setting for simplicity. Nevertheless, the proposed model has no methodological restrictions preventing an extension to joint prediction.

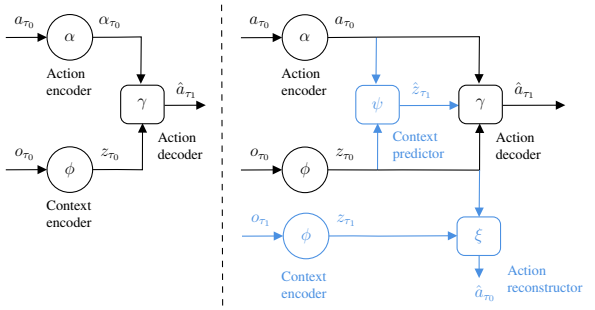


Fig. 2: Left: the (non-self-supervised) FF-ASP [4], conceptually very common in the literature (standard encoder-decoder structure) when action-spaces are excluded. Past actions a_{τ_0} and observations o_{τ_0} are encoded via an action encoder α and a context encoder ϕ into features α_{τ_0} (with a slight abuse of notation) and z_{τ_0}. Future actions $\hat{a}_{\tau_1}$ are predicted by the decoder γ. Right: self-supervised model with additional components in blue. The model additionally predicts future latent context $\hat{z}_{\tau_1}$ with the context predictor ψ and trains it against the encoding z_{τ_1} as pseudo-ground-truth. Furthermore, it reconstructs past actions $\hat{a}_{\tau_0}$ with an inverse model ξ and trains them against the past ground-truth a_{τ_0}. Multi-modal predicted actions and kinematic models converting actions to positions are omitted for clarity.

The proposed model uses a segment-wise prediction formulation where a time segment is a sequence of time steps and a full T time step prediction consists of N equal length segments[2]. Thus, we introduce supporting notation:

- i is the segment index, $i \in [1, ..., N]$
- t is the number of time steps in a segment, $T = N \cdot t$
- τ_i describes a future time segment i, $((i-1) \cdot t : it]$
- τ_0 describes a single past time segment, $(-t : 0]$

B. Self-Supervised Action-Space Predictor (SS-ASP)

The SS-ASP model [4] is the basis for developing the multi-segment model proposed in this work. It is an action-space prediction model, i.e. it predicts actions (accelerations and steering angles) and obtains positions via a kinematic model. At a high-level, it uses encoders for capturing past environment context information (e.g. a CNN encoding birds-eye-view grids or a GNN operating on graphs) into a latent (context) feature vector. Furthermore, it uses an action-based encoder for encoding past actions (e.g. RNN) and a multi-modal action-based decoder for regressing future actions (e.g. RNN). This does not separate it conceptually from a multitude of state-of-the-art approaches (irrespective of the action-space), since a vast majority uses a similar setup of encoding past information (context, trajectories) and predicting future trajectories. The described architecture is depicted in the left part of Fig. 2 (so-called Feed-Forward Action-Space Predictor (FF-ASP) [4]), where the context and action encoders, and action decoder are parameterized by ϕ, α, and γ, respectively.

The SS-ASP model stands out in the sense that, additionally to the aforementioned encoder and decoder components, it predicts a latent future context prior to predicting future actions. It trains this predicted future context against its own encoding of the future context. Furthermore, it reconstructs

[2]In this sense, a one-shot prediction is a single-segment prediction.

past actions via an inverse model taking in future actions. These two self-supervised tasks serve as additional regularization for the model. The SS-ASP model is depicted in the right part of Fig. 2, where the new context predictor component is parameterized by ψ, and the action reconstructor with ξ. For encoding the future context, the same past context encoder ϕ is reused, in this case receiving future information during training. The loss function of the model is

$$\mathcal{L}_{SS-ASP} = \mathcal{L}_{traj} + \mathcal{L}_{class} + \mathcal{L}_{context} + \mathcal{L}_{recon} \ , \qquad (1)$$

with weights omitted for clarity. The trajectory regression loss is $\mathcal{L}_{traj} = \|\hat{Y}_{\tau_1}(\hat{z}_{\tau_1}) - Y^*_{\tau_1}\| + \|\hat{Y}_{\tau_1}(z_{\tau_1}) - Y^*_{\tau_1}\|$; its two terms reflect the fact that the action decoder γ in Fig. 2 is called with both predicted and encoded future context (Fig. 2 only shows the former) in training to promote consistency between components. The loss function (1) considers multi-modal outputs via the winner-takes-all [33] approach. The classification loss function $\mathcal{L}_{class}$ considers mode probabilities via cross-entropy. The context loss $\mathcal{L}_{context} = \|\hat{z}_{\tau_1} - z_{\tau_1}\|$ penalizes the mismatch between encoded and predicted future context, while the reconstruction $\mathcal{L}_{recon} = \|\hat{Y}_{\tau_0}(\hat{z}_{\tau_1}) - Y_{\tau_0}\| + \|\hat{Y}_{\tau_0}(z_{\tau_1}) - Y_{\tau_0}\|$ considers past 'predictions' (two terms promoting consistency similar to $\mathcal{L}_{traj}$). For more details, see [4].

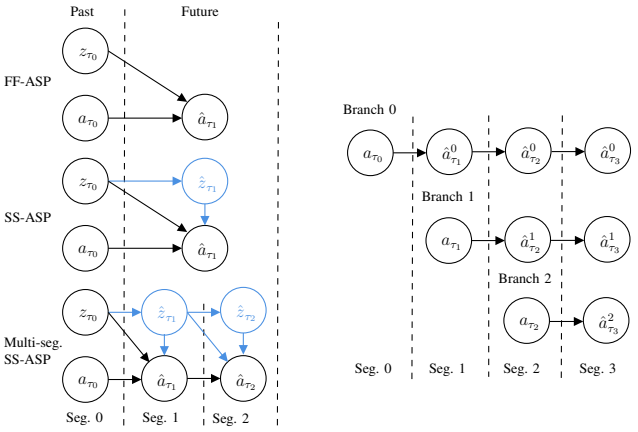
C. Multi-Segment SS-ASP

The SS-ASP model can be extended into an autoregressive formulation with repeated calls of its components over successive time-segments. The components model the interplay between context and actions over a certain time-segment and chaining multiple calls is expected to perform reasonably well in inference. This extension is depicted in Fig. 3a. However, this naive formulation (partly presented in [4]) actually regresses the performance due to the induced distribution drift of chaining predictions on top of predictions [4], see Sec. IV-C.

In addition to the distribution drift, chaining multi-modal predictions along trajectory segments is non-trivial. If the action decoder γ generates k modes per segment, a decision has to be made on which modes to expand in the next segment. If the prediction is continued for a single mode in a segment, diversity is suppressed, while considering all permutations results in k^N trajectories. Therefore, different strategies for combining multi-modal predictions can be employed in order to traverse the tree and select K out of possible k^N modes.

In the following, six combination strategies are investigated, visualized in Fig. 4. In implementing different strategies, we make use of mode probabilities. These probabilities can be either generated by the action decoder in addition to each predicted mode, or by a separate learned classification component. Conceptually, all strategies can be placed between the All-Modes strategy, considering k^N modes, and the Single-Mode strategy that selects the highest probability mode and discards others[3]. Start-k and End-k strategies take k modes at the start or the end, and a single mode otherwise. In Best-m-of-all, the product of the probabilities of previous and subsequent modes

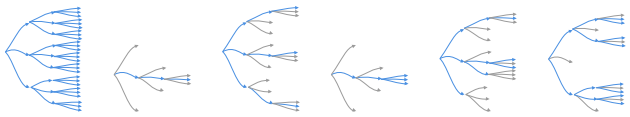
[3]Single-Mode can be viewed as an application of Best-first-search.



(a) High-level model comparison in terms of context and (uni-modal) action prediction. The FF-ASP uses only past features and actions to predict future actions. The SS-ASP predicts future features (blue) prior to future actions (action reconstruction not depicted). The Multi-Segment SS-ASP splits the future into segments and chains successive feature/action predictions. This induces a distribution shift however, since predictions are chained on top of predictions.

(b) High-level visualization of branched overshooting [20] over $N = 3$ future segments for predicted actions. At each intermediate future segment, a new prediction branch is started (denoted with superscript b, $\hat{a}^b_{\tau_i}$), with a shifted history and a shorter future prediction. In addition to predicting future actions in a branched manner, we branch context features as well as reconstructed actions (not visualized).

Fig. 3: Multi-segment (a) and multi-branch (b) SS-ASP depictions.



(a) All-Modes (b) Single-Mode (c) Start-k (d) End-k (e) Best-m-of-All (f) Best-m-of-Pred.

Fig. 4: Six investigated combination strategies for $N = 3$ segments. The selected modes are depicted in blue and non-selected in gray. The number of predicted modes in a segment is $k = 3$. Best-m-of-All uses $m = 3$ and Best-m-of-Prediction uses $m = 2$.

is calculated and the prediction is only continued for m most likely modes[4] of all modes within a segment. The last strategy is Best-m-of-Prediction, where the prediction is continued for $m \leq k$ most likely modes in a single multi-modal prediction, disregarding probabilities of earlier segments.

Each strategy has unique advantages and disadvantages. To quantify them, three properties are identified, see Tab. I. The property (i) is the maximum number of multi-modal prediction calls n for a sample, which serves as a proxy for the required computation time[5]. The property (ii) is the total number of obtained modes K over the entire prediction horizon. Property (iii) is qualitative and describes mode diversity via {*diverse*, *partialy-diverse*, *unclear*} qualifiers. In *diverse* strategies the K resulting modes share no segment trajectory subsets, in *partialy-diverse* strategies the modes share a least one segment trajectory subset, and for the *unclear* strategies the number

[4]Best-m-of-all corresponds to the Beam search heuristic.

[5]n can be larger than the number of segments N in the prediction horizon, e.g. $n = 13$ multi-modal prediction calls are required in the All-Modes visualization of Fig. 4.

TABLE I: Different strategies categorized by the maximum number of multi-modal prediction calls n for a sample, the total number of predicted modes K, and the mode diversity.

Strategy	K	n	Diversity
All-Modes	k^N	$\sum_{i=0}^{N-1} k^i$	*partially-diverse*
Single-Mode	1	N	*diverse*
Start-k	k	$1 + k \cdot (N-1)$	*diverse*
End-k	k	N	*partially-diverse*
Best-m-of-All	m	$1 + \sum_{i=1}^{N-1} m$	*unclear*
Best-m-of-Prediction	m^N	$\sum_{i=0}^{N-1} m^i$	*partially-diverse*

of shared subsets varies per sample. The presented strategies of combining multi-modal prediction are general and can be integrated with different multi-modal decoders, i.e. do not depend on the specific action decoder used in this work.

D. Multi-Branch SS-ASP

The autoregressive formulation of the Multi-Segment SS-ASP opens room for advanced training methods able to reduce the distribution drift of chaining multiple predictions. In the following, branched overshooting, termed in [20], is used as well as a novel combination of context aggregation and prediction, designed to cope with the partial observability of the state through the autoregressive formulation. The resulting model aims to answer *research question (i)* from Sec. I; it is termed as the Multi-Branch SS-ASP.

1) Branched overshooting: It refers to a training method in which additional prediction branches starting from intermediate future time steps (segments) are trained in conjunction with the main branch [16], [20], visualized in Fig. 3b. A prediction branch is simply a prediction from a start segment to an end segment. For example, branch 0 refers to the full N segment main branch, as used in the Multi-Segment SS-ASP, while subsequent prediction branches start from shifted time segments. This allows the model to perform $N-1$ additional, shorter predictions (of lengths 1 to $N-1$) in addition to the N-segment prediction covering the entire prediction horizon. As a result, additional training is performed without adding training data. To the best of the authors' knowledge, this is the first work that applies such overshooting methods in trajectory prediction. We apply it for action prediction and action reconstruction, as well as latent context prediction.

Incorporating multiple segments as well as multiple branches into the self-supervised loss of Eq. (1) extends the loss function over time segments and branches. For example, the trajectory loss $\mathcal{L}_{traj}$ can be extended to a sum of losses per branch b, $\mathcal{L}_{traj} = \sum_{b=0}^{N-1} \mathcal{L}_{traj}^{b}$. A similar extension can be performed for the classification, context, and reconstruction components of Eq. (1). Thus, the overall loss in Eq. (1) can be extended over branches and λ_i-weighted segments

$$\mathcal{L}_{MB-SS} = \sum_{b=0}^{N-1} \sum_{i=1}^{N-b} \lambda_i \mathcal{L}_{SS-ASP}^{i,b} \ . \tag{2}$$

Compared to Eq. (1), the loss function above provides significant additional training of the model on the same data.

2) Combining context aggregation and prediction: In autoregressive models, information from a previous recurrence step is used to predict the next step. An agent's observable state (e.g. its dynamics) does not constitute sufficient statistics for its behavior – to address this limitation, autoregressive predictions can learn additional latent features. A natural framework for modeling such problems where partial observability occurs is the Partially Observable Markov Decision Process (POMDP). Furthermore, since historical behavior beyond the previous recurrence step is relevant to determining the prediction, an accumulation of 'intent' over multiple recurrence steps should be possible. In this way, the future development of the scene can be modeled following Markovian assumptions.

The partial observability of intent over the entire prediction horizon can be handled through the use of recurrence over the latent context, which motivates the usage of a context aggregator component. For a segment i, it accumulates the encoded as well as predicted contexts of previous segments into an aggregated latent context $\bar{z}_i$, $\{z_{\tau_0}, ..., \hat{z}_{\tau_{i-1}}, \hat{z}_{\tau_i}\} \xrightarrow{\chi} \bar{z}_i$ (parameterized by χ). The aggregated context $\bar{z}_i$ can be considered as a latent context state that merges scene information of multiple consecutive segments. Thus, we use it as a stand-in for wherever predicted latent context is used, either as input to an action predictor or reconstructor component (e.g. in Fig. 2 and Fig. 3a). A concept similar to incorporating recurrence in latent context prediction are the Recurrent State Space Models (RSSM) in [16], which include stochastic components as opposed to the fully deterministic Multi-Branch SS-ASP.

E. Prediction uncertainty estimation

In this section, we present novel prediction uncertainty estimation techniques for the deterministic Multi-Branch SS-ASP model, as well as a corresponding evaluation procedure. Since we perform prediction over time segments, we model the change in prediction error between successive segments i and $i-1$ to capture the possibility of early predictions being accurate and later ones deviating significantly from the ground-truth. Specifically, we model the relative change in the Minimum Average Displacement Error (minADE) for k modes

$$\Delta\text{minADE}_k(i, i-1) = \text{minADE}_k(i) - \text{minADE}_k(i-1) \ , \tag{3}$$

and find correlations to the deterministic model uncertainty. In this sense, we aim to capture the model's changing confidence over time, and answer the *research question (ii)* from Sec. I.

The first uncertainty estimation metric is the **reconstruction error**. This metric aims to capture disagreement between model components generating predictions and reconstructions (past 'predictions') in inference. If the trajectory prediction of a segment i significantly deviates from the reconstruction in the same segment, it indicates the epistemic uncertainty of the overall model. Specifically, we measure the deviation between reconstructed xy positions $\hat{Y}_{\tau_i,\xi}$ of a segment i, obtained through the action reconstruction of the inverse model ξ (see Fig. 2), and the prediction $\hat{Y}_{\tau_i,\gamma}$, obtained through the action predictor γ (in case of segment 0 the ground-truth history Y_{τ_0})

$$\Delta_{recon}(i) = \begin{cases} \|\hat{Y}_{\tau_i,\xi} - \hat{Y}_{\tau_i,\gamma}\| & i \geq 1 \ , \\ \|\hat{Y}_{\tau_0,\xi} - Y_{\tau_0}\| & i = 0 \ . \end{cases} \tag{4}$$

In this implementation, the reconstruction error is coupled to the SS-ASP architecture due to a prerequisite for an inverse

(reconstruction) model. However, adding the auxiliary task of inverse predictions to any prediction model can serve as a relatively straightforward-to-use additional regularization.

The second metric is the **mean of mode variances**. The metric is based on the application of Monte Carlo dropout in order to obtain an uncertainty estimate, naturally provided by Bayesian inference [31]. Here, we estimate the variances of predicted trajectories $\hat{Y}$ under the dropout parameter distribution $q(w|\mathcal{D}_{train})$ of weights w given the training data $\mathcal{D}_{train}$. For a single xy position in a mode j of segment i, it is

$$\sigma^2_{\hat{Y}_{t,j,i}} = \frac{1}{2}\left(\mathrm{Var}\left(\hat{x}_{t,j,i}\right) + \mathrm{Var}\left(\hat{y}_{t,j,i}\right)\right) \ . \tag{5}$$

In practice, the variances are Monte-Carlo approximated by drawing multiple samples $w \sim q(w|\mathcal{D}_{train})$ with different weights dropped out [31]. In Eq. (5), we match same modes between different inference runs (by output order). We assume that variance between modes in a single run does not change significantly by applying dropout due to the inherent model determinism. The overall metric is obtained by averaging over modes and time steps (assuming independence)

$$\Delta_{mode-var}(i) = \sum_j^k \sum_t^T \frac{\sigma^2_{\hat{Y}_{t,j,i}}}{kT} \ . \tag{6}$$

The metric in Eq. (6) serves as an estimate of the covariance within a Bayesian network model whose weights are Gaussian distributed, which is theoretically grounded in [31]. Therefore, it serves as an indicator of the epistemic model uncertainty.

IV. RESULTS

A. Implementation

In implementing the Multi-Branch SS-ASP model, we use various network types to implement the components in Fig. 2. The context encoder ϕ embeds semantic images containing minimal driving context information (see Fig. 5) via a ResNet18 CNN with output feature dimension 256. The action encoder α is a 1D-CNN ActorNet model adapted from [34] generating 128-dim. output features. The action and context predictors γ and ψ, as well as the reconstructor ξ, are realized by three linear layers of dimensions $\{512, 256, 256\}$ (with $\tanh$ activation). The predictors γ and ψ have an additional two-layer Gated Recurrent Unit (GRU) with hidden state dimension 256, called iteratively three times. At the output of the action predictor γ, we use an additional linear layer and a `softmax` operation to map the feature vector to pseudo-probabilities of predicted modes. We use the same kinematic bicycle model setup as in [4] to obtain positions.

In training the model on the loss function in Eq. (2), we used the Huber loss function for all loss components in Eq. (1) (equal weights). For the multi-segment formulation, we used the segment length of 1s since we found it strikes a balance in capturing rich information on a short time interval. The loss values of different tasks (trajectory prediction and reconstruction, and context prediction) within a segment are averaged. This ensures that a segment is not over-represented in the overall loss since more terms can be present in later segments depending on the combination strategy. For efficiency, we batch different component calls over modes and among different prediction branches; we observed an approximately 1.75-times increase in training time over SS-ASP.

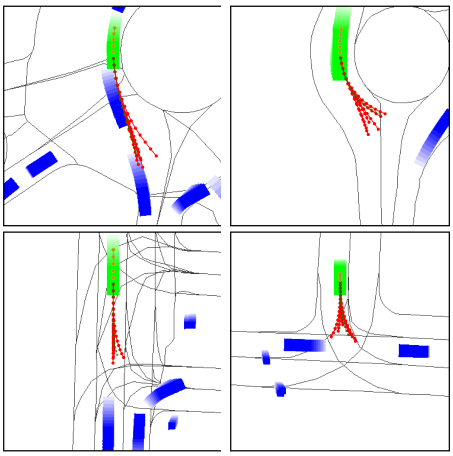


Fig. 5: Multi-Branch SS-ASP predictions on the INTERACTION dataset. The model uses a simplistic image-based representation of the driving context as input, representing past agent tracks by faded bounding boxes (prediction-ego in green, other agents in blue). The 6 predicted modes are shown in red and the ground-truth in green. The past reconstruction is shown within the prediction-ego depiction.

B. Datasets and training setup

The models are trained on the INTERACTION [18] and inD [35] datasets (using the same partition as in [6]), with 3s predictions based on 1s history ($N = 3$ in the multi-segment model). Implementation is done in PyTorch [36] with Adam optimizer [37] in training over 20 epochs and batch size 32, lasting two days for INTERACTION on a single Nvidia V100 GPU. The learning rate is set to 10^{-4} and multiplied with 0.5 if no improvement is observed in two consecutive epochs.

C. Prediction performance

We ablate the different multi-modal trajectory combination strategies in Tab. II. We see that Start-k outperforms others; this is expected due to its largest diversity in the first segment. Additionally, we offer an ablation study of the proposed approaches in Tab. III. It can be seen that (i) multi-branching and (ii) context aggregation bring boosts in metrics while multi-segmenting regresses the performance unless augmented with (i) and (ii). This is consistent with the results in [4], where a naive formulation with an End-k-like strategy is proposed. Overall, the Multi-Branch SS-ASP model brings a significant improvement of almost 25 % over the basic model. It shows that introducing additional training tasks without modifying more problem-relevant aspects (e.g. interaction modeling) can greatly improve prediction performance.

We compare the Multi-Branch-SS-ASP prediction results to reported results of other state-of-the-art models on the INTERACTION validation dataset in Tab. IV. Furthermore, we evaluate the model on the INTERACTION test set online leaderboard[6], where it achieves a competitive *3rd* place in minADE_6 and minFDE_6. However, it scores *9th* in Miss Rate (MR); this is understandable since the model components are inherently ill-equipped to handle interaction modeling due to

[6]http://challenge.interaction-dataset.com/leader-board as of 01-Feb-2023

TABLE II: Comparison of combination strategies for multi-segment multi-modal prediction. The parameters k and m are chosen such that the same number of resulting modes is obtained, $K = 8$.

Method	Configuration	inD [35]	
		minADE_6	minFDE_6
All-Modes	$k = 2$	0.25	0.61
Start-k	$k = 8$	**0.20**	**0.51**
End-k	$k = 8$	**0.20**	0.53
Best-m-of-All	$m = 8$	0.24	0.59
Best-m-of-Pred.	$m = 2$	0.22	0.54

TABLE III: Ablation study of proposed approaches: self-supervision, multi-segment chaining (with Start-k strategy, $k = 9$), branched overshooting, and context aggregation. Multi-branch* denotes branched overshooting (Sec. III-D1) without context aggregation (Sec. III-D2).

Model	Self-Sup.	Multi-seg.	Multi-branch	Context agg.	inD [35]	
					min-ADE_9	min-FDE_9
FF-ASP [4]	✗	✗	✗	✗	0.22	0.56
SS-ASP [4]	✓	✗	✗	✗	0.19	0.50
Multi-seg. SS-ASP	✓	✓	✗	✗	0.20	0.52
Multi-branch* SS-ASP	✓	✓	✓	✗	0.17	0.45
Multi-branch SS-ASP	✓	✓	✓	✓	**0.17**	**0.43**

TABLE IV: Minimal displacement metrics on the INTERACTION validation dataset. All methods predict $K = 6$ modes. We do not include [6], [41] due to a different number of predicted modes.

	INTERACTION [18]	
	minADE_6	minFDE_6
TNT [42]	0.21	0.67
STG-DAT [43]	0.29	0.54
ITRA [13]	0.17	0.49
GOHOME [7]	-	0.45
FF-ASP [4]	0.12	0.35
DIPA [44]	0.11	0.34
SS-ASP [4]	0.11	0.33
Multi-Branch SS-ASP	**0.10**	**0.30**

the very low-information-density environment representation and simplistic CNN encoding. For such purposes, many state-of-the-art approaches use graph- or Transformer-based [38] architectures in their encoders [6], [39] as well as target selection heuristics in their decoders [3], [40]. Such approaches could be easily integrated into the overall architecture. The generality of the self-supervision, segment-wise prediction, and branched training does not preclude component-level improvements.

D. Prediction uncertainty estimation

We evaluate the uncertainty quantification strategies from Sec. III-E by observing whether they correlate with the change in prediction error over successive segments. We quantify the prediction error by the ΔminADE in Eq. (3). In this way, a high value of the uncertainty metric could indicate that the model's predictions will deteriorate over time.

We calculate the metrics from Sec. III-E for each predicted segment on a randomly chosen $10\,\%$ inD subset. The results are visualized in Fig. 6. To ensure comparability between the two methods, we group the (sorted) obtained values into four quarters, where each quarter contains $25\,\%$ of the overall values (the first quarter is equivalent to the first quantile). Then, within each quarter we approximate the ΔminADE_k error distribution by a four bin histogram (lightest to darkest blue in Fig. 6). The bin intervals are determined by the quarters of the ΔminADE_k error distribution on the validation set.

Interpreting Fig. 6, we see that the change in the error distribution between quarters is evident. For example, quartile 1 of segment 1 (Fig. 6a) contains the lowest-reconstruction-error and more than $60\,\%$ of its values lie in the low ΔminADE_k range (lightest blue). Similarly, in quarter 4 of Fig. 6a (containing the highest metric values) the histogram distribution is biased towards high ΔminADE_k samples (darkest blue). Therefore, a correlation between the metric and the actual change in prediction error over segments can be confirmed. Similar relationships can be found for the dropout-based mean-of-mode-variances, where 20 Monte Carlo runs are performed (we dropped-out the two linear layers before the action predictor with $p = 0.5$). Furthermore, we observe that in both metrics the histogram distribution favors higher ΔminADE_k at later segments, which is reasonable.

V. CONCLUSION

In this paper, we investigated connections between one-shot and autoregressive trajectory prediction models. We deliberately focused on the structure of output representations and the training approach, as opposed to more problem-relevant aspects such as driving context and interaction modeling, in order to better see the effects of the proposed approach. We found significant gains by converting an existing one-shot predictor into a novel, segment-wise prediction trained with self-supervision and overshooting. Furthermore, we proposed two epistemic uncertainty measures for deterministic predictors. In combination with the segment-wise output structure, they pave way for prediction of a variable time horizon with the goal of providing only confident predictions to a downstream planner.

REFERENCES

[1] J. Gao, C. Sun, H. Zhao, Y. Shen, D. Anguelov, C. Li, and C. Schmid, "VectorNet: Encoding HD Maps and Agent Dynamics from Vectorized Representation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[2] Y. Yuan, X. Weng, Y. Ou, and K. Kitani, "Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting," *CoRR*, vol. abs/2103.14023, 2021. [Online]. Available: https://arxiv.org/abs/2103.14023

[3] B. Varadarajan, A. Hefny, A. Srivastava, K. S. Refaat, N. Nayakanti, A. Cornman, K. Chen, B. Douillard, C. P. Lam, D. Anguelov *et al.*, "Multipath++: Efficient information fusion and trajectory aggregation for behavior prediction," *arXiv preprint arXiv:2111.14973*, 2021.

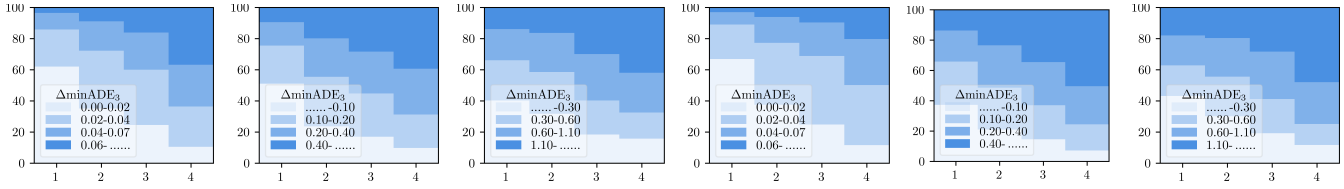
[4] F. Janjoš, M. Dolgov, and M. J. Zöllner, "Self-Supervised Action-Space Prediction for Automated Driving," in *2021 IEEE Intelligent Vehicles Symposium (IV)*, 2021.

[5] Q. Lu, W. Han, J. Ling, M. Wang, H. Chen, B. Varadarajan, and P. Covington, "Kemp: Keyframe-based hierarchical end-to-end deep model for long- term trajectory prediction," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022.

[6] F. Janjoš, M. Dolgov, and M. J. Zöllner, "StarNet: Joint Action-Space Prediction with Star Graphs and Implicit Global-Frame Self-Attention," *arXiv preprint arXiv:2111.13566*, 2021.

[7] T. Gilles, S. Sabatini, D. Tsishkou, B. Stanciulescu, and F. Moutarde, "Gohome: Graph-oriented heatmap output for future motion estimation," *arXiv preprint arXiv:2109.01827*, 2021.

[8] S. Casas, C. Gulino, S. Suo, K. Luo, R. Liao, and R. Urtasun, "Implicit Latent Variable Model for Scene-Consistent Motion Forecasting," *arXiv preprint arXiv:2007.12036*, 2020.



(a) Rec. error seg. 1 (b) Rec. error seg. 2 (c) Rec. error seg. 3 (d) Mode var. seg. 1 (e) Mode var. seg. 2 (f) Mode var. seg. 3

Fig. 6: Evaluation of reconstruction error and mean-of-mode-variances uncertainty estimation metrics over three predicted segments. The metric values are split into four quarters (horizontal axes) and the distribution of each quarter w.r.t. ΔminADE_k (Eq. (3)) is shown in different shades of blue (vertical axes indicate %). The metrics correlate with prediction error change – it can be seen that lower metric quarters contain mostly lower ΔminADE_k values (lighter shades of blue) and higher metrics contain higher ΔminADE_k (darker shades).

[9] X. Huang, G. Rosman, I. Gilitschenski, A. Jasour, S. G. McGill, J. J. Leonard, and B. C. Williams, "Hyper: Learned hybrid trajectory prediction via factored inference and adaptive sampling," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022.

[10] N. Rhinehart, R. McAllister, K. Kitani, and S. Levine, "PRECOG: Prediction Conditioned on Goals in Visual Multi-Agent Settings," in *Proceedings of the IEEE Int. Conf. on Computer Vision*, 2019.

[11] C. Tang and R. R. Salakhutdinov, "Multiple Futures Prediction," in *Advances in Neural Information Processing Systems*, 2019.

[12] R. Mahjourian, J. Kim, Y. Chai, M. Tan, B. Sapp, and D. Anguelov, "Occupancy flow fields for motion forecasting in autonomous driving," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, apr 2022. [Online]. Available: https://doi.org/10.1109%2Flra.2022.3151613

[13] A. Scibior, V. Lioutas, D. Reda, P. Bateni, and F. Wood, "Imagining The Road Ahead: Multi-Agent Trajectory Prediction via Differentiable Simulation," *arXiv preprint arXiv:2104.11212*, 2021.

[14] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proc. of the 14th international conf. on artificial intelligence and statistics*, 2011.

[15] A. Venkatraman, M. Hebert, and J. A. Bagnell, "Improving multi-step prediction of learned time series models," in *AAAI*, 2015.

[16] D. Hafner, T. P. Lillicrap, I. Fischer, R. Villegas, D. Ha, H. Lee, and J. Davidson, "Learning latent dynamics for planning from pixels," *CoRR*, vol. abs/1811.04551, 2018. [Online]. Available: http://arxiv.org/abs/1811.04551

[17] M. Lutter, L. Hasenclever, A. Byravan, G. Dulac-Arnold, P. Trochim, N. Heess, J. Merel, and Y. Tassa, "Learning dynamics models for model predictive agents," *CoRR*, vol. abs/2109.14311, 2021. [Online]. Available: https://arxiv.org/abs/2109.14311

[18] W. Zhan, L. Sun, D. Wang, H. Shi, A. Clausse, M. Naumann, J. Kummerle, H. Konigshof, C. Stiller, A. de La Fortelle *et al.*, "INTERACTION Dataset: An INTERnational, Adversarial and Cooperative MoTION Dataset in Interactive Driving Scenarios with Semantic Maps," *arXiv preprint arXiv:1910.03088*, 2019.

[19] A. Hu, G. Corrado, N. Griffiths, Z. Murez, C. Gurau, H. Yeo, A. Kendall, R. Cipolla, and J. Shotton, "Model-based imitation learning for urban driving," *arXiv preprint arXiv:2210.07729*, 2022.

[20] A. H. Li, P. Wu, and M. Kennedy, "Replay overshooting: Learning stochastic latent dynamics with the extended kalman filter," in *2021 IEEE International Conf. on Robotics and Automation (ICRA)*, 2021.

[21] D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi, "Dream to control: Learning behaviors by latent imagination," *arXiv:1912.01603*, 2019.

[22] H. Ma, Y. Sun, J. Li, and M. Tomizuka, "Multi-agent driving behavior prediction across different scenarios with self-supervised domain knowledge," in *2021 IEEE Intelligent Transportation Systems (ITSC)*.

[23] M. Geisslinger, P. Karle, J. Betz, and M. Lienkamp, "Watch-and-learn-net: Self-supervised online learning for probabilistic vehicle trajectory prediction," in *2021 IEEE international conference on systems, man, and cybernetics (SMC)*, 2021.

[24] M. Ye, J. Xu, X. Xu, T. Cao, and Q. Chen, "Dcms: Motion forecasting with dual consistency and multi-pseudo-target supervision," *arXiv preprint arXiv:2204.05859*, 2022.

[25] J. Gao, C. Sun, H. Zhao, Y. Shen, D. Anguelov, C. Li, and C. Schmid, "Vectornet: Encoding HD maps and agent dynamics from vectorized representation," *CoRR*, vol. abs/2005.04259, 2020. [Online]. Available: https://arxiv.org/abs/2005.04259
dreamer: World models for physical robot learning," *arXiv preprint arXiv:2206.14176*, 2022.

[26] P. Wu, A. Escontrela, D. Hafner, K. Goldberg, and P. Abbeel, "Day-

[27] J. Y. Koh, H. Lee, Y. Yang, J. Baldridge, and P. Anderson, "Pathdreamer: A world model for indoor navigation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

[28] P. Agrawal, A. V. Nair, P. Abbeel, J. Malik, and S. Levine, "Learning to Poke by Poking: Experiential Learning of Intuitive Physics," in *Advances in Neural Information Processing Systems*, 2016.

[29] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell, "Curiosity-driven exploration by self-supervised prediction," in *International conference on machine learning*, 2017.

[30] A. Filos, P. Tigkas, R. McAllister, N. Rhinehart, S. Levine, and Y. Gal, "Can autonomous vehicles identify, recover from, and adapt to distribution shifts?" in *Int. Conf. on Machine Learning (ICML)*, 2020.

[31] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ser. ICML'16. JMLR.org, 2016.

[32] A. Nayak, A. Eskandarian, and Z. Doerzaph, "Uncertainty estimation of pedestrian future trajectory using bayesian approximation," *IEEE Open Journal of Intelligent Transportation Systems*, 2022.

[33] S. Khandelwal, W. Qi, J. Singh, A. Hartnett, and D. Ramanan, "What-If Motion Prediction for Autonomous Driving," *arXiv preprint arXiv:2008.10587*, 2020.

[34] M. Liang, B. Yang, R. Hu, Y. Chen, R. Liao, S. Feng, and R. Urtasun, "Learning Lane Graph Representations for Motion Forecasting," in *European Conference on Computer Vision*, 2020.

[35] J. Bock, R. Krajewski, T. Moers, S. Runde, L. Vater, and L. Eckstein, "The ind dataset: A drone dataset of naturalistic road user trajectories at german intersections," *CoRR*, vol. abs/1911.07602, 2019. [Online]. Available: http://arxiv.org/abs/1911.07602

[36] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," *arXiv preprint arXiv:1912.01703*, 2019.

[37] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[38] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention Is All You Need," in *Advances in Neural Information Processing Systems*, 2017.

[39] N. Nayakanti, R. Al-Rfou, A. Zhou, K. Goel, K. S. Refaat, and B. Sapp, "Wayformer: Motion forecasting via simple & efficient attention networks," *arXiv preprint arXiv:2207.05844*, 2022.

[40] J. Gu, C. Sun, and H. Zhao, "Densetnt: End-to-end trajectory prediction from dense goal sets," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

[41] F. Janjoš, M. Dolgov, M. Kurić, Y. Shen, and J. M. Zöllner, "San: Scene anchor networks for joint action-space prediction," in *2022 IEEE Intelligent Vehicles Symposium (IV)*, 2022.

[42] H. Zhao, J. Gao, T. Lan, C. Sun, B. Sapp, B. Varadarajan, Y. Shen, Y. Shen, Y. Chai, C. Schmid *et al.*, "TNT: Target-driveN Trajectory Prediction," *arXiv preprint arXiv:2008.08294*, 2020.

[43] J. Li, H. Ma, Z. Zhang, J. Li, and M. Tomizuka, "Spatio-Temporal Graph Dual-Attention Network for Multi-Agent Prediction and Tracking," *arXiv preprint arXiv:2102.09117*, 2021.

[44] A. Knittel, M. Hawasly, S. V. Albrecht, J. Redford, and S. Ramamoorthy, "Dipa: Diverse and probabilistically accurate interactive prediction," *arXiv preprint arXiv:2210.06106*, 2022.

Paper III

- Title: *StarNet: Joint Action-Space Prediction with Star Graphs and Implicit Global-Frame Self-Attention*
- Authors: Faris Janjoš and Maxim Dolgov and J. Marius Zöllner
- Venue: 2022 IEEE Intelligent Vehicles Symposium (IV)
- Best Paper Runner-Up Award

2022 IEEE Intelligent Vehicles Symposium (IV)
June 5-9, 2022. Aachen, Germany

StarNet: Joint Action-Space Prediction with Star Graphs and Implicit Global-Frame Self-Attention

Faris Janjoš[1], Maxim Dolgov[1], and J. Marius Zöllner[2]

***Abstract*— In this work, we present a novel multi-modal multi-agent trajectory prediction architecture, focusing on map and interaction modeling using graph representation. For the purposes of map modeling, we capture rich topological structure into vector-based star graphs, which enable an agent to directly attend to relevant regions along polylines that are used to represent the map. We denote this architecture StarNet, and integrate it into a single-agent prediction setting. As the main result, we extend this architecture to joint scene-level prediction, which produces multiple agents' predictions simultaneously. The key idea in joint-StarNet is integrating the awareness of one agent in its own reference frame with how it is perceived from the points of view of other agents. We achieve this via masked self-attention. Both proposed architectures are built on top of the action-space prediction framework introduced in our previous work, which ensures kinematically feasible trajectory predictions. We evaluate the methods on the interaction-rich inD and INTERACTION datasets, with both StarNet and joint-StarNet achieving improvements over state of the art.**

I. INTRODUCTION

Accurate prediction of the driving situation is a major cornerstone for achieving performant full autonomy of self-driving cars. Despite a strong research and industry focus, there are many problems to be solved, such as understanding complex social interactions among different agents and effectively incorporating rich topological information. Other important aspects are prediction of multi-modal trajectories, conditioning predictions on assumed goals of given agents, as well as achieving reasonable long-term predictions. In tackling these challenges, Deep Neural Networks (DNN) have shown great results over classical robotics approaches, especially for the use-case of urban driving.

The challenges of environment representation and interaction modeling are tightly coupled, e.g. the maneuvers of two negotiating vehicles in a highly-interactive situation are constrained by the road topology. Therefore, a learned model must consider the effects of topology on the driving situation. This makes the representation of map information paramount: it should enable explicit relationships to map elements such as lane centerlines and road boundaries, as well as segments along these elements. Furthermore, it must allow the model to discern between more and less important segments. In this sense, a direct, explicit representation of map geometry serves as a context for the social interaction.

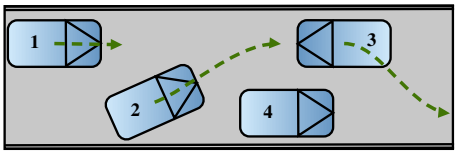


Fig. 1: Example of a highly interactive situation requiring joint scene prediction: vehicle 1 slows down to allow vehicle 2 to overtake parked vehicle 4; vehicle 3 must reverse behind vehicle 4.

Predictions for interacting vehicles in a driving scene must be consistent. Therefore, it is desirable to do a joint prediction and to consider mutual social interaction (example in Fig. 1). In contrast, predicting separately for each agent assumes that each vehicle considers others as part of its local environment. In this way, the scene at hand is considered from the point of view of each of the vehicles while marginalizing other vehicles, which is redundant and may lead to inconsistent predictions. This brings the question, how can we share this mutual local information? A potential solution is predicting jointly in a global reference frame; this eliminates the redundancy, but at the cost of an arbitrary dependency on the origin placement. Otherwise, performing individual, marginal predictions hinders sharing of intention information between local representations of the same agent.

In handling the joint nature of the prediction problem, many works use representations centered around the autonomous vehicle (AV) [1], [2], [3], [4], [5], in the form of sensor data or generated Bird's Eye View (BEV) images. In order to obtain agent-specific features, they extract patches around the agents of interest, derive corresponding latent information, and perform individual or joint predictions, provided that the latent features are combined. This introduces twofold disadvantages. First, grid-based environment representation assumes that the model extracts map objects and agents implicitly from individual pixels while inferring semantic knowledge. And second, the networks are tasked with associating information about the same agents from different sets of latent features, which might vary if the agents' perspectives are different. In particular, if there is no overlap among the initial patches, it might be difficult to infer relational information between two agents if they don't exist in each other's immediate environment representations.

In our approach, we help our models learn by explicitly relating the same agents from other agents' perspectives. Furthermore, we enable them to learn to attend to the most relevant map elements and their segments directly. The contributions of our work are the following:

[1] Robert Bosch GmbH, Corporate Research, Advanced Autonomous Systems, 71272 Renningen, Germany. {faris.janjos, maxim.dolgov}@de.bosch.com

[2] Research Center for Information Technology (FZI), 76131 Karlsruhe, Germany. zoellner@fzi.de

This work was financially supported by the Federal Ministry of Economic Affairs and Energy of Germany, grant number 19A20026H, based on a decision of the German Bundestag.

2022 IEEE Intelligent Vehicles Symposium (IV) | 978-1-6654-8821-1/22/$31.00 ©2022 IEEE | DOI: 10.1109/IV51971.2022.9827091

- A novel, star-graph polyline representation, with unconstrained field of view (FoV) (given a map description) and direct modeling of most relevant segments.
- A mechanism to explicitly combine features from multiple reference frames via self-attention [6] while masking out irrelevant information, enabling joint scene-level prediction.
- State-of-the-art performance on the INTERACTION dataset [7], containing challenging roundabout, intersection, and highway merge scenarios.

II. RELATED WORK

The field of trajectory prediction has generated an exhaustive literature [8]. Within the field of autonomous driving, the multifaceted nature of the problem yields works that focus on specific challenges within the overall landscape. They include but are not limited to: environment representation [9], [10], [11], multi-agent interaction [12], [13], [14], multi-modality [15], [16], [17], goal-conditioning [18], [19], [20], and kinematic constraints [21], [22], [23]. Furthermore, some works integrate prediction with detection and planning [24], [25], [3], as an important step in achieving end-to-end self-driving. Our work focuses on **map representation** and **joint interaction modeling**. In addressing these challenges, approaches using Convolutional Neural Networks (CNN) [26] and Graph Neural Networks (GNN) [27] are prevalent, with GNNs increasingly used over CNNs.

A. Graph-based map representation

Encoding map information into graphs and using GNNs offers several advantages over image-based inputs used in conjunction with CNNs. CNNs extract locational features from rasterized BEV images or grids with LiDAR point clouds. In contrast, GNNs can learn directly from graph-based representations, which encode the geometric structure of the road into nodes and edges. In doing so, they alleviate the need to infer objects from pixels, as well as improve efficiency due to fewer weights. Furthermore, graphs benefit from a larger FoV than rasters, which are usually restricted by image dimensions and resolution. These factors contribute to a substantially higher representation density.

VectorNet [9] is a seminal work that uses graph-based map representations. This approach fits polylines to map elements and dissects them into their constituent vectors. Then, fully-connected graphs are constructed per map element (e.g. lanes and boundaries) and aggregated by a GNN into a feature vector. This procedure is used as a basis in further works [16], [14], [17]. Alternative approaches are [10] and [28], redefining the graph convolution operation with additional adjacency matrices in order to capture a larger receptive field. Consequently, they are able to capture a longer range longitudinally, as well as account for the different semantic meaning of lateral lanes. Another class of works is [29] and [19], which model attention to specific lanes or sections along a reference polyline, constructed by concatenating individual polyline points. In the case of [19], this enables placement of hypothetical goals along the polyline to condition the trajectory prediction.

Works like VectorNet construct fully-connected graphs for each map element in order to mitigate the GNN information bottleneck[1] [30] problem. This enables each node in the graph to be no more than one hop away from any other node. However, as a result, unnecessary information is shared between vectors that are not physically close but are part of the same polyline. Thus, [9] only considers the nodes within a distance threshold to the predicted vehicle, in turn limiting the receptive field in aggregation. Furthermore, [9] includes ordering information into the node attributes via an integer index, which raises correctness questions since integers are combined with floating point features such as xy positions.

Regarding the **map-representation** aspect of our work, instead of connecting vectors by their polyline membership, we ask the question, which parts of a polyline are the most relevant *to an agent*? We task a Graph Attention Network (GAT) [31] with the answer; the attention mechanism learns to determine the most relevant vectors without artificially limiting the receptive field. Furthermore, our map representation is simpler to pre-process than [10], [28] since we use standard graph convolutions, as well as [19], [29], since we do not manually select the reference polyline and allow other map element types to be attended to as well.

B. Joint graph-based interaction modeling

Social interaction in a driving scene can inherently be represented as a natural graph, where nodes are agents and edges model their (weighted) connections. Hence, virtually all recent state-of-the-art approaches use graph-based learned models such as GNNs and Multi-Head Attention (MHA), which is related to the Transformer architecture[2] [6]. For the sake of brevity, we limit our review to joint prediction works [12], [18], [33], [2], [3], [4], [34], [35], [20], [5].

Among these approaches, [12], [18], [2], [3], [4], [34] use deep generative models. Here, they capture the uncertainty in future development of a scene into a set of latent variables and sample from the latent space. The major drawback of such approaches is the randomness of their outputs because they require sampling from a latent distribution during inference. Thus, likelihood estimation of the predicted trajectory distribution is difficult. This is exacerbated in the single-step prediction approaches [12], [18], which construct a trajectory iteratively. Another drawback is limited map information; approaches either don't consider the map at all [33], or use representations centered around the AV [2], [3], [4]. As mentioned in Sec. I, the AV-centered view in conjunction with a CNN implementation offered in these approaches requires extracting local patches around each agent of interest and relating information implicitly under a limited FoV. The AV-centering has another disadvantage in that the interaction between agents has to be inferred indirectly through the AV

[1]Modeling the full map connectivity as a mesh-like natural graph and interweaving the agents nodes would result in long chains of propagated information. Learning over such a graph would necessitate many iterations of message passing and yield over-smoothed node embeddings.

[2]Incidentally, the basic Transformer layer is equivalent to the GNN GAT layer, in the case of multiple attention heads and a fully-connected underlying graph [32].

perspective (even if the agents themselves are not interacting with the AV). As an alternative, [20] and [35] perform joint prediction in the global reference frame, but reduce the effects of arbitrary origin placement by injecting the global-frame map via cross-attention and Long Short-Term Memory (LSTM)-like [36] implicit gating, respectively.

Regarding the **joint interaction modeling** aspect of our work, we start with deterministic one-shot prediction outputs given deterministic inputs. We model the map explicitly from local-frame graph representations without limiting the FoV. Our work is closest to the prediction model in [20], however, instead of using global reference frames, we ask the question, how is an agent jointly perceived by other agents? We arrive at a MHA model with masking, which combines multiple local representations to construct an implicit global frame. Furthermore, our model is smaller since it uses two GAT and two MHA layers to model the map and social interaction, compared to 18 MHA layers in [20]. Finally, we frame the model in the action-space framework of our previous work [23], ensuring kinematically feasible outputs.

III. METHOD

A. Background

Consider the task of vehicle trajectory prediction in a driving situation with N heterogeneous interacting agents. We define the single-agent prediction problem as predicting the distribution of future waypoints $\hat{\mathbf{Y}}_i$ of vehicle i. It can be framed in an imitation learning setting, where a learned model parameterizes the distribution

$$\hat{Y}_i \sim P(\hat{\mathbf{Y}}_i|\mathcal{D}^i) \ , \tag{1}$$

conditioned on the local context $\mathcal{D}^i$ of vehicle i. Here, the superscript indicates that the values are represented in the local reference frame of vehicle i, subscript indicates prediction for agent i, while $\hat{\cdot}$ denotes predicted future values. We simplify the problem in (1) by predicting a sample $\hat{Y}_i$, e.g. a $2 \times T$ matrix of xy coordinates at T future time steps.

The context $\mathcal{D}^i = \{\mathcal{M}^i, \mathcal{T}^i\}$ contains the map $\mathcal{M}^i$ and past position tracks $\mathcal{T}^i$ of vehicle i and its neighboring $N-1$ agents, with $\mathcal{T}^i = \{X_j^i\}_{j=1}^N$. Each track X_j^i is a $3 \times T$ matrix of xy coordinates of agent j over T past time steps (in the reference frame of vehicle i at the current time step) and a padding row, since the agent might not be present in the scene for each time step. The padding row contains zeros for non-existent points (and zero positions) and ones otherwise.

In joint prediction, we consider the task of predicting K ($K \leq N$) vehicles' trajectories simultaneously. Thus, the learned model parameterizes the distribution

$$\hat{Y} \sim P(\hat{\mathbf{Y}}|\mathcal{D}) \ . \tag{2}$$

Therefore, we predict a sample $\hat{Y}$ of $\hat{\mathbf{Y}}$ for future trajectories of all K vehicles, given their contexts $\mathcal{D}$, where $\hat{Y} = \{\hat{Y}_k\}_{k=1}^K$ and $\mathcal{D} = \{\mathcal{D}^k\}_{k=1}^K$, respectively. Each $\mathcal{D}^k$ can be separated into its map and tracks components, $\mathcal{M}^k$ and $\mathcal{T}^k = \{X_j^k\}_{j=1}^N$. Note that tracks $\mathcal{T}^k$ contain trajectories for all N agents, including those who are not considered in prediction such as pedestrians and bicycles[3].

[3]In principle, bicycles can considered in action-space prediction (Sec. III-B) since in this model they share the same action/state space as vehicles.

In parameterizing the distributions in (1) and (2), we use a general encoder–decoder structure, with a context encoder and an output decoder. The encoder reasons about the context $\mathcal{D}$ while the decoder generates predicted trajectories $\hat{Y}$. Furthermore, we make use of track encoders, e.g. we encode each agent track X_j^i ($i \in [1, ..., K]$, $j \in [1, ..., N]$) into a feature vector z_j^i via a 1D CNN network.

B. Action-space prediction

We use positional representations in modeling the environment $\mathcal{D}$ within the context encoder. Both the polyline map $\mathcal{M}$ (as will be shown later), as well as the tracks $\mathcal{T}$, contain xy coordinates describing polyline control points and past trajectories, respectively. Hence, in generating learned feature representations of the environment, map-dependent interactions are inferred from data in the Euclidean space.

When generating the predicted trajectories $\hat{Y}$ via the decoder, we shift the learning problem into the action-space of accelerations and steering angles. We provide past actions as action tracks A_i^i ($i \in [1, ..., K]$) and generate future actions with a Gated Recurrent Unit (GRU) [37]. This is consistent with the action-space prediction framework [23] and guarantees that a learned model does not have to capture motion models, as well as ensuring kinematic feasibility (with a bicycle kinematic model in the output). Similarly to position tracks, we encode past action tracks A_i^i into feature vectors w_i^i with the same network.

C. Single-agent prediction with StarNet

In this section, we present a single-agent prediction method for regressing the future trajectory of vehicle i (prediction-ego). We approximate (1) via a deterministic encoder–decoder model. The encoder consists of a star graph map model and a map-dependent interaction model, while the decoder is a multi-modal action predictor, directly generating m samples from the distribution (1).

1) Star graph map model: Key component of single-agent StarNet is its representation of map elements via star graphs. First, each map element, such as a sidewalk, lane center-line, or a traffic island, is approximated by a polyline consisting of fixed-length vectors, similarly as in VectorNet [9]. Thus, the representation $\mathcal{M}^i$ of the map in the agent i's reference frame consists of Q polylines

$$\mathcal{M}^i = \{q_j^i\}_{j=1}^Q \ . \tag{3}$$

In turn, each polyline consists of L vectors comprising their start and end xy coordinates and one-hot type encoding

$$q_j^i = \{v_{jl}^i\}_{l=1}^L \ , \tag{4}$$

$$v_{jl}^i = [v_{\text{start}}, v_{\text{end}}, v_{\text{type}}]^T \ . \tag{5}$$

Given this polyline representation, we construct a directed star graph for each polyline q_j^i. In this structure, the past prediction-ego track is the central node with embedded track features z_i^i and edges connecting each vector v_{jl}^i to the central node. To ensure message passing compatibility, we embed the vector nodes (5) into the same dimensionality as the features z_i^i using a linear layer. Finally, we feed this graph to a 2-layer GAT and aggregate the nodes via max-pooling to

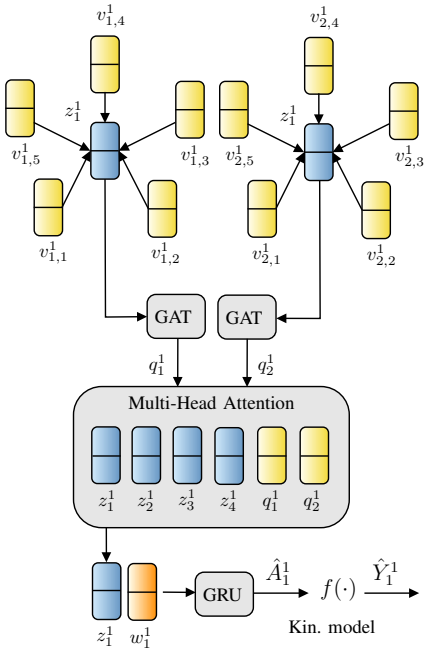
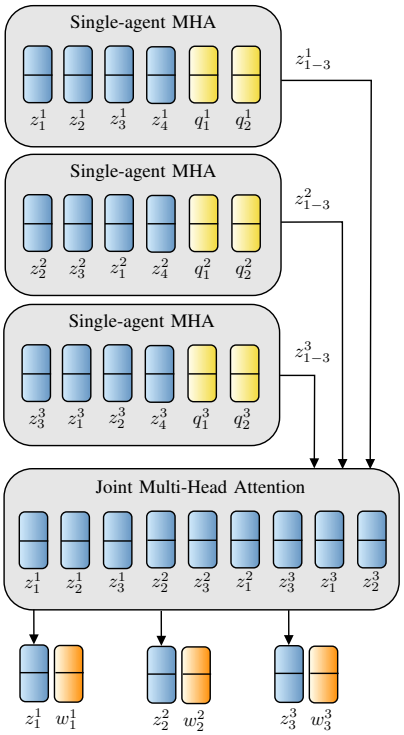


Fig. 2: StarNet single-agent architecture for the driving scene in Fig. 1. Vehicle 1 is the prediction-ego, and two polylines and three neighboring agents are in its local context. First, the two polylines determined by five vectors each are represented in a directed star graph with prediction-ego track embedding z_1^1 in the center and vector embeddings $v_{\{1,2\},\{1,5\}}^1$ outward. Polyline-level embeddings are generated by a 2-layer GAT, followed by a 1-layer Multi-Head Attention modeling map-dependent social interaction via aggregating all agents' and polylines' embeddings, $z_{\{1,4\}}^1$ and $q_{\{1,2\}}^1$ respectively. Then, prediction-ego embedding is selected, concatenated with its action track embedding w_1^1, and fed through an action-prediction GRU to predict future actions $\hat{a}_1^1$. Finally, positions $\hat{x}_1^1$ are obtained via a kinematic model transformation.



(a) Joint-StarNet with both map element star-graphs and action decoder blocks omitted: Single-agent Multi-Head Attention blocks model the local context for the joint prediction candidate vehicles $1-3$ (4 is not considered for prediction), with $K = 3$ and $N = 4$. Features z_{1-3}^1, z_{1-3}^2, z_{1-3}^3 are selected and fed into the Joint Multi-Head Attention block, which aggregates local features to construct an implicit global frame. After this operation, features z_i^i ($i \in [1, 2, 3]$) are selected and together with their action embeddings w_i^i fed into the action decoder block (not shown).

$$\left[\begin{array}{ccc|ccc|ccc} 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 2 & 0 & 2 & 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 3 & 0 & 3 & 0 & 3 & 0 & 0 \\ \hline 0 & 2 & 0 & 2 & 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 3 & 0 & 3 & 0 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ \hline 0 & 0 & 3 & 0 & 3 & 0 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 2 & 0 & 2 & 0 & 0 & 0 & 0 & 2 \end{array}\right]$$

(b) Depicted is the corresponding 9×9 attention mask for the three vehicles from Fig 3a. Here, each non-zero number denotes attention to a feature vector of a vehicle in a row (the number indicates the vehicle in question), while zero denotes no attention. Each 3×3 matrix in a block-row can be obtained by left-shifting the left-neighbor and up-shifting the upper-neighbor 3×3 matrices by one.

Fig. 3: Joint-StarNet in Fig. 3a (for the scene in Fig. 2) and the attention mask for the Joint Multi-Head Attention block in Fig. 3b.

obtain polyline-level embeddings q_j^i of same dimensionality as the nodes. This structure is depicted in Fig. 2.

The star graph and the accompanying GAT model the relationship between the ego track and the map. Contrary to VectorNet's fully-connected graphs, we assume that there is more information contained in the vehicle's direct attention (represented by its past track embedding) to a specific vector within a polyline rather than between the polyline vectors themselves. This allows us to expand the receptive field, since the attention mechanism will learn to ignore distant vectors, and to consider them proportionally to their weights in aggregation. Furthermore, the learning model is simplified by removing the need to include artificial ordering into the vector (5) that is needed in VectorNet [9] in order to help convey the polyline geometry.

2) Map-dependent interaction model: In the single-agent StarNet, we model map-dependent social interaction with a MHA [6], see Fig. 2. We combine vehicle i's past track embedding z_i^i with polyline embeddings q_j^i ($j \in [1, ..., Q]$), as well as track embeddings z_j^i ($j \in [1, ..., N-1]$) of each agent sharing the scene with i. Then, we stack the embedding vectors into a matrix with $N + Q$ rows and feed it into a single MHA layer. Here, linear projections of the input are generated in the form of query, key, and value matrices. Then, the self-attention operation [6] is applied in order to infer the relationships between the embeddings. The output of the MHA is of the same dimensionality as the stacked input matrix, and we select the row that corresponds to the vehicle i. Through the GAT and MHA layers, the obtained embedding is able to capture the map-dependent social interaction in the local context of the prediction-ego. The next step is feeding it into the action output decoder.

3) Multi-modal action decoder: The action decoder combines the positional embedding of the prediction-ego, aggregated to consider the map-dependent social interaction, with its action embedding. We concatenate z_i^i with w_i^i and feed it into the action decoder, which is the same GRU network

as in [23] (depicted in Fig. 2). It generates steering angles and accelerations, directly predicting m action modes (action trajectories and softmax scores, which can be interpreted as probabilities). The modes are converted to predicted vehicle positions via a bicycle kinematic model, fully capturing kinematic characteristics of motion.

We train the whole pipeline with the loss

$$\mathcal{L} = \mathcal{L}_{\text{reg}} + \beta \mathcal{L}_{\text{class}} \ , \tag{6}$$

where $\mathcal{L}_{\text{reg}}$ considers the mismatch to the ground-truth future trajectory and $\mathcal{L}_{\text{class}}$ considers the mode probability via cross-entropy, same as in [23], with β set to 1.

D. Joint prediction with joint-StarNet

In this section, we present a joint prediction method for regressing the future trajectories of K vehicles in a driving scene ($k \in [1, ..., K], K \leq N$). We directly model the joint distribution (2) without factorizing individual agents. The joint-StarNet architecture builds on StarNet from Sec. III-C and achieves joint prediction by aggregating local features into an implicit global frame via masked self-attention.

1) Implicit global frame self-attention: The joint-StarNet is an extension to single-agent StarNet. After determining the joint prediction candidates, we perform the first two steps of the single-agent StarNet pipeline separately for each vehicle. We construct local map element star graphs, aggregate them with GATs and combine them with locally embedded tracks in the Single-Agent MHA blocks. Then, we select the positional embeddings corresponding to each of the joint agents, in each joint agent's local context, obtaining K^2 feature vectors $\{\{z_j^k\}_{j=1}^K\}_{k=1}^K$. An example is provided in Fig. 3a. This combination of features contains mutual local information about each joint prediction candidate, at the cost of quadratically increasing number of features. Nevertheless, we do not observe a computational bottleneck due to efficient batching in training, described in Sec. IV-A.

Given the individual local features, we now construct an implicit global frame by combining features from each local frame. We achieve this with another MHA block (denoted as Joint Multi-Head Attention in Fig. 3a), taking in features $\{\{z_j^k\}_{j=1}^K\}_{k=1}^K$ stacked into a matrix (assuming a single batch element). In the output, we select the rows corresponding to features $\{z_k^k\}_{k=1}^K$ (in their respective reference frames) and feed them in a batched manner into an action decoder block. We train with the same loss (6) as in single-agent StarNet training.

2) Attention-mask: In combining the local contexts into an implicit global context, the embeddings corresponding to a single vehicle in different local frames should attend only to themselves. We achieve this by limiting the self-attention with a $K^2 \times K^2$ mask. It ensures that only features $\{z_j^k\}_{k=1}^K$ for the agent j in different frames are considered, in each row of the stacked input matrix. This is exemplified in Fig. 3b.

The joint-StarNet architecture with masking allows to explicitly combine multiple local interaction models and integrate them into an implicit global interaction model while accounting for non-symmetric attention. Each local, single-agent model uses direct map representations that condition the local social interaction.

	inD [38]		INTERACTION [7]	
	ADE	FDE	ADE	FDE
FFW-ASP [23]	0.37	1.02	0.24	0.63
VectorNet [9]	0.37	1.03	0.22	0.63
StarNet	0.35	0.97	0.16	0.49
joint-StarNet-*no-mask*	0.36	1.02	0.15	0.43
joint-StarNet	**0.32**	**0.89**	**0.13**	**0.38**

TABLE I: Comparison of StarNet and joint-StarNet (also without masking from Sec. III-D.2) with FFW-ASP [23] and VectorNet [9] (own implementation), taking the best out of $m = 3$ modes.

	INTERACTION [7]	
	ADE	FDE
DESIRE [39]	0.32	0.88
MultiPath [15]	0.30	0.99
STG-DAT [34]	0.29	0.54
TNT [16]	0.21	0.67
ReCoG [40]	0.19	0.66
HEAT-I-R [35]	0.19	0.65
ITRA [5]	0.17	0.49
StarNet	0.16	0.49
joint-StarNet	**0.13**	**0.38**

TABLE II: Comparison of StarNet and joint-StarNet with approaches in literature (reported results). The values for [39] and [15] are given in [16]. Since [34] reports results for different map types separately, we computed the aggregate value by combining the ratios of specific map types in the validation dataset.

IV. RESULTS

A. Implementation

For implementing the StarNet (Sec. III-C) and joint-StarNet (Sec. III-D) architectures we used several network types: 1D CNNs, GATs, MHA, and GRUs. The track encoder 1D CNNs are adapted from the ActorNet model in [10] and embed position and action tracks to 128- and 64-dimensional vectors, respectively. The position embeddings are then used as nodes in the GAT and MHA networks, which are both realized with 8 attention heads. In the action decoder block, the concatenated position and action track embeddings are first transformed by two linear layers of sizes $\{512, 256\}$ (with batch normalization and $\tanh$ activation) before being fed into the GRU network. The GRU iterates these transformed features three times through two layers of 512 hidden units, directly predicting m future action modes.

The joint-StarNet has several important practical considerations. Within the model, each joint candidate is first fed through the first two steps of the single-agent StarNet (Sec. III-C.1 and Sec. III-C.2), and then the features from multiple local contexts are aggregated in the Joint MHA, as exemplified in Fig. 3a. In a single batch element, this induces linearly growing complexity in the GAT and Single-agent MHA and quadratically growing complexity in the Joint MHA, with the number of joint candidates. However, the computational load does not grow equally due to efficient batching of different scenes. In the GAT case, we aggregate different star graphs into a single graph with a block-diagonal adjacency matrix. Similarly, in both MHA blocks we feed inputs from different scenes together in a batch, but use additional batch-wise attention mask. As a result, this brings a higher utilization of GPU memory. However, reasonably-sized batches are made possible by the compact input representations.

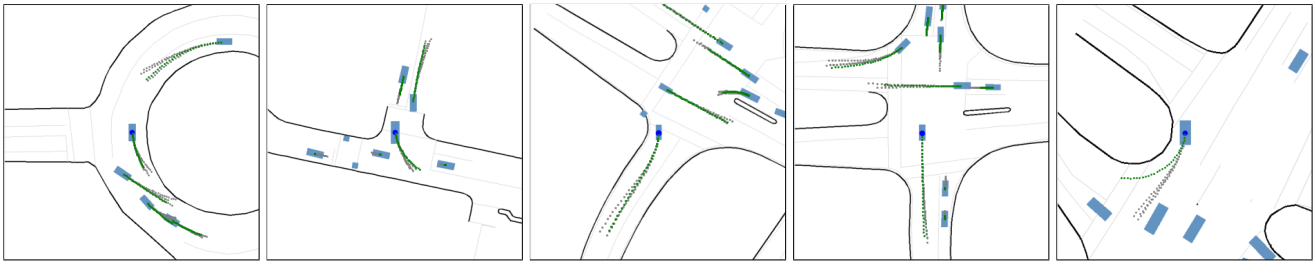


Fig. 4: Qualitative results of joint-StarNet on INTERACTION [7] validation subset. Rectangles indicate vehicles, while squares are pedestrians/bicycles. Gray trajectories are predictions ($m = 3$ modes), with the ground truth overlaid in green. The right-most prediction is among the 50 worst individual predictions according to the FDE metric.

B. Datasets

Datasets must allow flexibility in choosing single or multiple prediction targets, in order to facilitate both single-agent and joint trajectory prediction. Therefore, we selected the inD [38] and the INTERACTION datasets [7] that provide joint tracked data over a whole recording. In both datasets, we generated individual samples for all agents except pedestrians and bicycles by extracting all 2.5+3s segments (2.5s past and 3s prediction recorded with 10Hz) with a 1.5s spacing between the samples. In generating map data, we used the provided lanelets [41] to extract two polyline types: road boundaries (e.g. curbstones) and road properties (e.g. lane center-lines). For inD, we used the same training/validation/testing split as in [23], and for INTERACTION we used the provided validation dataset.

In the joint-StarNet setting, it is non-trivial to determine the joint prediction candidates. We perform this by first selecting one vehicle in the scene that is present over an entire prediction time interval as the virtual-ego. Then, we determine all other vehicles that exist throughout the time interval and that at any time during the interval come close to the virtual-ego within a certain threshold; better approaches to filter relevant agents are left for future work. In this sense, the virtual-ego mimics an AV that predicts trajectories of its nearby vehicles (and its own trajectory). Thus, when another vehicle is chosen to be the virtual-ego, a different set of joint candidates could emerge. This allows us to effectively upsample highly interactive training data.

C. Training setup

We implemented our models in PyTorch [42] and trained with the Adam optimizer [43] over 10 epochs with batch size 8. The learning rate was set to 10^{-4} and multiplied by 0.2 for every two consecutive epochs with no improvement outside an ϵ region. For joint-StarNet, the training took around five days to complete on a single RTX 3090 GPU.

D. Performance

We compared our approaches against the raster-based Feed-Forward Action-Space Prediction (FFW-ASP) architecture from our previous work [23] and our own implementation of VectorNet [9], with the results shown in Tab. I. To ensure a fair comparison, we used our multi-modal action output decoder in VectorNet (originally uni-modal) and did not incorporate the self-supervised map completion task from [9] within the MHA blocks of any architecture. Similarly, we adapted the FFW-ASP to use action track encoders mentioned in Sec. III-B. As metrics, we computed the Average Displacement Error (ADE) and Final Displacement Error (FDE), defined as in [5].

As seen in Tab. I, StarNet's graph-based map modeling and map-conditioned interaction modeling already brings improvements over the baseline methods. Similarly, joint-StarNet improves on StarNet's performance, as expected. Tab. I can also be interpreted as an ablation study, where the effects of removing the graph-based map modeling (by FFW-ASP/StarNet results) and masked joint prediction (by StarNet/joint-StarNet(*-no-mask*) results). We also compared against reported results in literature on the INTERACTION validation dataset, see Tab. II. To the best of our knowledge, joint-StarNet achieves state-of-the-art performance.

Examples of predicted trajectories with joint-StarNet are shown in Fig. 4. It can be seen that overall the predictions accurately model the interaction in a scene. Regarding multi-modality, we observe that in most cases the modes are distributed realistically, i.e. in straight driving they are spread longitudinally and slight turns result in relatively narrow dispersion. However, among the shown worst prediction example, turns with a larger radius result in occasional missed modes, indicating room for improvement.

V. CONCLUSION

In this work, we presented an attention-based approach to directly represent map elements and explicitly model mutual social interaction. We offered two novel architectures, the single-agent StarNet that models map elements as star graphs, and a joint prediction extension via an additional MHA layer. The joint-StarNet can handle a variable number of agents and integrate their local awarenesses into an implicit global model. In this sense, it contributes an important step towards joint scene understanding.

In future work, we will focus on the multi-modality aspect of joint prediction and address the shortcomings mentioned in Sec. IV-D. We plan to improve on the implicit modeling of multi-modality within the action output decoder, which does not condition predicted modes on other vehicles' predicted modes. Furthermore, we plan to integrate the presented architectures with the self-supervised long-term prediction framework of [23], which predicts future context representations prior to trajectories. We expect that the denser map and joint interaction modeling will lead to improved context prediction, bringing further performance improvements.

REFERENCES

[1] S. Casas, C. Gulino, R. Liao, and R. Urtasun, "SPAGNN: Spatially-Aware Graph Neural Networks for Relational Behavior Forecasting from Sensor Data," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020.

[2] S. Casas, C. Gulino, S. Suo, K. Luo, R. Liao, and R. Urtasun, "Implicit Latent Variable Model for Scene-Consistent Motion Forecasting," *arXiv preprint arXiv:2007.12036*, 2020.

[3] A. Cui, A. Sadat, S. Casas, R. Liao, and R. Urtasun, "LookOut: Diverse Multi-Future Prediction and Planning for Self-Driving," *arXiv preprint arXiv:2101.06547*, 2021.

[4] S. Suo, S. Regalado, S. Casas, and R. Urtasun, "TrafficSim: Learning to Simulate Realistic Multi-Agent Behaviors," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

[5] A. Scibior, V. Lioutas, D. Reda, P. Bateni, and F. Wood, "Imagining The Road Ahead: Multi-Agent Trajectory Prediction via Differentiable Simulation," *arXiv preprint arXiv:2104.11212*, 2021.

[6] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention Is All You Need," in *Advances in Neural Information Processing Systems*, 2017.

[7] W. Zhan, L. Sun, D. Wang, H. Shi, A. Clausse, M. Naumann, J. Kummerle, H. Konigshof, C. Stiller, A. de La Fortelle *et al.*, "INTERACTION Dataset: An INTERnational, Adversarial and Cooperative MoTION Dataset in Interactive Driving Scenarios with Semantic Maps," *arXiv preprint arXiv:1910.03088*, 2019.

[8] A. Rudenko, L. Palmieri, M. Herman, K. M. Kitani, D. M. Gavrila, and K. O. Arras, "Human Motion Trajectory Prediction: A Survey," *The International Journal of Robotics Research*, vol. 39, no. 8, 2020.

[9] J. Gao, C. Sun, H. Zhao, Y. Shen, D. Anguelov, C. Li, and C. Schmid, "VectorNet: Encoding HD Maps and Agent Dynamics from Vectorized Representation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[10] M. Liang, B. Yang, R. Hu, Y. Chen, R. Liao, S. Feng, and R. Urtasun, "Learning Lane Graph Representations for Motion Forecasting," in *European Conference on Computer Vision*. Springer, 2020.

[11] Y. Hu, W. Zhan, and M. Tomizuka, "Scenario-Transferable Semantic Graph Reasoning for Interaction-Aware Probabilistic Prediction," *arXiv preprint arXiv:2004.03053*, 2020.

[12] C. Tang and R. R. Salakhutdinov, "Multiple Futures Prediction," in *Advances in Neural Information Processing Systems*, 2019.

[13] J. Li, F. Yang, H. Ma, S. Malla, M. Tomizuka, and C. Choi, "RAIN: Reinforced Hybrid Attention Inference Network for Motion Forecasting," *arXiv preprint arXiv:2108.01316*, 2021.

[14] E. Tolstaya, R. Mahjourian, C. Downey, B. Vadarajan, B. Sapp, and D. Anguelov, "Identifying Driver Interactions via Conditional Behavior Prediction," *arXiv preprint arXiv:2104.09959*, 2021.

[15] Y. Chai, B. Sapp, M. Bansal, and D. Anguelov, "MultiPath: Multiple Probabilistic Anchor Trajectory Hypotheses for Behavior Prediction," *arXiv preprint arXiv:1910.05449*, 2019.

[16] H. Zhao, J. Gao, T. Lan, C. Sun, B. Sapp, B. Varadarajan, Y. Shen, Y. Shen, Y. Chai, C. Schmid *et al.*, "TNT: Target-driveN Trajectory Prediction," *arXiv preprint arXiv:2008.08294*, 2020.

[17] Y. Liu, J. Zhang, L. Fang, Q. Jiang, and B. Zhou, "Multimodal Motion Prediction with Stacked Transformers," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

[18] N. Rhinehart, R. McAllister, K. Kitani, and S. Levine, "PRECOG: Prediction Conditioned on Goals in Visual Multi-Agent Settings," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019.

[19] S. Khandelwal, W. Qi, J. Singh, A. Hartnett, and D. Ramanan, "What-If Motion Prediction for Autonomous Driving," *arXiv preprint arXiv:2008.10587*, 2020.

[20] J. Ngiam, B. Caine, V. Vasudevan, Z. Zhang, H.-T. L. Chiang, J. Ling, R. Roelofs, A. Bewley, C. Liu, A. Venugopal *et al.*, "Scene Transformer: A Unified Multi-Task Model for Behavior Prediction and Planning," *arXiv preprint arXiv:2106.08417*, 2021.

[22] T. Phan-Minh, E. C. Grigore, F. A. Boulton, O. Beijbom, and E. M. Wolff, "CoverNet: Multimodal Behavior Prediction Using Trajectory Sets," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[21] H. Cui, T. Nguyen, F.-C. Chou, T.-H. Lin, J. Schneider, D. Bradley, and N. Djuric, "Deep Kinematic Models for Physically Realistic Prediction of Vehicle Trajectories," *arXiv preprint arXiv:1908.00219*, 2019.

[23] F. Janjoš, M. Dolgov, and M. J. Zöllner, "Self-Supervised Action-Space Prediction for Automated Driving," in *2021 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2021.

[24] W. Zeng, W. Luo, S. Suo, A. Sadat, B. Yang, S. Casas, and R. Urtasun, "End-to-End Interpretable Neural Motion Planner," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

[25] W. Zeng, S. Wang, R. Liao, Y. Chen, B. Yang, and R. Urtasun, "DSDNet: Deep Structured Self-Driving Network," in *European Conference on Computer Vision*. Springer, 2020.

[26] Y. LeCun, Y. Bengio, and G. Hinton, "Deep Learning," *Nature*, vol. 521, no. 7553, 2015.

[27] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A Comprehensive Survey on Graph Neural Networks," *IEEE transactions on neural networks and learning systems*, vol. 32, no. 1, 2020.

[28] W. Zeng, M. Liang, R. Liao, and R. Urtasun, "LaneRCNN: Distributed Representations for Graph-Centric Motion Forecasting," *arXiv preprint arXiv:2101.06653*, 2021.

[29] J. Pan, H. Sun, K. Xu, Y. Jiang, X. Xiao, J. Hu, and J. Miao, "Lane-Attention: Predicting Vehicles' Moving Trajectories by Learning Their Attention Over Lanes," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020.

[30] U. Alon and E. Yahav, "On the Bottleneck of Graph Neural Networks and its Practical Implications," *arXiv preprint arXiv:2006.05205*, 2020.

[31] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph Attention Networks," *arXiv preprint arXiv:1710.10903*, 2017.

[32] W. L. Hamilton, "Graph Representation Learning," *Synthesis Lectures on Artifical Intelligence and Machine Learning*, vol. 14, no. 3, 2020.

[33] J. Mercat, T. Gilles, N. El Zoghby, G. Sandou, D. Beauvois, and G. P. Gil, "Multi-Head Attention for Multi-Modal Joint Vehicle Motion Forecasting," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020.

[34] J. Li, H. Ma, Z. Zhang, J. Li, and M. Tomizuka, "Spatio-Temporal Graph Dual-Attention Network for Multi-Agent Prediction and Tracking," *arXiv preprint arXiv:2102.09117*, 2021.

[35] X. Mo, Y. Xing, and C. Lv, "Heterogeneous Edge-Enhanced Graph Attention Network For Multi-Agent Trajectory Prediction," *arXiv preprint arXiv:2106.07161*, 2021.

[36] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural computation*, vol. 9, no. 8, 1997.

[37] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning Phrase Representations Using RNN Encoder-Decoder for Statistical Machine Translation," *arXiv preprint arXiv:1406.1078*, 2014.

[38] J. Bock, R. Krajewski, T. Moers, S. Runde, L. Vater, and L. Eckstein, "The inD Dataset: A Drone Dataset of Naturalistic Road User Trajectories at German Intersections," *arXiv preprint arXiv:1911.07602*, 2019.

[39] N. Lee, W. Choi, P. Vernaza, C. B. Choy, P. H. Torr, and M. Chandraker, "DESIRE: Distant Future Prediction in Dynamic Scenes with Interacting Agents," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[40] X. Mo, Y. Xing, and C. Lv, "ReCoG: A Deep Learning Framework with Heterogeneous Graph for Interaction-Aware Trajectory Prediction," *arXiv preprint arXiv:2012.05032*, 2020.

[41] F. Poggenhans, J.-H. Pauls, J. Janosovits, S. Orf, M. Naumann, F. Kuhnt, and M. Mayr, "Lanelet2: A High-Definition Map Framework for the Future of Automated Driving," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018.

[42] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," *arXiv preprint arXiv:1912.01703*, 2019.

[43] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv preprint arXiv:1412.6980*, 2014.

Paper IV

- Title: *SAN: Scene Anchor Networks for Joint Action-Space Prediction*
- Authors: Faris Janjoš and Maxim Dolgov and Muhamed Kurić and Yinzhe Shen and J. Marius Zöllner
- Venue: 2022 IEEE Intelligent Vehicles Symposium (IV) workshop: From Benchmarking Behavior Prediction to Socially Compatible Behavior Generation in Autonomous Driving

2022 IEEE Intelligent Vehicles Symposium (IV)
June 5-9, 2022. Aachen, Germany

SAN: Scene Anchor Networks for Joint Action-Space Prediction

Faris Janjoš[1], Maxim Dolgov[1], Muhamed Kurić[2], Yinzhe Shen[1], and J. Marius Zöllner[3]

***Abstract*— In this work, we present a novel multi-modal trajectory prediction architecture. We decompose the uncertainty of future trajectories along higher-level scene characteristics and lower-level motion characteristics, and model multi-modality along both dimensions separately. The scene uncertainty is captured in a joint manner, where diversity of *scene modes* is ensured by training multiple separate anchor networks which specialize to different scene realizations. At the same time, each network outputs multiple trajectories that cover smaller deviations given a scene mode, thus capturing *motion modes*. In addition, we train our architectures with an outlier-robust regression loss function, which offers a trade-off between the outlier-sensitive L_2 and outlier-insensitive L_1 losses. Our scene anchor model achieves improvements over the state of the art on the INTERACTION dataset, outperforming the StarNet architecture from our previous work.**

I. INTRODUCTION

Autonomous vehicles rely on accurate predictions of other agents' future motions in order to plan safe maneuvers. Many different sub-problems have to be addressed while generating future trajectory predictions, such as building an environment representation, modeling interaction between entities on the road, as well as capturing the uncertainty of future motion. For vehicle trajectory prediction, inferring different potential future realizations of a single observed scene is especially important. In tackling this problem, learned methods have shown great promise as they are able to aggregate this heterogeneity from datasets of naturalistic driving.

The uncertainty of future motions can be captured by learning a distribution over the trajectories of each agent in a driving scene, given training examples. If well-trained, this distribution would model the *aleatoric* uncertainty, a statistical uncertainty which represents the variability of different instances in the training data. For example, a model trained on intersection data would capture the likelihoods of making each turn. In contrast, *epistemic* uncertainty is systemic and would quantify unseen instances; in the given example, the model would most likely fail when presented with roundabout data at inference. Hence, it is important to train the model on data from multiple contexts.

[1] Robert Bosch GmbH, Corporate Research, Advanced Autonomous Systems, 71272 Renningen, Germany. {first-name.last-name}@de.bosch.com

[2] Virtual Vehicle Research GmbH, 8010 Graz, Austria. muhamed.kuric@v2c2.at

[3] Research Center for Information Technology (FZI), 76131 Karlsruhe, Germany. zoellner@fzi.de

This work was financially supported by the Federal Ministry of Economic Affairs and Energy of Germany, grant number 19A20026H, based on a decision of the German Bundestag. Virtual Vehicle Research GmbH has received funding within COMET Competence Centers for Excellent Technologies from the Austrian Federal Ministry for Climate Action, the Austrian Federal Ministry for Digital and Economic Affairs, the Province of Styria (Dept. 12) and the Styrian Business Promotion Agency (SFG). The Austrian Research Promotion Agency (FFG) has been authorised for the programme management.

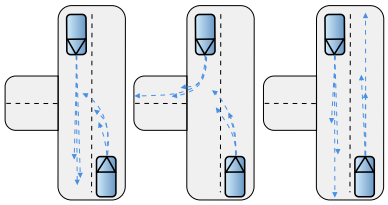


Fig. 1: Depicted are three joint predictions of an interactive driving scene. In each scene-level prediction, a consistent resolution is reached w.r.t. the intent of each agent. However, different motions can be performed to realize a given intent. This motivates the distinction between *scene modes* and *motion modes*, where motion modes are conditioned on scene modes.

An essential attribute of the distribution of future motions for each vehicle is its multi-modality, where each mode clusters similar potential future trajectories. In modeling this distribution, a major challenge is ensuring consistency between individual modes of different interacting agents. Methods that perform single-vehicle prediction and marginalize the distribution over (future) scene outcomes usually cannot ensure this consistency. Thus, it is important to consider multi-modality in a joint manner while performing scene-level predictions. In this sense, a scene mode would consider multiple vehicles at once and gather their individual modes that together characterize a consistent forecast of the entire scene.

Within a single scene mode, there still could be variation among the motions of each agent, especially at high velocities. For example, two vehicles negotiating an intersection in an interactive manner (depicted in Fig. 1) could plan considerably different trajectories that achieve the same goal. Therefore, a learned prediction method should be flexible enough to characterize this motion uncertainty as well.

In our work, we utilize scene anchor networks to tackle both scene-level and motion-level multi-modality. The approach is built on top of our joint, graph-based action-space prediction method [1], which ensures kinematically feasible predictions. The contributions of our work are the following:

- A dual representation of future motion uncertainty and multi-modality via *scene modes* and *motion modes*. In learning this duality, we employ Scene Anchor Networks (SAN), which specialize to different scene-wise realizations as well as output multiple motions given a scene mode.
- Advocating for the usage of a novel robust loss function [2] for regressing predicted trajectories, as an alternative to standard L_1 and L_2 losses.
- State-of-the-art performance on the challenging INTERACTION validation dataset [3], containing roundabout, intersection, and highway merge scenarios.

2022 IEEE Intelligent Vehicles Symposium (IV) | 978-1-6654-8821-1/22/$31.00 ©2022 IEEE | DOI: 10.1109/IV51971.2022.9827239

II. RELATED WORK

The uncertainty of future motions can be decomposed into its aleatoric and epistemic components. Works considering epistemic uncertainty [4], [5] are relatively rare and usually frame it within the context of ensemble networks [6]. The vast majority of approaches emphasizing multi-modality of the distribution over (future) scene outcomes analyze it in the context of the aleatoric uncertainty within the training data. Likewise, our work considers this type of uncertainty. Among this class of approaches, it is important to distinguish between single-agent and joint prediction methods.

Single-agent methods usually cannot guarantee consistency between multiple predicted modes for different agents, since they marginalize the joint distribution and predict individually for each agent of interest [1]. Nevertheless, the higher-level intent and lower-level motion uncertainty for a single agent can be captured in different ways. In describing motion-level uncertainty, a common choice is directly predicting the parameters of a Gaussian distribution [7], [4], thus modeling trajectories as a set of independent Gaussians around each point. However, this imposes an unnecessary symmetry on the shape of the uncertainty around each point. When it comes to modeling intent-level uncertainty, many works define intent w.r.t. desired locations on the road and thus opt for appropriate heuristics that aid in inferring meaningful modes. For example, anchor trajectories can be used, either in position space [7], [8], [9], or latent space [4]. Similarly, the road structure is used to find candidate regions or destination targets [10], [11], [12], [13], [14], [15]. Interestingly, [15] predicts a heatmap that in addition to intent-level uncertainty also covers motion uncertainty in its non-parametric output representation. Nevertheless, in order to incorporate mode heuristics, many heuristic-based methods require accurate map information such as drivable lane space or connectivity in order to define map-level goals. Our joint-prediction approach relies only on polyline descriptions without defining any explicit intent-level goal heuristics.

Among joint prediction approaches, sampling-based methods utilizing generative models are common [16], [17], [18], [19], [20], [21], [22], [23], [24]. Although powerful in modeling multi-modal distributions, models such as Variational Autoencoders (VAE) cannot guarantee diversity [22], [23], and thus rely on additional diversity-inducing sampling functions [22], [23] or auxiliary losses [19]. Furthermore, sampling introduces randomness in outputs, which is undesirable in autonomous driving applications. A notable exception is the single-sample model in [19] using auxiliary planning losses. We offer a deterministic model using standard regression and classification loss terms (described in Sec. III-D).

A special consideration among multi-modal approaches is mode collapse. Approaches that are especially susceptible to mode collapse are ones that predict multiple trajectories with the same output network [25], [26], [1]. Usually, such approaches employ a winner-takes-all loss which pushes the predicted modes to average out – in effect, the model trades off performance on rarer, harder-to-solve cases to get incremental improvements on more common, easier-to-solve cases. To avoid this issue, the heuristically inclined approaches [10], [11], [15] combat mode collapse by conditioning predicted trajectories on environment elements such as target regions or lane sections. As an alternative, [27] solves the problem in the loss space by ensuring fair coverage. When it comes to generative models, they are also susceptible to mode collapse as well as the problem of yielding out-of-distribution outputs "in-between" the modes [28]. To tackle these problems, [28] trains multiple generators in their pedestrian prediction model, one for each mode.

In our approach, we combat mode collapse by training separate anchor networks which specialize to different scene-level realizations via *scene-modes*. We reach similar insights as [28], but for the deterministic case and in joint vehicle trajectory prediction. In addition to scene uncertainty, our approach also models motion uncertainty by having each anchor network output multiple motion modes. Furthermore, a novel loss function [2] helps in robustifying the training to outliers. Our approach is a multi-modality focused extension of our previous approach [26] that performs joint prediction, does not use any explicit map-level goal heuristics, and ensures kinematically feasible outputs. Together, these factors contribute to setting new state-of-the-art performance on the INTERACTION dataset [3].

III. METHOD

We present our method as follows. First, the trajectory prediction problem is described in Sec. III-A. Then, we introduce the joint-StarNet model [1] in Sec. III-B, since our method is its extension. The sections III-C and III-D describe the SAN architecture and the training procedure.

A. Problem definition

We frame the task of joint vehicle trajectory prediction in an imitation learning setting. For a driving scene with N agents (vehicles, pedestrians, cyclists), we jointly predict the trajectories of K ($K \leq N$) vehicles of interest. Thus, we learn a model that parameterizes the joint distribution of predicted future trajectories

$$\hat{Y} \sim P(\hat{\mathbf{Y}}|\mathcal{D}) \ . \tag{1}$$

We predict a sample $\hat{Y} = \{\hat{Y}_k\}_{k=1}^{K}$ of future trajectories for K vehicles, where $\hat{Y}$ consists of xy positions. The context information $\mathcal{D}$ contains the map and track history; the map consists of road boundaries and lane markings represented with polylines (sets of vectors), whereas tracks are past xy positions for each of the N agents.

In parameterizing the distribution (1), we use a standard deterministic encoder-decoder structure. Here, the encoder jointly aggregates context information into a scene-relevant feature vector, while the decoder generates future trajectories for all vehicles of interest based on the scene features.

B. joint-StarNet

As a basis for our method, we use the joint-StarNet architecture from [1] and the action-space prediction paradigm from [26]. The joint-StarNet models the distribution (1) w.r.t. the same context data $\mathcal{D}$. Here, we will shortly describe its encoder and decoder components, with a detailed example

depicted in Fig. 2. For more information, we invite the reader to visit [1] and [26].

In the encoder component of joint-StarNet (Fig. 2c), polyline-vectors and track history embeddings are used as nodes for star graphs. The graphs model the attention of each vehicle to individual polyline elements and are aggregated via Graph Attention Networks (GAT) [29]. Then, a Multi-Head Attention (MHA) [30] block models the map-dependent local interaction in the local reference frame of each vehicle. Finally, another MHA block models the global interaction by combining the information from local reference frames. This enables the model to capture non-symmetric attention between agents (by attending in local frames), as well as to solve the problem of choosing a reference frame in joint prediction (by aggregating local information). The output of the model is a set of feature vectors z_i for each joint prediction candidate vehicle $i \in K$.

Another characteristic of joint-StarNet is the action-space prediction in the decoder depicted in Fig. 2d. By using each vehicle's action history to predict future actions (consisting of accelerations and steering angles), it is possible to completely decouple kinematic motion modeling from the learned prediction task [26]. The predicted future actions are feasible since they are mapped to a reasonable range. Furthermore, kinematically feasible predictions are ensured since the predicted actions are propagated through a differentiable kinematic model.

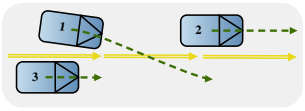
The joint-StarNet model can be considered multi-modal, since the action decoder outputs multiple action trajectory modes. However, its multi-modality modeling abilities are limited since it is trained with a simple winner-takes-all loss that induces mode collapse (elaborated in Sec II). Herein lies the major drawback of joint-StarNet; we attempt to address it with the proposed Scene Anchor Networks.

C. Scene Anchor Networks

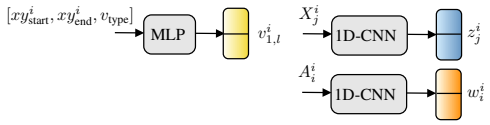
On a high level, the SAN model serves as an alternative decoder within the joint-StarNet architecture. As an input, it uses the encoder outputs z_i (depicted in bottom left of Fig. 2c) for each joint prediction candidate vehicle, and it outputs future actions. However, its structure enables it to handle multi-modality in a more principled manner.

The SAN model is depicted in Fig. 3. It is realized as a two-stage architecture, where in the first stage, scene-specific information is computed using the feature vectors z_i. The features z_i are generated by considering upstream map information, local interaction, and global interaction of the entire scene, and thus can be used to describe the driving scene. To generate a scene description, we join the vectors into an undirected graph and aggregate it via a GAT to obtain a scene feature vector z.

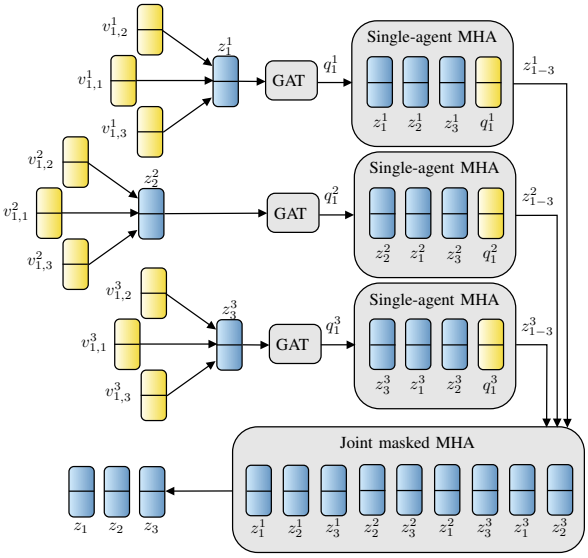
In the second stage, output trajectories are generated. Here, an anchor-network is associated with each scene mode. The S anchor networks take in the same scene feature vector z and action history embeddings w_i for each joint prediction vehicle $i \in K$, and each predict a single future realization of the driving scene for the K vehicles. In addition, the anchors



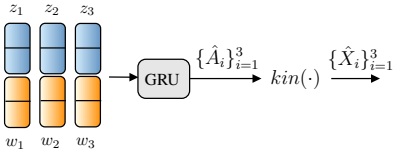
(a) A driving scene with three interacting vehicles and a single polyline consisting of three vectors.



(b) Embedding the elements from the driving scene into feature vectors. Each of the three vectors is embedded by a single linear layer encoding its start and end positions (in the frame of vehicle $i \in [1, 2, 3]$), and one-hot encoded polyline type. Each vehicle's track history (for vehicle j, in its own and any other vehicle i's frame $i, j \in [1, 2, 3]$), as well as action history (for vehicle i in its own frame) is processed via a 1D-Convolutional Neural Network (CNN).



(c) Joint-StarNet encoder: First, polyline-level graphs are constructed for the polyline 1 consisting of vectors $\{v^i_{1,l}\}^3_{l=1}$ (in the reference frames of vehicles $i \in [1, 2, 3]$). The vector nodes are connected to the vehicle track embeddings $\{z^i_i\}^3_{i=1}$ (in their own frames) and the graphs are processed with a GAT to obtain polyline-level embeddings for each vehicle q^i_1. Then, the polyline embeddings are processed together with the track embeddings $z^i_{1,2,3}$ (in the frame of each vehicle i) in three separate single-agent MHA blocks. Finally, the joint masked MHA block aggregates the track embeddings for the three vehicles from different frames to obtain final feature vectors z_i for each vehicle i.



(d) Joint-StarNet decoder: First, the feature vectors z_i for each vehicle i are concatenated with the respective action embeddings w_i. Then, future actions $\hat{A}_i$ are predicted and converted to future positions $\hat{X}_i$ via a kinematic model.

Fig. 2: Depicted are a driving scene, the necessary embeddings, and the joint-StarNet encoder and decoder models, for the task of jointly predicting the trajectories of three vehicles ($K = 3$).

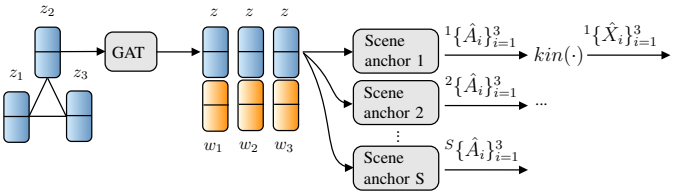


Fig. 3: SAN architecture: an addition to the joint-StarNet encoder from Fig. 2c, and an alternative decoder to the one in Fig. 2d. The S anchor networks receive same inputs, the concatenated scene-level features z and action history embeddings of the $K = 3$ joint prediction candidate vehicles. Each $\hat{A}_i$ output from each anchor network contains M motion modes for the vehicle i.

are flexible enough to model motion-level uncertainty within the scene mode. Each anchor can be outfitted to predict M *motion modes* for each vehicle, as well as the softmax scores of each motion mode (similar to the multi-modality concept in [1]). This method is potentially more versatile than predicting the parameters of a Gaussian distribution to describe motion-level uncertainty. It does not impose an artificial symmetry onto the shape of motion uncertainty but rather represents it by a set of trajectories.

Scene anchors have the potential to combat mode collapse by training separate networks to specialize to different scene-level realizations. This can be illustrated by the problem of learning a Gaussian mixture on data with well-separated clusters. If each cluster is trained with a separate network, the separability is preserved as each network can place the mode of the cluster independently of the others. In contrast, one network would struggle in ensuring separable modes since it would be forced to update its weights to improve performance on all examples, resulting in some form of averaging. Thus, while training SAN, we need a mechanism to train a single anchor at a time. We achieve this by updating only the anchor with the lowest average regression loss across the entire scene (definition in Eq. (2) in Sec. III-D). In our experiments in Sec. IV-C, we show that this anchor-wise winner-takes-all loss outperforms a single network model.

D. Training with the robust regression loss

We train the entire pipeline with the following loss

$$\mathcal{L} = \min_m \sum_{s=1}^{S} I_{s=s^\dagger} (\mathcal{L}_{\text{reg}}^{s,m} + \beta \mathcal{L}_{\text{class}}^{s,m}) \ , \qquad (2)$$

where I is a binary indicator function selecting the anchor (denoted by $\dagger$) whose motion modes are on average closest to the ground-truth for all K predicted agents. The motion mode score $\mathcal{L}_{\text{class}}$ is a cross-entropy term with $\beta = 1$. For the regression loss $\mathcal{L}_{\text{reg}}$, we use a novel general loss function [2]. We motivate the usage with robustness to outliers; we posit that outliers can appear in the context of trajectory prediction, especially when vehicles perform unorthodox motions.

In state-of-the-art prediction methods, it is common to use L_2, L_1, or smoothed versions of the L_1 loss such as Huber [31] or pseudo-Huber [32] for penalizing regressed trajectories that deviate from the ground-truth [18], [19]. These loss functions behave differently w.r.t. outliers — the gradient of the function at a distant data point can vary greatly. The Huber loss for example saturates the gradient

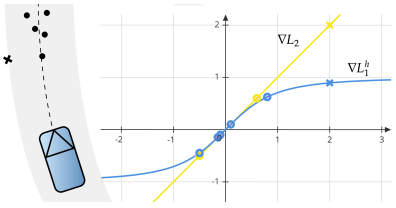


Fig. 4: Depicted on the left side are a number of ground-truth future positions, one of which is an outlier (cross). In training, this example will incur different gradient values compared to the inliers, depending on the loss function. Depicted on the right side, the gradient of the pseudo-Huber L_1^h loss at the outlier will be similar to an inlier's, while the L_2 gradient will be drastically larger.

the farther the outlier is, while the L_2 linearly increases the gradient value and thus the influence of the outlier on the network weights. This is illustrated in Fig. 4.

The aforementioned loss functions, as well as others such as Cauchy [33] and Welsh loss [34], are members of a family of functions that can be described by a single general loss [2]

$$\rho(x, \alpha) = \frac{|2-\alpha|}{\alpha} \left(\left(\frac{x^2}{|2-\alpha|} + 1 \right)^{\alpha/2} - 1 \right) . \qquad (3)$$

In this loss, it is possible to reconstruct individual losses by setting the parameter $\alpha \in \mathbb{R}$ and thus tune the sensitivity to outliers[1]. However, [2] goes one step further – it offers a mechanism to learn the parameter α in training and determine the specific loss (which might not be any of the standard loss functions) that fits the nature of the data.

When outliers appear in trajectory prediction, it is unclear whether they should be harshly penalized (as the L_2 loss does), or relatively ignored in favor of other samples (as the pseudo-Huber loss does). Therefore, by using the learned adaptive loss (3), one can leave this decision to the optimizer. Motivated by this reasoning, we use the robust loss function from [2] in training our SAN model. The adaptive loss enables the model to independently determine its sensitivity to outliers in training, and thus increase its robustness in cases that are harder to predict. Therefore, we advocate for its wider usage in trajectory prediction algorithms.

IV. RESULTS

A. Implementation

The SAN model is built on top of the joint-StarNet encoder (depicted in Fig. 2c), consisting of multiple networks types such as 1D-CNNs, GATs, and MHA layers. Compared to [1], we extend the MHA layers with an MLP layer and a residual connection, same as in [35]. In the SAN decoder (depicted in Fig. 3), we process the graph of 128-dim feature vector nodes via a 2-layer GAT and aggregate the features via max-pooling. The scene anchor networks are implemented as Gated Recurrent Units (GRU) and are identical to the decoder from Fig. 2d. If not stated otherwise, the number of scene modes S and motion modes M is set to three in our experiments. We found that it achieves a good trade-off between diversity and performance. All models are implemented in PyTorch [36].

[1]For example, L_2 can be obtained with $\alpha = 2$ in the limit, and pseudo-Huber loss with $\alpha = 1$.

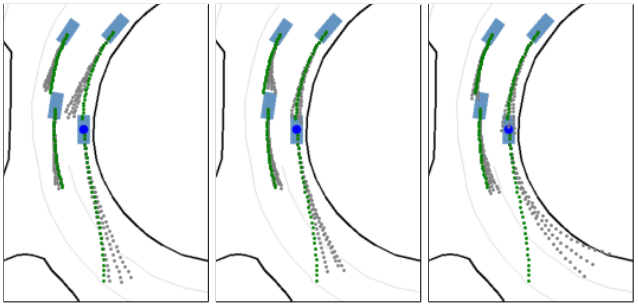


Fig. 5: Depicted are three possible scene realizations coming from three SANs. Gray trajectories are the motion modes, the ground truth is in green. The anchors predict cutting-in and lane change where reasonable. Importantly, they are realistic and do not yield spurious modes.

	inD [37]		INTERACTION [3]	
	ADE	FDE	ADE	FDE
FFW-ASP [26]	0.35	0.96	0.18	0.51
StarNet [1]	0.32	0.88	0.15	0.45
joint-StarNet [1]	0.29	0.82	0.12	0.35
SAN + adaptive loss	**0.24**	**0.64**	**0.10**	**0.29**

TABLE I: Comparison of the adaptive loss SAN model with several baselines. The results for the baseline methods are better here than reported in [1], where they are run for fewer epochs.

	INTERACTION [3]	
	ADE	FDE
CBP-CGM [24]	0.47	0.89
DESIRE [38]	0.32	0.88
MultiPath [7]	0.30	0.99
STG-DAT [39]	0.29	0.54
TNT [10]	0.21	0.67
ReCoG [40]	0.19	0.66
HEAT-I-R [41]	0.19	0.65
ITRA [21]	0.17	0.49
StarNet [1]	0.15	0.45
joint-StarNet [1]	0.12	0.35
SAN + adaptive loss	**0.10**	**0.29**

TABLE II: Comparison of the adaptive loss SAN model with approaches from the literature (reported results). The values for [38] and [7] are taken from [10]. The approach in [39] reports results by map type; we computed the overall value by accounting for the ratios of each type in the validation dataset. Another approach that reports INTERACTION results is [42]; we do not compare it with the others since it reports results only for a subset of five scenarios.

	inD [3]	
	ADE	FDE
SAN Huber loss	0.26	0.69
SAN L_2 loss	0.26	0.68
SAN adaptive loss	**0.24**	**0.64**

TABLE III: Ablation study of the adaptive loss function effects, compared to commonly used Huber and L_2 losses.

B. Datasets and training setup

We trained our models on the INTERACTION [3] and inD [37] datasets of interactive urban driving. The former provides a large set of challenging intersection, roundabout, and highway merge scenarios. Importantly, both provide tracked data of an entire recording, making it straightforward to design a joint prediction experiment. To prepare the data, we followed the same approach as in [1], extracting all 2.5+3s history and future vehicle trajectory segments as the ground-truth (with 10Hz sampling and 1.5s spacing between samples). We trained with the Adam optimizer [43] for 20 epochs with batch size eight and reduced the learning rate by a factor of five for every two epochs without improvement.

	inD [37]	
	ADE	FDE
FFW-ASP: 1 anchor $\times$ 3 motion modes	0.35	0.96
FFW-ASP: 3 anchors $\times$ 1 motion mode	0.29	0.80
FFW-ASP: 3 anchors $\times$ 3 motion modes	0.28	0.73

TABLE IV: Effects of adding anchor decoders on a raster-based single-agent prediction method from [26]: comparison of vanilla FFW-ASP (1-anchor / 3-motion-modes), 3-anchor / 1-motion-mode variant (same number of predictions as vanilla), and 3-anchor / 3-motion-mode model.

C. Performance

As baselines we used the joint-StarNet as well as the Feed-Forward Action-Space Prediction (FFW-ASP) [26] and StarNet [1] models. FFW-ASP is a raster-based action-space model, while StarNet is the single-agent prediction version of joint-StarNet. As metrics, we used Average Displacement Error (ADE) and Final Displacement Error (FDE) (defined as in [21]). We compared all approaches on a manually chosen inD test set (approx. 10% share) and the validation set provided with the INTERACTION dataset. In inference, we activate all three anchors and pick the best scene/motion mode combination. As seen in Tab. I, the SAN model with adaptive loss brings a significant 20% performance improvement over the closest baseline, especially in the FDE metric. We also compared with reported results of other approaches in literature, see Tab II. To the best of our knowledge, SAN achieves state-of-the-art performance on the INTERACTION validation dataset. In Fig. 5, we see example joint predictions for an interactive roundabout scenario from the dataset.

In tables III and IV, we can obtain further insights into the model's performance. In the adaptive loss ablation study in Tab. III, we see that the L_2 and Huber loss achieve worse performance than the adaptive loss, indicating that a better specialization of the loss (3) than the two losses exists. In the experiment in question, the α parameter in (3) converged to approximately 1.5. In Tab. IV, we investigated the effects of adding multiple anchor decoders to a raster-based single-agent prediction method that does not consider the scene jointly. Even for such a model, the hypothesized benefits of multiple anchors are confirmed, evidenced by a large performance improvement. Furthermore, Tab. IV also compares a multi-anchor / single-motion-mode setup with the same number of predictions as a single-anchor model, confirming that simply decoupling the predictions across multiple anchors brings a performance improvement. We also experimented with a simple classifier network to select an anchor, however, the anchor-wise winner-takes-all loss showed superior performance.

V. CONCLUSION

In this work, we presented scene anchor networks, an easy-to-use and effective method to improve multi-modality modeling in trajectory prediction models. We observed great benefits in training multiple decoders that allow the model to specialize to different scene-level realizations of the scene at hand, while also accounting for multiple motions within the scene. Further work may include strategies on evaluating the likelihood of individual scene modes, which could be used to identify "meaningful" futures of a scene.

REFERENCES

[1] F. Janjoš, M. Dolgov, and M. J. Zöllner, "StarNet: Joint Action-Space Prediction with Star Graphs and Implicit Global-Frame Self-Attention," *arXiv preprint arXiv:2111.13566*, 2021.

[2] J. T. Barron, "A general and adaptive robust loss function," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

[3] W. Zhan, L. Sun, D. Wang, H. Shi, A. Clausse, M. Naumann, J. Kummerle, H. Konigshof, C. Stiller, A. de La Fortelle *et al.*, "INTERACTION Dataset: An INTERnational, Adversarial and Cooperative MoTION Dataset in Interactive Driving Scenarios with Semantic Maps," *arXiv preprint arXiv:1910.03088*, 2019.

[4] B. Varadarajan, A. Hefny, A. Srivastava, K. S. Refaat, N. Nayakanti, A. Cornman, K. Chen, B. Douillard, C. P. Lam, D. Anguelov *et al.*, "Multipath++: Efficient information fusion and trajectory aggregation for behavior prediction," *arXiv preprint arXiv:2111.14973*, 2021.

[5] A. Filos, P. Tigkas, R. McAllister, N. Rhinehart, S. Levine, and Y. Gal, "Can autonomous vehicles identify, recover from, and adapt to distribution shifts?" in *Int. Conf. on Machine Learning (ICML)*, 2020.

[6] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," *Advances in neural information processing systems*, vol. 30, 2017.

[7] Y. Chai, B. Sapp, M. Bansal, and D. Anguelov, "MultiPath: Multiple Probabilistic Anchor Trajectory Hypotheses for Behavior Prediction," *arXiv preprint arXiv:1910.05449*, 2019.

[8] T. Phan-Minh, E. C. Grigore, F. A. Boulton, O. Beijbom, and E. M. Wolff, "CoverNet: Multimodal Behavior Prediction Using Trajectory Sets," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[9] H. Berkemeyer, R. Franceschini, T. Tran, L. Che, and G. Pipa, "Feasible and adaptive multimodal trajectory prediction with semantic maneuver fusion," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021.

[10] H. Zhao, J. Gao, T. Lan, C. Sun, B. Sapp, B. Varadarajan, Y. Shen, Y. Shen, Y. Chai, C. Schmid *et al.*, "TNT: Target-driveN Trajectory Prediction," *arXiv preprint arXiv:2008.08294*, 2020.

[11] J. Gu, C. Sun, and H. Zhao, "Densetnt: End-to-end trajectory prediction from dense goal sets," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

[12] S. Khandelwal, W. Qi, J. Singh, A. Hartnett, and D. Ramanan, "What-If Motion Prediction for Autonomous Driving," *arXiv preprint arXiv:2008.10587*, 2020.

[13] H. Song, D. Luan, W. Ding, M. Y. Wang, and Q. Chen, "Learning to predict vehicle trajectories with model-based planning," in *Conference on Robot Learning*. PMLR, 2022.

[14] M. Liang, B. Yang, R. Hu, Y. Chen, R. Liao, S. Feng, and R. Urtasun, "Learning Lane Graph Representations for Motion Forecasting," in *European Conference on Computer Vision*. Springer, 2020.

[15] T. Gilles, S. Sabatini, D. Tsishkou, B. Stanciulescu, and F. Moutarde, "Gohome: Graph-oriented heatmap output for future motion estimation," *arXiv preprint arXiv:2109.01827*, 2021.

[16] C. Tang and R. R. Salakhutdinov, "Multiple Futures Prediction," in *Advances in Neural Information Processing Systems*, 2019.

[17] N. Rhinehart, R. McAllister, K. Kitani, and S. Levine, "PRECOG: Prediction Conditioned on Goals in Visual Multi-Agent Settings," in *Proceedings of the IEEE Int. Conf. on Computer Vision*, 2019.

[19] A. Cui, A. Sadat, S. Casas, R. Liao, and R. Urtasun, "LookOut: Diverse Multi-Future Prediction and Planning for Self-Driving," *arXiv preprint arXiv:2101.06547*, 2021.

[18] S. Casas, C. Gulino, S. Suo, K. Luo, R. Liao, and R. Urtasun, "Implicit Latent Variable Model for Scene-Consistent Motion Forecasting," *arXiv preprint arXiv:2007.12036*, 2020.

[20] S. Suo, S. Regalado, S. Casas, and R. Urtasun, "TrafficSim: Learning to Simulate Realistic Multi-Agent Behaviors," in *Proceedings of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, 2021.

[21] A. Scibior, V. Lioutas, D. Reda, P. Bateni, and F. Wood, "Imagining The Road Ahead: Multi-Agent Trajectory Prediction via Differentiable Simulation," *arXiv preprint arXiv:2104.11212*, 2021.

[22] Y. Yuan and K. Kitani, "Diverse trajectory forecasting with determinantal point processes," *arXiv preprint arXiv:1907.04967*, 2019.

[23] ——, "Dlow: Diversifying latent flows for diverse human motion prediction," in *European Conf. on Computer Vision*. Springer, 2020.

[24] H. Ma, Y. Sun, J. Li, M. Tomizuka, and C. Choi, "Continual multi-agent interaction behavior prediction with conditional generative memory," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, 2021.

[25] H. Cui, V. Radosavljevic, F.-C. Chou, T.-H. Lin, T. Nguyen, T.-K. Huang, J. Schneider, and N. Djuric, "Multimodal Trajectory Predictions for Autonomous Driving Using Deep Convolutional Networks," in *2019 Int. Conf. on Robotics and Automation (ICRA)*. IEEE, 2019.

[26] F. Janjoš, M. Dolgov, and M. J. Zöllner, "Self-Supervised Action-Space Prediction for Automated Driving," in *2021 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2021.

[27] S. Narayanan, R. Moslemi, F. Pittaluga, B. Liu, and M. Chandraker, "Divide-and-conquer for lane-aware diverse trajectory prediction," in *IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, 2021.

[28] P. Dendorfer, S. Elflein, and L. Leal-Taixé, "Mg-gan: A multi-generator model preventing out-of-distribution samples in pedestrian trajectory prediction," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

[29] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph Attention Networks," *arXiv:1710.10903*, 2017.

[30] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention Is All You Need," in *Advances in Neural Information Processing Systems*, 2017.

[31] P. J. Huber, "Robust estimation of a location parameter," in *Breakthroughs in statistics*. Springer, 1992.

[32] P. Charbonnier, L. Blanc-Feraud, G. Aubert, and M. Barlaud, "Two deterministic half-quadratic regularization algorithms for computed imaging," in *Proceedings of 1st International Conference on Image Processing*, vol. 2. IEEE, 1994.

[33] M. J. Black and P. Anandan, "The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields," *Computer vision and image understanding*, vol. 63, no. 1, 1996.

[34] J. E. Dennis Jr and R. E. Welsch, "Techniques for nonlinear least squares and robust regression," *Communications in Statistics-simulation and Computation*, vol. 7, no. 4, 1978.

[35] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[36] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," *arXiv preprint arXiv:1912.01703*, 2019.

[37] J. Bock, R. Krajewski, T. Moers, S. Runde, L. Vater, and L. Eckstein, "The ind dataset: A drone dataset of naturalistic road user trajectories at german intersections," *arXiv preprint arXiv:1911.07602*, 2019.

[38] N. Lee, W. Choi, P. Vernaza, C. B. Choy, P. H. Torr, and M. Chandraker, "DESIRE: Distant Future Prediction in Dynamic Scenes with Interacting Agents," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[39] J. Li, H. Ma, Z. Zhang, J. Li, and M. Tomizuka, "Spatio-Temporal Graph Dual-Attention Network for Multi-Agent Prediction and Tracking," *arXiv preprint arXiv:2102.09117*, 2021.

[40] X. Mo, Y. Xing, and C. Lv, "ReCoG: A Deep Learning Framework with Heterogeneous Graph for Interaction-Aware Trajectory Prediction," *arXiv preprint arXiv:2012.05032*, 2020.

[41] ——, "Heterogeneous Edge-Enhanced Graph Attention Network For Multi-Agent Trajectory Prediction," *arXiv:2106.07161*, 2021.

[42] H. Ma, Y. Sun, J. Li, and M. Tomizuka, "Multi-agent driving behavior prediction across different scenarios with self-supervised domain knowledge," in *2021 IEEE Intelligent Transportation Systems (ITSC)*.

[43] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv preprint arXiv:1412.6980*, 2014.

Paper V

- Title: Unscented Autoencoder
- Authors: Faris Janjoš and Lars Rosenbaum and Maxim Dolgov and J. Marius Zöllner
- Venue: 2023 International Conference on Machine Learning (ICML)

Unscented Autoencoder

Faris Janjoš [1] **Lars Rosenbaum** [1] **Maxim Dolgov** [1] **J. Marius Zöllner** [2]

Abstract

The Variational Autoencoder (VAE) is a seminal approach in deep generative modeling with latent variables. Interpreting its reconstruction process as a nonlinear transformation of samples from the latent posterior distribution, we apply the Unscented Transform (UT) – a well-known distribution approximation used in the Unscented Kalman Filter (UKF) from the field of filtering. A finite set of statistics called sigma points, sampled deterministically, provides a more informative and lower-variance posterior representation than the ubiquitous noise-scaling of the reparameterization trick, while ensuring higher-quality reconstruction. We further boost the performance by replacing the Kullback-Leibler (KL) divergence with the Wasserstein distribution metric that allows for a sharper posterior. Inspired by the two components, we derive a novel, deterministic-sampling flavor of the VAE, the Unscented Autoencoder (UAE), trained purely with regularization-like terms on the per-sample posterior. We empirically show competitive performance in Fréchet Inception Distance (FID) scores over closely-related models, in addition to a lower training variance than the VAE[1].

1. Introduction

The Variational Autoencoder (VAE) (Rezende et al., 2014; Kingma et al., 2015) is a widely used method for learning deep latent variable models via maximization of the data likelihood using a reparametrized version of the Evidence Lower Bound (ELBO). Deep latent variable models are used as generative models in a variety of application domains such as image (Vahdat & Kautz, 2020), language (Bowman et al., 2015; Kusner et al., 2017), and dynamics modeling (Karl et al., 2016). A good generative model requires the VAE to produce high-quality samples from the prior latent variable distribution and a disentangled latent representation is desired to control the generation process (Higgins et al., 2017). Another important application of deep latent variable models is representation learning, where the goal is to induce a latent representation facilitating downstream tasks (Bengio et al., 2013; Townsend et al., 2019; Tripp et al., 2020; Rombach et al., 2022). In many of these tasks a good sample quality, as well as a 'well-behaved' latent representation with a high reconstruction accuracy is desired.

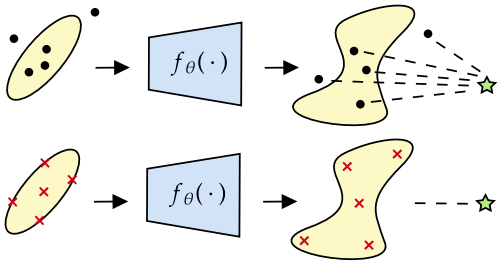


Figure 1: The VAE decoder $f_\theta(\cdot)$ can be interpreted as a nonlinear mapping of the Gaussian posterior distribution generated by the encoder, resulting in a non-Gaussian output distribution. The standard VAE (top) samples randomly from the posterior (black points) and matches each decoded sample to the ground truth (green star). Our model (bottom) samples and transforms fixed posterior sigma points (red) instead. By matching the mean of the transformed points, we push the entire output distribution to resemble the ground truth.

Since their introduction, VAEs have been one of the methods of choice in generative modeling due to their comparatively easy training and the ability to map data to a lower dimensional representation as opposed to generative adversarial networks (Goodfellow et al., 2014). However, despite their popularity there are still open challenges in VAE training addressed by recent works. A major problem of VAEs is their tendency to have a trade-off between the quality of samples from the prior and the reconstruction qual-

[1]Robert Bosch GmbH, Corporate Research, 71272 Renningen, Germany [2]Research Center for Information Technology (FZI), 76131 Karlsruhe, Germany. Correspondence to: <firstname.last-name@de.bosch.com>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

[1]Code available at: https://github.com/boschresearch/unscented-autoencoder

ity. This trade-off can be attributed to overly simplistic priors (Bauer & Mnih, 2019), encoder/decoder variance (Dai & Wipf, 2019), weighting of the KL divergence regularization (Higgins et al., 2017; Tolstikhin et al., 2018), or the aggregated posterior not matching the prior (Tolstikhin et al., 2018; Ghosh et al., 2019). Furthermore, the VAE objective can be prone to spurious local maxima leading to posterior collapse (Chen et al., 2017; Lucas et al., 2019; Dai et al., 2020), which is characterized by the latent posterior (partially) reducing to an uninformative prior. Finally, the variational objective requires approximations of expectations by sampling, which causes increased gradient variance (Burda et al., 2016) and makes the training sensitive to several hyperparameters (Bowman et al., 2015; Higgins et al., 2017).

Our main technical contributions are two modifications to the original VAE objective resulting in an improved sample and reconstruction quality. We propose to use a well-known algorithm from the filtering and control literature, the Unscented Transform (UT) (Uhlmann, 1995), to obtain lower-variance, albeit potentially biased gradient estimates for the optimization of the variational objective. A lower variance is achieved by only sampling at the sigma points of the variational posterior and transforming these points with a deterministic decoder. In this context, we show that reconstructing the entire posterior distribution via its sigma points (visualized in Fig. 1) is superior in resulting image quality to reconstructing individual random samples. Furthermore, we observe that the regularization toward a standard normal prior using a KL divergence often harshly penalizes low variance along some components even though the low variance is usually beneficial for reconstruction. Thus, we use a different regularization based on the Wasserstein metric (Patrini et al., 2020). To account for resulting sharper posteriors, we add a regularizer for decoder smoothness around the mean encoded value, similar to (Ghosh et al., 2019). We conduct rigorous experiments on several standard image datasets to compare our modifications against the VAE baseline, the closely-related Regularized Autoencoder (RAE) (Ghosh et al., 2019), the Importance-Weighted Autoencoder (IWAE) (Burda et al., 2016), as well as the Wasserstein Autoencoder (WAE) (Tolstikhin et al., 2018).

2. Related Work

Many recent works on VAEs focus on understanding and addressing still existing problems like undesired posterior collapse (Dai et al., 2020), trade-off between sample and reconstruction quality (Tolstikhin et al., 2018; Bauer & Mnih, 2019), or non-interpretable latent representations (Rolinek et al., 2019; Higgins et al., 2017). Other recent works suggest to move from the probabilistic VAE models to deterministic models, such as the RAE in (Ghosh et al., 2019); our model can be considered as part of this class. As previously mentioned, we employ two major modifications to the VAE, namely the **Unscented Transform** and the **Wasserstein metric**, as well as **decoder regularization**; we outline the section accordingly.

We use the **Unscented Transform** (Uhlmann, 1995) from the field of nonlinear filtering within signal processing. In this context, the signal state estimate is often assumed to be Gaussian in order to maintain tractability. However, nonlinear prediction and measurement models always invalidate this assumption at each time step so that a re-approximation becomes necessary. A commonly used approach is the Extended Kalman Filter (EKF), where a linearization of the models is employed so that the Gaussian state remains Gaussian during filtering. In contrast, alternative approaches that represent the Gaussian state (assuming application in the context of the VAE posterior) with samples for propagation and update have emerged. These approaches can be clustered according to the employed sampling method – random as in (Gaussian) particle filters (Doucet & Johansen, 2011) or deterministic, e.g. in the UKF (Julier et al., 2000). In the UKF, the n-dimensional Gaussian is approximated with $2n + 1$ deterministic samples, which can be propagated through the nonlinearities and are sufficient for computing the statistics of a Gaussian distribution, i.e. its mean and covariance. This procedure is referred to as the Unscented Transform (UT).

The use of deterministic sampling[2] aims to achieve a good coverage of the distribution represented with the mean and covariance. Although this approach produces biased estimates of the involved expectations compared to random sampling due to non-i.i.d. samples, it often captures well the nonlinearities applied to the distribution, for a finite, small set of samples in the filtering context. This observation can transfer to neural networks due to their Lipschitz continuity (Khromov & Singh, 2023). Our UT experiments empirically underline this expectation. For a more comprehensive overview of the UT and the UKF, we refer the reader to (Menegaz et al., 2015).

The UT uses several samples to get an estimate of the moments of a nonlinearly transformed probability distribution. Along those lines, our method also relates to the IWAE (Burda et al., 2016) and some of its extensions (Tucker et al., 2018). IWAE uses importance weighting of K posterior samples to obtain a variational distribution closer to the true posterior (Cremer et al., 2017). The method is known to have a diminishing gradient signal for the inference network (Rainforth et al., 2018) if no additional improvements are used (Tucker et al., 2018). Using the Wasserstein metric, the inference distribution is sharp,

[2] Sampling from a set of points at fixed locations in the domain.

so practically there is not much gain in a more complex distribution. However, multiple samples can help to obtain lower variance gradient estimates, which also applies to the IWAE by taking a multiple of K samples. Sampling only at the sigma points reduces this variance even more and is known to empirically work well in filtering and control.

The **Wasserstein metric** is used in (Tolstikhin et al., 2018; Patrini et al., 2020) to regularize the aggregated posterior $q_{\text{agg}}(\mathbf{z}) = \mathbb{E}_{p(\mathbf{x})}\left[q(\mathbf{z}|\mathbf{x})\right]$ toward the standard normal prior. The authors also show that such an objective is an upper bound to the Wasserstein distance between the sampling distribution of the generative model and the data distribution if the regularization is scaled by the Lipschitz constant of the generator. In contrast, we do not regularize the aggregated posterior, but use the Wasserstein distance to weakly regularize the mean and variance of the encoder, such that neither explodes and we can do ex-post density estimation. From a theoretical point of view, we do not fix the prior but learn the manifold; the aggregated posterior is learned by fitting a mixture to the encoded data points.

Finally, our work incorporates several ideas from the recently published RAE (Ghosh et al., 2019). We also use a **decoder regularization** term based on the decoder Jacobian in our loss, which promotes smoothness of the latent space. In contrast to the RAE however, we generalize the term from a deterministic to a stochastic encoder as not every data point might be encoded with the same fidelity. Furthermore, we employ ex-post density estimation as we do not explicitly regularize the aggregated posterior toward a prior. Conceptually, the UAE can be placed between the VAE, characterized by significant sampling variance, and the purely deterministic RAE.

3. Problem Description

Most generative models take a max-likelihood approach to model a real-world distribution $p(\mathbf{x})$ via the θ-parameterized probabilistic generator model $p_\theta(\mathbf{x})$

$$\theta \leftarrow \arg\max_\theta \quad \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[\log p_\theta(\mathbf{x})] \,. \tag{1}$$

In this setting, latent variable generative approaches assume an underlying structure in $p(\mathbf{x})$ not directly observable from the data and model this structure with a latent variable $\mathbf{z}$, which is well-motivated by de Finetti's theorem (Accardi, 2001). As a result, the distribution $p(\mathbf{x})$ can be represented as a product of tractable distributions. However, directly incorporating $\mathbf{z}$ via an integral $\int p_\theta(\mathbf{x}|\mathbf{z})p(\mathbf{z})d\mathbf{z}$ is intractable; thus, one introduces an amortized variational distribution $q_\phi(\mathbf{z}|\mathbf{x})$ (Zhang et al., 2018) and obtains

$$\log p_\theta(\mathbf{x}) = \log \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}\left[\frac{p_\theta(\mathbf{x},\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}\right] \,. \tag{2}$$

This model assumption is the basis of variational inference. Applying Jensen's inequality yields the well-known ELBO, denoted by $\mathcal{L}$

$$\begin{aligned}\log p_\theta(\mathbf{x}) \geq \mathcal{L} = \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - \\ - D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|p(\mathbf{z})) \,,\end{aligned} \tag{3}$$

which is maximized w.r.t. θ and ϕ. The first term accounts for the quality of reconstructed samples and the $D_{\mathrm{KL}}(...)$ term pushes the approximate posterior to mimic the prior, i.e. it enforces a $p(\mathbf{z})$-like structure to the latent space.

Training on $\mathcal{L}$ in Eq. (3) requires computing gradients w.r.t. θ and ϕ. This is relatively straightforward for the generator parameters, however, requiring a high-variance policy gradient for the posterior parameters. To avoid this issue in practice, the reparameterization trick (Kingma et al., 2015) is used to simplify the sampling of the approximate posterior by means of an easy-to-sample distribution. Assuming a Gaussian posterior $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, we can sample a multivariate normal and obtain the latent feature vector via the deterministic transformation

$$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{L}\boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \quad \boldsymbol{\Sigma} = \boldsymbol{L}\boldsymbol{L}^T \,. \tag{4}$$

With the help of the reparameterization trick, the VAE (Kingma & Welling, 2013) provides a framework for optimizing the loss function from the condition in Eq. (3) via an encoder–decoder generative latent variable model. The encoder $E_\phi(\mathbf{x}) = \{\boldsymbol{\mu}_\phi(\mathbf{x}), \boldsymbol{\Sigma}_\phi(\mathbf{x})\}$ parameterizes a multivariate Gaussian $q_\phi(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}|\boldsymbol{\mu}_\phi(\mathbf{x}), \boldsymbol{\Sigma}_\phi(\mathbf{x}))$, where $\boldsymbol{\Sigma}_\phi$ is usually a diagonal matrix, $\boldsymbol{\Sigma}_\phi = \mathrm{diag}(\boldsymbol{\sigma}_\phi)$. The decoder $D_\theta(\mathbf{z}) = \boldsymbol{\mu}_\theta(\mathbf{z})$ is in practice rendered deterministic: $p_\theta(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_\theta(\mathbf{z}), \mathbf{0})$, reducing the reconstruction term in Eq. (3) to a simple mean-squared error under the expectation of the posterior $\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}\|\mathbf{x} - \boldsymbol{\mu}_\theta(\mathbf{z})\|_2^2$. The VAE uses the reparameterization trick for efficient sampling from the posterior q_ϕ (in practice providing only a single sample to the decoder), which enables a lower-variance gradient backpropagation through the encoder.

The deterministic decoder and the reparameterization trick allow for a slightly different interpretation of the reconstruction/generation process: a (highly) nonlinear transformation of an input distribution, represented (usually) only by a single stochastic sample. The sample is white noise[3], scaled and shifted by the posterior moments. This interpretation serves as the basis for our work, where the unscented transform of the input distribution serves as an alternative to the single-stochastic-sample representation. In the next section, we outline the unscented transform representation of the input to the decoder via a set of deterministically computed and sampled sigma points.

[3]The white-noise interpretation is used in (Ghosh et al., 2019) to justify regularization as an alternative to the noise sampling.

4. Unscented Transform of the Posterior

4.1. Background

The unscented transform (Uhlmann, 1995) is a method to evaluate a nonlinear transformation of a distribution characterized by its first two moments. Assume a known deterministic function $\boldsymbol{f}$ applied to a distribution $P(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ with mean and covariance $\boldsymbol{\mu} \in \mathbb{R}^n$ and $\boldsymbol{\Sigma} \in \mathbb{R}^{n \times n}$. If $\boldsymbol{f}$ is a linear transformation, one can describe the distribution $Q(\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\Sigma}})$ at the output via $\hat{\boldsymbol{\mu}} = \boldsymbol{f}\boldsymbol{\mu}$ and $\hat{\boldsymbol{\Sigma}} = \boldsymbol{f}\boldsymbol{\Sigma}\boldsymbol{f}^T$. Similarly, for a nonlinear transformation $\boldsymbol{f}$ but a zero covariance matrix $\boldsymbol{\Sigma} = \mathbf{0}$, the mean of the transformed distribution is $\hat{\boldsymbol{\mu}} = \boldsymbol{f}(\boldsymbol{\mu})$. However, in the general case it is not possible to determine $\hat{\boldsymbol{\mu}}$ and $\hat{\boldsymbol{\Sigma}}$ of the $\boldsymbol{f}$-transformed distribution given $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ since the result depends on higher-order moments. Thus, the unscented transform is useful; it provides a mechanism to obtain this result via an approximation of the input distribution while assuming full knowledge of $\boldsymbol{f}$.

In computing the unscented transform, first a set of sigma points characterizing the input $P(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is chosen. The most common approach (Menegaz et al., 2015) is to take a set $\{\boldsymbol{\chi}_i\}_{i=0}^{2n}$, $\boldsymbol{\chi}_i \in \mathbb{R}^n$ of $2n+1$ symmetric points centered around the mean (incl. the mean), e.g. for $1 \leq i \leq n$,

$$\begin{aligned} \boldsymbol{\chi}_0 &= \boldsymbol{\mu} \,, \\ \boldsymbol{\chi}_i &= \boldsymbol{\mu} + \sqrt{(\kappa+n)\boldsymbol{\Sigma}}\big|_i \,, \\ \boldsymbol{\chi}_{i+n} &= \boldsymbol{\mu} - \sqrt{(\kappa+n)\boldsymbol{\Sigma}}\big|_i \,, \end{aligned} \tag{5}$$

where $\kappa > -n$ is a real constant and $\big|_i$ denotes the i-th column. The approximation in Eq. (5) is unbiased; the mean and covariance of the sigma points are $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$. Thus, one can compute the transformation $\hat{\boldsymbol{\chi}}_i = \boldsymbol{f}(\boldsymbol{\chi}_i)$ and estimate the mean and covariance of the $\boldsymbol{f}$-transformed distribution

$$\hat{\boldsymbol{\mu}} = \tfrac{1}{2n+1} \textstyle\sum_{i=0}^{2n} \hat{\boldsymbol{\chi}}_i \,, \tag{6}$$

$$\hat{\boldsymbol{\Sigma}} = \tfrac{1}{2n+1} \textstyle\sum_{i=0}^{2n} (\hat{\boldsymbol{\chi}}_i - \hat{\boldsymbol{\mu}})(\hat{\boldsymbol{\chi}}_i - \hat{\boldsymbol{\mu}})^T \,. \tag{7}$$

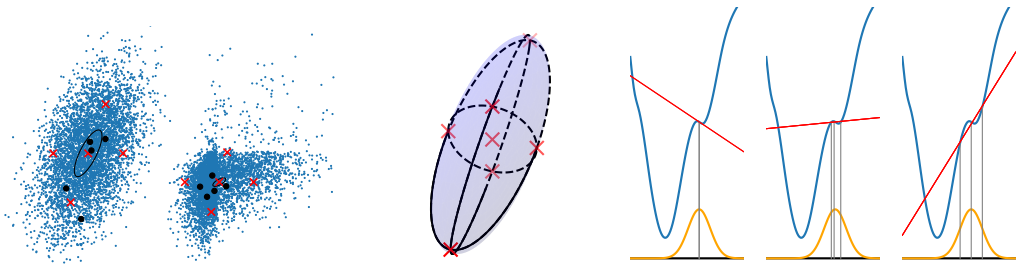
A visualization of the sigma points and their transformation is depicted in Fig. 2a. The procedure in Eq. (5-7) effectively applies the fully-known function $\boldsymbol{f}$ to an approximating set of points whose mean and covariance equal the original distribution's. Therefore, in the context of the commonly used VAE decoder nonlinearities, the mean and covariance of the transformed sigma points can be closer to the true transformed mean and covariance compared to the ones computed by propagating the same number of random samples from the original distribution.

4.2. Unscented Transform in the VAE

In an ELBO maximization setting from Eq. (3), the nonlinear transformation of the posterior in the decoder lends itself straightforwardly to the unscented transform approximation. Given any posterior defined by $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$, we can compute the sigma points (for example according to Eq. (5)) and provide them to the decoder. In a VAE, the sigma points provide a deterministic-sampling alternative to the reparameterization-trick-computed random samples of the latent space. Furthermore, computing the average reconstruction of the sigma points at the output of the decoder provides an approximation of the mean of the entire transformed posterior distribution in Eq. (6), while implicitly taking into account the variance in Eq. (7), as opposed to the per-sample reconstructions.

The choice of the number of sigma points provided to the decoder is similar to the sampling in Eq. (4), where one can realize a single latent vector with a single sample from $\mathcal{N}(\mathbf{0}, \mathbf{I})$ or multiple latents, resulting in a trade-off between reconstruction quality and computation demands (Ghosh et al., 2019). However, taking a single or few random samples in the VAE setting can produce instances very far from the mean, especially in high dimensional spaces. In contrast, sampling sigma points produces a more controlled overall estimate of the posterior (as well as producing a more accurate transformed posterior, see Eq. (6-7)) since the samples lie on the border of a hyperellipsoid induced by the covariance matrix $\boldsymbol{\Sigma}$ (example in Fig. 2b). Thus, while computing the loss function gradients (which are a function of the samples), the sigma-sampling has the potential to bring a more accurate and lower-variance estimate when all the sigma points are considered. This is illustrated in Fig 2c. Further empirical arguments validating the lower gradient variance claim are provided in Appendix B.

The sigma-sampling of the UT can be applied to any learned posterior described by its first two moments (as common in generative models), not only the VAE standard normal. With this description, the sigma points cannot be the uniquely optimal representation of the distribution since there is an infinite number of distributions that share the first two moments. However, the UT has shown superior empirical performance over other representations in extensive experiments in (Julier et al., 2000) and (Zhang et al., 2009), under various distributions and nonlinear functions, and especially for the case of differentiable functions. This has led to the UKF, built on this paradigm, being one of the major algorithms in filtering and control. Guided by the success of the method, we hypothesize that applying the UT in the VAE setting has the potential to, for a finite set of samples, provide a better approximation of the learned two-moment Gaussian posterior than the ubiquitous independent random sampling and reconstruction. With these insights, we develop the UAE model presented in the next section.



(a) (b) (c)

Figure 2: (best viewed in color) (a) **(transforming 2D sigma points)** Left: a Gaussian with its Monte Carlo approximation (blue), sigma points computed according to Eq. (5) (red), and five random samples (black points). Right: nonlinear RReLU activation (Xu et al., 2015) applied to the distribution, sigma points, and the random samples. In this example, the five sigma points provide a better approximation of the transformed distribution than the five random samples.
(b) **(3D sigma points)** Sigma points (red) on an ellipsoid spanned by a 3×3 covariance matrix, consisting of a central sigma point and a pair of sigma points on each axis.
(c) **(gradient variance)** Left: loss function (blue) at a sample (gray) corresponding to the standard normal (yellow) mean. The gradient of the loss function (red) at the mean is not representative of the true gradient. Middle: a high-variance gradient computed from the gradients at the three random samples drawn from the standard normal, potentially far away from the true gradient. Right: gradient of the loss function computed from the gradients at the three sigma points; although the estimate is potentially biased due to the applied nonlinear transformation, it has lower variance than if computed from the random points. The three provided examples can be interpreted as the RAE-(Ghosh et al., 2019), VAE-, and UAE-like sampling procedures.

5. Unscented Autoencoder (UAE)

The UAE is a deterministic-sampling autoencoder model maximizing the ELBO. It addresses the maximum likelihood optimization problem from Sec. 3, namely the $\mathcal{L}$ maximization from Eq. (3), by computing the UT of the posterior $q_\phi(\mathbf{z}|\mathbf{x})$ parameterized by the encoder $E_\phi(\mathbf{x}) = \{\boldsymbol{\mu}_\phi(\mathbf{x}), \boldsymbol{\Sigma}_\phi(\mathbf{x})\}$ (see Eq. (5-7)). The latent features $\mathbf{z}$ can be obtained by deterministically sampling multiple sigma points, resulting in a lower variance sampling than of the reparameterization trick in Eq. (4). Good performance of the model is further boosted by replacing the vanilla KL divergence with the Wasserstein distribution metric, which effectively performs a regularization of the posterior moments. The decoder regularization applies an additional smoothing effect on the latent space – it is formally derived in Sec. 5.2. The full training objective consists of optimizing $\phi, \theta \leftarrow \arg\min_{\phi,\theta} \mathcal{L}_{\text{UAE}}$,

$$\mathcal{L}_{\text{UAE}} = \mathbb{E}_{\mathbf{x}\sim p_{\text{data}}} \mathcal{L}_{\text{REC}} + \beta \mathcal{L}_W + \gamma \mathcal{L}_{D_\theta \text{REG}} \,, \tag{8}$$

where β (from the β-VAE (Higgins et al., 2017)) and γ are weights.

The **reconstruction term** $\mathcal{L}_{\text{REC}}$ is an L_2 loss function incorporating the average of decoded sigma points

$$\begin{aligned} \mathcal{L}_{\text{REC}} &= \|\mathbf{x} - \tfrac{1}{K}\textstyle\sum_{k=1}^{K} D_\theta(\mathbf{z}_k)\|_2^2 \,, \\ \mathbf{z}_k &\sim \{\boldsymbol{\chi}_i(\boldsymbol{\mu}_\phi, \boldsymbol{\Sigma}_\phi)\}_{i=0}^{2n} \,, \end{aligned} \tag{9}$$

where K n-dimensional vectors $\mathbf{z}_k$ are sampled from the set of sigma points, $K \leq 2n+1$. Various sampling heuristics are investigated in Appendix C. Note that this reconstruction loss function differs from the commonly used $\frac{1}{K}\sum_{k=1}^{K} \|\mathbf{x} - D_\theta(\mathbf{z}_k)\|_2^2$, where each decoded sample is matched to the ground truth. This strategy, employed in the standard multi-sample VAE, aims at getting the same output image for different samples thus demanding a certain attenuation property from the deterministic decoder. In contrast, Eq. (9) is motivated by the application of the UT in filtering where after propagating the sigma points through a nonlinear function a Gaussian is fit to the posterior (see Eq. (6-7)). By applying the loss to the mean output image, we essentially maintain a probability distribution at the output.

We use the **Wasserstein metric term** $\mathcal{L}_{\text{W}}$ as an alternative to the KL divergence. For a multivariate posterior and a multivariate normal prior, the KL divergence is defined as

$$\mathcal{L}_{\text{KL}} = \|\boldsymbol{\mu}_\phi\|_2^2 + \text{tr}(\boldsymbol{\Sigma}_\phi) - n - 2\text{tr}(\log \boldsymbol{L}_\phi) \,, \tag{10}$$

in the general case[4] of a full-covariance matrix $\boldsymbol{\Sigma}_\phi = \boldsymbol{L}_\phi \boldsymbol{L}_\phi^T$. Instead, due to favorable optimization properties and higher-quality reconstruction, we use the Wasserstein metric between distributions. This metric effectively replaces the covariance part of the KL term, $\text{tr}(\boldsymbol{\Sigma}_\phi) - 2\text{tr}(\log \boldsymbol{L}_\phi)$, with the squared Frobenius norm of the mismatch between the lower triangular matrix and the identity

[4] Derived from $D_{\text{KL}}(\mathcal{N}_0 \| \mathcal{N}_1) = \frac{1}{2}(\text{tr}(\boldsymbol{\Sigma}_1^{-1}\boldsymbol{\Sigma}_0) - n + (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)^T \boldsymbol{\Sigma}_1^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) + \log(\frac{\det \boldsymbol{\Sigma}_1}{\det \boldsymbol{\Sigma}_0}))$ for $\mathcal{N}_1(\mathbf{0}, \mathbf{I})$ and $\boldsymbol{\Sigma} = \boldsymbol{L}\boldsymbol{L}^T$.

$$\mathcal{L}_W = \|\boldsymbol{L}_\phi - \mathbf{I}\|_F^2 = \mathrm{tr}(\boldsymbol{\Sigma}_\phi) - 2\mathrm{tr}(\boldsymbol{L}_\phi)\,. \tag{11}$$

It differs from the original objective in Eq. (10) only in the lack of a logarithm while sharing the same global minimum. Further details are provided in Sec. 5.3. Such a loss function allows the variance to approach zero (which is instead strongly penalized by the logarithm in Eq. (10)), yielding a sharper posterior.

The **decoder regularization term** $\mathcal{L}_{D_\theta\mathrm{REG}}$ is a generalization of the gradient penalty term in (Ghosh et al., 2019), accounting for a fully probabilistic formulation. It can be realized as a penalty on the input–output gradient of the posterior mean, weighted by the largest eigenvalue of the covariance matrix

$$\mathcal{L}_{D_\theta\mathrm{REG}} = \lambda_{\max}(\boldsymbol{\Sigma}_\phi)\|\nabla_{\boldsymbol{\mu}_\phi} D_\theta(\boldsymbol{\mu}_\phi)\|_2^2\,. \tag{12}$$

We approximate $\lambda_{\max}(\boldsymbol{\Sigma}_\phi)$ by the largest diagonal, which is correct for a diagonal $\boldsymbol{\Sigma}_\phi$.

We provide an overview of the VAE, RAE, and UAE loss functions in Tab. 1, together with the models that are conceptually between the VAE and UAE. Additional models employing different combinations of the loss function components are provided in Appendix D, Tab. 7.

5.1. Sampling From the Prior-Less UAE

Since the UAE model doesn't regularize the aggregated posterior toward the prior using the KL divergence (Hoffman & Johnson, 2016) or the Wasserstein metric (Patrini et al., 2020) (we use the per-posterior Wasserstein metric), it is not equipped with an easy-to-use sampling procedure as the VAE. To remedy this, we use the straightforward ex-post density estimation procedure described in (Ghosh et al., 2019) for the deterministic RAE model. We fit the latent means $\boldsymbol{\mu}_\phi$ for each input sample $\mathbf{x}$ to a 10-component Gaussian Mixture Model (GMM) (which has shown good performance and generalization ability in the experiments of (Ghosh et al., 2019) even for VAE models) and use the mixture to sample from the latent space. For a fair comparison, we utilize this procedure in all models.

5.2. ELBO Derivation

In the following, we analytically derive the UAE model in Eq. (8). The derivation is largely inspired from (Ghosh et al., 2019), with a few crucial differences allowing for greater generalizability and less restrictive assumptions. We start with the general ELBO minimization formulation in Eq. (3), augmented with a constraint

$$\arg\min_{\phi,\theta} \quad E_{x\sim p_{\text{data}}}\mathcal{L}_{\text{REC}} + \mathcal{L}_{\text{KL}} \tag{13}$$

$$\text{s.t.} \quad \|D_\theta(\mathbf{z}_1) - D_\theta(\mathbf{z}_2)\|_p < \epsilon, \quad \mathbf{z}_1, \mathbf{z}_2 \sim q_\phi(\mathbf{z}|\mathbf{x}),\ \forall \mathbf{x} \sim p_{\text{data}}. \tag{14}$$

Here, the decoder outputs given any two latent vectors $\mathbf{z}_1$ and $\mathbf{z}_2$ (any two draws from the posterior $q_\phi(\mathbf{z}|\mathbf{x})$) are bounded via their p-norm difference, for a deterministic decoder D_θ. It was shown in (Ghosh et al., 2019) that the constraint in Eq. (14) can be reformulated as

$$\sup\{\|\nabla_{\mathbf{z}} D_\theta(\mathbf{z})\|_p\} \cdot \sup\{\|\mathbf{z}_1 - \mathbf{z}_2\|_p\} < \epsilon\,. \tag{15}$$

We provide the full derivation in Appendix E. In Eq. (15), $\nabla_{\mathbf{z}} D_\theta(\mathbf{z})$ is the derivative of the decoder output w.r.t. its input (not the parameterization θ). The second term in the product depends on the parameterization of the posterior $q_\phi(\mathbf{z}|\mathbf{x})$. For a Gaussian, $\sup\{\|\mathbf{z}_1 - \mathbf{z}_2\|_p\}$ becomes a functional r of the posterior entropy, $r(\mathbb{H}(q_\phi(\mathbf{z}|\mathbf{x})))$. At this point, the RAE derivation from (Ghosh et al., 2019) takes a strong simplifying assumption of constant entropy for all samples $\mathbf{x}$, effectively asserting constant variance in the posterior. This allows to incorporate a simplified version of Eq. (15) into Eq. (13) via the Lagrange multiplier γ, obtaining the following RAE loss function[5]

$$\mathcal{L}_{\text{RAE}} = \|\mathbf{x} - D_\theta(\mathbf{z})\|_2^2 + \beta\|\mathbf{z}\|_2^2 + \gamma\|\nabla_{\mathbf{z}} D_\theta(\mathbf{z})\|_2^2\,. \tag{16}$$

Here, the KL-term from Eq. (13) is approximated by $\|\mathbf{z}\|_2^2$ due to the constant variance assumption.

In the UAE formulation, the samples $\mathbf{z}_1$ and $\mathbf{z}_2$ in Eq. (15) simply correspond to the sigma points of $q_\phi(\mathbf{z}|\mathbf{x})$ parameterized by $E_\phi(\mathbf{x}) = \{\boldsymbol{\mu}_\phi(\mathbf{x}), \boldsymbol{\Sigma}_\phi(\mathbf{x})\}$. Therefore, the term $\sup\{\|\mathbf{z}_1 - \mathbf{z}_2\|_p\}$ can be computed analytically as the largest eigenvalue $\lambda_{\max}$ of the covariance matrix $\boldsymbol{\Sigma}_\phi$. We regularize the decoder in an RAE-manner around the posterior mean with $\|\nabla_{\boldsymbol{\mu}_\phi} D_\theta(\boldsymbol{\mu}_\phi)\|_p$ to enforce smoothness. Finally, the UAE does not require the constant variance assumption; we can incorporate a posterior KL-term or the Wasserstein metric used in Eq. (8). Thus, we arrive at the following analytical UAE loss function from Eq. (8)

$$\begin{aligned}\mathcal{L}_{\text{UAE}} = E_{\mathbf{x}\sim p_{\text{data}}}\mathcal{L}_{\text{REC}} + \beta\mathcal{L}_{\text{W}} + \\ +\gamma\lambda_{\max}(\boldsymbol{\Sigma}_\phi)\|\nabla_{\boldsymbol{\mu}_\phi} D_\theta(\boldsymbol{\mu}_\phi)\|_p\,,\end{aligned} \tag{17}$$

where a more general form of the Eq. (15) constraint is used than in Eq. (16).

It follows from the derivation that the major difference between the RAE on the one hand and VAE and UAE on the other is that the RAE assumes constant variance in mapping the training data distribution into the latent space, thus not including any variance-compensating terms in the loss function. In effect, the RAE considers all the dimensions equally and cannot take into account that the encoder might have different uncertainty per dimension and data point.

[5] In (Ghosh et al., 2019), the decoder gradient penalty from Eq. (16) is the analytically derived regularization; alternatives such as weight decay and spectral norm are offered as well and can also be used in the UAE.

Table 1: A comparison of the VAE, RAE-GP (employing a Gradient Penalty (GP) on the decoder, a less general version of Eq. (12)), and UAE loss functions, including the intermediate models UT-VAE, VAE*, UT-VAE*, (weights omitted for clarity). UT-VAE uses the unscented transform in the VAE, VAE* uses the Wasserstein metric from Eq. (11), and UT-VAE* differs from the UAE only in the lack of a decoder regularization term. All models use a diagonal posterior representation (except RAE, which does not model uncertainty). The terms $\mathbf{z}$, $\boldsymbol{\mu}_\phi$, and $\boldsymbol{\sigma}_\phi$ are realized given the sample $\mathbf{x}$.

	Loss function	Posterior sampling
$\mathcal{L}_{\text{VAE}}$	$\frac{1}{K}\sum_{k=1}^{K}\|\mathbf{x}-D_\theta(\mathbf{z}_k)\|_2^2+\|\boldsymbol{\mu}_\phi\|_2^2-n+\sum_i\sigma_{\phi,i}^2-2\log\sigma_{\phi,i}$	$\mathbf{z}_k=\boldsymbol{\mu}_\phi+\boldsymbol{\sigma}_\phi\odot\boldsymbol{\epsilon}_k,\ \boldsymbol{\epsilon}_k\sim\mathcal{N}(\mathbf{0},\mathbf{I})$
$\mathcal{L}_{\text{UT-VAE}}$	$\|\mathbf{x}-\frac{1}{K}\sum_{k=1}^{K}D_\theta(\mathbf{z}_k)\|_2^2+\|\boldsymbol{\mu}_\phi\|_2^2-n+\sum_i\sigma_{\phi,i}^2-2\log\sigma_{\phi,i}$	$\mathbf{z}_k\sim\{\boldsymbol{\chi}_i(\boldsymbol{\mu}_\phi,\text{diag}(\boldsymbol{\sigma}_\phi^2))\}_{i=0}^{2n}$
$\mathcal{L}_{\text{RAE-GP}}$	$\|\mathbf{x}-D_\theta(\mathbf{z})\|_2^2+\|\mathbf{z}\|_2^2+\|\nabla_\mathbf{z}D_\theta(\mathbf{z})\|_2^2$	None, $\mathbf{z}=\boldsymbol{\mu}_\phi$
$\mathcal{L}_{\text{VAE*}}$	$\frac{1}{K}\sum_{k=1}^{K}\|\mathbf{x}-D_\theta(\mathbf{z}_k)\|_2^2+\|\boldsymbol{\mu}_\phi\|_2^2+\|\text{diag}(\boldsymbol{\sigma}_\phi^2)-\mathbf{I}\|_F^2$	$\mathbf{z}_k=\boldsymbol{\mu}_\phi+\boldsymbol{\sigma}_\phi\odot\boldsymbol{\epsilon}_k,\ \boldsymbol{\epsilon}_k\sim\mathcal{N}(\mathbf{0},\mathbf{I})$
$\mathcal{L}_{\text{UT-VAE*}}$	$\|\mathbf{x}-\frac{1}{K}\sum_{k=1}^{K}D_\theta(\mathbf{z}_k)\|_2^2+\|\boldsymbol{\mu}_\phi\|_2^2+\|\text{diag}(\boldsymbol{\sigma}_\phi^2)-\mathbf{I}\|_F^2$	$\mathbf{z}_k\sim\{\boldsymbol{\chi}_i(\boldsymbol{\mu}_\phi,\text{diag}(\boldsymbol{\sigma}_\phi^2))\}_{i=0}^{2n}$
$\mathcal{L}_{\text{UAE}}$	$\|\mathbf{x}-\frac{1}{K}\sum_{k=1}^{K}D_\theta(\mathbf{z}_k)\|_2^2+\|\boldsymbol{\mu}_\phi\|_2^2+\|\text{diag}(\boldsymbol{\sigma}_\phi^2)-\mathbf{I}\|_F^2+\max(\boldsymbol{\sigma}_\phi^2)\|\nabla_{\boldsymbol{\mu}_\phi}D_\theta(\boldsymbol{\mu}_\phi)\|_2^2$	$\mathbf{z}_k\sim\{\boldsymbol{\chi}_i(\boldsymbol{\mu}_\phi,\text{diag}(\boldsymbol{\sigma}_\phi^2))\}_{i=0}^{2n}$

Additionally, the difference between VAE and UAE is that the VAE incorporates a sampling procedure with higher variance than the deterministic sigma-point sampling used in the unscented transform. Therefore, loss function-wise, the UAE can be regarded as a middle-ground between the VAE and RAE – deterministic and lower-variance in training than the VAE, but with greater generalization capabilities than the RAE due to the probabilistic formulation.

5.3. Posterior Regularization via the Wasserstein Metric

The usage of the Wasserstein metric is motivated by practical properties of VAE model optimization. The training can be sensitive to the weighting of the KL divergence term, which can lead to posterior collapse (Dai et al., 2020). The main factor is the strong variance regularization of the KL divergence with its log term, which can be written as

$$\mathcal{L}_{\text{KL}} = \|\boldsymbol{\mu}_\phi\|_2^2 + \text{tr}(\boldsymbol{\Sigma}_\phi) - n - 2\sum_i \log L_{\phi,ii} \tag{18}$$

If the posterior gets more peaked, which might be necessary for good reconstructions, the divergence quickly grows toward infinity. We observed such problems in particular with full-covariance posteriors (see Appendix F).

Despite these problems the KL divergence is theoretically sound. It was shown in (Hoffman & Johnson, 2016) that $D_{\text{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|p(\mathbf{z}))$ can be reformulated into two terms, one that weakly pushes toward overlapping per-sample posterior distributions and a KL divergence between the aggregated posterior and the prior. The latter is required if samples are drawn from the prior and the former prevents the latent encoding from becoming a lookup table (Mathieu et al., 2019). Replacing the KL divergence with the Wasserstein-2 metric preserves the tendency toward overlapping posteriors, but does not match the aggregated posterior to a predefined prior. However, a simple connection can be found to such models, see Appendix G. Nevertheless, this matching is not required in our setup due to the ex-post density estimation. Furthermore, successful practical approaches like Stable Diffusion (Rombach et al., 2022) only require correctly learning the manifold and therefore do not need a certain aggregated posterior to sample from.

We use the Wasserstein-2 metric between two Gaussian distributions. Mathematically, it can be written as

$$\begin{aligned} W_2(\mathcal{N}_1,\mathcal{N}_2) &= \|\boldsymbol{\mu}_\phi\|_2^2 + \text{tr}(\boldsymbol{\Sigma}_\phi) + n - 2\text{tr}(\boldsymbol{\Sigma}_\phi^{1/2}) \\ &= \|\boldsymbol{\mu}_\phi\|_2^2 + \text{tr}(\boldsymbol{\Sigma}_\phi) + n - 2\text{tr}(\boldsymbol{L}_\phi)\ , \end{aligned} \tag{19}$$

for $\mathcal{N}_1 = \mathcal{N}(\boldsymbol{\mu}_\phi, \boldsymbol{\Sigma}_\phi)$ and $\mathcal{N}_2 = \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. The last three terms can be reformulated into Eq. (11)

$$\begin{aligned} &\text{tr}(\boldsymbol{\Sigma}_\phi) + n - 2\text{tr}(\boldsymbol{L}_\phi) = \text{tr}(\boldsymbol{L}_\phi^T\boldsymbol{L}_\phi - 2\boldsymbol{L}_\phi + \boldsymbol{I}) = \\ &= \text{tr}((\boldsymbol{L}_\phi - \mathbf{I})^T(\boldsymbol{L}_\phi - \mathbf{I})) = \|\boldsymbol{L}_\phi - \mathbf{I}\|_F^2\ . \end{aligned} \tag{20}$$

Disregarding the constant terms, it is clear that Eq. (18) and Eq. (19) differ in the lack of the log term that infinitely penalizes zero-variance latents. In contrast, the Wasserstein metric even allows the posterior variance to approach zero if it helps to significantly reduce the reconstruction loss. This is evidenced in the aggregated posterior visualization of our model provided in Appendix H.

Naturally, the reduced reconstruction losses brought on by the *per-sample* Wasserstein metric in place of the KL divergence come at the cost of losing the ELBO formulation of the overall optimization problem. Furthermore, the Wasserstein distance between the *aggregated* posterior and the standard normal prior (Patrini et al., 2020) is not optimized either. Nevertheless, our empirical analysis shows that replacing the KL divergence with a Wasserstein metric regularization of the per-sample posterior results in significantly better reconstruction performance.

6. Results

In the following, we present quantitative and qualitative results of the UAE and its precursors compared to the VAE and RAE baselines on Fashion-MNIST (Xiao et al., 2017), CIFAR10 (Krizhevsky et al., 2009), and CelebA (Liu et al., 2015). We aim to delineate the effects of the UT (along with the reconstruction loss in Eq. (9), Wasserstein metric, and the decoder regularization. Furthermore, we investigate multi-sampling and various sigma-point heuristics in Appendix C and ablate the entire loss function from Eq. (8) in Appendix D. In addition to evaluating the reconstruction and sampling quality (using a mixture for all models, see Sec. 5.1), we investigate if sampling only at the sigmas in training preserves the latent space structure (e.g. does not create 'holes') by evaluating interpolated samples. The metric is the widely-used FID (Heusel et al., 2017), which quantifies the distance between two distributions of images. Detailed information about the network architecture, training, and the choice of FID datasets is given in Appendix A.

The main results are provided in Tab. 2. The table is divided into three parts: the first part shows the effects of applying the Unscented Transform to the vanilla VAE model; the second part shows the baseline results of the RAE, while the third part shows the results of Wasserstein metric models. In the UT-VAE row of Tab. 2, *we tweak the VAE sampling to select instances at the sigma points while averaging the resulting images in the reconstruction loss, as consistent with the definition in Eq. (5-6). This simple change brings a remarkable near 40% improvement on Fashion-MNIST on average, near 15% on CIFAR10, and near 30% on CelebA*. It provides strong evidence that a higher-quality, lower-variance representation of the posterior distribution results in higher-quality decoded images.

The deterministic baseline RAE model in Tab. 2 sets the context with a significantly higher performance than the vanilla VAE. The Wasserstein metric of the VAE*, which preserves the latent space regularization in spirit of the RAE but extends it to a probabilistic, non-constant variance setting, can be considered close to the non-regularized RAE: outperforms it on CIFAR10 while being behind on Fashion-MNIST and CelebA. More importantly, the VAE* model also achieves a large improvement over the classical VAE in all metrics and on all datasets, achieved effectively only by replacing the logarithm term with a linear term. This indicates that the rigidity of the KL divergence w.r.t. posterior variance potentially harms the quality of decoded samples, particularly on the richer CIFAR10 and CelebA.

Observing the UT-VAE* row in Tab. 2, it can be seen that the unscented transform (UT) sampling in the VAE* context gives a further, albeit lesser boost in most metrics than with the KL divergence. Due to the Wasserstein metric's ability to shrink the posterior variance while approaching convergence, the effect of any sampling is reduced. Nevertheless, it provides a considerable, approximately 10% boost on CelebA and Fashion-MNIST as well as a larger relative improvement with multiple samples than in VAE* (see Tab. 5, 6 in Appendix C). Finally, the generalized decoder regularization from Eq. (12) of the UAE applies a strong smoothing effect and further boosts the performance on CelebA and especially CIFAR10. Surprisingly, it yields a regression on Fashion-MNIST; similar effect of the gradient penalty harming the RAE performance compared to no-regularization is observable in (Ghosh et al., 2019) MNIST experiments. Overall, compared to the RAE, the UAE achieves significant improvements on CIFAR10 and a minor improvement on CelebA, while interestingly, the best model on Fashion-MNIST can be considered the UT-VAE.

In Tab. 3, we take a deeper look at the performance of the UT reconstruction loss term from Eq. (9). We empirically compare two strategies for designing the loss function: (i) use the mean reconstruction loss of images for each selected sample from the posterior (consistent with the standard VAE reconstruction loss) and (ii) apply the reconstruction loss to the mean image of samples from the posterior. Quantitative results in Tab. 3 consistently show the advantages of strategy (ii) for both the VAE and UT-VAE models using random samples and sigma points, respectively.

CelebA qualitative results are shown in Fig. 3 and reflect the FID scores: the UAE images appear similar to the RAE but significantly more realistic than the VAE. Fashion-MNIST and CIFAR10 images are provided in Appendix I.

7. Conclusion

In this paper, we introduced a novel VAE architecture employing the Unscented Transform, a lower-variance alternative to the reparameterization trick. We have challenged one of the core components of the VAE by showing that a sigma-point transform of the posterior significantly outperforms propagating random samples through the decoder. This was empirically shown for a small number of sigma points (2, 4, and 8) while taking more becomes impractical due to computationally-intensive training. Additionally, we proposed to use the Wasserstein metric, which does not optimize the ELBO. Although it can be considered as the main theoretical limitation of our model, it is a sound practical alternative to the KL divergence. By breaking its rigidity w.r.t. posterior variance, we unlocked performance improvements brought on by sharper posteriors that preserve a smooth latent space. Our work contributes an important step toward establishing competitive deterministic and deterministic-sampling generative models. Future work will thus focus on expanding the classes of supported generative models and on evaluation of further deterministic and quasi-deterministic sampling methods.

Table 2: Comparison of the architectures from Tab. 1. In all sampling instances, we select 8 random samples or sigma points. In the unscented transform models (UT-VAE, UT-VAE*, UAE), we select random sigma points on all datasets apart from CIFAR10, where pairs of sigma points along the largest eigenvalue axes are selected (see Appendix C). All RAE variants from (Ghosh et al., 2019) are provided: RAE-no-reg. without decoder regularization, RAE-GP with the Gradient Penalty (GP) from Eq. (16), RAE-L2 with decoder weight decay, and RAE-SN with spectral normalization.

	Fashion-MNIST			CIFAR10			CelebA		
	Rec.	Sample	Interp.	Rec.	Sample	Interp.	Rec.	Sample	Interp.
$\text{VAE}_{8\text{x}}$	44.29	48.73	61.99	110.0	120.6	118.3	65.86	68.53	68.75
$\text{UT-VAE}_{8\text{x}}$	27.79	**30.39**	**39.92**	91.04	111.7	104.3	50.11	54.15	54.32
RAE-no-reg.	21.56	34.79	50.27	86.79	102.1	96.80	40.79	47.88	49.97
RAE-GP	22.91	33.80	50.74	85.70	100.7	96.06	39.89	46.67	46.18
RAE-L2	**20.28**	32.06	48.52	84.27	99.26	94.23	38.78	46.44	50.33
RAE-SN	21.40	33.50	49.60	85.75	101.1	96.48	41.23	48.39	50.23
$\text{VAE*}_{8\text{x}}$	27.36	36.63	52.61	82.22	99.11	92.84	45.02	50.81	53.64
$\text{UT-VAE*}_{8\text{x}}$	23.64	31.51	48.06	81.12	100.6	93.80	40.18	47.39	49.62
$\text{UAE}_{8\text{x}}$	25.07	35.19	54.24	**71.97**	**89.91**	**83.50**	**38.48**	**45.60**	**45.88**

Table 3: Comparison of a VAE model using the reconstruction loss of the mean image of random samples from the posterior: $\|\mathbf{x} - \frac{1}{K}\sum_{k=1}^{K} D_\theta(\mathbf{z}_k)\|_2^2,\ \mathbf{z}_k = \boldsymbol{\mu}_\phi + \boldsymbol{\sigma}_\phi \odot \boldsymbol{\epsilon}_k,\ \boldsymbol{\epsilon}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, denoted by $\text{VAE}^{\dagger}_{2\text{x}}$, and a model with the mean reconstruction loss of sigma points from the posterior: $\frac{1}{K}\sum_{k=1}^{K} \|\mathbf{x} - D_\theta(\mathbf{z}_k)\|_2^2,\ \mathbf{z}_k \sim \{\boldsymbol{\chi}_i(\boldsymbol{\mu}_\phi, \text{diag}(\boldsymbol{\sigma}_\phi^2))\}_{i=0}^{2n}$, denoted by $\text{UT-VAE}^{\ddagger}_{2\text{x}}$. The $\text{UT-VAE}_{2\text{x}}$ uses the full unscented transform with the reconstruction loss of the mean image of sigma points from the posterior: $\|\mathbf{x} - \frac{1}{K}\sum_{k=1}^{K} D_\theta(\mathbf{z}_k)\|_2^2,\ \mathbf{z}_k \sim \{\boldsymbol{\chi}_i(\boldsymbol{\mu}_\phi, \text{diag}(\boldsymbol{\sigma}_\phi^2))\}_{i=0}^{2n}$, as consistent with the Unscented Transform in Eq. (5-6). In the sigma-point variants of $\text{UT-VAE}^{\ddagger}_{2\text{x}}$ and $\text{UT-VAE}_{2\text{x}}$, random sigma points are selected for Fashion-MNIST and CelebA, while largest-eigenvalue pairs are used in CIFAR10.

	Fashion-MNIST			CIFAR10			CelebA		
	Rec.	Sample	Interp.	Rec.	Sample	Interp.	Rec.	Sample	Interp.
$\text{VAE}_{2\text{x}}$	43.66	49.01	61.03	112.7	123.2	120.6	67.29	69.92	70.00
$\text{VAE}^{\dagger}_{2\text{x}}$	42.22	47.33	59.47	110.0	121.6	118.6	61.71	65.77	65.29
$\text{UT-VAE}^{\ddagger}_{2\text{x}}$	46.79	52.87	74.11	115.2	128.2	124.7	54.61	61.03	59.49
$\text{UT-VAE}_{2\text{x}}$	**36.25**	**40.30**	**53.10**	**95.70**	**115.4**	**107.3**	**51.61**	**57.42**	**56.56**

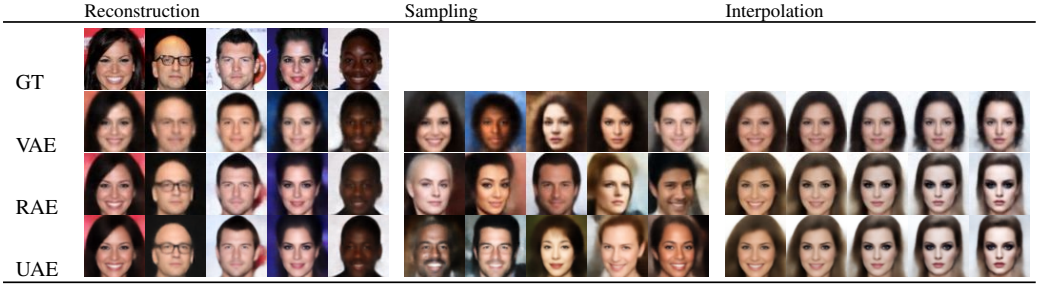


Figure 3: Qualitative results on the CelebA dataset of the $\text{VAE}_{8\text{x}}$, RAE-L2, and $\text{UAE}_{8\text{x}}$ models.

References

Accardi, L. De Finetti Theorem. *Hazewinkel, Michiel, Encyclopaedia of Mathematics, Kluwer Academic Publishers*, 2001.

Bauer, M. and Mnih, A. Resampled Priors for Variational Autoencoders. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 66–75. PMLR, 2019.

Bengio, Y., Courville, A. C., and Vincent, P. Representation Learning: A Review and New Perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(8):1798–1828, 2013. doi: 10.1109/TPAMI.2013.50. URL https://doi.org/10.1109/TPAMI.2013.50.

Bowman, S. R., Vilnis, L., Vinyals, O., Dai, A. M., Jozefowicz, R., and Bengio, S. Generating Sentences From a Continuous Space. *arXiv preprint arXiv:1511.06349*, 2015.

Burda, Y., Grosse, R. B., and Salakhutdinov, R. Importance Weighted Autoencoders. In Bengio, Y. and LeCun, Y. (eds.), *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016. URL http://arxiv.org/abs/1509.00519.

Chen, X., Kingma, D. P., Salimans, T., Duan, Y., Dhariwal, P., Schulman, J., Sutskever, I., and Abbeel, P. Variational Lossy Autoencoder. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=BysvGP5ee.

Cremer, C., Morris, Q., and Duvenaud, D. Reinterpreting Importance-Weighted Autoencoders. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=Syw2ZgrFx.

Dai, B. and Wipf, D. Diagnosing and Enhancing VAE Models. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=B1e0X3C9tQ.

Dai, B., Wang, Y., Aston, J., Hua, G., and Wipf, D. Connections with Robust PCA and the Role of Emergent Sparsity In Variational Autoencoder Models. *The Journal of Machine Learning Research*, 19(1):1573–1614, 2018.

Dai, B., Wang, Z., and Wipf, D. The Usual Suspects? Reassessing Blame for VAE Posterior Collapse. In *International Conference on Machine Learning*, pp. 2313–2322. PMLR, 2020.

Doucet, A. and Johansen, A. M. A Tutorial On Particle Filtering and Smoothing: Fifteen Years Later. *Oxford Handbook of Nonlinear Filtering*, 2011.

Ghosh, P., Sajjadi, M. S., Vergari, A., Black, M., and Schölkopf, B. From Variational to Deterministic Autoencoders. *arXiv preprint arXiv:1903.12436*, 2019.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. C., and Bengio, Y. Generative Adversarial Networks. *CoRR*, abs/1406.2661, 2014. URL http://arxiv.org/abs/1406.2661.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Klambauer, G., and Hochreiter, S. GANs Trained by a Two Time-scale Update Rule Converge to a Nash Equilibrium. *arXiv preprint arXiv:1706.08500*, 12(1), 2017.

Higgins, I., Matthey, L., Pal, A., Burgess, C. P., Glorot, X., Botvinick, M. M., Mohamed, S., and Lerchner, A. Beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=Sy2fzU9gl.

Hoffman, M. D. and Johnson, M. J. ELBO Surgery: Yet Another Way to Carve Up the Evidence Lower Bound. In *Proc. Workshop Adv. Approx. Bayesian Inference*, pp. 2, 2016.

Julier, S., Uhlmann, J., and Durrant-Whyte, H. F. A New Method for the Nonlinear Transformation of Means and Covariances In Filters and Estimators. *IEEE Transactions on automatic control*, 45(3):477–482, 2000.

Karl, M., Soelch, M., Bayer, J., and Van der Smagt, P. Deep Variational Bayes Filters: Unsupervised Learning of State Space Models From Raw Data. *arXiv preprint arXiv:1605.06432*, 2016.

Khromov, G. and Singh, S. P. Some fundamental aspects about lipschitz continuity of neural network functions, 2023.

Kingma, D. P. and Welling, M. Auto-encoding Variational Bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Kingma, D. P., Salimans, T., and Welling, M. Variational Dropout and the Local Reparameterization Trick. *Advances in neural information processing systems*, 28, 2015.

Krizhevsky, A., Hinton, G., et al. Learning Multiple Layers of Features From Tiny Images. 2009.

Kusner, M. J., Paige, B., and Hernández-Lobato, J. M. Grammar Variational Autoencoder. In Precup, D. and Teh, Y. W. (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1945–1954. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/kusner17a.html.

Liu, Z., Luo, P., Wang, X., and Tang, X. Deep Learning Face Attributes In the Wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

Lucas, J., Tucker, G., Grosse, R. B., and Norouzi, M. Understanding Posterior Collapse In Generative Latent Variable Models. In *Deep Generative Models for Highly Structured Data, ICLR 2019 Workshop, New Orleans, Louisiana, United States, May 6, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=r1xaVLUYuE.

Mathieu, E., Rainforth, T., Siddharth, N., and Teh, Y. W. Disentangling Disentanglement In Variational Autoencoders. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 4402–4412. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/mathieu19a.html.

Menegaz, H. M., Ishihara, J. Y., Borges, G. A., and Vargas, A. N. A Systematization of the Unscented Kalman Filter Theory. *IEEE Transactions on automatic control*, 60 (10):2583–2598, 2015.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. Pytorch: An Imperative Style, High-performance Deep Learning Library. *Advances in neural information processing systems*, 32, 2019.

Patrini, G., van den Berg, R., Forre, P., Carioni, M., Bhargav, S., Welling, M., Genewein, T., and Nielsen, F. Sinkhorn Autoencoders. In *Uncertainty in Artificial Intelligence*, pp. 733–743. PMLR, 2020.

Rainforth, T., Kosiorek, A., Le, T. A., Maddison, C., Igl, M., Wood, F., and Teh, Y. W. Tighter Variational Bounds Are Not Necessarily Better. In *International Conference on Machine Learning*, pp. 4277–4285. PMLR, 2018.

Rezende, D. J., Mohamed, S., and Wierstra, D. Stochastic Backpropagation and Approximate Inference In Deep Generative Models. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, pp. II–1278–II–1286. JMLR.org, 2014.

Rolinek, M., Zietlow, D., and Martius, G. Variational Autoencoders Pursue PCA Directions (by Accident). In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12406–12415, 2019.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution Image Synthesis with Latent Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695, 2022.

Seitzer, M. Pytorch-fid: FID Score for PyTorch. https://github.com/mseitzer/pytorch-fid, August 2020. Version 0.2.1.

Tolstikhin, I. O., Bousquet, O., Gelly, S., and Schölkopf, B. Wasserstein Auto-Encoders. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=HkL7n1-0b.

Townsend, J., Bird, T., and Barber, D. Practical Lossless Compression with Latent Variables Using Bits Back Coding. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=ryE98iR5tm.

Tripp, A., Daxberger, E. A., and Hernández-Lobato, J. M. Sample-Efficient Optimization In the Latent Space of Deep Generative Models Via Weighted Retraining. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

Tucker, G., Lawson, D., Gu, S., and Maddison, C. J. Doubly Reparameterized Gradient Estimators for Monte Carlo Objectives. *arXiv preprint arXiv:1810.04152*, 2018.

Uhlmann, J. *Dynamic Map Building and Localization: New Theoretical Foundations.* PhD thesis, University of Oxford, 1995.

Vahdat, A. and Kautz, J. NVAE: A Deep Hierarchical Variational Autoencoder. In *Neural Information Processing Systems (NeurIPS)*, 2020.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Xu, B., Wang, N., Chen, T., and Li, M. Empirical Evaluation of Rectified Activations In Convolutional Network. *arXiv preprint arXiv:1505.00853*, 2015.

Zhang, C., Bütepage, J., Kjellström, H., and Mandt, S. Advances In Variational Inference. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):2008–2026, 2018.

Zhang, W., Liu, M., and Zhao, Z.-g. Accuracy Analysis of Unscented Transformation of Several Sampling Strategies. In *2009 10th ACIS International Conference on Software Engineering, Artificial Intelligences, Networking and Parallel/Distributed Computing*, pp. 377–380. IEEE, 2009.

Zietlow, D., Rolinek, M., and Martius, G. Demystifying Inductive Biases for (Beta-) VAE Based Architectures. In *International Conference on Machine Learning*, pp. 12945–12954. PMLR, 2021.

Appendix

A. Network Architecture and Training

Table 4: Network architectures of the implemented VAE, RAE, and UAE models. Batch dimensions omitted for clarity.

VAE, UAE: $\mathbf{x}_{C\times W\times H} \rightarrow$ ENCODER $\rightarrow \{\text{FC}_{1024\times n} : \boldsymbol{\mu}_\phi, \text{FC}_{1024\times n} : \log \boldsymbol{\sigma}^2_\phi\} \rightarrow \mathbf{z} \rightarrow$ DECODER $\rightarrow \hat{\mathbf{x}}$
RAE: $\mathbf{x}_{C\times W\times H} \rightarrow$ ENCODER $\rightarrow \{\text{FC}_{1024\times n} : \mathbf{z}_\phi\} \rightarrow$ DECODER $\rightarrow \hat{\mathbf{x}}$
ENCODER: $\text{CONV}_{32\times 64} \rightarrow \text{CONV}_{64\times 128} \rightarrow \text{CONV}_{128\times 256} \rightarrow \text{CONV}_{256\times 512} \rightarrow \text{CONV}_{512\times 1024} \rightarrow$ FLATTEN
DECODER: $\text{FC}_{n\times 1024\cdot 8\cdot 8} \rightarrow \text{TCONV}_{1024\times 512} \rightarrow \text{TCONV}_{512\times 256}[\rightarrow \text{TCONV}_{256\times 128}]^{\text{CelebA}} \rightarrow \text{TCONV}_{256\text{ or }128\times C}$
MNIST: $C = 1, W = H = 32, n = 64$ CIFAR10: $C = 3, W = H = 32, n = 128$ CELEBA: $C = 3, W = H = 64, n = 64$

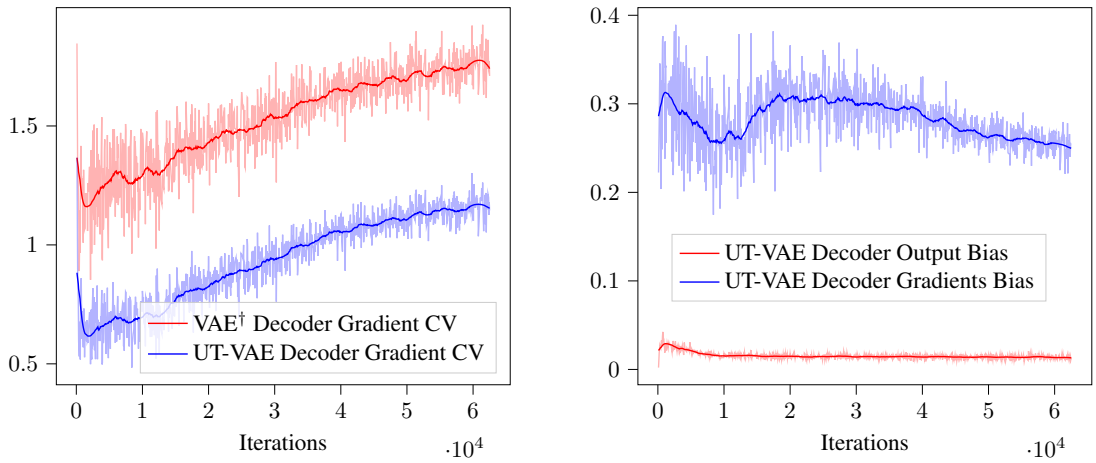
Network architectures are given in Tab. 4 and largely follow the architecture in (Ghosh et al., 2019). For consistency, all models share the same encoder/decoder structure. All encoder 2D convolution blocks contain 3×3 kernels, stride 2, and padding 1, followed by a 2D batch normalization and a Leaky-ReLU activation. The decoder transposed convolutions share the same parameters as the encoder convolutions apart from using a 4×4 kernel. The last transposed convolution (mapping to channel dimension) however has a 3×3 kernel and is followed by a $\tanh$ activation (without batch normalization).

The dataset preprocessing procedure is the following. The Fashion-MNIST images are scaled from 28×28 to 32×32. For the training dataset, we use 50k out of the 60k provided examples, leaving the remaining 10k for the validation dataset. For the test dataset, we use the provided examples. In CIFAR10, we perform a random horizontal flip on the training data followed by a normalization for all dataset subsets. We use the same training/validation/test split method as in Fashion-MNIST. In CelebA, we perform a 148×148 center crop and resize the images to 64×64. We use the provided training/validation/testing subsets.

All models are implemented in PyTorch (Paszke et al., 2019) and use the library provided in (Seitzer, 2020) for FID computation. The models are trained for 100 epochs, starting with a 0.005 learning rate that is then halved after every five epochs without improvement. The weights used in the loss functions are the following: KL-divergence (or the Wasserstein metric) terms are weighted with $\beta = 2.5e^{-4}$ in the case of VAE and UAE and $\beta = 1e^{-4}$ for the RAE. The decoder regularization terms are weighted with $\gamma = 1e^{-6}$ for both RAE and UAE. We performed minimal hyperparameter search over the weights.

In computing the FID scores, we follow the same procedure as in (Ghosh et al., 2019). In the three cases of reconstruction, sampling, and interpolation, we evaluate the FID to the test set image reconstructions as the ground-truth. In the reconstruction metric, we use the validation set image reconstructions. In sampling, we fit the training dataset latent features to a GMM (see Sec. 5.1) and sample and reconstruct the same number of elements as in the test set. In interpolation, we apply mid-point spherical interpolation between a random pair of validation set embeddings. In all cases, we generate a single image per input; this image corresponds to the posterior mean of the latent distribution. This mean latent feature vector is also used in sampling and interpolation while fitting a mixture ex-post or interpolating the latent space vectors. Thus, the resulting number of generated images for FID computation is the same regardless of the number of sigma points or samples used in training. In all experiments, the average FID score of three runs is reported, while observing a similar variation between scores of individual runs among the models employing the UT compared to the vanilla VAE. In contrast, the scores of RAE and VAE* modes were significantly more consistent.

The network architectures largely follow the structure adopted by (Ghosh et al., 2019), with the difference of the added first two encoder layers. Nevertheless, in Tab. 2, we did not manage to reproduce the FID values reported in (Ghosh et al., 2019) on CelebA and CIFAR10, even observing that removing the first two encoder layers reduces the overall performance. We suspect that it is due to the differing Tensorflow and PyTorch model implementations as well as the FID computation libraries. However, in most cases, our implementation of the RAE attains a significantly larger performance gain over the VAE than reported in (Ghosh et al., 2019).



(a) Median of the decoder gradient CV for UT-VAE and VAE†

(b) Median relative bias based on an estimate of the true gradient using 200 random samples

Figure 4: Comparison of the variance and bias trade-off for the VAE† (employing the decoder output mean instead of the sample mean, see Tab. 3) and UT-VAE across approx. 60k training steps (100 epochs) on the CIFAR10 dataset. The data is based on a single training of an UT-VAE where every 50th epoch the gradient variance and bias was estimated using different sampling schemes. In case of VAE†, two random points are sampled (in accordance with the reparameterization trick), while in case of UT-VAE, a single sigma point pair is sampled.

B. Gradient Variance and Bias

In this section, we investigate the gradient variance and bias of the proposed base UT-VAE model. Compared to random sampling of the reparameterization trick, using a different integration scheme like sampling sigma points can be biased. It can nevertheless achieve lower variance depending on the nonlinear function of the decoder. Thus, for our decoder setup, we compare the gradient variance and bias of the UT-VAE (with random sigma pair sampling) and the VAE† (with random sampling) employing the decoder output mean instead of the sample mean[6] (see Tab. 3 for a performance comparison) in order to isolate the effect of sampling sigma points.

We train both models and estimate the gradient variance and bias every 50th iteration. For UT-VAE we independently sample 50 sigma point pairs, pass them through the decoder, and calculate the gradients' mean m_j and standard deviation σ_j. For VAE† we draw 2 random samples 200 times and perform the same steps to obtain m'_j and σ'_j. We calculate the median Coefficient of Variation (CV) of the gradients for both models, assuming that m'_j computed with 200 random samples is a good enough estimate of the true gradient. Furthermore, we compute the median relative bias b_{rel} for the decoder gradients and output of the UT-VAE. The CV (for UT-VAE) and b_{rel} (for decoder gradients bias) are computed as follows

$$\mathrm{CV} = \mathrm{median}\left\{\frac{\sigma_j}{|m_j|}\right\} \qquad b_{rel} = \mathrm{median}\left\{\frac{|m_j - m'_j|}{|m'_j|}\right\}. \tag{21}$$

The gradient variance results are depicted in Fig. 4a. The variance of the sigma pair sampling of the UT-VAE is consistently lower than the gradient variance of the random sampling within VAE†. Interestingly, for the VAE† the standard deviation of the gradients is on average larger than the magnitude of the gradient during the whole training, whereas for the UT-VAE this is only the case at the end of the training. Fig. 4b shows the relative decoder output bias as well as the relative gradient bias of the UT-VAE at the same iterations. Whereas the relative bias at the decoder output is below 3% throughout the whole training, the bias of the gradients is around 30% of their magnitude. It is unclear whether such a substantial gradient bias is behind the good performance of the UT-VAE or if there is a performance trade-off between variance and bias. Nevertheless, our experiments show that, under a common decoder architecture, integration schemes like the UT can exhibit lower variance and higher bias while outperforming the standard VAE sampling scheme. Thus, investigating alternative integration schemes for VAEs can be a promising research direction.

[6] Reconstruction loss function of the VAE†: $\|\mathbf{x} - \frac{1}{K}\sum_{k=1}^{K} D_\theta(\mathbf{z}_k)\|_2^2,\ \mathbf{z}_k = \boldsymbol{\mu}_\phi + \boldsymbol{\sigma}_\phi \odot \boldsymbol{\epsilon}_k,\ \boldsymbol{\epsilon}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

Table 5: Analysis of the number of sampled sigma points and different heuristics, where the mean image of multiple sigma points is matched to the ground truth in the reconstruction loss. The three investigated heuristics are sampling random sigma points, random pairs of sigma points along an axis, and pairs of sigma points along axes with largest eigenvalues.

	Fashion-MNIST			CIFAR10			CelebA		
	Rec.	Samp.	Interp.	Rec.	Samp.	Interp.	Rec.	Samp.	Interp.
UT-VAE$_{1x,rand.}$	47.27	52.10	67.16	119.9	129.8	127.9	55.93	62.13	60.54
UT-VAE$_{2x,rand.}$	36.25	40.30	53.10	111.5	124.7	121.0	51.61	57.42	56.56
UT-VAE$_{4x,rand.}$	32.13	36.41	47.30	105.9	119.8	115.9	50.85	55.82	55.99
UT-VAE$_{8x,rand.}$	27.79	**30.39**	**39.92**	95.40	110.8	106.4	50.11	54.15	44.32
UT-VAE*$_{2x,rand.}$	28.26	36.36	50.69	85.88	103.7	96.90	44.32	50.33	52.40
UT-VAE*$_{4x,rand.}$	24.38	32.75	49.40	81.99	100.6	93.52	42.52	49.21	51.35
UT-VAE*$_{8x,rand.}$	**23.64**	31.51	48.06	81.10	99.87	92.48	**40.18**	**47.39**	**49.62**
UT-VAE$_{2x,rand.\ pairs}$	102.1	115.1	112.8	102.3	119.6	114.0	150.0	150.4	151.3
UT-VAE$_{4x,rand.\ pairs}$	96.85	110.1	107.3	101.0	119.5	113.4	224.3	225.0	225.4
UT-VAE$_{8x,rand.\ pairs}$	90.14	103.6	101.5	100.3	119.2	113.2	173.2	175.4	175.8
UT-VAE*$_{2x,rand.\ pairs}$	32.66	38.68	58.72	85.64	102.3	97.00	45.96	53.16	51.49
UT-VAE*$_{4x,rand.\ pairs}$	32.85	38.58	57.70	84.62	102.2	96.14	252.9	254.8	253.8
UT-VAE*$_{8x,rand.\ pairs}$	30.65	36.88	56.42	**80.51**	**98.40**	**91.96**	141.9	144.3	147.4
UT-VAE$_{2x,larg.\ \lambda\ pairs}$	106.6	118.6	115.7	95.70	115.4	107.3	54.02	60.29	60.26
UT-VAE$_{4x,larg.\ \lambda\ pairs}$	108.3	120.1	117.2	92.56	111.6	104.2	46.37	53.53	52.62
UT-VAE$_{8x,larg.\ \lambda\ pairs}$	115.5	128.8	126.3	91.04	111.7	104.3	48.59	55.22	55.29
UT-VAE*$_{2x,larg.\ \lambda\ pairs}$	33.49	42.63	61.57	82.17	100.7	93.80	55.57	61.42	61.53
UT-VAE*$_{4x,larg.\ \lambda\ pairs}$	34.94	43.18	67.65	81.61	101.3	94.11	48.41	54.70	54.80
UT-VAE*$_{8x,larg.\ \lambda\ pairs}$	31.08	41.06	64.58	81.12	100.6	93.80	45.08	51.45	52.05

C. Additional Results: Multi-Sigma Heuristics and Multi-Sample Models

The UT-VAE loss function defined in Tab. 1 samples K sigma points in the reconstruction term. Increasing the number of sigma points (up to $2n + 1$) improves the estimate of the transformed posterior distribution and thus the resulting reconstruction quality, at the expense of an approximately linear increase in training time. We observed this in most cases when training on 2, 4, and 8 sigma points, see Tab. 5. However, a much larger number of sigma points might not result in expected additional performance improvement due to significantly larger batch size, which could be mitigated by constructing approaches to select and train on a fixed, smaller batch size.

For K selected sigma points, various strategies can be used instead of sampling a discrete uniform distribution. For example, only pairs of sigma points along an axis can be chosen, conveying the width of the posterior distribution in the given dimension. This strategy can be adapted to select pairs along axes with largest eigenvalues. Tab. 5 also explores different sampling heuristics in the case of UT-VAE and UT-VAE*. We have observed that models trained with KL divergence exhibit larger variation in results w.r.t. the sampling heuristic, which is reasonable since the Wasserstein metric's posterior variance suppression diminishes the effect of sampling. The choice of the sigma-point selection heuristic turns out to have a large effect on the overall performance given a dataset. We have observed that a random selection of sigma points performs consistently well across all datasets while selecting random pairs generates reasonable results only in the case of CIFAR10. Interestingly, random-pairs performs very poorly on Fashion-MNIST and CelebA while largest eigenvalue pairs shows very good performance in the UT-VAE case on CIFAR10. In the main experiments of Tab. 2, we used a random selection for the Fashion-MNIST and CelebA models and largest-eigenvalue pairs for CIFAR10, due to its superior performance in the UT-VAE case.

Tab. 6 analyzes models using multiple samples in training. We compare the VAE* and the UAE with the classical VAE and the IWAE (Burda et al., 2016) as a baseline where multiple importance-weighted posterior samples help achieve a tighter lower bound. Observing the results, it is clear that models employing the Wasserstein metric can benefit from increasing the number of samples in training despite their ability to reduce the latent space variance, while significantly outperforming the baselines.

Table 6: Comparison of models employing multiple samples in training. The UAE uses random sigma points on Fashion-MNIST and CelebA and largest-eigenvalue pairs on CIFAR10.

	Fashion-MNIST			CIFAR10			CelebA		
	Rec.	Sample	Interp.	Rec.	Sample	Interp.	Rec.	Sample	Interp.
VAE_{1x}	45.64	49.99	61.33	116.4	126.8	124.2	68.32	71.05	71.16
VAE_{2x}	43.66	49.01	61.03	112.7	123.2	120.6	67.29	69.92	70.00
VAE_{4x}	44.94	49.51	62.29	111.7	121.3	119.5	66.32	68.87	69.06
VAE_{8x}	44.29	48.73	61.99	110.0	120.6	118.3	65.86	68.53	68.75
$IWAE_{1x}$	49.27	53.71	64.50	111.7	121.6	119.6	68.28	71.16	71.17
$IWAE_{2x}$	48.21	53.11	65.69	112.1	122.4	119.8	66.85	69.81	69.74
$IWAE_{4x}$	47.40	51.77	64.10	110.6	120.6	118.2	66.01	68.82	68.90
$IWAE_{8x}$	46.16	50.91	63.68	108.9	118.9	116.9	64.83	67.96	67.86
VAE^*_{1x}	31.62	38.44	52.33	83.49	101.5	94.56	44.69	50.55	53.18
VAE^*_{2x}	30.07	37.92	**52.15**	84.57	102.2	95.61	45.18	50.97	53.73
VAE^*_{4x}	28.98	41.35	52.17	84.64	102.3	95.96	45.03	50.59	53.32
VAE^*_{8x}	27.36	36.63	52.61	82.22	99.11	92.84	45.02	50.81	53.64
UAE_{2x}	29.29	37.59	53.69	77.71	96.37	89.71	40.07	47.28	50.51
UAE_{4x}	27.11	38.03	53.11	75.63	93.02	86.41	39.48	46.35	50.94
UAE_{8x}	**25.07**	**35.19**	54.24	**71.97**	**89.91**	**83.50**	**38.48**	**45.60**	**45.88**

D. Additional Results: Ablation Study of the Loss Components

This section provides an additional ablation study of the loss components used in the UAE model. The loss functions considered are provided in the upper half of Tab. 7 and the obtained results are in Tab. 8. There are three dimensions along which the results can be interpreted: Wasserstein metric, unscented transform, and the generalized decoder regularization (gradient penalty).

Tab. 8 is divided into two parts: the top part models use the analytical form of the KL divergence in Eq. (10) while the bottom part use the Frobenius norm mismatch derived from the Wasserstein metric in Eq. (11). It is clearly visible that the latter models strongly outperform the former, in all datasets and configurations. The loss function allows for a sharper posterior and thus larger expressiveness of the model (see Appendix H).

Similarly, the unscented transform models UT-VAE and UT-VAE* clearly outperform the random sampling and per-sample reconstruction counterparts of VAE and VAE*. In the latter case, the differences are smaller due to the sharper posterior of the VAE*. An ablation study of the unscented transform components can be found in Tab. 3.

Considering the gradient penalty models, interesting interplays can be noticed. Applying the decoder regularization on the vanilla VAE and the VAE* (this model can be considered closest to the RAE-GP) brings only minor improvements in the case of CIFAR10 and CelebA for each of the models respectively. The strong smoothing of the latent space however seems detrimental when combined with the unscented transform and the KL divergence training. One can conclude that only the latent space regularization models (such as the Wasserstein metric VAE* or the deterministic RAE) can benefit from decoder regularization. Furthermore, the effect appears to be dataset-dependent since the Fashion-MNIST VAE* and UT-VAE* slightly regress when augmented with decoder regularization.

Table 7: The loss functions used for the models in Tab. 8 and Tab. 9. The upper and lower half of the table contain diagonal and full-covariance posterior models, respectively.

	Loss function	Posterior sampling
$\mathcal{L}_{\text{VAE}}$	$\frac{1}{K}\sum_{k=1}^{K}\|\mathbf{x}-D_\theta(\mathbf{z}_k)\|_2^2+\|\boldsymbol{\mu}_\phi\|_2^2-n+\sum_i\sigma_{\phi,i}^2-2\log\sigma_{\phi,i}$	$\mathbf{z}_k=\boldsymbol{\mu}_\phi+\boldsymbol{\sigma}_\phi\odot\boldsymbol{\epsilon}_k,\ \boldsymbol{\epsilon}_k\sim\mathcal{N}(\mathbf{0},\mathbf{I})$
$\mathcal{L}_{\text{VAE-GP}}$	$\frac{1}{K}\sum_{k=1}^{K}\|\mathbf{x}-D_\theta(\mathbf{z}_k)\|_2^2+\|\boldsymbol{\mu}_\phi\|_2^2+\sum_i\sigma_{\phi,i}^2-2\log\sigma_{\phi,i}+\max(\boldsymbol{\sigma}_\phi)\|\nabla_{\boldsymbol{\mu}_\phi}D_\theta(\boldsymbol{\mu}_\phi)\|_2^2$	$\mathbf{z}_k=\boldsymbol{\mu}_\phi+\boldsymbol{\sigma}_\phi\odot\boldsymbol{\epsilon}_k,\ \boldsymbol{\epsilon}_k\sim\mathcal{N}(\mathbf{0},\mathbf{I})$
$\mathcal{L}_{\text{UT-VAE}}$	$\|\mathbf{x}-\frac{1}{K}\sum_{k=1}^{K}D_\theta(\mathbf{z}_k)\|_2^2+\|\boldsymbol{\mu}_\phi\|_2^2+\sum_i\sigma_{\phi,i}^2-2\log\sigma_{\phi,i}$	$\mathbf{z}_k\sim\{\boldsymbol{\chi}_i(\boldsymbol{\mu}_\phi,\text{diag}(\boldsymbol{\sigma}_\phi^2))\}_{i=0}^{2n}$
$\mathcal{L}_{\text{UT-VAE-GP}}$	$\|\mathbf{x}-\frac{1}{K}\sum_{k=1}^{K}D_\theta(\mathbf{z}_k)\|_2^2+\|\boldsymbol{\mu}_\phi\|_2^2+\sum_i\sigma_{\phi,i}^2-2\log\sigma_{\phi,i}+\max(\boldsymbol{\sigma}_\phi)\|\nabla_{\boldsymbol{\mu}_\phi}D_\theta(\boldsymbol{\mu}_\phi)\|_2^2$	$\mathbf{z}_k\sim\{\boldsymbol{\chi}_i(\boldsymbol{\mu}_\phi,\text{diag}(\boldsymbol{\sigma}_\phi^2))\}_{i=0}^{2n}$
$\mathcal{L}_{\text{VAE*}}$	$\frac{1}{K}\sum_{k=1}^{K}\|\mathbf{x}-D_\theta(\mathbf{z}_k)\|_2^2+\|\boldsymbol{\mu}_\phi\|_2^2+\|\text{diag}(\boldsymbol{\sigma}_\phi^2)-\mathbf{I}\|_F^2$	$\mathbf{z}_k=\boldsymbol{\mu}_\phi+\boldsymbol{\sigma}_\phi\odot\boldsymbol{\epsilon}_k,\ \boldsymbol{\epsilon}_k\sim\mathcal{N}(\mathbf{0},\mathbf{I})$
$\mathcal{L}_{\text{VAE*-GP}}$	$\frac{1}{K}\sum_{k=1}^{K}\|\mathbf{x}-D_\theta(\mathbf{z}_k)\|_2^2+\|\boldsymbol{\mu}_\phi\|_2^2+\|\text{diag}(\boldsymbol{\sigma}_\phi^2)-\mathbf{I}\|_F^2+\max(\boldsymbol{\sigma}_\phi)\|\nabla_{\boldsymbol{\mu}_\phi}D_\theta(\boldsymbol{\mu}_\phi)\|_2^2$	$\mathbf{z}_k=\boldsymbol{\mu}_\phi+\boldsymbol{\sigma}_\phi\odot\boldsymbol{\epsilon}_k,\ \boldsymbol{\epsilon}_k\sim\mathcal{N}(\mathbf{0},\mathbf{I})$
$\mathcal{L}_{\text{UT-VAE*}}$	$\|\mathbf{x}-\frac{1}{K}\sum_{k=1}^{K}D_\theta(\mathbf{z}_k)\|_2^2+\|\boldsymbol{\mu}_\phi\|_2^2+\|\text{diag}(\boldsymbol{\sigma}_\phi^2)-\mathbf{I}\|_F^2$	$\mathbf{z}_k\sim\{\boldsymbol{\chi}_i(\boldsymbol{\mu}_\phi,\text{diag}(\boldsymbol{\sigma}_\phi^2))\}_{i=0}^{2n}$
$\mathcal{L}_{\text{UT-VAE*-GP}}$	$\|\mathbf{x}-\frac{1}{K}\sum_{k=1}^{K}D_\theta(\mathbf{z}_k)\|_2^2+\|\boldsymbol{\mu}_\phi\|_2^2+\|\text{diag}(\boldsymbol{\sigma}_\phi^2)-\mathbf{I}\|_F^2+\max(\boldsymbol{\sigma}_\phi)\|\nabla_{\boldsymbol{\mu}_\phi}D_\theta(\boldsymbol{\mu}_\phi)\|_2^2$	$\mathbf{z}_k\sim\{\boldsymbol{\chi}_i(\boldsymbol{\mu}_\phi,\text{diag}(\boldsymbol{\sigma}_\phi^2))\}_{i=0}^{2n}$
$\mathcal{L}_{\text{VAE-full}\boldsymbol{\Sigma}_\phi}$	$\frac{1}{K}\sum_{k=1}^{K}\|\mathbf{x}-D_\theta(\mathbf{z}_k)\|_2^2+\|\boldsymbol{\mu}_\phi\|_2^2+\text{tr}(\boldsymbol{\Sigma}_\phi)-2\text{tr}(\log\boldsymbol{L}_\phi)$	$\mathbf{z}_k=\boldsymbol{\mu}_\phi+\boldsymbol{L}_\phi\boldsymbol{\epsilon}_k,\ \ \boldsymbol{\epsilon}_k\sim\mathcal{N}(\mathbf{0},\mathbf{I})$
$\mathcal{L}_{\text{VAE-full}\boldsymbol{\Sigma}_\phi\text{-GP}}$	$\frac{1}{K}\sum_{k=1}^{K}\|\mathbf{x}-D_\theta(\mathbf{z}_k)\|_2^2+\|\boldsymbol{\mu}_\phi\|_2^2+\text{tr}(\boldsymbol{\Sigma}_\phi)-2\text{tr}(\log\boldsymbol{L}_\phi)+\lambda_{\max}(\boldsymbol{\Sigma}_\phi)\|\nabla_{\boldsymbol{\mu}_\phi}D_\theta(\boldsymbol{\mu}_\phi)\|_2^2$	$\mathbf{z}_k=\boldsymbol{\mu}_\phi+\boldsymbol{L}_\phi\boldsymbol{\epsilon}_k,\ \ \boldsymbol{\epsilon}_k\sim\mathcal{N}(\mathbf{0},\mathbf{I})$
$\mathcal{L}_{\text{UT-VAE-full}\boldsymbol{\Sigma}_\phi}$	$\|\mathbf{x}-\frac{1}{K}\sum_{k=1}^{K}D_\theta(\mathbf{z}_k)\|_2^2+\|\boldsymbol{\mu}_\phi\|_2^2+\text{tr}(\boldsymbol{\Sigma}_\phi)-2\text{tr}(\log\boldsymbol{L}_\phi)$	$\mathbf{z}_k\sim\{\boldsymbol{\chi}_i(\boldsymbol{\mu}_\phi,\boldsymbol{\Sigma}_\phi)\}_{i=0}^{2n}$
$\mathcal{L}_{\text{UT-VAE-full}\boldsymbol{\Sigma}_\phi\text{-GP}}$	$\|\mathbf{x}-\frac{1}{K}\sum_{k=1}^{K}D_\theta(\mathbf{z}_k)\|_2^2+\|\boldsymbol{\mu}_\phi\|_2^2+\text{tr}(\boldsymbol{\Sigma}_\phi)-2\text{tr}(\log\boldsymbol{L}_\phi)+\lambda_{\max}(\boldsymbol{\Sigma}_\phi)\|\nabla_{\boldsymbol{\mu}_\phi}D_\theta(\boldsymbol{\mu}_\phi)\|_2^2$	$\mathbf{z}_k\sim\{\boldsymbol{\chi}_i(\boldsymbol{\mu}_\phi,\boldsymbol{\Sigma}_\phi)\}_{i=0}^{2n}$
$\mathcal{L}_{\text{VAE*-full}\boldsymbol{\Sigma}_\phi}$	$\frac{1}{K}\sum_{k=1}^{K}\|\mathbf{x}-D_\theta(\mathbf{z}_k)\|_2^2+\|\boldsymbol{\mu}_\phi\|_2^2+\|\boldsymbol{L}_\phi-\mathbf{I}\|_F^2$	$\mathbf{z}_k=\boldsymbol{\mu}_\phi+\boldsymbol{L}_\phi\boldsymbol{\epsilon}_k,\ \ \boldsymbol{\epsilon}_k\sim\mathcal{N}(\mathbf{0},\mathbf{I})$
$\mathcal{L}_{\text{VAE*-full}\boldsymbol{\Sigma}_\phi\text{-GP}}$	$\frac{1}{K}\sum_{k=1}^{K}\|\mathbf{x}-D_\theta(\mathbf{z}_k)\|_2^2+\|\boldsymbol{\mu}_\phi\|_2^2+\|\boldsymbol{L}_\phi-\mathbf{I}\|_F^2+\lambda_{\max}(\boldsymbol{\Sigma}_\phi)\|\nabla_{\boldsymbol{\mu}_\phi}D_\theta(\boldsymbol{\mu}_\phi)\|_2^2$	$\mathbf{z}_k=\boldsymbol{\mu}_\phi+\boldsymbol{L}_\phi\boldsymbol{\epsilon}_k,\ \ \boldsymbol{\epsilon}_k\sim\mathcal{N}(\mathbf{0},\mathbf{I})$
$\mathcal{L}_{\text{UT-VAE*-full}\boldsymbol{\Sigma}_\phi}$	$\|\mathbf{x}-\frac{1}{K}\sum_{k=1}^{K}D_\theta(\mathbf{z}_k)\|_2^2+\|\boldsymbol{\mu}_\phi\|_2^2+\|\boldsymbol{L}_\phi-\mathbf{I}\|_F^2$	$\mathbf{z}_k\sim\{\boldsymbol{\chi}_i(\boldsymbol{\mu}_\phi,\boldsymbol{\Sigma}_\phi)\}_{i=0}^{2n}$
$\mathcal{L}_{\text{UT-VAE*-full}\boldsymbol{\Sigma}_\phi\text{-GP}}$	$\|\mathbf{x}-\frac{1}{K}\sum_{k=1}^{K}D_\theta(\mathbf{z}_k)\|_2^2+\|\boldsymbol{\mu}_\phi\|_2^2+\|\boldsymbol{L}_\phi-\mathbf{I}\|_F^2+\lambda_{\max}(\boldsymbol{\Sigma}_\phi)\|\nabla_{\boldsymbol{\mu}_\phi}D_\theta(\boldsymbol{\mu}_\phi)\|_2^2$	$\mathbf{z}_k\sim\{\boldsymbol{\chi}_i(\boldsymbol{\mu}_\phi,\boldsymbol{\Sigma}_\phi)\}_{i=0}^{2n}$

Table 8: Full ablation study of the models between the VAE and UAE (in the UT-VAE*-GP row), using the Wasserstein metric denoted by *, unscented transform (UT), and the decoder gradient penalty (GP) components. See the upper half Tab. 7 for the loss function definitions.

	Fashion-MNIST			CIFAR10			CelebA		
	Rec.	Sample	Interp.	Rec.	Sample	Interp.	Rec.	Sample	Interp.
VAE_{2x}	43.66	49.01	61.03	112.7	123.2	120.6	67.29	69.92	70.00
VAE-GP_{2x}	44.17	48.63	59.58	108.9	120.3	117.5	66.94	70.16	69.77
UT-VAE_{2x}	36.25	40.30	53.10	95.70	115.4	107.4	51.61	57.42	56.56
UT-VAE-GP_{2x}	47.77	65.24	72.43	102.6	118.6	113.1	100.4	102.2	100.3
VAE*_{2x}	30.07	37.92	52.15	84.57	102.2	95.61	45.18	50.97	53.73
VAE*-GP_{2x}	29.40	38.53	53.88	85.19	103.7	96.66	41.69	48.77	51.29
UT-VAE*_{2x}	**28.26**	**36.36**	**50.69**	82.17	100.7	93.80	44.32	50.33	52.40
UT-VAE* -GP_{2x}	29.29	37.59	53.69	**77.71**	**96.37**	**89.71**	**40.07**	**47.28**	**50.51**

E. ELBO Constraint Derivation

In this section, we complete the derivation of the constraint in Eq. (14) to the reformulated version in Eq. (15). The constraint in Eq. (14) can be bounded by the maximum of the decoder output in a single dimension i, multiplied by the number of dimensions

$$\|D_\theta(\mathbf{z}_1) - D_\theta(\mathbf{z}_2)\|_p \leq \dim(\mathbf{x}) \cdot \sup_i \{\|d_i(\mathbf{z}_1) - d_i(\mathbf{z}_2)\|_p\} < \epsilon \,. \tag{22}$$

Using the mean value theorem, the term $\sup_i\{\|d_i(\mathbf{z}_1) - d_i(\mathbf{z}_2)\|_p\}$ can be reduced to

$$\sup_i \{\|\nabla_t d_i((1-t)\mathbf{z}_1 + t\mathbf{z}_2)\|_p \cdot \|\mathbf{z}_1 - \mathbf{z}_2\|_p\} < \epsilon \,, \tag{23}$$

Since $\mathbf{z}_1$ and $\mathbf{z}_2$ are arbitrary, the first part can be simplified and generalized over all dimensions while separating the overall product using the Cauchy-Schwarz inequality

$$\sup_i \{\|\nabla_\mathbf{z} d_i(\mathbf{z})\|_p \cdot \|\mathbf{z}_1 - \mathbf{z}_2\|_p\} < \epsilon \tag{24}$$

$$\sup\{\|\nabla_\mathbf{z} D_\theta(\mathbf{z})\|_p\} \cdot \sup\{\|\mathbf{z}_1 - \mathbf{z}_2\|_p\} < \epsilon \,, \tag{25}$$

obtaining the form in Eq. (15).

F. Full-Covariance Posterior

In this section, we aim to investigate the performance of full-covariance posterior models. The non-diagonal posterior representation is naturally supported by the unscented transform and common in filtering. However, it is seldom in VAEs – one of the key ingredients of the standard VAE model is its diagonal Gaussian posterior approximation. The induced orthogonality can implicitly have positive effects on the structure of the latent space and the decoder (Zietlow et al., 2021; Rolinek et al., 2019), but such effects highly depend on implicit biases present in the dataset (Zietlow et al., 2021). Furthermore, the diagonal posterior together with the KL regularization allows for pruning unnecessary latent dimensions, also known as desired posterior collapse (Dai et al., 2020). A full-covariance posterior does not have such implicit biases and pruning properties, but it can have a positive effect on the optimization of the variational objective, as it connects otherwise disconnected global optima (Dai et al., 2018). Furthermore, it allows for modeling correlations in the posterior. We are not aware of a work successfully employing a full-covariance posterior.

The full-covariance representation can be practically realized by predicting n-dimensional standard deviations $\boldsymbol{\sigma}_\phi$ as well as $n(n-1)/2$-dimensional correlation factors $\boldsymbol{r}_\phi$ (followed by a $\tanh$ projection into the valid $[-1, 1]$ range), and building the lower triangular covariance matrix[7] $\boldsymbol{L}_\phi$. In this way, the full-covariance matrix $\boldsymbol{\Sigma}_\phi = \boldsymbol{L}_\phi \boldsymbol{L}_\phi^T$ is ensured to be symmetric and positive semi-definite.

The results of the full-covariance models are shown in the bottom half of Tab. 9. In all KL divergence instances, the performance of the models regresses significantly compared to their counterparts in Tab. 8. This indicates that, despite its theoretical potential to connect disconnected global optima of the optimization objective, a non-diagonal latent space is nevertheless difficult to train with KL divergence, regardless of the sampling method. However, the Wasserstein metric models receive a surprising performance boost. In some cases, they significantly outperform the models from Tab. 8 on Fashion-MNIST and CelebA while achieving similar results on CIFAR10, which has less structure in its input data. It is evident that the Wasserstein metric and potentially its lower posterior variance can enable a successful utilization of correlations in the posterior.

[7] In the 3-dimensional case: $\boldsymbol{L}_\phi = [\sigma_1 \quad 0 \quad 0; \quad r_1\sigma_2\sigma_1 \quad \sigma_2 \quad 0; \quad r_2\sigma_3\sigma_1 \quad r_3\sigma_3\sigma_2 \quad \sigma_3]$.

Table 9: Ablation study of the models in Tab. 8 in a full-covariance setting. See Tab. 7 for the loss function definitions.

	Fashion-MNIST			CIFAR10			CelebA		
	Rec.	Sample	Interp.	Rec.	Sample	Interp.	Rec.	Sample	Interp.
VAE-full $\mathbf{\Sigma}_{\phi 2\text{x}}$	79.01	83.15	91.01	123.8	132.6	130.2	99.72	100.9	99.96
VAE-full $\mathbf{\Sigma}_{\phi}$-GP$_{2\text{x}}$	180.0	181.5	184.4	158.3	165.8	164.0	244.2	244.6	241.8
UT-VAE-full $\mathbf{\Sigma}_{\phi 2\text{x}}$	57.93	58.87	64.86	129.6	141.2	138.2	132.1	132.4	136.0
UT-VAE-full $\mathbf{\Sigma}_{\phi}$-GP$_{2\text{x}}$	133.6	136.7	136.9	208.9	217.7	212.2	303.5	304.5	303.3
VAE*-full $\mathbf{\Sigma}_{\phi 2\text{x}}$	31.16	40.99	54.73	85.47	103.9	96.55	42.07	48.59	50.72
VAE*-full $\mathbf{\Sigma}_{\phi}$-GP$_{2\text{x}}$	**19.86**	**32.71**	48.84	84.19	102.9	95.63	39.69	46.76	49.70
UT-VAE*-full $\mathbf{\Sigma}_{\phi 2\text{x}}$	21.96	34.17	**48.32**	**79.51**	**98.32**	**91.82**	41.54	48.32	50.29
UT-VAE*-full $\mathbf{\Sigma}_{\phi}$-GP$_{2\text{x}}$	24.37	34.43	51.58	82.15	100.9	94.65	**39.48**	**46.60**	**48.97**

G. Connection to Wasserstein Autoencoders

Wasserstein-distance autoencoders (Patrini et al., 2020; Tolstikhin et al., 2018) use the Wasserstein distance $W_p(q_{\text{agg}}(\mathbf{z}), p(\mathbf{z}))$ to regularize the aggregated posterior $q_{\text{agg}}(\mathbf{z})$ toward the prior $p(\mathbf{z}) = \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. Instead, we use the Wasserstein distance as a simple regularization of the per-sample posterior. However, there is a simple connection of our posterior regularization to the aggregated posterior regularization. Assuming standard normal posteriors, the aggregated posterior can be represented as a mixture

$$q_{\text{agg}}(\mathbf{z}) = \frac{1}{N}\sum_n q(\mathbf{z}|\mathbf{x}_n) = \frac{1}{N}\sum_n \mathcal{N}(\boldsymbol{\mu}_n, \mathbf{\Sigma}_n). \tag{26}$$

In the one-dimensional case (generalizable to multiple dimensions) the mean and variance of the mixture are

$$\mathcal{N}(\mu_n, \sigma_n^2) \overset{i.d.}{=} \mathcal{N}\left(\frac{1}{N}\sum_n \mu_n, \frac{1}{N}\sum_n \left(\sigma_n^2 + \mu_n^2\right) - \left(\frac{1}{N}\sum_n \mu_n\right)^2\right) . \tag{27}$$

Thus, the aggregated posterior Wasserstein metric can be represented as

$$\begin{aligned} W_2(q_{\text{agg}}(\mathbf{z}), p(\mathbf{z})) &= \left(\frac{1}{N}\sum_n \mu_n\right)^2 + \frac{1}{N}\sum_n \left(\sigma_n^2 + \mu_n^2\right) - \left(\frac{1}{N}\sum_n \mu_n\right)^2 - 2\sqrt{\frac{1}{N}\sum_n \left(\sigma_n^2 + \mu_n^2\right) - \left(\frac{1}{N}\sum_n \mu_n\right)^2} = \\ &= \frac{1}{N}\sum_n \left(\sigma_n^2 + \mu_n^2\right) - 2\sqrt{\frac{1}{N}\sum_n \left(\sigma_n^2 + \mu_n^2\right) - \left(\frac{1}{N}\sum_n \mu_n\right)^2} , \end{aligned} \tag{28}$$

in the case $p = 2$ and while discarding constants. Similarly, the average per-sample posterior metric is

$$\frac{1}{N}\sum_n W_2(q_{\text{pp}}(\mathbf{z}|\mathbf{x}), p(\mathbf{z})) = \frac{1}{N}\sum_n \left(\mu_n^2 + \sigma_n^2 - 2\sigma_n\right) = \frac{1}{N}\sum_n \mu_n^2 + \frac{1}{N}\sum_n \sigma_n^2 - 2\frac{1}{N}\sum_n \sigma_n \, . \tag{29}$$

Table 10: Comparison of the Wasserstein autoencoder that utilizes the aggregated posterior Wasserstein metric, and the VAE*, utilizing the per-sample posterior Wasserstein metric in the loss.

	Fashion-MNIST			CIFAR10			CelebA		
	Rec.	Sample	Interp.	Rec.	Sample	Interp.	Rec.	Sample	Interp.
WAE-MMD	47.58	62.44	73.94	88.31	**100.35**	94.78	67.54	75.92	73.21
VAE*$_{1x}$	**31.62**	**38.44**	**52.33**	**83.49**	101.5	**94.56**	**44.69**	**50.55**	**53.18**

Comparing the aggregated posterior metric with the average per-sample posterior metric yields

$$\frac{1}{N}\sum_n \left(\sigma_n^2+\mu_n^2\right) - 2\sqrt{\frac{1}{N}\sum_n \left(\sigma_n^2+\mu_n^2\right) - \left(\frac{1}{N}\sum_n \mu_n\right)^2} \leq \frac{1}{N}\sum_n \mu_n^2 + \frac{1}{N}\sum_n \sigma_n^2 - 2\frac{1}{N}\sum_n \sigma_n \tag{30}$$

$$-2\sqrt{\frac{1}{N}\sum_n \left(\sigma_n^2+\mu_n^2\right) - \left(\frac{1}{N}\sum_n \mu_n\right)^2} \leq -2\frac{1}{N}\sum_n \sigma_n \tag{31}$$

$$\sqrt{\frac{1}{N}\sum_n \left(\sigma_n^2+\mu_n^2\right) - \left(\frac{1}{N}\sum_n \mu_n\right)^2} \geq \frac{1}{N}\sum_n \sigma_n \tag{32}$$

$$\frac{1}{N}\sum_n \left(\sigma_n^2+\mu_n^2\right) - \left(\frac{1}{N}\sum_n \mu_n\right)^2 \geq \left(\frac{1}{N}\sum_n \sigma_n\right)^2 \tag{33}$$

$$\frac{1}{N}\sum_n \left(\sigma_n^2+\mu_n^2\right) \geq \left(\frac{1}{N}\sum_n \mu_n\right)^2 + \left(\frac{1}{N}\sum_n \sigma_n\right)^2 . \tag{34}$$

Eq. (34) can be regarded as two Jensen's inequalities $f(\mathbb{E}[x]) \leq \mathbb{E}[f(x)]$, where $f(x) = x^2$, and $\mathbb{E}[x] = \frac{1}{N}\sum_n x_n$. Thus, the initial inequality holds. It shows that the per-sample posterior Wasserstein metric is an upper bound to the aggregated posterior Wasserstein metric, commonly used in the WAE (Tolstikhin et al., 2018). Therefore, we can guarantee that the Wasserstein distance of the aggregated posterior to the assumed standard normal prior will not be larger than than the average distance of per-sample posteriors.

In addition to the theoretical argument, in Tab. 10 we offer an empirical comparison of the VAE* with the WAE-MMD model from (Tolstikhin et al., 2018) with aggregated posterior weight $\lambda = 10$. We observed that the per-sample posterior regularization significantly outperforms the WAE on Fashion-MNIST and CelebA, while being on par on CIFAR10.

H. Wasserstein Metric Aggregated Posterior Visualization

In Fig. 5 we present detailed plots on the posterior distributions of VAE and VAE* for the first 16 dimensions. The VAE clearly shows signs of posterior collapse (so-called *polarized regime* (Rolinek et al., 2019)); we have observed that more than half of the 128 dimensions are nearly equal to the prior. This considerably hurts the generative power of the VAE model. In contrast, the VAE* model has very low variance in all dimensions, which reflects a nearly deterministic encoder at the end of the training.

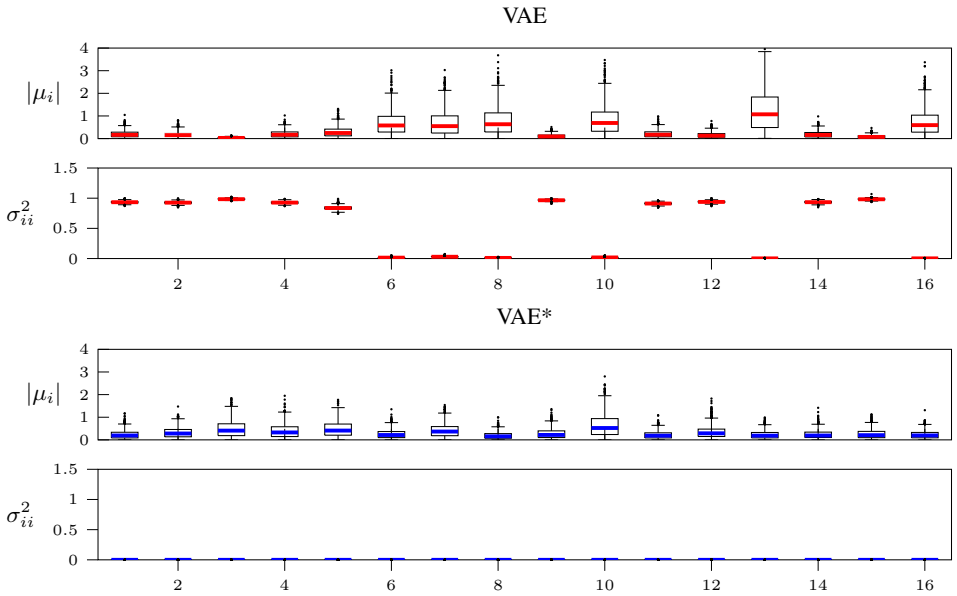


Figure 5: Comparison of the distribution of absolute means and variances of 1000 posterior samples for the VAE_{1x} and the VAE*_{1x} models trained with 100 epochs on the CIFAR10 dataset. Top rows show the absolute means and the lower rows the variances of the first 16 dimensions. For the VAE*_{1x} all means differ from zero while the variances are close to zero, whereas for the VAE_{1x}, 10 of 16 dimensions are effectively deactivated.

I. Qualitative Results on Fashion-MNIST and CIFAR10

Qualitative results on Fashion-MNIST and CIFAR10 are provided in Fig. 6 and Fig. 7. The same setup as in Fig. 3 is employed. It can be seen that the CIFAR10 images appear considerably richer and sharper, consistent with the results in Tab. 2 and Tab. 6.

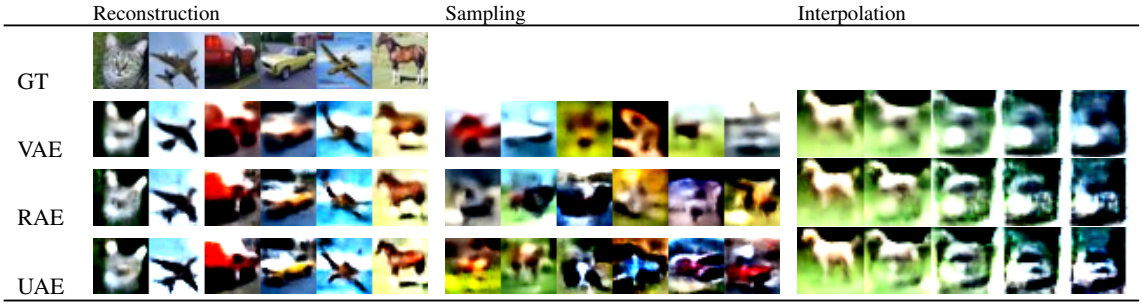


Figure 6: Qualitative results on the CIFAR10 dataset.

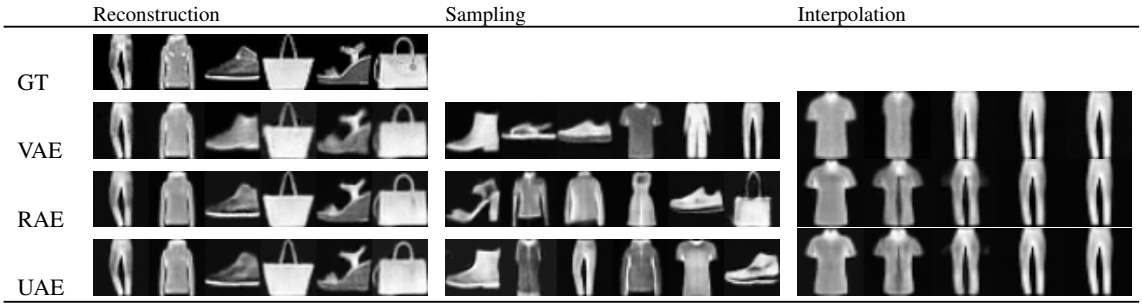


Figure 7: Qualitative results on the Fashion-MNIST dataset.

Paper VI

- Title: *Conditional Unscented Autoencoders for Trajectory Prediction*
- Authors: Faris Janjoš and Marcel Hallgarten and Anthony Knittel and Maxim Dolgov and Andreas Zell and J. Marius Zöllner
- Venue: submitted to 2024 European Conference on Computer Vision (ECCV) workshop: Event Detection for Situation Awareness in Autonomous Driving

Conditional Unscented Autoencoders for Trajectory Prediction

Faris Janjoš[1], Marcel Hallgarten[1,2], Anthony Knittel[1,3], Maxim Dolgov[1], Andreas Zell[2], and J. Marius Zöllner[4]

[1] Robert Bosch GmbH, Corporate Research, Renningen, Germany
first.last-name@bosch.com
[2] University of Tübingen, Tübingen, Germany
[3] Five AI Ltd, Cambridge, United Kingdom
[4] FZI Research Center for Information Technology, Karlsruhe, Germany

Abstract. The Conditional Variational Autoencoder (CVAE) is one of the most widely-used models in trajectory prediction for Autonomous Driving (AD). It captures the interplay between a driving context and its ground-truth future into a probabilistic latent space and uses it to produce predictions. In this paper, we challenge key components of the CVAE. We leverage recent advances in the space of the Variational Autoencoder (VAE), the foundation of the CVAE, which show that a simple change in the sampling procedure can greatly benefit performance. We find that unscented sampling, which draws samples from any learned distribution in a deterministic manner, can naturally be better suited to trajectory prediction than potentially dangerous random sampling. We go further and offer additional improvements including a more structured Gaussian mixture latent space, as well as a novel, potentially more expressive way to do inference with CVAEs. We show wide applicability of our models by evaluating them on the INTERACTION prediction dataset, outperforming the state of the art, as well as at the task of image modeling on the CelebA dataset, outperforming the baseline vanilla CVAE. Code is available at: https://github.com/boschresearch/cuae-prediction.

Keywords: Trajectory Prediction · Autonomous Driving · Conditional Variational Autoencoder (CVAE)

1 Introduction

Predicting the motion of human-driven vehicles sharing an environment with an autonomous system is a key enabler for fully-automated driving. Rich environment contexts present in urban driving and the prevalent interaction between traffic participants make it imperative to model the uncertainty in future trajectories. In this task of probabilistic trajectory prediction, machine learning models have proven indispensable. By learning a probability distribution, either in the space of the model's internal representations or the model's output, they capture the uncertainty inherent to the problem.

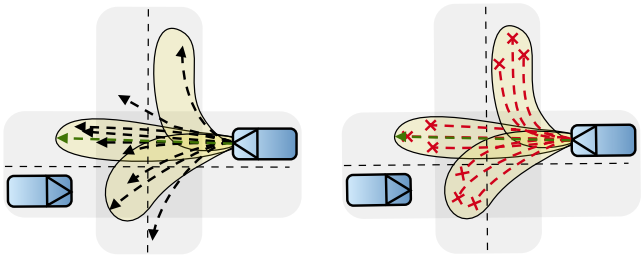


Fig. 1: Assume a trajectory predictor learned a multi-modal distribution (yellow), either by propagating its latent space or directly in the output space. Random sampling (black) can bring unsafe, unlikely, or in-between-mode outputs. In contrast, the unscented sampling (red), realized by computing sigma points of the distribution, brings structure to the learned stochasticity.

In addressing the challenges of probabilistic trajectory prediction, many approaches use established generative models such as a CVAE, a Generative Adversarial Network (GAN), or a Normalizing Flow (NF). The CVAE is especially useful; its powerful latent space model represents the underlying structure present in the relationship between a future trajectory and the potentially high-dimensional historical context that induces it. This real-world joint distribution is compressed into a tractable, relatively low-dimensional latent space Gaussian, amenable to sampling. Generating future predictions involves simply drawing samples from the latent space and transforming them into trajectories. The tasks of compressing inputs into the latent space and decompressing predictions from it are delegated to the CVAE's encoder and decoder, which can leverage powerful GNN or Transformer models. Thus, it has been a method of choice in state-of-the-art probabilistic prediction [6, 9, 27, 42, 55, 64].

Despite its wide appeal, the CVAE has certain shortcomings when applied in trajectory prediction. It does not provide an out-of-the-box means to evaluate the likelihoods of its trajectories. Further, since the distribution of future motion is highly multi-modal (usually involving distinct behaviors), recovering it from a smooth Gaussian latent space used in continuous CVAEs can bring unreasonable in-between outputs. Finally, the randomness inherent to the model is at odds with the primacy of safety and reproducibility. Random sampling of the latent space in inference can generate spurious and potentially dangerous trajectories (see Fig. 1) as well as miss critical trajectories, in addition to bringing a high gradient variance in training (a pitfall of the VAE itself). This can have serious ramifications on a downstream planner fed CVAE-predicted futures; these might differ significantly over consecutive prediction calls. Overall, these issues can be traced back to the CVAE's overly simplistic latent space and the unreliable random sampling.

Our work challenges well-established assumptions surrounding the CVAE and its Gaussian latent space. We aim to answer the following two research questions: *(i) Can the random sampling and propagation be replaced by more structured selection?*, *(ii) Are there effective alternatives to the simplistic latent space in training and inference, especially considering multi-modality of the output space?* In answering *(i)*, we leverage recent advancements in the base VAE [32] while for *(ii)*, we use more expressive distributions. As we improve core aspects of the

CVAE, we also evaluate our models on image generation tasks. Our contributions are:

- Unscented sampling and transformation of CVAE distributions as an alternative to random sampling for trajectory prediction, tackling *(i)*. As part of this contribution, we develop a novel Conditional Unscented Autoencoder (CUAE) model with deterministic sampling.
- A CVAE extension toward a mixture model latent space in place of a Gaussian latent space (usable with both random and unscented sampling). It promotes multi-modality in the output-space and tackles *(ii)*.
- A novel approach for inference with CVAEs via conditional ex-post estimation, inspired by [32] and tackling *(ii)*. It preserves latent space training but circumvents the need to use it in inference by building and sampling a more expressive distribution instead.

2 Related Work

In the following, we discuss approaches to probabilistic trajectory prediction, i.e. modeling the conditional distribution $\mathcal{P}(\mathbf{y}|\mathbf{x})$ of future trajectories $\mathbf{y}$ given a generic context $\mathbf{x}$. A popular choice is to represent $\mathcal{P}(\mathbf{y}|\mathbf{x})$ by a **set of trajectories**. Here, a fixed number of modes with associated probabilities is regressed either individually per agent [11, 16, 18, 20, 66] or jointly for all agents in a scene [10, 19, 30, 43, 47]. Commonly, Winner-Takes-All (WTA) loss functions only consider the predicted mode closest to the ground truth, exhibiting low sample efficiency. Moreover, as non-winner modes are not penalized, the predicted set can contain unrealistic and inadmissible trajectories (*e.g.* off-road). Many approaches address such issues by explicitly conditioning on map elements [14, 22, 24, 46, 66].

Other classes of models attempt to directly capture $\mathcal{P}(\mathbf{y}|\mathbf{x})$ into a **parametric distribution** such as a Gaussian Mixture Model (GMM). Here, trajectories are considered means of mixture components and (co)variances are learned separately [7, 36, 40, 50, 60], thus modeling the uncertainty of the underlying problem more accurately. Moreover, loss functions can consider the entire distribution (via the Negative Log Likelihood (NLL)), increasing sample efficiency and reducing inadmissible predictions. However, these models are theoretically limited since they do not reason about the generative process of the data, i.e. $\mathcal{P}(\mathbf{x}, \mathbf{y})$.

In contrast, **generative models** such as NFs [45, 51, 56], GANs [13, 21, 23, 41, 53], or CVAEs [6, 9, 27, 42, 55, 64], attempt to first learn a proxy for the joint data distribution and then obtain the predictive distribution $\mathcal{P}(\mathbf{y}|\mathbf{x})$. By sampling a prior and propagating the samples into the output space, they can implicitly capture rich non-parametric distributions. Among these models, NFs have limited expressiveness for high-dimensional data distributions found in trajectory prediction as well as strict architectural constraints, although they provide tractable likelihoods. Among GANs, prevalent issues include lack of diversity and mode collapse [54], out-of-distribution samples [37, 59], and training instability [1, 2]. Moreover, GANs learn a continuous transformation and are unable to model

disconnected manifolds [12, 37, 59], which is often necessary in prediction. To mitigate this, [12] uses multiple generators. Recently, diffusion models have also shown success in prediction [8, 33]. However, they are less established in AD applications due to practical issues w.r.t. training complexity and hyperparameter sensitivity, as well the lack of an explicitly parameterized latent space.

Similarly to GANs, CVAEs can struggle to model output distributions with disconnected modes [52]. The decoder transformation is continuous and the latent distribution capturing multiple futures is commonly modeled as a multivariate Gaussian. The approaches in [26, 55] address the issue by using a discrete latent variable mapped to a GMM output, which also facilitates integration over the conditional prior distribution. However, this limits the expressiveness of the latent space. In this work, we approach this problem by leveraging more expressive latent distributions such as GMMs in both training and inference. Furthermore, many models adapt the CVAE to output likelihoods by additional classifier networks, a common approach across the trajectory prediction landscape [6, 9, 40]. Another pitfall of CVAEs is that propagating the latent distribution to the output space involves drawing and decoding random samples. The randomness can result in bad coverage of the true output distribution, especially with few samples. To mitigate this, [9] and [64] employ diversity sampling techniques in training, while [9] off-loads the modeling of distinct futures to a GNN decoder. In inference, [9] uses only the latent mean and abandons the rich learned latent space. In contrast, we use deterministic sampling [32] to obtain diverse and representative samples from the learned latent space.

3 Method

We consider the task of modeling $\mathcal{P}(\mathbf{y}|\mathbf{x})$, where $\mathbf{y}$ are future vehicle trajectories $\mathbf{y}$, i.e. a $T\times 2$ matrix of T future positions, and $\mathbf{x}$ is a generic context. In addressing this task, we leverage the CVAE framework to construct and sample an expressive latent space. Thus, we outline the presentation of our approach along the research questions posed in Sec. 1. Sec. 3.1 provides a CVAE background and proposes alternatives to random sampling and transformation of the latent space, tackling *(i)*, while Sec. 3.2 offers alternatives in latent space modeling and using it for inference, tackling *(ii)*. Sec. 3.3 discusses the generation of output trajectories given the choices in Sec. 3.1 and Sec. 3.2.

3.1 Latent Space Sampling and Transformation

CVAE Background CVAEs [58] are generative models that can capture a conditional distribution $\mathcal{P}(\mathbf{y}|\mathbf{x})$. They model the relationship between pairs of high-dimensional inputs $\mathbf{x}$ and $\mathbf{y}$ by projecting them into a lower-dimensional latent space $\mathbf{z}$, see Fig. 2. An encoder parameterized by ϕ learns the latent posterior distribution $q_\phi(\mathbf{z}|\mathbf{x},\mathbf{y};\boldsymbol{\mu}_\phi,\boldsymbol{\Sigma}_\phi)$, commonly modeled as a multivariate Gaussian. Then, a θ-parameterized decoder is tasked with estimating the true output distribution $\mathcal{P}(\mathbf{y}|\mathbf{x})$. This is done by conditioning on $\mathbf{x}$ and drawing random samples $\mathbf{z}$ from q_ϕ to first produce $p_\theta(\mathbf{y}|\mathbf{x},\mathbf{z})$ and thus marginalize out

$\mathbf{z}$. In inference, since the ground-truth $\mathbf{y}$ is not available, the model instead samples a surrogate, γ-parameterized latent prior $p_\gamma(\mathbf{z}|\mathbf{x}; \boldsymbol{\mu}_\gamma, \boldsymbol{\Sigma}_\gamma)$. The posterior q_ϕ and prior p_γ are trained to be consistent. Thus, the loss function minimizes $\mathcal{L}_{\text{CVAE}} = \mathcal{L}_{\text{REC}} + \mathcal{L}_{\text{KL}}$ (and maximizes the Evidence Lower Bound (ELBO) [39]),

$$\mathcal{L}_{\text{REC}} = -\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x},\mathbf{y})} \left[\log p_\theta(\mathbf{y}|\mathbf{x}, \mathbf{z})\right] \ , \tag{1}$$

$$\mathcal{L}_{\text{KL}} = \beta D_{\text{KL}}(q_\phi(\mathbf{z}|\mathbf{x}, \mathbf{y}) \| p_\gamma(\mathbf{z}|\mathbf{x})) \ . \tag{2}$$

Eq. (1) promotes consistency between the decoder output and the observed ground truth, while Eq. (2) brings the posterior and prior distributions together by minimizing their Kullback-Leibler (KL) divergence and is weighted by β. In practice, K samples $\mathbf{z}_k$ from q_ϕ are drawn and a deterministic decoder function f_θ maps $(\mathbf{x}, \mathbf{z}_k)$ to an output trajectory $\mathbf{y}_k = f_\theta(\mathbf{x}, \mathbf{z}_k)$. Thus, no parametric form of $\mathcal{P}(\mathbf{y}|\mathbf{x})$ is estimated and the output distribution is represented by a set of K samples. In this way, Eq. (1) can be approximated by the NLL[5] of reconstructed samples $\mathbf{y}_k$ under the ground-truth distribution $\mathcal{N}(\mathbf{y}, \sigma^2 \boldsymbol{I})$, yielding

$$\mathcal{L}_{\text{REC-samples}} = -\tfrac{1}{K} \textstyle\sum_k^K \log \mathcal{N}(\mathbf{y}_k; \mathbf{y}, \sigma^2 \boldsymbol{I}) \ . \tag{3}$$

A common approximation when predicting entire trajectories is to assume independence across time steps [40] or a fixed diagonal covariance matrix $\sigma \boldsymbol{I}$ [12].

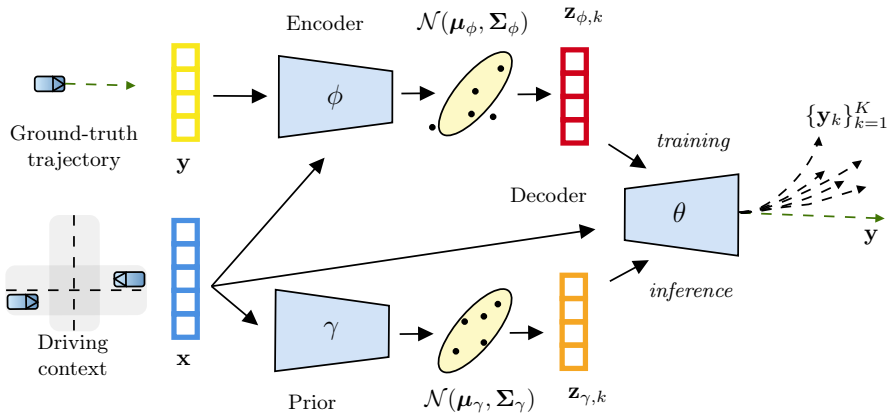


Fig. 2: CVAE: in training, the model captures the joint distribution of the ground-truth trajectory and driving context via the encoder network ϕ, samples randomly, and reconstructs trajectories via the decoder network θ. In inference, the prior network γ replaces ϕ and is sampled instead.

Unscented Transform of the Latent Space A key component of the CVAE (and its VAE foundation) is random sampling of the latent space. It is a feature of the reparameterization trick [39], employed in order to sample the latent Gaussian posterior and efficiently compute gradients w.r.t. ϕ in Eq. (1). However, it exhibits high variance in training. Therefore, a deterministic-sampling alternative based on the Unscented Transform (UT) [35] (prominent in filtering and control) has emerged in the Unscented Autoencoder (UAE) [32]. It is motivated by the fact that the decoder is a nonlinear function of the posterior

[5] In image modeling, it is usually the Mean Squared Error (MSE) instead.

distribution. Thus, a set of representative points in the latent space can be chosen and transformed to approximate the output distribution, which is difficult in practice by transforming a few random samples.

The UT application in the VAE context can be straightforwardly extended to the CVAE. The CUAE model is shown in Fig. 3. The model analytically computes the sigma points of the Gaussian posterior $\mathcal{N}(\boldsymbol{\mu}_\phi, \boldsymbol{\Sigma}_\phi)$, $\boldsymbol{\mu}_\phi \in \mathbb{R}^n$. The $2n{+}1$ sigmas $\{\boldsymbol{\chi}_i\}_{i=0}^{2n}$ are the mean $\boldsymbol{\chi}_n{=}\boldsymbol{\mu}_\phi$ and a pair on each axis $\boldsymbol{\chi}_{n\pm j}{=}\boldsymbol{\mu}_\phi{\pm}\sqrt{n\boldsymbol{\Sigma}_\phi}\big|_j$, $1{\leq}j{\leq}n$, where $\big|_j$ indicates the j-th column. For a commonly used diagonal $\boldsymbol{\Sigma}_\phi$, there is no computational overhead since the Cholesky decomposition $\sqrt{\boldsymbol{\Sigma}_\phi}$ can be directly obtained from predicted log variances.

Since the sigmas fully describe the latent distribution[6], they usually also describe the output distribution well w.r.t commonly used decoder nonlinearities [34]. Thus, going one step further, we can approximate the expectation in Eq. (1) by the mean of the transformed sigmas. With this, we push the entire output distribution (w.r.t. its mean) toward the ground truth instead of the individually transformed samples. Thus,

$$\mathcal{L}_{\text{REC-dist.}} = -\log \mathcal{N}\big(\tfrac{1}{K}\textstyle\sum_k^K \mathbf{y}_k; \mathbf{y}, \sigma^2 \boldsymbol{I}\big) \ , \tag{4}$$

where each $\mathbf{y}_k$ comes from a latent space sigma point. In practice, due to a large dimensionality n, we select $K{<}2n{+}1$ random pairs of sigma points on the same covariance axis [32].

In the context of trajectory prediction, sigma points have the potential to reasonably cover the latent space with few samples, train the entire output distribution accordingly, and prevent spurious and unlikely samples in inference.

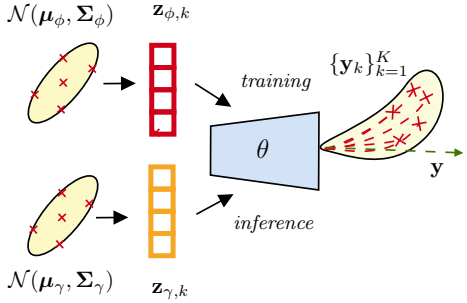


Fig. 3: CUAE: instead of sampling the latent space randomly (in both training and inference), the model analytically computes sigma points of the ϕ and γ distributions and transforms them instead.

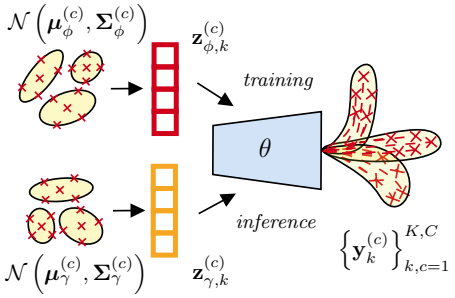


Fig. 4: GMM-CUAE: it structures the latent space into a GMM and separately transforms its components (sigma points shown). Compared to Fig. 3, it can better model multi-modality.

3.2 Expressive Latent Spaces in Training and Inference

Many use-cases within probabilistic trajectory prediction necessitate a disjoint output with well-separated modes such as turning left or right. Here, CVAEs

[6] Their first two moments (the mean and covariance) equal the original distribution's first two moments.

struggle due to the continuous latent distribution that is decoded as a continuous distribution of trajectories. Therefore, we propose two methods to promote a multi-modal output space. Both use GMMs: the first trains a GMM latent space and the second uses a separately-constructed GMM purely for inference.

Mixture Distributions in Training: GMM Latent Space The mixture prior model attempts to capture distinct modes of behavior using a GMM for the prior and posterior distributions in the latent space, see Fig. 4. The GMM components can correspond with modes of behavior, while the distribution of each can represent the variation within each mode. For example, one mode may correspond to a right-turn behavior whose speed or path variation is given by the variance. The two GMMs with C components are described by $\sum_c^C w_\phi^{(c)} \mathcal{N}(\boldsymbol{\mu}_\phi^{(c)}, \boldsymbol{\Sigma}_\phi^{(c)})$ and $\sum_c^C w_\gamma^{(c)} \mathcal{N}(\boldsymbol{\mu}_\gamma^{(c)}, \boldsymbol{\Sigma}_\gamma^{(c)})$, for fixed C. Sampling is performed independently for each component; we draw K random samples or sigma points from each, totaling $K{\cdot}C$. Then, we compute the centroid $\mathbf{y}_{\boldsymbol{\mu}}^{(c)}$ and covariance $\mathbf{y}_{\boldsymbol{\Sigma}}^{(c)}$ of the decoded trajectories for each mode to obtain an output-space GMM, with the corresponding component weights carried over.

The loss functions in Eq. (1) and Eq. (2) are adapted as follows to be compatible with a GMM representation. As the KL divergence between Gaussian mixtures is not analytically defined, we apply it individually (according to Eq. (2)) between each corresponding posterior and prior component and their discrete mixture distributions

$$\mathcal{L}_{\text{KL-GMM}} = \sum_c^C \mathcal{L}_{\text{KL}}^{(c)} + D_{\text{KL}}(w_\phi \| w_\gamma) \ . \tag{5}$$

The approximation in Eq. (5) is introduced in [15] and offers an upper bound. The reconstruction loss minimizes the NLL of the ground-truth trajectory under the predicted future distribution, represented by the output GMM. This way, we train the component whose centroid trajectory is closest to the ground-truth $\mathbf{y}$ (denoted by c^*) and a one-hot distribution $w_{\mathbf{y}}$ of the closest component,

$$\mathcal{L}_{\text{REC-GMM}} = -\log \mathcal{N}(\mathbf{y}; \mathbf{y}_{\boldsymbol{\mu}}^{(c^*)}, \mathbf{y}_{\boldsymbol{\Sigma}}^{(c^*)}) + D_{\text{KL}}(w_\phi \| w_{\mathbf{y}}) \ . \tag{6}$$

Note that this form of the reconstruction loss function is closer to the UT-like computation in Eq. (4) than the vanilla CVAE-like computation in Eq. (3), since the likelihood is computed for the centroid trajectory of the winner latent GMM component rather than in expectation over multiple individual latent samples. The choice between random and unscented sampling determines the method to obtain the output trajectories (that approximate the true distribution), which are averaged to compute the centroid.

Mixture Distributions in Inference: Conditional Ex-Post (CXP) Estimation In this section, we present an alternative to using the trained latent space for inference, which is universal among VAEs and CVAEs. Termed ex-post (XP) estimation, it involves training a latent space but not using it directly

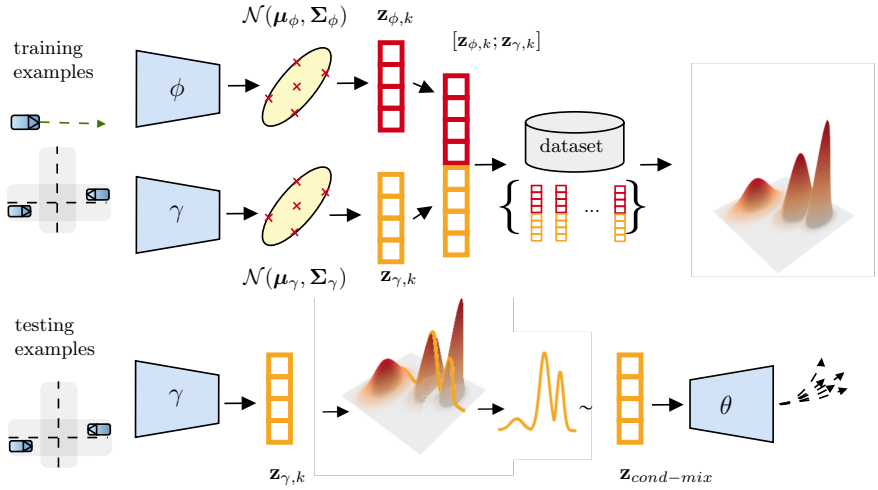


Fig. 5: Illustration of conditional ex-post (CXP) joint mixture construction and conditional sampling. Top: all posterior and prior sigma points in training are concatenated, collected, and used to fit a mixture. Bottom: given a new example's prior encoding, the mixture is conditioned (intuitively, it is "cut"). The resulting lower-dim. mixture is sampled as input for the decoder.

in inference. For the VAE, an empirically-obtained distribution constructed after training is sampled instead of the theoretically-imposed standard normal prior [17]. The advantage is that it addresses the well-known VAE posterior mismatch[7] as well as enables deterministic autoencoders (lacking a probabilistic latent space) to generate samples. It can be realized by collecting a dataset of posterior encodings during training $\mathcal{D}_{\text{ex-post}}=\{\mathbf{z}_\phi^i\}$ (i for training example) and using an off-the-shelf tool (*e.g.* [49]) to fit another more expressive distribution. For example, a GMM with C Gaussians, $p(\mathbf{z}_{\phi\text{mix}})=\sum_c^C w^{(c)}\mathcal{N}(\mathbf{z}_{\phi\text{mix}};\boldsymbol{\mu}_{\phi\text{mix}},\boldsymbol{\Sigma}_{\phi\text{mix}})$, where $\mathbf{z}_{\phi\text{mix}}$ is the GMM random variable, $w^{(c)}$ are the obtained weights, and C is a priori defined. This empirical distribution is then sampled instead.

We extend the original method to the CVAE and CUAE by introducing a conditional ex-post (CXP) estimated density. The vanilla XP sampling is inadequate in the CVAE case since using a mixture built only from the posterior encodings $\mathbf{z}_\phi$ precludes conditioning, *e.g.* the driving context encountered in a test set example. Therefore, we incorporate the prior encoding $\mathbf{z}_\gamma$ (obtained through the conditional prior γ, see Fig. 2). First, we collect a set of concatenated posterior-prior pairs $\mathcal{D}_{\text{cond-ex-post}}=\{[\mathbf{z}_\phi^i;\mathbf{z}_\gamma^i]\}$ and then fit a GMM $p(\mathbf{z}_{\phi\gamma\text{mix}})$. See Fig. 5 (top) for an illustration. These pairs are a dataset of latent-space relationships between future trajectories and the associated context. Thus, concatenating them and fitting a GMM models the joint distribution between the ground-truth-future-posterior and the context-prior that preceded it. However, sampling given a new context $\mathbf{x}$ requires conditioning the joint mixture on $\mathbf{z}_\gamma$. In the following, we lay out the necessary steps.

[7] In practice, the average posterior over the entire training set does not fully match the assumed $\mathcal{N}(\mathbf{0},\mathbf{I})$ prior, leading to lower sample quality.

Assume that the mixture of posterior-prior encodings is parameterized by C Gaussians along with their weights $w^{(c)}$

$$p(\mathbf{z}_{\phi\gamma\text{mix}}) = \sum_c^C w^{(c)} \mathcal{N}(\mathbf{z}_{\phi\gamma\text{mix}}; \boldsymbol{\mu}^{(c)}_{\phi\gamma\text{mix}}, \boldsymbol{\Sigma}^{(c)}_{\phi\gamma\text{mix}}) \ . \tag{7}$$

The random variable realization $\mathbf{z}_{\phi\gamma\text{mix}}$ can be split into $\mathbf{z}_{\phi\gamma\text{mix}}{=}[\mathbf{z}_1; \mathbf{z}_2]$, with $\dim(\mathbf{z}_1){=}\dim(\mathbf{z}_\phi)$ and $\dim(\mathbf{z}_2){=}\dim(\mathbf{z}_\gamma)$. Note that $\mathbf{z}_1$ and $\mathbf{z}_2$ "belong" to $p(\mathbf{z}_{\phi\gamma\text{mix}})$ and are not the same as $\mathbf{z}_\phi$ and $\mathbf{z}_\gamma$. Thus, each component c is factored as

$$\boldsymbol{\mu}^{(c)}_{\phi\gamma\text{mix}} = \begin{bmatrix} \boldsymbol{\mu}^{(c)}_1 \\ \boldsymbol{\mu}^{(c)}_2 \end{bmatrix}, \quad \boldsymbol{\Sigma}^{(c)}_{\phi\gamma\text{mix}} = \begin{bmatrix} \boldsymbol{\Sigma}^{(c)}_{11} & \boldsymbol{\Sigma}^{(c)}_{12} \\ \boldsymbol{\Sigma}^{(c)}_{21} & \boldsymbol{\Sigma}^{(c)}_{22} \end{bmatrix} \ . \tag{8}$$

The aim is to compute the conditional mixture distribution $p(\mathbf{z}_1|\mathbf{z}_2){=}\frac{p(\mathbf{z}_1,\mathbf{z}_2)}{p(\mathbf{z}_2)}$. In [3], the conditional distribution of a component in a multivariate Gaussian is

$$\mathcal{N}(\mathbf{z}_1|\mathbf{z}_2; \boldsymbol{\mu}^{(c)}_{\phi\gamma\text{mix}}, \boldsymbol{\Sigma}^{(c)}_{\phi\gamma\text{mix}}) = \mathcal{N}(\mathbf{z}_1; \boldsymbol{\mu}^{(c)}_{1|2}, \boldsymbol{\Sigma}^{(c)}_{1|2}) \ , \tag{9}$$

$$\boldsymbol{\mu}^{(c)}_{1|2} = \boldsymbol{\mu}^{(c)}_1 + \boldsymbol{\Sigma}^{(c)}_{12}(\boldsymbol{\Sigma}^{(c)}_{22})^{-1}(\mathbf{z}_2 - \boldsymbol{\mu}^{(c)}_2) \ , \tag{10}$$

$$\boldsymbol{\Sigma}^{(c)}_{1|2} = \boldsymbol{\Sigma}^{(c)}_{11} - \boldsymbol{\Sigma}^{(c)}_{12}(\boldsymbol{\Sigma}^{(c)}_{22})^{-1}\boldsymbol{\Sigma}^{(c)}_{21} \ . \tag{11}$$

The marginal distribution $p(\mathbf{z}_2)$ is given simply by $\sum_c^C w^{(c)}\mathcal{N}(\mathbf{z}_2; \boldsymbol{\mu}^{(c)}_{22}, \boldsymbol{\Sigma}^{(c)}_{22})$. Thus, $p(\mathbf{z}_1|\mathbf{z}_2)$ is computed by

$$p(\mathbf{z}_1|\mathbf{z}_2){=}\sum_c^C \frac{w^{(c)}\mathcal{N}(\mathbf{z}_2;\boldsymbol{\mu}^{(c)}_{22},\boldsymbol{\Sigma}^{(c)}_{22})}{p(\mathbf{z}_2)}\mathcal{N}(\mathbf{z}_1; \boldsymbol{\mu}^{(c)}_{1|2}, \boldsymbol{\Sigma}^{(c)}_{1|2}) \ , \tag{12}$$

where the fraction provides the new mixture weights that are normalized by the density of the marginal $p(\mathbf{z}_2)$. In this manner, we can sample $\mathbf{z}_{\text{cond-mix}}{\sim}p(\mathbf{z}_1|\mathbf{z}_2{=}\mathbf{z}_\gamma)$ and feed the decoder with $\mathbf{z}_{\text{cond-mix}}$ instead of $\mathbf{z}_\gamma$, $p_\theta(\mathbf{y}|\mathbf{x}, \mathbf{z}_{\text{cond-mix}})$. Thus, $p(\mathbf{z}_1|\mathbf{z}_2)$ serves as a link to obtain a latent vector close to what would be a posterior encoding (through their joint mixture relationship), considering that the posterior is not available in inference. See Fig. 5 (bottom) for an illustration.

The conditional mixture in Eq. (12) provides a more expressive sampling distribution than the simplistic Gaussian prior. The conditioning by the prior sample results in a distribution that contains similar ground-truth training posteriors. Further, the weights of the conditional mixture (the fraction in Eq. (12)), different to $w^{(c)}$ in Eq. (7), can effectively prune irrelevant components by assigning low values, potentially providing $\ll C$ components with non-negligible weights. In this way, a variable number of components in the latent space can be modeled based on the encountered context.

In the context of the CXP-CUAE model, an open question is choosing the specific $\mathbf{z}_\gamma$ vector to condition the joint mixture. Considering that the prior γ can provide sigma points, each of the $2n{+}1$ distinct points would result in a different conditional mixture. We choose the sigma point that incurs the largest density in the marginal distribution $p(\mathbf{z}_2)$ (the denominator term in Eq. (12)). Intuitively, such a sigma point would cut the joint mixture where it is most data-rich.

Model	sampling	latent space	inference via	K	M	minADE	minFDE	mixture-NLL			winner-NLL		
								1s	2s	3s	1s	2s	3s
CVAE	random	Gaussian	latent space	6	6	0.149	0.478	-	-	-	1.846	1.877	2.167
CVAE	random	Gaussian	latent space	65	65	0.078	0.209	-	-	-	1.845	1.854	1.955
CUAE	unscented	Gaussian	latent space	6	6	0.145	0.452	-	-	-	1.846	1.872	2.131
CUAE	unscented	Gaussian	latent space	65 (all)	65	0.087	0.170	-	-	-	1.845	1.846	1.899
CVAE+*clusters*	random	Gaussian	latent space	65	6	0.134	0.454	1.842	1.928	2.607	-2.685	-0.382	1.980
CUAE+*clusters*	unscented	Gaussian	latent space	65 (all)	6	0.130	0.400	1.850	2.010	2.780	**-2.701**	**-1.358**	**1.011**
CXP-CVAE+*clusters*	random	Gaussian	cond. ex-post	65	6	0.128	0.427	**-2.556**	3.400	6.608	-	-	-
CXP-CUAE+*clusters*	unscented	Gaussian	cond. ex-post	65 (all)	6	0.122	0.379	-2.431	0.336	2.792	-	-	-
GMM-CVAE	random	GMM	latent space	65	6	0.101	0.301	-0.437	**-0.016**	1.152	-	-	-
GMM-CUAE	unscented	GMM	latent space	65 (all)	6	**0.097**	**0.283**	-0.430	0.053	**1.150**	-	-	-

Table 1: Left half: Breakdown of the approaches described in Sec. 3. Note that all of the approaches in the gray-colored rows below the CVAE rows are proposed in this work. Right half: Corresponding INTERACTION experiment results. Legend: K – number of samples/sigmas from the latent space (6 or 65 due to a 32-dim. latent space), M – number of final provided trajectories, *+clusters* – K decoded samples/sigmas in the output are clustered into M trajectories, GMM latent space – M trajectories are given by the centroids of K decoded samples/sigmas per each GMM component.

3.3 Output Trajectory Generation

Commonly used metrics such as Minimum Average Displacement Error (minADE) and Minimum Final Displacement Error (minFDE) (see [40] for definitions) necessitate a fixed number of M candidate trajectories, *e.g.* M=6. CVAEs inherently exhibit large variance on such metrics due to the random sampling. Even the deterministic sampling of the CUAE poses the question of which M sigmas to provide among $2n$+1 choices, $M \ll 2n$+1. Therefore, we investigate a simple way to provide a more consistent output. We first draw a large number of K random samples (CVAE) or take all K=$2n$+1 latent sigmas (CUAE). Then, we cluster them into M clusters with a k-means procedure and provide only the centroids. A similar approach is explored in [14]. Thus, we also evaluate the clustering-enhanced CVAE, CUAE, and the CXP-CUAE, detailed in Sec. 3.1, 3.1, and 3.2, respectively. We do not apply it to the latent space GMM from Sec 3.2 since it already has a mechanism to provide fixed M trajectories through the C=M components. We expect that this approach especially boosts the performance of the CUAE in training, since it translates its structured latent space coverage into the output space. Overall, as we offer multiple models touching different facets of the CVAE, we summarize our proposed approaches in Tab. 1 (left).

4 Results

Here, we describe the experimental setup and present the results of our proposed CVAE trajectory prediction models. As CVAEs are used beyond this task, our architectural improvements are not limited to prediction – the primary evaluation setting. Thus, to better understand our models, we extend the evaluation with the secondary setting of classical image modeling on the CelebA [44] dataset.

4.1 Implementation

The network architectures of our CVAE approaches explicitly follow the StarNet model [30]. It is a deterministic, single-agent[8] predictor that uses a graph-based map and trajectory history context. We use a shared StarNet encoder, comprising a 1D-CNN trajectory history network, GNN map network and an attention-based [61] agent interaction network as a basis for the posterior ϕ and prior γ distributions. Since the posterior ϕ additionally receives the ground-truth future trajectory, we reuse the 1D-CNN. Thus, in both ϕ and γ the StarNet encoder produces a single feature vector passed onto a two-layer $[128, 128]$ MLP with batch normalization and ReLU activation. The output of this MLP is passed onto two separate 32-dim. layers producing $\boldsymbol{\mu}_\phi$ or $\boldsymbol{\mu}_\gamma$ and $\log \boldsymbol{\sigma}^2_\phi$ or $\log \boldsymbol{\sigma}^2_\gamma$ (used to construct diagonal covariance matrices). In the GMM-CVAE[9] in Sec. 3.2, an additional 64-dim. layer produces weights w_ϕ or w_γ from concatenated means and variances as input. The StarNet decoder θ predicts future trajectories in an action-based manner[10] [29]. We emphasize that other more sophisticated models can be used for the encoder/decoder, which is orthogonal to our top-level CVAE.

In image modeling experiments, we use the identical setup from [32] and extend it with a prior γ network. It encodes the conditioning $\mathbf{x}$ in the CelebA dataset consisting of a 40-dim. binary vector of face attributes (whereas the ground-truth $\mathbf{y}$ acc. to Fig. 2 is an image). The prior γ is realized as a $[64, 64]$ MLP for both $\boldsymbol{\mu}_\gamma$ and $\log \boldsymbol{\sigma}^2_\gamma$ with a shared first layer.

4.2 Datasets and Training Setup

We trained and evaluated our trajectory prediction models on the INTERACTION [65] dataset, since the base StarNet model [30] was evaluated on this dataset. It contains a large number of highly interactive driving scenarios with merges, roundabouts, and intersections. We used the official training and validation splits, predicting 3s trajectories (T=30 at 10Hz) given a 1s history. In image modeling, we used the rich CelebA dataset [44] of human faces, containing 64×64×3 images pre-processed the same way as in [32] and 40-dim. one-hot annotations[11].

The prediction models are trained for 30 epochs with Adam [38], starting from a $1e^{-4}$ learning rate and halving it for epochs 10, 15, 20, and 25. CelebA

[8] Our CVAE-level improvements have no inherent restrictions toward a joint prediction extension, which is a more sound approach to the problem.

[9] The term GMM-CVAE was previously used in [26] in a prediction context and in [62] in image modeling. The former uses a categorical latent space and regresses an output-level GMM. The latter uses pre-defined clusters as modes of the GMM, where cluster labels are already available instead of being learned in the training process. Since both approaches significantly differ from ours, we reuse the term for our latent-space GMM.

[10] The model first predicts future actions (acceleration and steering angle) and then unrolls them into future positions (starting from the current position) using a kinematic bicycle model.

[11] Examples include: *Smiling*, *Eyeglasses*, *Young*, *Blond_Hair*.

experiments use the same setup as in [32]: 100 epochs and a learning rate halving on loss plateau. All models are implemented in PyTorch [48]. In CXP estimation, we used [4] (compatible with [49]) for fast GPU-based GMM fitting after training, taking around 30 min. over the entire training set. In both use-cases, the models took around 1.5 days to train on a single Nvidia 3090 GPU.

4.3 Image Modeling Performance

In this secondary evaluation setting, our aim is to assess the proposed models' ability to generate realistic images. One goal is reconstructing existing images by compressing and decompressing them from the latent space (a task trajectory prediction models are not evaluated on). More specifically, a trained CVAE encodes a ground-truth image $\mathbf{y}$ and its attributes as context $\mathbf{x}$ into the posterior distribution $\mathcal{N}(\mathbf{z}_\phi|\mathbf{x},\mathbf{y};\boldsymbol{\mu}_\phi,\boldsymbol{\Sigma}_\phi)$. Then, it feeds a random sample (or sigma) $\mathbf{z}_{\phi,k}$ to the decoder, which reconstructs the output image $\mathbf{y}_k{=}f_\theta(\mathbf{x},\mathbf{z}_{\phi,k})$. This process is conceptually the same as the CVAE in Fig. 2. Furthermore, we evaluate the ability to generate realistic new image samples. In this manner, the model only receives the context $\mathbf{x}$ encoded into the prior $\mathcal{N}(\mathbf{z}_\gamma|\mathbf{x};\boldsymbol{\mu}_\gamma,\boldsymbol{\Sigma}_\gamma)$. Then, the decoder produces an image $\mathbf{y}_k{=}f_\theta(\mathbf{x},\mathbf{z}_{\gamma,k})$ using a sample $\mathbf{z}_{\gamma,k}$. In contrast, the CXP-CVAE produces an image in inference using a sample from the conditional mixture, $\mathbf{y}_k{=}f_\theta(\mathbf{x},\mathbf{z}_{\text{cond-mix}})$. For all models, we use four samples (or random sigmas) in training. In both reconstruction and sampling, we evaluate image realism with the established Fréchet Inception Distance (FID) [25], which computes the Wasserstein metric between sets of real and sampled images.

Tab. 2 shows quantitative results comparing the vanilla CVAE, CUAE, and both with CXP estimation ($C{=}10$, see Eq. (7)). We do not include GMM-CVAE since generating multiple image outputs is usually not relevant to the problem (which is stationary). We additionally ablate the baseline XP estimation from [32], [17], which does not condition on attributes $\mathbf{x}$; it fits the mixture only on $\mathbf{z}_{\phi,k}$ and directly samples $\mathbf{z}_{\phi\text{mix}}$ to feed the decoder $\mathbf{y}_k{=}f_\theta(\mathbf{x},\mathbf{z}_{\phi\text{mix}})$. We observe that the best scores are achieved by CUAE models with (C)XP estimation. Fig. 6 qualitatively corroborates the results from Tab. 2; it is evident that such models generate sharper and more realistic images than prior inference models. As expected though, the XP-CVAE struggles to include the queried attribute into the image (since the sample $\mathbf{z}_{\phi\text{mix}}$ does not contain it), something vanilla CVAE and our CXP-CVAE are well capable of.

4.4 Trajectory Prediction Performance

In this section, we pit our proposed CVAE improvements against each other: CUAE (Sec. 3.1), GMM latent space (Sec. 3.2), CXP (Sec. 3.2), as well as output-level clustering (Sec. 3.3) on the primary, trajectory prediction evaluation setting. We aim to evaluate the quality of multi-modal predictions on a trajectory and distribution level using the minADE and minFDE metrics as well as the distributional NLL. Comprehensive results are shown in Tab. 1 (right). Since our approaches relate to core CVAE aspects, our main baseline is a vanilla

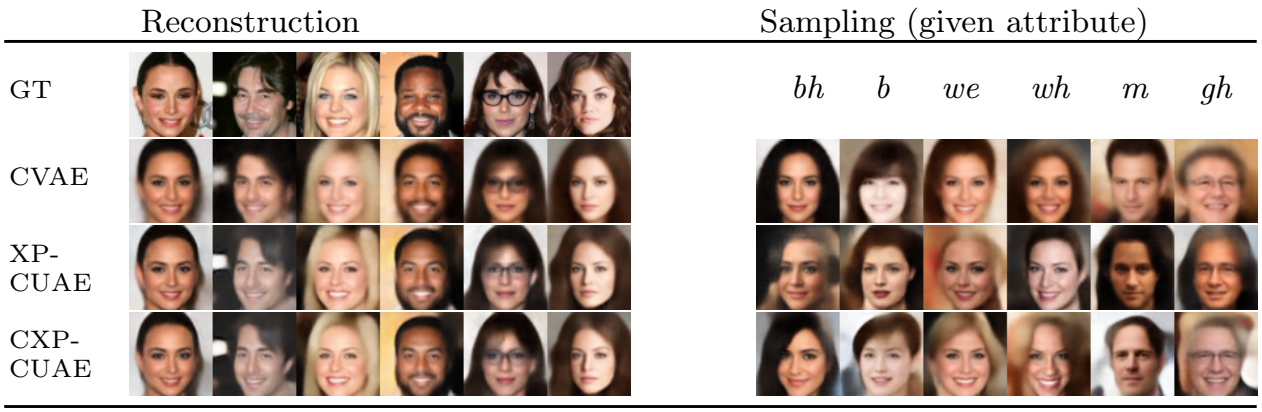


Fig. 6: Example reconstructed and sampled images. The XP-CUAE and CXP-CUAE inference models generate sharper images than vanilla CVAE, as also evidenced by the quantitative results of corresponding rows in Tab. 2. Furthermore, the proposed CXP inference succeeds at incorporating the attributes into samples, while the baseline XP inference struggles with it. Attributes: *Black_Hair* (*bh*), *Bangs* (*b*), *Wearing_Earrings* (*we*), *Wavy_Hair* (*wh*), *Male* (*m*), *Gray_Hair* (*gh*).

Model	sampling	inference via	image modeling	
			reconstr.	sampling
CVAE	random	latent space	59.74	62.61
CUAE	unscented	latent space	47.92	98.50
XP-CVAE	random	ex-post	59.29	63.70
XP-CUAE	unscented	ex-post	40.67	48.83
CXP-CVAE	random	cond. ex-post	59.32	63.53
CXP-CUAE	unscented	cond. ex-post	**40.44**	**48.52**

Table 2: Quantitative image modeling results on FID (lower is better). We ablate the proposed unscented sampling, ex-post estimation (XP) [17, 32], and the proposed conditional ex-post estimation (CXP).

Model	minADE_6	minFDE_6
ITRA [57]	0.17	0.49
GOHOME [20]	-	0.45
joint-StarNet [30]	0.13	0.38
DiPA [40]	0.11	0.34
MB-SS-ASP [31]	0.10	0.30
SAN [28]	0.10	0.29
GMM-CUAE	**0.097**	**0.283**

Table 3: Comparison of the best model in Tab. 1 with models from the literature on the INTERACTION validation dataset.

CVAE, however, Tab. 3 shows a comparison of our highest-performing model with non-CVAE approaches in literature.

The first four rows of Tab. 1 show the results of CVAE and CUAE models from Sec. 3.1. We trained with σ=1.0 in Eq. (3) and Eq. (4) (it has been shown to work well for sample-based predictors [12]). Since it is nontrivial to compute the full NLL, we compute it only for the closest mode to the ground truth (σ=1.0). We see that the CUAE provides a 5% boost over CVAE in trajectory metrics, however, qualitative results in Fig. 7 show the potential of sigma points to provide good coverage, as illustrated in Fig. 1. Further, both CVAE and CUAE benefit from increasing the number of samples or using all sigma points (32-dim.

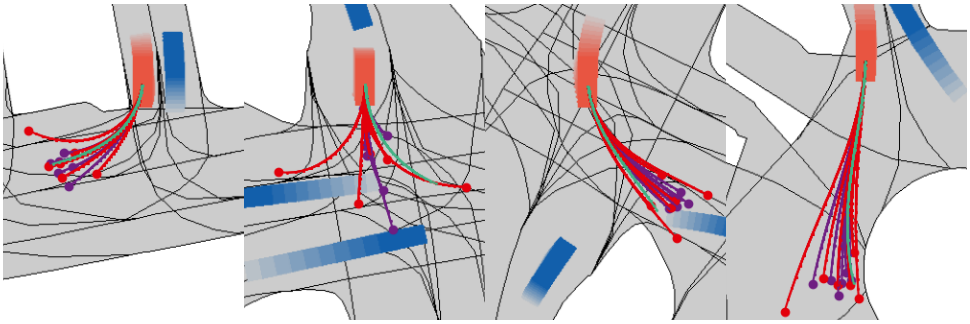


Fig. 7: Qualitative comparison of sampling choices (best viewed in color): trajectories reconstructed from sigma points (red) are significantly more diverse than random samples (purple). Surrounding traffic is depicted in blue and the predicted vehicle in red.

latent space yields K=65), though these models are not comparable with the rest. The output clustering from Sec. 3.3 provides M=6 trajectories based on the K samples/sigmas. It especially benefits the CUAE; its thorough output-space coverage is encapsulated into few higher quality candidates than the CVAE's clustered outputs. The >50% lower winner-NLL (only the cluster centroid closest to the ground truth is evaluated) shows that clusters on sigma trajectories are more meaningful. However the mixture-NLL, computed using ratios of members in each cluster as weights, is high. It shows that such weights are not a good proxy for the actual mixture distribution.

CXP provides an alternative to the latent prior in inference. We apply it to the clustering models in which we fit a C=50-component GMM from the training set posterior and prior encodings after training. Then, we condition it as described in Sec. 3.2 and draw K=65 samples from the conditional mixture, clustered into M=6 trajectories. CXP brings additional gains in trajectory metrics, showing that a more expressive inference distribution is beneficial. However, it does not provide an easy way to compute the mixture NLL. We approximate it via the conditional mixture weights and component mean sigma trajectories. Finally, the GMM latent model (with C=M=6) provides the best scores, significantly outperforming the previous models in trajectory- and distribution-level metrics. It shows that inducing a more expressive latent space in a CVAE context (while accordingly approximating the ELBO) brings significant performance improvements. Here, random and unscented sampling perform similarly well. However, it is important to note that the GMM reconstruction loss function in Eq. (6) already incorporates a crucial aspect of the UT, as described in Sec. 3.1. The reconstruction error is computed in a CUAE-like manner (see Eq. (4)), using the likelihood of the centroid trajectory of the winner GMM component (where the centroid is computed either by transforming random samples or sigmas), rather than in a CVAE-like manner (see Eq. (3)), using the expectation of individually reconstructed trajectories.

Overall, the obtained results comprehensively demonstrate the efficacy of our proposed alternatives to key components of the trajectory prediction CVAE, answering the research questions *(i)* and *(ii)* posed in Sec. 1. First, the unscented sampling and latent space transformation can bring greater diversity in output trajectories while outperforming the ubiquitous random sampling. Second, al-

ternative inference procedures, validated in non-prediction contexts as well, can utilize substantially more expressive distributions to draw higher-quality samples, while requiring no training overhead. Finally, greatest advantages can be found in directly training a more expressive Gaussian mixture latent space whose components map to a structured multi-modal output distribution.

5 Conclusion

In this paper, we investigated important shortcomings of the CVAE in trajectory prediction. We answered questions surrounding latent space assumptions by showing that unscented sampling and mixture models in training and inference provide high performance alternatives to existing structures. Overall, we anticipate that our findings will lead to a more effective usage of CVAE models in prediction and beyond. For future work, we plan to demonstrate our CVAE improvements on other prediction datasets (*e.g.* nuScenes [5], Argoverse [63]), using state-of-the-art models on those datasets as bases.

References

1. Arjovsky, M., Bottou, L.: Towards Principled Methods for Training Generative Adversarial Networks. arXiv preprint arXiv:1701.04862 (2017) 3
2. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein Generative Adversarial Networks. In: International conference on machine learning. PMLR (2017) 3
3. Bishop, C.M., Nasrabadi, N.M.: Pattern Recognition and Machine Learning. Springer (2006) 9
4. Borchert, O.: PyCave: Traditional Machine Learning Models for Large-Scale Datasets In PyTorch. https://github.com/borchero/pycave (2022), release used: v3.2.1 12
5. Caesar, H., Bankiti, V., Lang, A.H., Vora, S., Liong, V.E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., Beijbom, O.: NuScenes: A Multimodal Dataset for Autonomous Driving. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (2020) 15
6. Casas, S., Gulino, C., Suo, S., Luo, K., Liao, R., Urtasun, R.: Implicit Latent Variable Model for Scene-consistent Motion Forecasting. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIII 16. Springer (2020) 2, 3, 4
7. Chai, Y., Sapp, B., Bansal, M., Anguelov, D.: MultiPath: Multiple Probabilistic Anchor Trajectory Hypotheses for Behavior Prediction. arXiv preprint arXiv:1910.05449 (2019) 3
8. Choi, Y., Mercurius, R.C., Shabestary, S.M.A., Rasouli, A.: Dice: Diverse Diffusion Model with Scoring for Trajectory Prediction. In: 2024 IEEE Intelligent Vehicles Symposium (IV). pp. 3023–3029. IEEE (2024) 4
9. Cui, A., Casas, S., Sadat, A., Liao, R., Urtasun, R.: LookOut: Diverse Multi-future Prediction and Planning for Self-driving. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (2021) 2, 3, 4
10. Cui, A., Casas, S., Wong, K., Suo, S., Urtasun, R.: GoReLa: Go Relative for Viewpoint-invariant Motion Forecasting. In: 2023 IEEE International Conference on Robotics and Automation (ICRA). IEEE (2023) 3

11. Cui, H., Nguyen, T., Chou, F.C., Lin, T.H., Schneider, J., Bradley, D., Djuric, N.: Deep Kinematic Models for Kinematically Feasible Vehicle Trajectory Predictions. In: 2020 IEEE International Conference on Robotics and Automation (ICRA). IEEE (2020) 3
12. Dendorfer, P., Elflein, S., Leal-Taixé, L.: MG-GAN: A Multi-generator Model Preventing Out-of-distribution Samples In Pedestrian Trajectory Prediction. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (2021) 4, 5, 13
13. Dendorfer, P., Osep, A., Leal-Taixé, L.: Goal-GAN: Multimodal Trajectory Prediction Based On Goal Position Estimation. In: Proceedings of the Asian Conference on Computer Vision (2020) 3
14. Deo, N., Wolff, E., Beijbom, O.: Multimodal Trajectory Prediction Conditioned On Lane-graph Traversals. In: Conference on Robot Learning. PMLR (2022) 3, 10
15. Do, M.N.: Fast Approximation of Kullback-Leibler Distance for Dependence Trees and Hidden Markov Models. IEEE signal processing letters **10**(4), 115–118 (2003) 7
16. Gao, J., Sun, C., Zhao, H., Shen, Y., Anguelov, D., Li, C., Schmid, C.: VectorNet: Encoding Hd Maps and Agent Dynamics From Vectorized Representation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2020) 3
17. Ghosh, P., Sajjadi, M.S., Vergari, A., Black, M.: From Variational to Deterministic Autoencoders. In: 8th International Conference on Learning Representations (ICLR) (2020) 8, 12, 13
18. Gilles, T., Sabatini, S., Tsishkou, D., Stanciulescu, B., Moutarde, F.: HOME: Heatmap Output for Future Motion Estimation. In: 2021 IEEE International Intelligent Transportation Systems Conference (ITSC). IEEE (2021) 3
19. Gilles, T., Sabatini, S., Tsishkou, D., Stanciulescu, B., Moutarde, F.: THOMAS: Trajectory Heatmap Output with Learned Multi-agent Sampling. arXiv preprint arXiv:2110.06607 (2021) 3
20. Gilles, T., Sabatini, S., Tsishkou, D., Stanciulescu, B., Moutarde, F.: GOHOME: Graph-oriented Heatmap Output for Future Motion Estimation. In: 2022 international conference on robotics and automation (ICRA). IEEE (2022) 3, 13
21. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative Adversarial Networks. Communications of the ACM (2020) 3
22. Gu, J., Sun, C., Zhao, H.: DenseTNT: End-to-end Trajectory Prediction From Dense Goal Sets. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (2021) 3
23. Gupta, A., Johnson, J., Fei-Fei, L., Savarese, S., Alahi, A.: Social GAN: Socially Acceptable Trajectories with Generative Adversarial Networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition (2018) 3
24. Hallgarten, M., Kisa, I., Stoll, M., Zell, A.: Stay On Track: A Frenet Wrapper to Overcome Off-road Trajectories In Vehicle Motion Prediction. arXiv preprint arXiv:2306.00605 (2023) 3
25. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: GANs Trained by a Two Time-scale Update Rule Converge to a Local Nash Equilibrium. Advances in neural information processing systems (2017) 12
26. Hong, J., Sapp, B., Philbin, J.: Rules of the Road: Predicting Driving Behavior with a Convolutional Model of Semantic Interactions. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2019) 4, 11

27. Ivanovic, B., Leung, K., Schmerling, E., Pavone, M.: Multimodal Deep Generative Models for Trajectory Prediction: A Conditional Variational Autoencoder Approach. IEEE Robotics and Automation Letters (2020) 2, 3
28. Janjoš, F., Dolgov, M., Kurić, M., Shen, Y., Zöllner, J.M.: SAN: Scene Anchor Networks for Joint Action-Space Prediction. In: 2022 IEEE Intelligent Vehicles Symposium (IV). IEEE (2022) 13
29. Janjoš, F., Dolgov, M., Zöllner, J.M.: Self-supervised Action-space Prediction for Automated Driving. In: 2021 IEEE Intelligent Vehicles Symposium (IV). IEEE (2021) 11
30. Janjoš, F., Dolgov, M., Zöllner, J.M.: StarNet: Joint Action-space Prediction with Star Graphs and Implicit Global-frame Self-attention. In: 2022 IEEE Intelligent Vehicles Symposium (IV). IEEE (2022) 3, 11, 13
31. Janjoš, F., Keller, M., Dolgov, M., Zöllner, J.M.: Bridging the Gap Between Multi-Step and One-Shot Trajectory Prediction Via Self-Supervision. In: 2023 IEEE Intelligent Vehicles Symposium (IV). IEEE (2023) 13
32. Janjoš, F., Rosenbaum, L., Dolgov, M., Zöllner, J.M.: Unscented Autoencoder. In: 40th International Conference on Machine Learning (ICML) (2023) 2, 3, 4, 5, 6, 11, 12, 13
33. Jiang, C., Cornman, A., Park, C., Sapp, B., Zhou, Y., Anguelov, D., et al.: Motiondiffuser: Controllable multi-agent motion prediction using diffusion. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9644–9653 (2023) 4
34. Julier, S., Uhlmann, J., Durrant-Whyte, H.F.: A New Method for the Nonlinear Transformation of Means and Covariances In Filters and Estimators. IEEE Transactions on automatic control (2000) 6
35. Julier, S.J., Uhlmann, J.K.: Unscented Filtering and Nonlinear Estimation. Proceedings of the IEEE (2004) 5
36. Khandelwal, S., Qi, W., Singh, J., Hartnett, A., Ramanan, D.: What-if Motion Prediction for Autonomous Driving. arXiv preprint arXiv:2008.10587 (2020) 3
37. Khayatkhoei, M., Singh, M.K., Elgammal, A.: Disconnected Manifold Learning for Generative Adversarial Networks. Advances in Neural Information Processing Systems (2018) 3, 4
38. Kingma, D.P., Ba, J.: Adam: A Method for Stochastic Optimization. arXiv preprint arXiv:1412.6980 (2014) 11
39. Kingma, D.P., Welling, M.: Auto-encoding Variational Bayes. arXiv preprint arXiv:1312.6114 (2013) 5
40. Knittel, A., Hawasly, M., Albrecht, S.V., Redford, J., Ramamoorthy, S.: DiPA: Probabilistic Multi-Modal Interactive Prediction for Autonomous Driving. IEEE Robotics and Aut. Letters (2023) 3, 4, 5, 10, 13
41. Kosaraju, V., Sadeghian, A., Martín-Martín, R., Reid, I., Rezatofighi, H., Savarese, S.: Social-BiGAT: Multimodal Trajectory Forecasting Using Bicycle-GAN and Graph Attention Networks. Advances in Neural Information Processing Systems (2019) 3
42. Lee, M., Sohn, S.S., Moon, S., Yoon, S., Kapadia, M., Pavlovic, V.: Muse-VAE: Multi-scale VAE for Environment-aware Long Term Trajectory Prediction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2022) 2, 3
43. Liang, M., Yang, B., Hu, R., Chen, Y., Liao, R., Feng, S., Urtasun, R.: Learning Lane Graph Representations for Motion Forecasting. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16. Springer (2020) 3

44. Liu, Z., Luo, P., Wang, X., Tang, X.: Deep Learning Face Attributes In the Wild. In: Proceedings of International Conference on Computer Vision (ICCV) (December 2015) 10, 11
45. Mészáros, A., Alonso-Mora, J., Kober, J.: Trajflow: Learning the Distribution Over Trajectories. arXiv preprint arXiv:2304.05166 (2023) 3
46. Narayanan, S., Moslemi, R., Pittaluga, F., Liu, B., Chandraker, M.: Divide-and-conquer for Lane-aware Diverse Trajectory Prediction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2021) 3
47. Ngiam, J., Vasudevan, V., Caine, B., Zhang, Z., Chiang, H.T.L., Ling, J., Roelofs, R., Bewley, A., Liu, C., Venugopal, A., et al.: Scene Transformer: A Unified Architecture for Predicting Future Trajectories of Multiple Agents. In: International Conference on Learning Representations (2021) 3
48. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: PyTorch: An Imperative Style, High-performance Deep Learning Library. Advances in neural information processing systems (2019) 12
49. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine Learning In Python. Journal of Machine Learning Research (2011) 8, 12
50. Phan-Minh, T., Grigore, E.C., Boulton, F.A., Beijbom, O., Wolff, E.M.: CoverNet: Multimodal Behavior Prediction Using Trajectory Sets. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (2020) 3
51. Rhinehart, N., McAllister, R., Kitani, K., Levine, S.: PRECOG: PREdiction Conditioned On Goals In Visual Multi-agent Settings. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (2019) 3
52. Rolfe, J.T.: Discrete Variational Autoencoders. arXiv preprint arXiv:1609.02200 (2016) 4
53. Sadeghian, A., Kosaraju, V., Sadeghian, A., Hirose, N., Rezatofighi, H., Savarese, S.: Sophie: An Attentive Gan for Predicting Paths Compliant to Social and Physical Constraints. In: Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Rrecognition (2019) 3
54. Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., Chen, X.: Improved Techniques for Training GANs. Advances in neural information processing systems (2016) 3
55. Salzmann, T., Ivanovic, B., Chakravarty, P., Pavone, M.: Trajectron++: Dynamically-feasible Trajectory Forecasting with Heterogeneous Data. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVIII 16. Springer (2020) 2, 3, 4
56. Schöller, C., Knoll, A.: Flomo: Tractable Motion Prediction with Normalizing Flows. In: 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE (2021) 3
57. Ścibior, A., Lioutas, V., Reda, D., Bateni, P., Wood, F.: Imagining the Road Ahead: Multi-agent Trajectory Prediction Via Differentiable Simulation. In: 2021 IEEE International Intelligent Transportation Systems Conference (ITSC). IEEE (2021) 13
58. Sohn, K., Lee, H., Yan, X.: Learning Structured Output Representation Using Deep Conditional Generative Models. Advances in neural information processing systems (2015) 4

59. Tanielian, U., Issenhuth, T., Dohmatob, E., Mary, J.: Learning Disconnected Manifolds: a No Gan's Land. In: International Conference on Machine Learning. PMLR (2020) 3, 4
60. Varadarajan, B., Hefny, A., Srivastava, A., Refaat, K.S., Nayakanti, N., Cornman, A., Chen, K., Douillard, B., Lam, C.P., Anguelov, D., et al.: MultiPath++: Efficient Information Fusion and Trajectory Aggregation for Behavior Prediction. In: 2022 International Conference on Robotics and Automation (ICRA). IEEE (2022) 3
61. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention Is All You Need. Advances in neural information processing systems (2017) 11
62. Wang, L., Schwing, A., Lazebnik, S.: Diverse and Accurate Image Description Using a Variational Auto-encoder with An Additive Gaussian Encoding Space. Advances in Neural Information Processing Systems **30** (2017) 11
63. Wilson, B., Qi, W., Agarwal, T., Lambert, J., Singh, J., Khandelwal, S., Pan, B., Kumar, R., Hartnett, A., Pontes, J.K., et al.: Argoverse 2: Next Generation Datasets for Self-driving Perception and Forecasting. arXiv preprint arXiv:2301.00493 (2023) 15
64. Yuan, Y., Weng, X., Ou, Y., Kitani, K.M.: Agentformer: Agent-aware Transformers for Socio-temporal Multi-agent Forecasting. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (2021) 2, 3, 4
65. Zhan, W., Sun, L., Wang, D., Shi, H., Clausse, A., Naumann, M., Kummerle, J., Konigshof, H., Stiller, C., de La Fortelle, A., et al.: Interaction Dataset: An International, Adversarial and Cooperative Motion Dataset In Interactive Driving Scenarios with Semantic Maps. arXiv preprint arXiv:1910.03088 (2019) 11
66. Zhao, H., Gao, J., Lan, T., Sun, C., Sapp, B., Varadarajan, B., Shen, Y., Shen, Y., Chai, Y., Schmid, C., et al.: TNT: Target-driven Trajectory Prediction. In: Conference on Robot Learning. PMLR (2021) 3