Generative deep learning for advanced battery materials

Zur Erlangung des akademischen Grades einer
DOKTORIN DER INGENIEURWISSENSCHAFTEN (Dr.-Ing.)

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

DISSERTATION

von

Deepalaxmi Rajagopal, M.Sc.

Tag der mündlichen Prüfung:	26. January 2026
Referentin:	Prof. Dr. Britta Nestler
Korreferent:	Prof. Dr. Helge Stein

Abstract

The development of advanced battery materials has relied on a combination of physics-based simulations and experimental techniques for material characterization and electrochemical testing. These methods have significantly contributed to understanding batteries, including electrochemical stability, safety, and degradation processes. While practical, these conventional approaches face long development cycles to achieve desired performance criteria. The research and development of battery materials can be expedited by leveraging data-driven techniques that derive insights across a broad material space. Among these data-driven approaches, generative deep learning captures complex, high-dimensional patterns in experimental and simulation data. In contrast, discriminative learning interprets and contracts high-dimensional information into meaningful, reduced representations. Coupling generative and discriminative deep learning frameworks enables both expansion and distillation of knowledge about the material design space. This thesis proposes a generative-discriminative deep learning framework to accelerate early-stage screening, characterization, and guided generation of advanced battery materials. The implemented framework is demonstrated across multiple application modes. The first demonstration of the framework focuses on a data-driven analysis to learn information-rich latent representations of the formation and evolution of solid electrolyte interphases using physical and data-driven properties derived from simulations. Later developments demonstrate automated structure and composition discovery from characterization data and early-stage screening of candidate battery materials based on functional properties, prior to any experiments or computational validation. By unifying analysis, characterization, and screening, the framework provides a cohesive approach to accelerate data-driven design of advanced battery materials.

Kurzfassung

Die Entwicklung neuartiger Batteriematerialien baut auf einer Kombination aus physikbasierten Simulationen sowie experimentellen Verfahren zur Materialcharakterisierung und elektrochemischen Untersuchung auf. Diese Verfahren zur Materialentwicklung haben wesentlich zum Verständnis von Batterien, einschließlich der elektrochemischen Stabilität, Sicherheit und Degradationsprozessen, beigetragen. Diese konventionellen Ansätze sind zwar praktikabel, erfordern jedoch einen langen Entwicklungsprozess, um die angestrebte Batterieleistung zu erreichen. Die Forschung und Entwicklung von Batteriematerialien können durch den Einsatz datengetriebener Techniken beschleunigt werden, um Erkenntnisse über einen umfangreichen Materialraum zu gewinnen. Unter den datengetriebenen Ansätzen erfasst generatives Deep Learning komplexe, hochdimensionale Verhaltensmuster aus experimentellen und simulativen Daten. Im Gegensatz dazu sind diskriminative Modelle darauf ausgelegt, diese hochdimensionalen Daten zu interpretieren und zu sinnvollen Vorhersagen zu verdichten. Die Kopplung generativer und diskriminativer Deep-Learning-Frameworks ermöglicht somit sowohl die Erweiterung als auch die Verdichtung des Wissens über den Materialdesignraum. Diese Dissertation demonstriert ein generativ-diskriminativeres Deep-Learning-Framework zur Beschleunigung der Früherkennung, Charakterisierung und gezielten Erzeugung neuartiger Batteriematerialien. Das implementierte Framework wird in mehreren Anwendungsmodi demonstriert. Die erste Demonstration des Frameworks konzentriert sich auf eine datengetriebene Analyse zum Erlernen informationsreicher latenter Abbildungen der Entstehung und Entwicklung von festen Elektrolytgrenzschichten in Batterien. Dieser Anwendungsmodus ist zur Darstellung und Charakterisierung der Bildung von festen

Elektrolytgrenzschichten auf Basis physikalischer und datengetriebener Eigenschaften aus Simulationen konzipiert. Die nachfolgenden Entwicklungen des Frameworks demonstrieren die automatisierte Ermittlung von Struktur und Zusammensetzung aus Charakterisierungsdaten sowie das frühzeitige, gezielte Screening von Batteriematerialkandidaten auf Basis der funktionalen Eigenschaften vor jeglichen Experimenten oder rechnerischen Validierungen. Durch die Vereinigung von Analyse-, Charakterisierungs- und Screening-Aufgaben bietet das Framework einen durchgängigen Ansatz zur Beschleunigung des datengetriebenen Designs neuartiger Batteriematerialien.

Acknowledgments

This thesis is the result of the collective efforts and support of many individuals. First of all, I would like to express my deepest gratitude to Univ.-Prof. Dr. rer. nat. Britta Nestler for providing me with the opportunity to pursue research at the institute and for her supervision throughout this work. I am also thankful to Univ.-Prof. Dr.-Ing. Helge Stein for his role as co-supervisor and for his insights, particularly during our collaboration on my first publication.

My heartfelt thanks go to my group leader, Dr.-Ing. Arnd Koeppe, for his constructive feedback, encouragement, and unwavering guidance throughout this work. His insights steered this work in the right direction and led to its successful completion. I would also like to thank Dr.-Ing. Michael Selzer, group leader of the Kadi4Mat team, for his support throughout this work.

To my colleagues, Giovanna Tosato, Lars Griem, Julian Grolig, Yinghan Zhao, Adrian Cierpka, Muhammed Kocak, and many others, I thank you not only for stimulating discussions and thoughtful advice but also for turning stressful research days into occasions for laughter. I would also like to thank the Deutsche Forschungsgemeinschaft's "Cluster of Excellence" POLiS for funding this research.

Above all, I owe so much to my parents, whose encouragement and sacrifices made it possible for me to chase my dreams in Germany. And to my husband, thank you for being there with endless patience, love, and support. I definitely couldn't have finished this without you.

Karlsruhe, November 2025 *Deepalaxmi Rajagopal*

Notations

The mathematical notations used in this thesis are formulated based on the following structure.

The $\boldsymbol{x}$ denotes a data sample belonging to a data domain $\mathcal{X} \subseteq \mathbb{R}^D$, drawn from a probability distribution $p(\boldsymbol{x})$, i.e., $\boldsymbol{x} \sim p(\boldsymbol{x})$.

In unsupervised learning, the dataset $\mathcal{D}$ contains N samples arranged in a matrix $\boldsymbol{X} \in \mathbb{R}^{N \times D}$, where each row is a sample and columns define its D-dimensional features. In supervised learning, each sample $\boldsymbol{x}$ has an associated label y.

The datasets in this dissertation contain multidimensional, multichannel data. For simplicity, spatial tensors are conceptually flattened so each input sample is a vector $\boldsymbol{x}$. A latent variable is denoted by $\boldsymbol{z} \in \mathcal{Z}$, drawn from a distribution $p(\boldsymbol{z})$.

Vectors are represented by bold lowercase Roman letters, e.g., $\boldsymbol{a}$. Matrices and higher-order tensors are written in bold uppercase Roman letters, e.g., $\boldsymbol{A}$.

Contents

1 Introduction

1.1 Motivation

Batteries may seem simple electrochemical systems composed of four components: two electrodes, an electrolyte, and a separator. However, developing efficient energy storage systems requires a deep understanding of how components interact and the kinetics of the electrochemical reactions involved. The performance of batteries depends on a network of interdependent physicochemical processes, including ion transport, phase transformations, interfacial reactions, and mechanical deformations. Each of these interdependent processes occurs over multiple length and time scales, from the nanoscale, where ion transport and interfacial reactions predominantly happen, to the mesoscale, where phase transformations and mechanical deformations are more significant [1–4]. Advancements in experimental and simulation techniques [5–7] have advanced the understanding of battery materials and their interactions across multiple scales. However, most experimental and simulation methods are constrained by instrumentation parameters and underlying physics, limiting their ability to derive insights to specific length and time scales [8].

Traditional materials discovery has been instrumental in driving progress, relying on careful experimentation and intuition developed through experience. This approach excels at uncovering valuable insights within well-explored regions of chemical space and has led to many foundational breakthroughs. However, as the number of possible material combinations continues to expand, it becomes challenging to explore new chemistries and understand complex physicochemical properties that govern battery performance [9–11].

Just as explorers once charted the unknown islands of distant seas, researchers now face vast, uncharted territories of chemical combinations, waiting to be discovered and understood. This adventure into the unexplored chemical "white space" holds the promise of unlocking innovative solutions for next-generation batteries.

Data-driven approaches empower systematic exploration of complex chemical spaces and extraction of transferable structure-property relationships in battery materials research. Machine Learning (ML), a subset of Artificial Intelligence (AI), employs a data-driven methodology replacing traditional understanding of fundamental physical processes with observations of the system under study [12–14]. In the scope of ML, data-driven models are widely classified into discriminative and generative models. Discriminative models distill information representation by learning predictive boundaries between classes, focusing on features relevant to a specific outcome. Generative models, in contrast, expand information representation by modeling entire data distributions to generate new, realistic samples that explore previously uncharted regions of chemical space [13, 3]. Leveraging a generative model in tandem with discrimination offers access to expansive and contractive knowledge of material design pathways.

Classical ML algorithms typically rely on predefined assumptions and manual feature engineering to model the relationship between input and output variables. While effective for structured, low-dimensional data, they often struggle with high-dimensional, heterogeneous datasets. Neural networks overcome these limitations by mimicking the way biological neurons work together to identify patterns, assess options, and reach conclusions. Each neural network is composed of layers of interconnected units, known as neurons, that process data and identify nonlinear patterns through weighted connections, bias terms, and nonlinear activation functions. Traditional shallow neural networks can model nonlinear relationships, but their limited depth limits their ability to capture complex patterns with a finite number of neurons.

Deep learning enhances traditional frameworks by incorporating architectures with multiple hidden layers that automatically learn hierarchical representations. The transition from standard neural networks to deep learning signifies a shift from constrained feature learning to hierarchical modeling. Such advancements reduce the need for manual feature engineering and improve performance on high-dimensional, unstructured data. Incorporating generative modeling concepts into deep learning frameworks enables learning the underlying data distribution, which in turn enables interpolation, optimization, and the exploration of novel materials for target applications through representation learning.

1.2 Objective

Addressing the limitations of traditional battery development, this thesis aims to develop a unified data-driven framework that integrates generative and discriminative methods to automate and accelerate the characterization, screening, and generation of advanced battery materials. By coupling forward characterization (discriminative learning) with inverse design (generative learning), the framework enables contractive-expansive navigation of the material design landscape, thereby accelerating early-stage battery material design (Figure 1.1). The practical outcome of the framework enables battery researchers to make informed decisions across vast material design spaces and to generate information-rich target battery chemistries.

The proposed framework establishes a latent-space-driven approach that links material configurations, physical descriptors, and electrochemical performance by capturing intrinsic correlations among these domains. Within this unified latent space, the discriminative component serves as a contractive pathway, encoding essential information to guide material design, while the generative component provides an expansive pathway that reconstructs or generates novel material configurations optimized for specific functional targets such as average voltage, capacity, and energy density. By implementing this bidirectional mapping, the framework enables automated screening, targeted generation, and characterization of structure-property linkages.

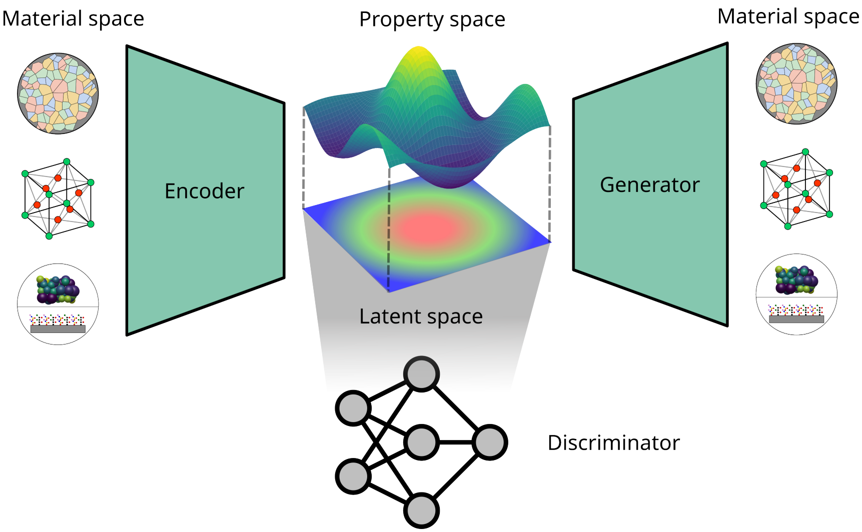


Figure 1.1: Schematic overview of generative-discriminative framework. The encoder and discriminator define the contraction of high-dimensional material data into information-rich, lower-dimensional latent knowledge. The generator defines the expansion of knowledge from the latent representation to high-dimensional data.

The generative-discriminative approach enables a data-centered understanding of battery material behavior and enhances interpretability across multiple scales. This framework accelerates predictive characterization, screening large material spaces, and generating material configurations that meet target physical and functional properties.

1.3 Overview on generative deep learning in battery research

In battery research, generative deep learning models are an emerging field that extends beyond predictive modeling to explore structure-property-function relationships. While experimental and high-throughput computational screening methods have accelerated the search for new materials, such approaches still only explore a small fraction of the vast chemical and structural design space. Generative deep learning presents a complementary inverse design approach, which captures the relationship between structures and properties within a continuous and information-rich latent space. Building on the review by Rajagopal et al., [15], this section of the thesis explores the generative deep learning frameworks developed in battery research at the material level design. In the context of battery materials, the crystal structures, molecular interphase design, and microstructure morphologies of the material dictate the electrochemical performance of the cell throughout its cycle life. Understanding the structure-function relationship is critical in defining the operational limitations and degradation pathways. The design and discovery of the materials that constitute the components of batteries can be studied in different forms of representations, such as molecular, chemical, and structural space.

The crystal structure of electrode materials affects battery performance, influencing ion diffusion and conductivity. The generative frameworks, such as iMatGen by Noh et al. [16] and CrystalGAN by Nouira et al. [17], demonstrate early efforts in the novel and stable crystal polymorphs of inorganic solids using Variational AutoEncoder (VAE) with a stability classifier and Generative Adversarial Network (GAN) based approaches, respectively. Building on this, Zhao et al. [18] developed CubicGAN, conditioned on crystal space group embedding and element embedding, to guide the density functional theory (DFT) validated screening of new cubic functional materials for applications such as solar panels. To capture periodicity and chemical invariance of materials, Xie et al. [19] developed the Crystal Diffusion Variational Autoencoder (CDVAE), integrating graph-based embeddings with an inductive bias toward stability and a diffusion refinement process to generate more stable materials. This enables CDVAE to reconstruct valid, diverse, and realistic materials, as well as generate materials with target properties. Yang et al. [20] introduced UniMat, a diffusion-based model designed to scale across the periodic table via a unified representation of materials. UniMat addresses challenges in generative design by handling systems with arbitrary chemical compositions and structural complexity, by encoding continuous values of atomic locations at the corresponding element entries in the periodic table as a four-dimensional tensor. The conditional generation with UniMat can scale to previously established crystal datasets, containing up to millions of crystal structures, and outperform random structure search in discovering new stable materials. MatterGen is another diffusion-based generative model developed by Microsoft AI4Science that uses a diffusion process tailored for the design of crystalline materials. In addition to the diffusion process, which gradually refines atom types, the coordinate and periodic lattice adaptation module is used to fine-tune the desired chemical composition, symmetry, and property constraints to generate stable, unique, and novel materials [21].

At the molecular and interphase scale, generative learning has been applied to the design of electrolytes, additives, and polymers, which are critical for ion transport, interphase stability, and safety in battery systems. Yoon et al. [22] developed a natural product VAE framework to generate novel electrolyte additives and simultaneously predict their electronic properties, such as the Highest Occupied Molecular Orbital (HOMO) and the Lowest Unoccupied Molecular Orbital (LUMO). The predicted HOMO and LUMO electron properties provide insight into the oxidative and reductive stability of the solid electrolyte interphase in batteries. Moving towards the design of polymer electrolytes, Khajeh et al. [23] utilized a minimal version of the generative pre-trained transformer (minGPT) to generate polymer electrolytes optimized for high ionic conductivity. Yang et al. [24] compared minGPT and diffusion approaches for polymer design, identifying candidates with improved conductivity and structural diversity.

The microstructural features of the electrode play an essential role in determining electrode performance. The microstructural features of the electrodes, including porosity, tortuosity, and particle connectivity, determine how ions and electrons move through the electrodes. Gayon Lombardo et al. [25] used a deep convolutional GAN (DCGAN) to generate three-dimensional multiphase electrode microstructures with periodic boundary conditions. Periodic boundaries in microstructures enable smaller yet representative simulation volumes, significantly reducing the computational cost of electrochemical simulations needed to evaluate microstructure performance during optimization. Following this, Kench et al. [26] developed SliceGAN to generate three-dimensional microstructures from two-dimensional images. Traditional two-dimensional imaging often overlooks important volumetric properties necessary for understanding electrochemical processes, such as ion transport and fluid flow in battery electrodes. Using SliceGAN, researchers can generate realistic 3D representations of battery microstructures, which enhances the simulations of electrochemical and mechanical behaviors within battery systems. Dahari et al. [27] extended on the earlier implementation by introducing a GAN-based fusion framework that combines complementary 2D and 3D imaging datasets to create high-resolution, multiphase reconstructions of battery cathode mesostructures. This fusion approach addresses the trade-off between field of view and resolution typically seen in conventional imaging, producing realistic mesoscale volumes that accurately depict pore networks and binder domains, which are important for transport and degradation studies.

1.4 Outline

This thesis builds upon recent advancements in generative deep learning applied to battery materials research by developing latent-powered generative-discriminative frameworks that facilitate continuous representation learning, automated material characterization, and data-driven design of battery materials. The application of the implemented framework is demonstrated through representation learning of solid electrolyte interphases, automated translation of battery electrodes into material descriptors, and application-targeted screening of battery materials based on their functional properties. The thesis is structured as follows:

- **Chapter 2** introduces the foundational concepts of data-driven modeling and machine learning, followed by the principles of generative modeling, the motivation for deep learning approaches, an overview of state-of-the-art generative model formulations, and strategies for model evaluation.

- **Chapter 3** establishes a methodological framework for applying generative deep learning to advanced battery materials, covering data acquisition and preprocessing, model selection criteria, and different application modes.
- **Chapter 4** demonstrates the results of the thesis contributions, including the characterization of latent space, the automated translation of characterization data into material descriptors, and the virtual screening of battery materials using generative workflows.
- **Chapter 5** concludes the thesis with a discussion on key findings and an outlook on future research directions for generative deep learning in advanced battery material design and analysis.

2 Theoretical foundation

2.1 Concepts of data-driven modeling

Data-driven modeling is defined as the process of using observed data of the system to discover, approximate, and predict the functional relationships of the system without requiring explicit knowledge of its internal physical mechanisms. In data-driven modeling, parametric and non-parametric approaches provide two different strategies to approximate the system behavior. Parametric models assume a specific functional form to fit the input-output relationship, defined by its set of parameters. Non-parametric models do not assume a fixed form and rely on the observed data of the system under consideration. This does not mean that the non-parametric model is free of parameters. Instead, the number of parameters in non-parametric data-driven modeling increases as the dataset size grows. Parameter estimation in data-driven modeling is formulated as an optimization problem to find the model hyperparameters that best represent the system's underlying behavior. Once the model structure is selected, the estimation algorithm searches for parameters that minimize a suitable loss function. As systems transition from linear to nonlinear and from low-dimensional to high-dimensional spaces, the complexity of estimation increases, necessitating diverse approaches that range from classical statistical methods to modern deep learning techniques. The estimation algorithm used for parameter optimization of data-driven models includes classical methods such as least squares and recursive least squares, as well as Computational Intelligence (CI) methods such as fuzzy logic systems, genetic algorithms, and Machine Learning (ML)[28, 29]. ML-based estimation methods involve algorithms that learn input-output relationships directly from data by minimizing a loss function over the model parameter space. Deep Learning (DL) is a subset of machine learning that utilizes multi-layer neural network architectures to capture complex nonlinear relationships in data.

2.1.1 Fundamentals of machine learning

ML is a subset of Artificial Intelligence (AI) that focuses on developing algorithms and models that enable systems to improve performance on a task by learning from experience without being explicitly programmed. In the context of ML, experience refers to the accumulation of past information, often in the form of large, electronically collected datasets. The quality of the collected data is a critical determinant of predictive success. ML provides insights into structures and patterns within these large datasets [12, 14]. The following is the list of definitions and terminology used in ML:

- **Samples**: Samples are instances of collected data utilized for learning or evaluation. Datasets are commonly partitioned into subsets: training samples are used to fit the model, validation samples are employed to tune and select model parameters, and test samples are reserved for evaluating the model's performance.

- **Features**: Features are attributes associated with each sample. These may include individual variables, derived variables, or composite attributes constructed from underlying data elements.

- **Label**: A label is the target value or category assigned to each data sample in supervised learning. Formally, the label represents the output variable that the model is trained to predict, given a set of input features.

- **Hyperparameters**: Hyperparameters are high-level parameters that govern the learning process of a model. They are adjusted to optimize model performance, training efficiency, and generalization capability.

- **Loss function**: The loss function quantifies model performance by measuring the discrepancy between predicted and target labels. Model optimization involves adjusting parameters to minimize the loss function, also known as the cost function or error function. Various types of loss functions are used depending on the specific ML task, which will be discussed in the Section 2.6.1

Assuming a dataset $\mathcal{D}$ consisting of N independently and identically distributed samples. Each sample is represented as a pair $(\boldsymbol{x}^{(i)}, y^{(i)})$, where $\boldsymbol{x}^{(i)}$ denotes the feature vector and $y^{(i)}$ is the corresponding label. The dataset of N samples can thus be written as $\mathcal{D} = \{(\boldsymbol{x}^{(1)}, y^{(1)}), (\boldsymbol{x}^{(2)}, y^{(2)}), \ldots, (\boldsymbol{x}^{(N)}, y^{(N)})\}$. The goal of ML is to find the optimal function $f^*(\boldsymbol{x})$ from a hypothesis space $\mathcal{F} = \{f(\boldsymbol{x}; \boldsymbol{\theta}) \mid \boldsymbol{\theta} \in \mathbb{R}^d\}$ that best approximates the actual mapping between $\boldsymbol{x}$ and y. The $\boldsymbol{\theta}$ represents the model parameters, and d is the number of parameters. The predicted value can be expressed as, $\hat{y} = f^*(\boldsymbol{x})$ and $p(y \mid \boldsymbol{x}) = f_y^*(\boldsymbol{x})$, where $f_y^*(\boldsymbol{x})$ represents the conditional probability of the label y given input $\boldsymbol{x}$. The process of finding an optimal function is called training.

The main components of building an ML algorithm include the model, the loss function for evaluating the model's performance, and the optimization algorithm.

The model defines the hypothesis space, which can be broadly classified into linear and nonlinear models. The hypothesis space of linear models is expressed through $f(\boldsymbol{x}; \boldsymbol{\theta}) = \boldsymbol{w}^T \boldsymbol{x} + b$ with parameters $\boldsymbol{\theta} = (\boldsymbol{w}, b)$. The $\boldsymbol{w}$ represents the weight vector and the bias b. The hypothesis space of nonlinear models can be expressed through a linear combination of multiple nonlinear basis functions: $f(\boldsymbol{x}; \boldsymbol{\theta}) = \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}) + b$, where $\boldsymbol{\phi}(\boldsymbol{x}) = [\phi_1(\boldsymbol{x}), \phi_2(\boldsymbol{x}), \ldots, \phi_K(\boldsymbol{x})]^T$ is a vector of K nonlinear mappings.

The learning criterion specifies how well a candidate model matches the true underlying relationship. The learning criterion is formalized by a loss function, which measures the discrepancy between predicted outputs and target labels. Assuming training data samples are drawn from an unknown but fixed joint probability distribution $p(\boldsymbol{x}, y)$, the learning objective is to approximate the true mapping function $y = g(\boldsymbol{x})$ or, equivalently, the conditional distribution $p(y \mid \boldsymbol{x})$. The performance of the model $f(\boldsymbol{x}; \boldsymbol{\theta})$ can be measured by the expected risk $\mathcal{R}(\boldsymbol{\theta})$, which is defined as $\mathcal{R}(\boldsymbol{\theta}) = \mathbb{E}_{(\boldsymbol{x},y)\sim p_r(\boldsymbol{x},y)}[\mathcal{L}(y, f(\boldsymbol{x}; \boldsymbol{\theta}))]$, where $p_r(\boldsymbol{x}, y)$ is the real data distribution, $\mathcal{L}\left(y, f(\boldsymbol{x}; \boldsymbol{\theta})\right)$ is the loss function, which is used to quantify the difference between two variables. Since the true data distribution is unknown, this expectation is approximated by the empirical risk computed on the available training dataset. The empirical risk is given by,

$$\mathcal{R}_{\text{emp}}(\boldsymbol{\theta}) = \frac{1}{N}\sum_{n=1}^{N} \mathcal{L}(y^{(n)}, f(\boldsymbol{x}^{(n)}; \boldsymbol{\theta})) \tag{2.1}$$

where $f(\boldsymbol{x}^{(n)}; \boldsymbol{\theta})$ denotes the model prediction for input $\boldsymbol{x}^{(n)}$ given parameters $\boldsymbol{\theta}$ and $\mathcal{L}(\cdot)$ represents the differentiable loss function. This empirical risk minimization principle forms the foundation of most machine learning algorithms, as it provides a practical way to optimize model performance using finite training data. The quality of this approximation depends on the size and diversity of the training dataset, with larger and more diverse datasets generally leading to better approximations of the true risk. The goal is to determine the optimal parameters $\boldsymbol{\theta}^*$ that minimize the empirical risk,

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathcal{R}_{\text{emp}}(\boldsymbol{\theta}) \tag{2.2}$$

However, this optimization problem is often challenging to solve in closed form, particularly for complex nonlinear models. Direct analytical solutions exist for linear models with simple loss functions, such as the ordinary least squares. However, for nonlinear models with a non-convex loss function or with certain type of regularization parameter, a closed-form solution is unavailable and iterative optimization methods are required [30, 31].

Gradient descent [32] is an iterative optimization technique to find the parameters of the model that minimize a given objective function. The first-order Taylor expansion of $\mathcal{R}_{\text{emp}}(\boldsymbol{\theta})$ around the current parameters $\boldsymbol{\theta}_t$ is given by,

$$\mathcal{R}_{\text{emp}}(\boldsymbol{\theta}) \approx \mathcal{R}_{\text{emp}}(\boldsymbol{\theta}_t) + \nabla_{\boldsymbol{\theta}}\mathcal{R}_{\text{emp}}(\boldsymbol{\theta}_t)^T(\boldsymbol{\theta} - \boldsymbol{\theta}_t) \tag{2.3}$$

The gradient $\nabla_{\boldsymbol{\theta}} \mathcal{R}_{\text{emp}}(\boldsymbol{\theta}_t)$ points in the direction of steepest increase of $\mathcal{R}_{\text{emp}}(\boldsymbol{\theta})$. To minimize the empirical risk, parameters are updated in the opposite direction. The update of model parameters is then written as,

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \nabla_{\boldsymbol{\theta}} \mathcal{R}_{\text{emp}}(\boldsymbol{\theta}_t), \tag{2.4}$$

Where $\eta > 0$ is the learning rate controlling the step size. Substituting the definition of empirical risk yields the explicit update rule,

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \frac{\eta}{N} \sum_{n=1}^{N} \nabla_{\boldsymbol{\theta}} \mathcal{L}\big(y^{(n)}, f(\boldsymbol{x}^{(n)}; \boldsymbol{\theta}_t)\big) \tag{2.5}$$

The performance of gradient descent largely depends on factors such as the choice of learning rate, the properties of the loss function, including convexity and smoothness, as well as the initial values of the model parameters

Different variants of gradient descent have been proposed to improve the convergence rate, convergence point, and scalability of the algorithm. Each variant has its own advantages and disadvantages, and the choice of which variant to use depends on the specific problem at hand.

Batch gradient descent calculates the gradient of the cost function using the entire dataset, which provides stable convergence by averaging over all samples and is preferred when the dataset is small and fits in memory. Stochastic gradient descent (SGD) [33], on the other hand, updates the parameters using a single randomly selected training sample at each iteration. This makes SGD computationally efficient and capable of escaping shallow local minima, the noisy updates often lead to fluctuating convergence and require careful tuning of the learning rate. Mini-batch gradient descent combines the advantages of batch and stochastic gradient descent methods by computing the gradient using a subset of training data at each step. The significant challenges in the above discussed gradient descent algorithms include choosing the correct learning rate. If the learning rate is too high, the model overshoots the minimum, and if the learning rate is too low, the model takes a longer time to converge. To address the issues of slow and unstable convergence, momentum-based variants of gradient descent algorithms are used. The momentum-based gradient descent method enhances convergence by utilizing a velocity vector that accumulates past gradients to accelerate consistent gradient directions and dampen oscillations in the optimization path. In addition to momentum-based methods, gradient descent with adaptive learning rates, such as Adam [34], Adagrad [35], and RMSprop [36], adjust the learning rate for each parameter based on the gradient history. Adaptive gradient methods allow faster and stable convergence across varying loss surfaces [37].

2.1.2 Need for deep learning

The performance of classical ML algorithms, such as logistic regression, depends heavily on the well-defined features. Raw data, such as pixel intensities in images or word frequencies in text, often exhibit high dimensionality, redundancy, and noise, which hinder both efficiency and generalization. Therefore, classical ML algorithms require handcrafted feature engineering based on feature selection and feature extraction to construct meaningful data representations.

Feature selection identifies informative subsets of features using various strategies. Filter-based feature selection strategies are model-agnostic, evaluating features based on statistical measures such as information gain, feature correlations, and variances, to rank and select the relevant features. Wrapper methods use specific ML algorithms to search for the best performing subset of features. Wrapper methods are computationally expensive, as they must train many ML models, yet provide high accuracy due to the evaluation of the actual model. Embedded methods incorporate feature selection as part of training through a regularization technique. The applied regularization during training penalizes unimportant features by driving the corresponding weights to zero, yielding a sparse model with only the most predictive inputs [38, 39].

Feature extraction is a complementary approach to feature selection, which focuses on transforming the original feature space into a new space that captures the most relevant information for the predictive task. Classical feature extraction techniques include principal component analysis (PCA), an unsupervised method that identifies orthogonal linear combinations of features that maximize the variance in the data, and linear discriminant analysis, a supervised method that separates multiple classes and features to solve classification problems. Manual feature selection and extraction can improve machine learning performance, but function as separate preprocessing steps. This separation reduces effectiveness and leads to a mismatch between the learned feature space and training objectives [40, 41].

Deep learning is a specialized subset of machine learning that uses neural networks with multiple layers to model complex patterns in large datasets. Unlike traditional machine learning, deep learning models automatically learn to extract relevant features from raw input data. This reduces the need for manual feature engineering, allowing the model to discover intricate patterns and representations that may not be easily discernible by humans [42, 43].

2.1.3 Neural networks

Neural networks are inspired by the human brain, which consists of interconnected nodes called neurons, organized into layers. The three primary components of a neural network architecture are the input layer, the hidden layer, and the output layer. The input layers comprise neurons that represent the features of the input data. The hidden layers are the intermediate layers between the input and output layers. The number and size of the hidden layers can vary, contributing to the depth of the network. Each hidden layer applies a set of weights and biases to the input data, followed by an activation to introduce nonlinearity. The output layer produces the final predictions. The number of neurons in the output layer depends on the nature of the problem [44, 45].

Figure 2.1 illustrates the standard model of neuron j, which comprises an affine transformation followed by a non-linear activation function. Given an input signal $\boldsymbol{x} \in \mathbb{R}^n$ and an output signal $\boldsymbol{h}_j$, the neuron output is computed as follows:

$$\boldsymbol{h}_j = g\Big(\sum_{i=1}^{n} \boldsymbol{w}_{ji}\boldsymbol{x}_i + \boldsymbol{b}_j\Big). \tag{2.6}$$

In this formulation, $\boldsymbol{w}_{j1}, \boldsymbol{w}_{j2}, \ldots, \boldsymbol{w}_{jn}$ denote the connection weights associated with the respective inputs $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n$. The weighted sum is combined with the bias term $\boldsymbol{b}_j$, and $g(\cdot)$ denotes the selected activation function [46].

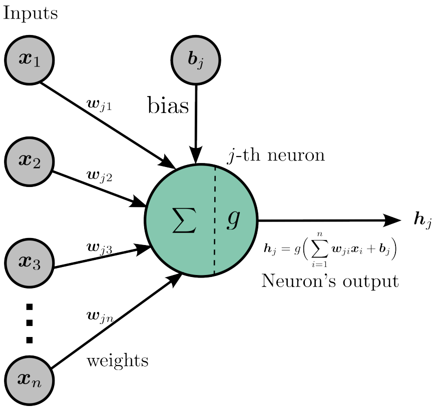


Figure 2.1: Structure of the artificial neuron. The $\boldsymbol{x}_i$ represents the input signal to the j-th neuron and $\boldsymbol{w}_{ji}$ defines the corresponding weights. The b_j represents the neuron's bias, and $\boldsymbol{h}_j$ is the neuron's output.

Activation function of the neuron introduces nonlinearity, allowing the network to learn complex patterns. A purely linear activation function, such as $g(z) = z$, reduces the network to a linear model regardless of its depth. To capture the non-linear relationship in the data, non-linear activation functions are used. The most commonly used non-linear activation functions include sigmoid, hyperbolic tangent (Tanh), rectified linear unit (ReLU), Leaky ReLU, softplus, and Swish [47, 48].

Sigmoid function is one of the earliest and simplest non-linear functions applied in neural networks. The sigmoid function is mathematically expressed as,

$$g(z) = \frac{1}{1 + e^{-z}}, \tag{2.7}$$

which maps the output to the range $[0, 1]$. The sigmoid function has an S-shaped curve with a smooth gradient, an important property for gradient-based optimization methods. One of the other important characteristics of the sigmoid function is that for z values between -2 and 2, the curve is steep, meaning that small changes in the input can produce large changes in the output, which facilitates rapid learning during training. However, the output of the sigmoid function is saturated for both higher and lower inputs, which leads to the vanishing gradient problem, where weight updates in earlier layers become negligible, slowing down the training of deep networks (Figure 2.2). Sigmoid functions are widely used in the output layer of the network, particularly in cases where the outputs need to be interpreted as probabilities, such as in classification problems.

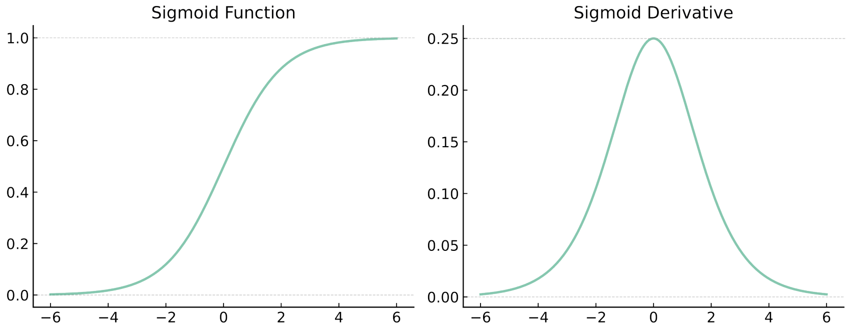


Figure 2.2: Sigmoid activation function and its derivative. The function produces smooth, bounded outputs suitable for probability modeling, but gradients shrink near the extremes, which can slow learning in deep networks.

Tanh function is another widely used non-linear function and can be interpreted as a scaled and shifted version of the sigmoid function. The Tanh function is mathematically expressed as

$$g(z) = \tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}, \tag{2.8}$$

The Tanh function also has a bounded output in the range of $[-1, 1]$, which makes it zero-centered. The symmetry in tanh functions ensures that the positive and negative activations are balanced. The tanh function has a steeper gradient than the sigmoid, allowing more effective weight updates and faster convergence. As shown in Figure 2.3, the zero-centered output balances activations, reducing bias in subsequent layers. However, in deep networks, gradients can vanish, slowing learning, so careful weight initialization and input normalization are essential. The tanh function is often preferred over the sigmoid function when zero-centered outputs and stronger gradients are needed.

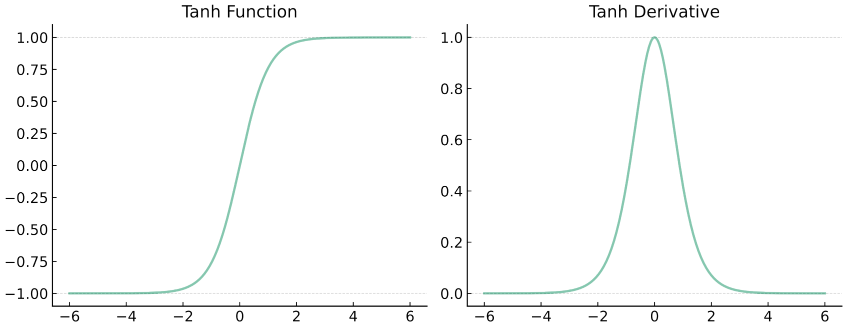


Figure 2.3: Tanh activation function and its derivative. The function is zero-centered and provides stronger gradients than sigmoid, though gradients still diminish for large input magnitudes.

ReLU is a piecewise linear function and is mathematically described as

$$g(z) = max(0, z) = \begin{cases} z, & \text{if } z \geq 0, \\ 0, & \text{otherwise} \end{cases} \tag{2.9}$$

which activates neurons only if the input is positive and zero otherwise, providing sparse and efficient representations. The range of ReLU is $[0, \infty)$. The downside of ReLU is the "dying ReLU" problem, where the input to a neuron falls into a negative region, resulting in an output of 0 (Figure 2.4). Hence, the gradients of the neuron will also be zero. As a result, during back propagation, the weights of that neuron remain unchanged. This can result in a portion of the neural network becoming inactive, reducing the network's representational capacity if a large fraction of neurons die.

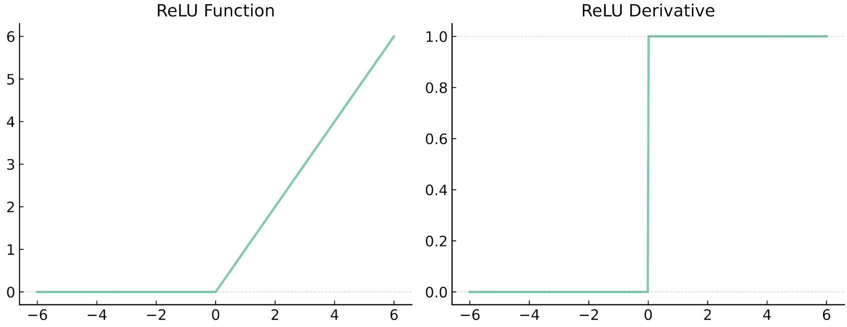


Figure 2.4: ReLU activation function and its derivative. The function efficiently propagates gradients for positive inputs, but neurons can become inactive when inputs remain negative.

LeakyReLU addresses the issue of dying ReLU by introducing a small negative slope α when $z < 0$, which can be expressed as

$$g(z) = \begin{cases} z, & \text{if } z \geq 0, \\ \alpha z, & z \leq 0 \end{cases} \tag{2.10}$$

where α is a small constant value. As shown in Figure 2.5, the small negative slope provides a pathway for gradients to flow, even when the neuron is inactive. Parametric ReLU extends LeakyReLU by making the negative slope learnable, by optimizing α during training. This adaptive mechanism enables the network to adjust its activation dynamics for improved performance.

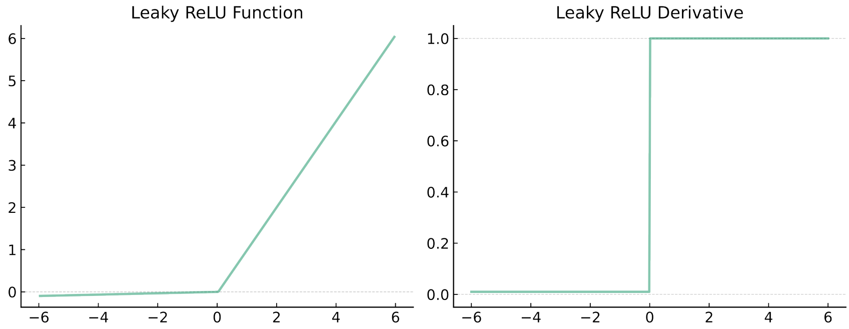


Figure 2.5: LeakyReLU activation function and its derivative. By allowing small negative gradients, the LeakyReLU function addresses the dead neuron problem and improves learning stability.

SoftPlus provides a smooth approximation of ReLU and is mathematically expressed as

$$g(z) = \frac{1}{\beta}\log(1 + e^{\beta z}) \tag{2.11}$$

where β is the scale factor, which controls how closely the function approximates ReLU. The higher the value of β, the closer the SoftPlus mimics the ReLU function. Figure 2.6 shows that SoftPlus provides smooth gradient flow, facilitating continuous optimization. The SoftPlus activation function always outputs positive values, making it suitable for cases where positive activation is required.

Swish is a smooth, non-monotonic function known as a self-gated activation function. Swish is mathematically expressed as,

$$g(z) = z \cdot \sigma(z) = \frac{z}{1 + e^{-z}}, \tag{2.12}$$

where $\sigma(z)$ denotes the sigmoid function. Swish is smooth, non-monotonic, unbounded in the positive region, and bounded in the negative region as shown in Figure 2.7. The non-monotonicity and smoothness properties of the swish activation function allow negative outputs, eliminating the dying ReLU and

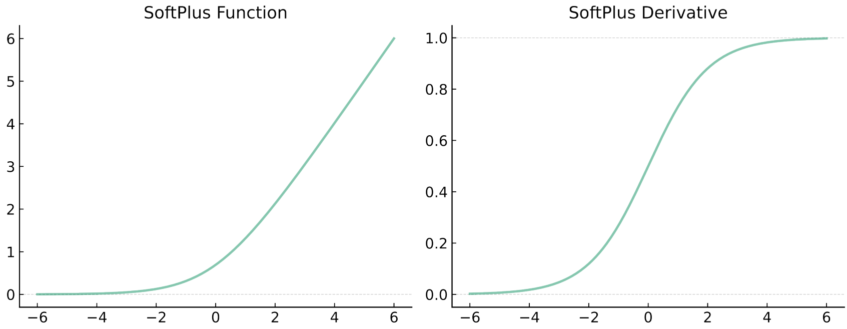


Figure 2.6: SoftPlus activation function and its derivative. This smooth approximation of ReLU ensures continuous gradients, supporting stable optimization across all input values.

promoting stable gradient flow. The swish function is computationally more expensive than ReLU due to the sigmoid term.

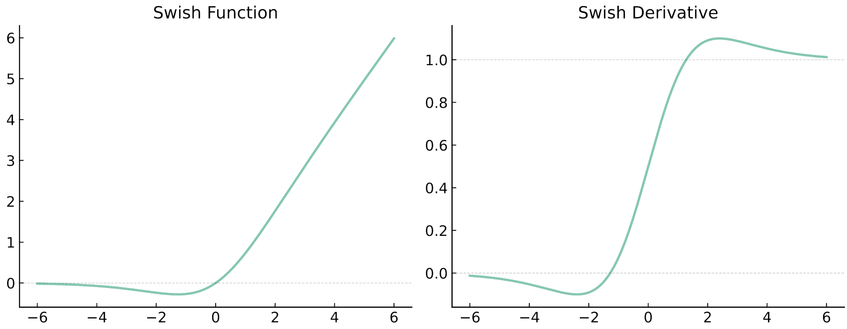


Figure 2.7: Swish activation function and its derivative. The self-gated, smooth function balances linear and nonlinear behavior, often improving gradient propagation and network performance.

Training algorithm of neural network: Forward and backpropagation

Training artificial neural networks can be divided into two stages: the forward pass and the backward pass, also known as backpropagation.

In the forward pass, the input is transformed into an output prediction by sequentially propagating through the network's layers. In a multi layer perceptron (MLP) (Figure 2.8) with N hidden layers, Mathematically, for an input vector $\boldsymbol{x}$, the output of each layer is given by,

$$\boldsymbol{r}^{(1)} = g\left(\boldsymbol{W}^{(1)}\boldsymbol{x} + \boldsymbol{b}^{(1)}\right), \tag{2.13}$$

$$\boldsymbol{r}^{(l)} = f\left(\boldsymbol{W}^{(l)}\boldsymbol{r}^{(l-1)} + \boldsymbol{b}^{(l)}\right), \quad 1 < l \leq L, \tag{2.14}$$

$$\hat{\boldsymbol{y}} = \boldsymbol{W}^{(L)}\boldsymbol{r}^{(L)} + \boldsymbol{b}^{(L)}. \tag{2.15}$$

Here $\boldsymbol{r}^{(l)}$ denotes the activation of the neurons in the l-th layer, $\boldsymbol{W}^{(l)}$ and $\boldsymbol{b}^{(l)}$ are the weight matrix and bias vector of the l-th layer, $g(\cdot)$ is a nonlinear activation function, and $\hat{\boldsymbol{y}}$ represents the network output.

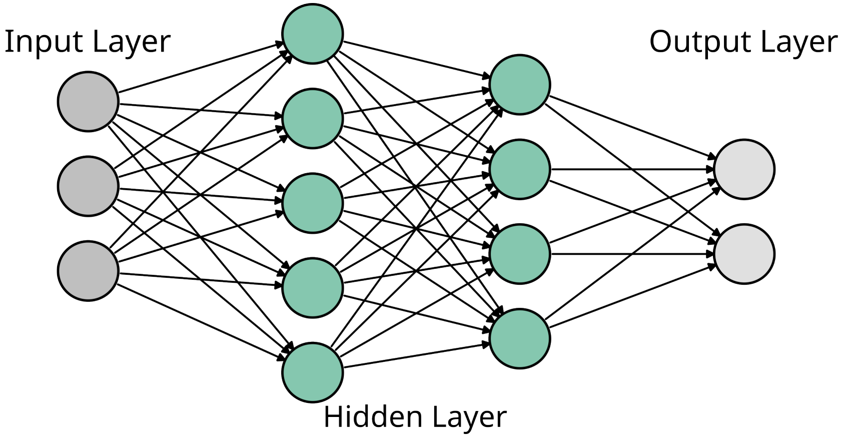


Figure 2.8: Structure of the multi-layer perceptron.

The loss function quantifies the difference between the predicted and target outputs. To quantify the network's performance for the forward pass, the predicted output $\hat{\boldsymbol{y}}$ is compared with the target $\boldsymbol{y}$ using a loss function $\mathcal{L}$.

In the backward pass, the gradient of the computed loss is used to adjust the network's parameters in the opposite direction, thereby minimizing the loss surface. The gradient computation in a feedforward neural network without any intermediate hidden layer is straightforward. In a multi-layer network, the gradient is computed using the backpropagation algorithm, which applies the chain rule of calculus to propagate gradients from the output layer to the input. To compute the loss $\mathcal{L}$, the network is run in a forward pass. To efficiently compute the exact gradient $\frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}}$, information about the loss needs to be passed in the opposite direction of the forward pass, hence the name backpropagation. To illustrate the principle of backpropagation, consider a simplified three-neuron chain network consisting of an input node $\boldsymbol{x}$, a hidden node $\boldsymbol{r}$, and an output $\hat{\boldsymbol{y}}$ (Figure 2.9). Assuming that

$$\mathcal{L}\left(\boldsymbol{y}, \hat{\boldsymbol{y}}\right) = \frac{1}{2}\left(\boldsymbol{y} - \hat{\boldsymbol{y}}\right)^2, \tag{2.16}$$

The change of the weight $\boldsymbol{w_r}$ is given by

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{w_r}} = -\left(\boldsymbol{y} - \hat{\boldsymbol{y}}\right)\boldsymbol{r}, \tag{2.17}$$

This weight update depends only on local information at the input and output units of this connection. For the input weight $\boldsymbol{w_x}$, the gradient is given by

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{w_x}} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{r}}\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{w_x}} = \frac{\partial \mathcal{L}}{\partial \hat{\boldsymbol{y}}}\frac{\partial \hat{\boldsymbol{y}}}{\partial \boldsymbol{r}}\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{w_x}}. \tag{2.18}$$

Here, the update of $\boldsymbol{w_x}$ is not local as it requires information from a downstream weight $\boldsymbol{w_r}$. In other words, the error $\frac{\partial \mathcal{L}}{\partial \boldsymbol{y}} = -\left(\boldsymbol{y} - \hat{\boldsymbol{y}}\right)$ is backpropagated from the output node to the input node. The backpropagation algorithm provides a method for determining how much $\boldsymbol{w_x}$ and $\boldsymbol{w_r}$ should be adjusted to improve the network's performance.

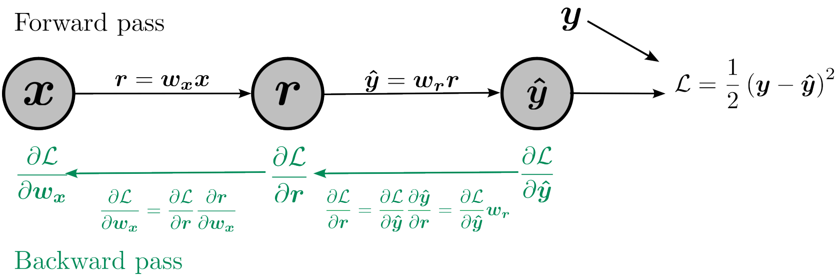

Forward pass
$\boldsymbol{y}$
$\boldsymbol{x}$
$\boldsymbol{r} = \boldsymbol{w}_{\boldsymbol{x}}\boldsymbol{x}$
$\boldsymbol{r}$
$\hat{\boldsymbol{y}} = \boldsymbol{w}_{\boldsymbol{r}}\boldsymbol{r}$
$\hat{\boldsymbol{y}}$
$\mathcal{L} = \frac{1}{2}(\boldsymbol{y} - \hat{\boldsymbol{y}})^2$
$\frac{\partial \mathcal{L}}{\partial \boldsymbol{w}_{\boldsymbol{x}}}$
$\frac{\partial \mathcal{L}}{\partial \boldsymbol{w}_{\boldsymbol{x}}} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{r}}\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{w}_{\boldsymbol{x}}}$
$\frac{\partial \mathcal{L}}{\partial \boldsymbol{r}}$
$\frac{\partial \mathcal{L}}{\partial \boldsymbol{r}} = \frac{\partial \mathcal{L}}{\partial \hat{\boldsymbol{y}}}\frac{\partial \hat{\boldsymbol{y}}}{\partial \boldsymbol{r}} = \frac{\partial \mathcal{L}}{\partial \hat{\boldsymbol{y}}}\boldsymbol{w}_{\boldsymbol{r}}$
$\frac{\partial \mathcal{L}}{\partial \hat{\boldsymbol{y}}}$
Backward pass

Figure 2.9: Illustration of the forward and backward propagation in a neural network. During the forward pass, input $\boldsymbol{x}$ is transformed through a hidden unit $\boldsymbol{r} = \boldsymbol{w}_{\boldsymbol{x}}\boldsymbol{x}$ to produce output $\hat{\boldsymbol{y}} = \boldsymbol{w}_{\boldsymbol{r}}\boldsymbol{r}$. The backward pass propagates the loss gradient from the output back to earlier layers, computing $\frac{\partial \mathcal{L}}{\partial \boldsymbol{w}_{\boldsymbol{x}}} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{r}}\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{w}_{\boldsymbol{x}}}$. Backpropagation highlights how weight updates depend on information transmitted across multiple layers. Inspired from [45]

2.2 Discriminative modeling

Discriminative modeling in machine learning focuses on learning a direct mapping from input observations to their associated outputs. As shown in Figure 2.10, the discriminative model learns a conditional probability distribution $p(y \mid \boldsymbol{x})$, where y can be a discrete class label for a classification task or a continuous value for a regression task. Consider a supervised dataset $\mathcal{D} = \{(\boldsymbol{x}^{(i)}, y^{(i)})\}_{i=1}^{N}$, where each input vector $\boldsymbol{x}^{(i)} \in \mathbb{R}^D$ and each output $y^{(i)}$ may be discrete or continuous. The likelihood of the model parameters $\boldsymbol{\theta}$ is

$$\mathscr{L}(\boldsymbol{\theta} \mid \mathcal{D}) = \prod_{i=1}^{N} p_{\boldsymbol{\theta}}(y^{(i)} \mid \boldsymbol{x}^{(i)}), \tag{2.19}$$

and the log-likelihood is

$$\ell(\boldsymbol{\theta} \mid \mathcal{D}) = \sum_{i=1}^{N} \log p_{\boldsymbol{\theta}}(y^{(i)} \mid \boldsymbol{x}^{(i)}). \tag{2.20}$$

The model parameters are estimated by maximizing the log-likelihood:

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \ \ell(\boldsymbol{\theta} \mid \mathcal{D}). \tag{2.21}$$

By focusing on $p(y \mid \boldsymbol{x})$, discriminative models effectively extract the relevant features of the input that best describe the output, whether the task is classification or regression.

2.3 Generative Modeling

Human reasoning often relies on the ability to envision possible scenarios without directly experiencing them. Machine learning adopts a similar concept through generative modeling, which aims to reproduce the mechanisms that generate observed data. A generative model aims to capture the process by which data is generated and can produce new instances similar to those in the training set. Formally, a generative model learns to approximate the probability distribution underlying a dataset. The key idea is to utilize this learned distribution to sample or generate new data and incorporate information.

In generative modeling, the sample space defines the complete set of possible outputs a model can, in principle, generate. To describe how likely different elements of the sample space are, the generative models rely on the probability density function $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ for unsupervised dataset $\mathcal{D} = \{\boldsymbol{x}^{(i)}\}_{i=1}^{N}$ or $p_{\boldsymbol{\theta}}(\boldsymbol{x}, y)$ for supervised dataset $\mathcal{D} = \{(\boldsymbol{x}^{(i)}, y^{(i)})\}_{i=1}^{N}$, parameterized by model parameters $\boldsymbol{\theta}$. For an unsupervised dataset, the Probability Density Function (PDF), expressed as $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ and parameterized by $\boldsymbol{\theta}$, specifies how probability mass is distributed across the sample space. Because infinitely many density functions could be employed to approximate the true distribution of the data, parametric modeling imposes structure by restricting the choice to families of functions with adjustable parameters. Assuming a Gaussian distribution allows the parameters that best describe the observed data to be estimated. The likelihood measures how well a given set of parameters explains the observed data, expressed as $p_{\boldsymbol{\theta}}\left(\boldsymbol{x}\right)$, which represents the probability of data $\boldsymbol{x}$ under the model parameters $\boldsymbol{\theta}$. The likelihood function can be expressed as,

$$\mathscr{L}(\boldsymbol{\theta} \mid \boldsymbol{x}) = p_{\boldsymbol{\theta}}\left(\boldsymbol{x}\right), \tag{2.22}$$

and for a unsupervised dataset $\mathcal{D} = \{\boldsymbol{x}^{(i)}\}_{i=1}^{N}$,

$$\mathscr{L}(\boldsymbol{\theta} \mid \mathcal{D}) = \prod_{\boldsymbol{x} \in \mathcal{D}} p_{\boldsymbol{\theta}}\left(\boldsymbol{x}\right). \tag{2.23}$$

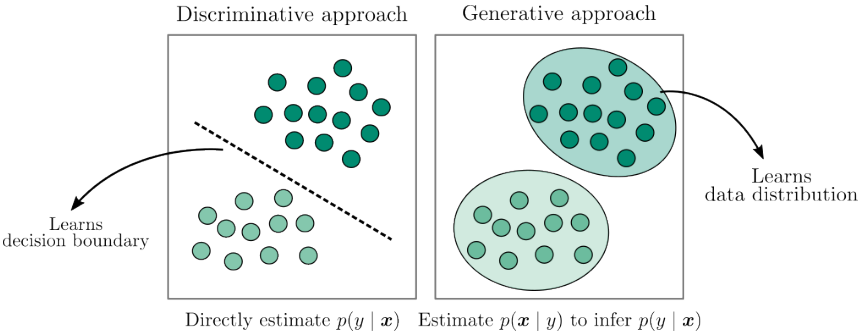

Discriminative approach
Generative approach
Learns decision boundary
Learns data distribution
Directly estimate $p(y \mid \boldsymbol{x})$
Estimate $p(\boldsymbol{x} \mid y)$ to infer $p(y \mid \boldsymbol{x})$

Figure 2.10: Comparison of discriminative and generative modeling approaches. In the context of a classification problem, discriminative models estimate $p(y \mid \boldsymbol{x})$ to directly discriminate between different data classes, while generative models learn $p(\boldsymbol{x} \mid y)$ to capture the underlying data distribution and infer $p(y \mid \boldsymbol{x})$.

Calculating likelihood in complex datasets, involving multiplying small probabilities, can lead to numerical underflow. To address this, the log-likelihood is often used, since it converts products into sums that are easier to compute. The log likelihood is given as

$$\ell\left(\boldsymbol{\theta} \mid \mathcal{D}\right) = \sum_{\boldsymbol{x} \in \mathcal{D}} \log p_{\boldsymbol{\theta}}\left(\boldsymbol{x}\right). \tag{2.24}$$

The model parameters are then optimized using the principle of maximum likelihood estimate (MLE), which selects the values of $\hat{\boldsymbol{\theta}}$ that maximize the log-likelihood. Mathematically, the maximum likelihood estimate $\hat{\boldsymbol{\theta}}$ is given as

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \ \ell\left(\boldsymbol{\theta} \mid \mathcal{D}\right). \tag{2.25}$$

While MLE provides a systematic method for parameter estimation, applying it directly to high-dimensional data can be computationally demanding. The number of parameters grows rapidly with the dimensionality, and unobserved points may be assigned a negligible probability. Naive Bayes offers a simpler alternative by assuming conditional independence among features, which drastically reduces the parameter count. However, independence assumptions limit the model's expressiveness. For example, Naive Bayes cannot capture pixel correlations in images. To overcome these limitations, modern generative modeling often relies on deep learning. As discussed in Section 2.1.2, Deep neural networks learn hierarchical, non-linear feature representations, allowing them to capture complex dependencies without restrictive independence assumptions. By leveraging multiple layers of abstraction, deep generative models provide a more powerful and flexible framework for modeling high-dimensional data [49].

2.4 Overview of generative deep learning

Integrating deep learning into generative modeling addresses the limitations of traditional generative modeling in representing complex, high-dimensional data. As discussed earlier, conventional generative modeling approaches, such as Naive Bayes models, rely on simple assumptions, including that the data features are independent. However, this assumption is not applicable when modeling image pixels. Generative DL models use Bayes theorem, which formalizes the reasoning under uncertainty, variance, and hidden structures in data. The Bayes theorem can be expressed as

$$p\,(H \mid E) = \frac{p\,(E, H)}{p\,(E)} = \frac{p\,(E \mid H)\; p\,(H)}{p\,(E)}. \tag{2.26}$$

The term $p(H)$ represents the prior distribution, which encodes the beliefs or hypotheses prior to any data being observed. The likelihood $p(E \mid H)$ represents the probability of observing evidence E given a hypothesis H. The evidence $p(E)$ represents the marginal probability of observing E for all the possible values, while the posterior $p(H \mid E)$ represents the belief after incorporating the evidence. Computing the evidence $p(E) = \int p(E \mid H)p(H)dH$ is often infeasible in high-dimensional spaces because the integral requires integrating over all possible configurations. In continuous high-dimensional spaces, the number of possible configurations grows exponentially, making this computationally infeasible. To address this challenge in Bayesian inference, generative models employ various approximation strategies, including variational inference and Markov chain Monte Carlo estimators [50, 43].

Generative models can be categorized based on how the model distributions $p_{\text{model}}(\boldsymbol{x})$ are learned. Explicit density models define a probability density function $p_{\text{model}}(\boldsymbol{x}; \boldsymbol{\theta})$ with parameters $\boldsymbol{\theta}$. The goal of an explicit model is to maximize the likelihood of the training data under this model. Under this category, the two main approaches include tractable and approximate density models. Tractable explicit density models define a density function $p_{\text{model}}(\boldsymbol{x}; \boldsymbol{\theta})$ that is computationally tractable, which directly calculates the likelihood $p_{\text{model}}(\boldsymbol{x}; \boldsymbol{\theta})$ for a given data point $\boldsymbol{x}$.

Autoregressive models, including PixelRNN [51] and PixelCNN [52], decompose the joint probability over dimensions into a product of conditional probabilities using the chain rule $p(\boldsymbol{x}) = \prod_i p(x_i \mid x_1, \ldots, x_{i-1})$ with each conditional modeled by a neural network. While the autoregressive models achieve high likelihood scores, sequential generation slows sample production [53–56]. Normalizing flow transforms a simple distribution, such as a Gaussian, into a complex data distribution using a series of invertible transformations with tractable Jacobians.

Normalizing flow offers exact likelihood estimation and efficient sampling once trained.

Approximate explicit density models also define $p_{\text{model}}(\boldsymbol{x}; \boldsymbol{\theta})$, which incorporates latent variables $\boldsymbol{z}$, making exact likelihood computation intractable. Variational Autoencoders (VAEs) introduce latent variables and assume data is generated via a conditional distribution $p(\boldsymbol{x} \mid \boldsymbol{z})$. An encoder network approximates the posterior distribution over latent variables, while a decoder reconstructs data from latent samples. Training maximizes a lower bound on the log-likelihood, known as the Evidence Lower Bound (ELBO), which balances the fidelity of reconstruction and divergence from a prior distribution over latent variables. These models provide a structured latent space, efficient sampling, and stable training, though samples may appear blurrier, and only a lower bound on likelihood is obtained [57].

Implicit density models learn to generate samples from $p_{\text{model}}(\boldsymbol{x})$ without explicitly defining the density function itself. Instead, implicit models provide a mechanism for directly sampling from the distribution. The generative adversarial network falls into this category, which uses a generator network G that maps a random noise vector $\boldsymbol{z}$ randomly sampled from a simple prior $p_{\boldsymbol{z}}(\boldsymbol{z})$ to a data sample $\boldsymbol{x} = G(\boldsymbol{z})$. A discriminator network D tries to distinguish real data from generated samples. The generator learns implicitly to transform the prior distribution $p_{\boldsymbol{z}}(\boldsymbol{z})$ into the target data distribution [58].

2.5 Generative models

2.5.1 Variational Autoencoders

A standard autoencoder consists of two neural networks: an encoder $\mathcal{E}$ and a decoder $\mathcal{G}$ (Figure 2.11). The encoder network maps the data points from the input space $\mathcal{X}$ to a lower-dimensional latent space $\mathcal{Z}$, while the decoder reconstructs the samples by mapping $\mathcal{Z}$ back to $\mathcal{X}$. The goal of an autoencoder is to encode and reconstruct a sample $\boldsymbol{x} \in \mathcal{X}$. The reconstruction of the sample is given by

$\hat{\boldsymbol{x}} = \mathcal{G}(\mathcal{E}(\boldsymbol{x}))$. The latent space $\mathcal{Z}$ creates a bottleneck at the encoder to compress input data into a meaningful latent representation of $\boldsymbol{x}$. The latent representation is a compressed representation of the salient features of the data, achieved by filtering out irrelevant features.

An autoencoder only with linear activation in both the encoder and decoder networks is mathematically equivalent to principal component analysis (PCA), which identifies the principal components of data through a linear transformation. The fixed latent vector in autoencoders is useful for tasks such as data compression and regeneration. Despite their effectiveness in learning compact encodings, the fixed latent vector representation prevents standard autoencoders from functioning as generative models. Since the latent space remains unconstrained, random sampling of $\boldsymbol{z} \in \mathcal{Z}$ fails to yield valid reconstructions, leaving discontinuities in the learned latent representation. This limitation of the autoencoder is addressed by probabilistic VAEs, which impose variational inference on the autoencoder. In VAEs, the encoder maps the input data to a latent distribution defined by its mean and variance, rather than a fixed latent vector. The variation inference estimates the distribution over $\boldsymbol{z}$ in the latent space given a data sample $\boldsymbol{x}$, in other words, the posterior probability distribution $p(\boldsymbol{z} \mid \boldsymbol{x})$. Then the $p(\boldsymbol{z} \mid \boldsymbol{x})$ can be computed using Bayesian theorem as,

$$p\left(\boldsymbol{z} \mid \boldsymbol{x}\right) = \frac{p\left(\boldsymbol{x} \mid \boldsymbol{z}\right)\ p\left(\boldsymbol{z}\right)}{p\left(\boldsymbol{x}\right)}. \tag{2.27}$$

For a continuous distribution, the integral is taken over all sample partitions:

$$p\left(\boldsymbol{x}\right) = \int p\left(\boldsymbol{x} \mid \boldsymbol{z}\right)\ p\left(\boldsymbol{z}\right)\ d\boldsymbol{z}. \tag{2.28}$$

The estimation of the posterior requires computing the marginal density or evidence $p\left(\boldsymbol{x}\right)$. Computation of the evidence requires integrating over the entire latent space, which becomes infeasible in high dimensions. To solve this challenge, the posterior $p(\boldsymbol{z} \mid \boldsymbol{x})$ is approximated by a family of tractable distributions $q(\boldsymbol{z} \mid \boldsymbol{x})$ from a known family of distributions known as an inference model. A tractable

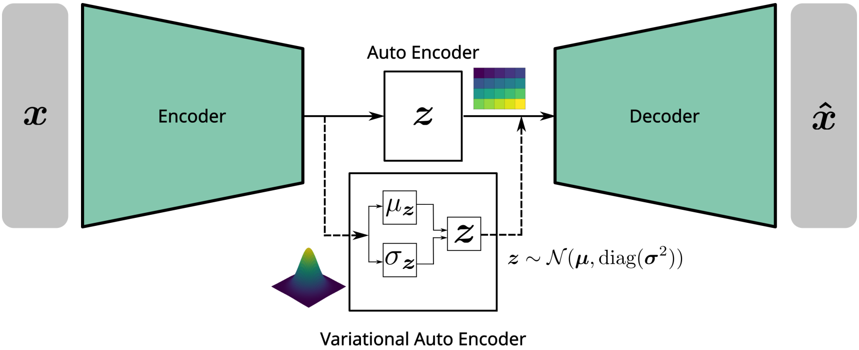


Figure 2.11: Schematic illustration of an autoencoder and a variational autoencoder. The autoencoder compresses input data into a latent vector and reconstructs it back to the original form. The VAE encodes the input into a probabilistic latent space defined by the mean $\mu_{\boldsymbol{z}}$ and variance $\sigma_{\boldsymbol{z}}$, yielding a continuous representation. The encoder and decoder are connected through a low-dimensional bottleneck, enabling the model to learn meaningful latent variables $\boldsymbol{z}$ for accurate data reconstruction $\boldsymbol{x}$.

training objective is obtained by taking a variational lower bound on $p\left(\boldsymbol{x}\right)$ using the approximate $q(\boldsymbol{z} \mid \boldsymbol{x})$:

$$\log p(\boldsymbol{x}) = \log \int q(\boldsymbol{z} \mid \boldsymbol{x}) \frac{p(\boldsymbol{x}, \boldsymbol{z})}{q(\boldsymbol{z} \mid \boldsymbol{x})} d\boldsymbol{z}. \tag{2.29}$$

Applying Jensen's inequality to the logarithm, yields the following variational lower bound, known as the evidence lower bound (ELBO):

$$\log p(\boldsymbol{x}) \geq \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})} \left[\log \frac{p(\boldsymbol{x}, \boldsymbol{z})}{q(\boldsymbol{z} \mid \boldsymbol{x})} \right]. \tag{2.30}$$

The difference between $\log p(\boldsymbol{x})$ and the ELBO corresponds to the Kullback-Leibler (KL) divergence between the approximate posterior $q(\boldsymbol{z} \mid \boldsymbol{x})$ and the true posterior $p(\boldsymbol{z} \mid \boldsymbol{x})$,

$$\log p(\boldsymbol{x}) = \mathrm{ELBO} + \mathrm{KL}\big(q(\boldsymbol{z} \mid \boldsymbol{x}) \mid\mid p(\boldsymbol{z} \mid \boldsymbol{x})\big). \tag{2.31}$$

Since $p(\boldsymbol{x})$ is independent of the variational parameters, maximizing the ELBO is equivalent to minimizing this KL divergence. The VAE architecture uses the variational inference framework by assigning the encoder $\mathcal{E}$ to parameterize $q(\boldsymbol{z} \mid \boldsymbol{x})$ and the decoder $\mathcal{G}$ to model $p(\boldsymbol{x} \mid \boldsymbol{z})$. Substituting $p(\boldsymbol{x}, \boldsymbol{z}) = p(\boldsymbol{x} \mid \boldsymbol{z})p(\boldsymbol{z})$ yields the following training objective:

$$\mathcal{L} = \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})}\big[\log p(\boldsymbol{x} \mid \boldsymbol{z})\big] - \mathrm{KL}\big(q(\boldsymbol{z} \mid \boldsymbol{x}) \mid\mid p(\boldsymbol{z})\big), \tag{2.32}$$

The term $\mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})}\big[\log p(\boldsymbol{x} \mid \boldsymbol{z})\big]$ corresponds to the reconstruction likelihood, and $\mathrm{KL}\big(q(\boldsymbol{z} \mid \boldsymbol{x}) \mid\mid p(\boldsymbol{z})\big)$ regularizes the latent distribution by penalizing deviations from the prior $p(\boldsymbol{z})$, typically a standard Gaussian $\mathcal{N}(0, \boldsymbol{I})$. The regularization constraint enforces smoothness in the latent space, ensuring that samples $\boldsymbol{z} \sim p(\boldsymbol{z})$ correspond to valid reconstructions through $\mathcal{G}$. Unlike standard autoencoders, where $\mathcal{E}$ maps each $\boldsymbol{x}$ to a fixed latent vector, VAEs map the input data to a probability distribution over the latent variables. The encoder outputs parameters $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ of a normal distribution $q(\boldsymbol{z} \mid \boldsymbol{x}) = \mathcal{N}(\boldsymbol{\mu}, \mathrm{diag}(\boldsymbol{\sigma}^2))$, from which latent variables are sampled. To maintain differentiability during training, the reparameterization trick is applied:

$$\boldsymbol{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon},\ \boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I}). \tag{2.33}$$

The reparametrization trick moves stochasticity to the $\boldsymbol{\epsilon}$ node that is independent of parameters by sampling it from a standard normal distribution with mean 0 and variance 1, enabling gradient-based optimization of $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$. For Gaussian posteriors and priors, the KL divergence admits a closed-form expression:

$$\mathrm{KL}(q(\boldsymbol{z} \mid \boldsymbol{x}) \mid\mid p(\boldsymbol{z})) = \frac{1}{2}\sum_{i=1}^{k}\Big(- \log \sigma_i^2 - 1 + \sigma_i^2 + \mu_i^2\Big). \tag{2.34}$$

While VAEs successfully resolve the discontinuities present in standard autoencoders by enforcing a smooth latent space, they often produce blurry reconstructions and generated samples. The strong regularization imposed by the Gaussian prior may lead to oversmoothing. The β-VAE framework mitigates this effect by introducing a weighting parameter β on the KL term:

$$\mathcal{L}_\beta = \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})}\big[\log p(\boldsymbol{x} \mid \boldsymbol{z})\big] - \beta\,\mathrm{KL}\big(q(\boldsymbol{z} \mid \boldsymbol{x}) \mid\mid p(\boldsymbol{z})\big). \tag{2.35}$$

Assigning a larger β value enables a disentangled latent space at the expense of reconstruction fidelity, whereas smaller values of β focus on perceptual quality. Popular generative models, such as Stable Diffusion, adopt an extremely small value of β, relying on massive training datasets to preserve generative capacity while maximizing visual quality [57, 59–63].

2.5.2 Generative adversarial networks

Generative Adversarial Networks (GANs), first introduced by Goodfellow et al. [58], represent a class of implicit density generative models that can synthesize high-quality data samples. As shown in Figure 2.12, the GAN framework consists of two neural networks: a generator G, which maps latent variables $\boldsymbol{z} \sim p_{\boldsymbol{z}}(\boldsymbol{z})$ into the data space, and a discriminator D, which evaluates whether an input originates from the real data distribution or the generator. The interaction between G and D represents the adversarial learning process.

The generator G takes a noise vector $\boldsymbol{z}$ as input to generate data $\hat{\boldsymbol{x}}$ that resembles the real data $\boldsymbol{x}$. The role of the discriminator D is to distinguish real samples $\boldsymbol{x}$ from generated samples $\hat{\boldsymbol{x}}$. During training, the generator and discriminator compete in a min-max optimization problem: the discriminator seeks to maximize the probability of correctly classifying real and generated samples, while the generator aims to minimize this probability. In practice, training alternates between updating D and G: when D is updated, G's weights remain fixed, and vice versa.

The interaction between the generator and the discriminator can be mathematically formalized as

$$\min_{G} \max_{D} V(D,G) := \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] \\ +\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}\left[\log\left(1 - D(G(\boldsymbol{z}))\right)\right].$$

Here, the function $V(D,G)$ consists of two terms that need to be optimized. $\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log(D\,(\boldsymbol{x}))]$ represents the average log probability produced by the discriminator when input is real and $\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}\left[\log\left(1 - D(G(\boldsymbol{z}))\right)\right]$ represents the average log probability produced by the discriminator when the input is generated [64–67].

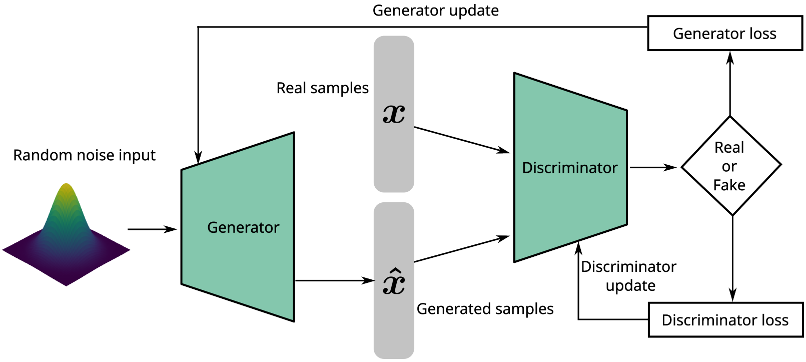


Figure 2.12: Schematic representation of a generative adversarial network (GAN). The generator creates realistic data samples from random noise vectors, while the discriminator differentiates between real and generated samples. Both networks are trained adversarially, improving performance through mutual competition.

Maximizing the objective function with respect to the discriminator leads to an optimal solution where real samples $\boldsymbol{x} \sim p_{\text{data}}$ are assigned a probability of one and generated samples $\hat{\boldsymbol{x}} \sim p_G$ are assigned a probability of zero. The generator, in contrast, influences only the second term of the objective, and minimizing this term drives it to produce samples that are indistinguishable from real data.

This process compels the generator to approximate the real data distribution p_{data}. Training proceeds through alternating updates to the two networks, where achieving optimal performance depends on maintaining a balance that ensures neither model overpowers the other [65–67].

2.5.3 Diffusion model

Diffusion Models (DMs) have emerged as a powerful class of generative models, recently surpassing Generative Adversarial Networks (GANs) in several benchmark tasks. Diffusion models are inspired by non-equilibrium thermodynamics, which define a Markov chain that gradually corrupts data by adding Gaussian noise via forward diffusion and subsequently learns to remove the noise via reverse diffusion [68]. The following formulation of the diffusion model is based on the definition of the denoising diffusion probabilistic model (DDPM). As depicted in the Figure 2.13, the forward process begins with a data point $\boldsymbol{x}_0 \sim q(\boldsymbol{x}_0)$ and introduces Gaussian noise over T discrete timesteps. Each step depends solely on the immediately preceding state, reflecting the Markovian property. The forward diffusion process introduces noise to the training data over a series of time steps, with the noise scale varying linearly with time. This is achieved by using linear noise scheduling at each step until the training data is corrupted, resulting in pure Gaussian noise. In the forward diffusion process, noise is introduced using a Markov chain, indicating that the current state of the training data relies solely on its most recent state. Let $q\left(\boldsymbol{x}_0\right)$ be the probability density of the training data, where the index 0 denotes the data before adding any noise. Given an uncorrupted training sample $\boldsymbol{x}_0 \sim q\left(\boldsymbol{x}_0\right)$, the noised versions $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_T$ are generated through the following Markovian process:

$$q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}) = \mathcal{N}\Big(\boldsymbol{x}_t;\, \mu = \sqrt{1-\beta_t}\,\boldsymbol{x}_{t-1},\ \Sigma = \beta_t \boldsymbol{I}\Big), \quad \forall t \in \{1, \ldots, T\}. \tag{2.36}$$

Here, T is the number of diffusion steps, $\beta_1, \ldots, \beta_T$ are the hyperparameters controlling the noise variance, and $\boldsymbol{I}$ is the identity matrix with dimension equal to the input data dimension. This setup enables the sampling of $\boldsymbol{x}_t$ when t is drawn from a uniform distribution $t \sim \mathcal{U}(\{1, \ldots, T\})$:

$$q(\boldsymbol{x}_t \mid \boldsymbol{x}_0) = \mathcal{N}\left(\boldsymbol{x}_t; \mu = \sqrt{\hat{\beta}_t}\boldsymbol{x}_0, \Sigma = (1 - \hat{\beta}_t)\boldsymbol{I}\right), \tag{2.37}$$

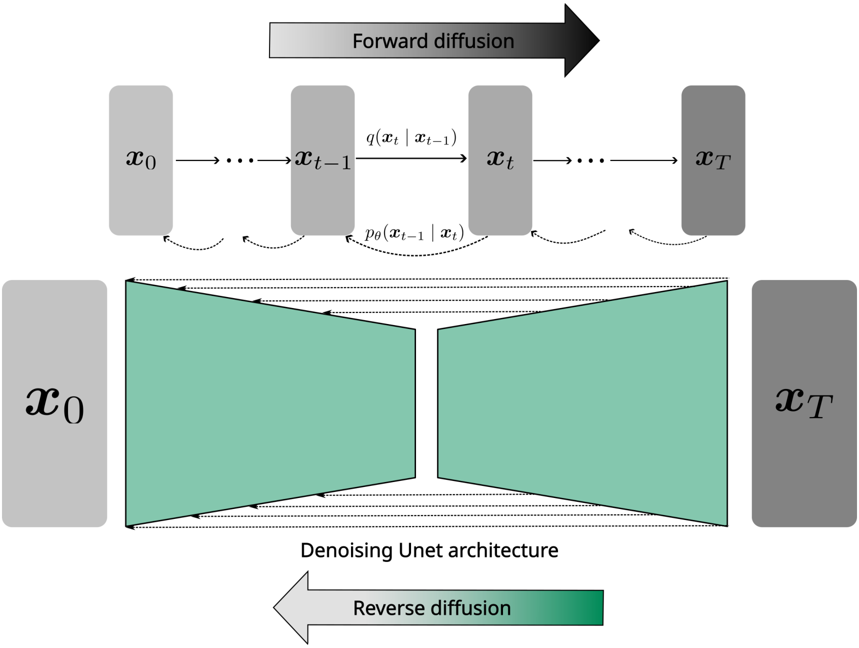


Figure 2.13: Schematic representation of diffusion model showing the forward and reverse diffusion process. In the forward process q, clean data $\boldsymbol{x}_0 \sim q(\boldsymbol{x})$ is progressively perturbed with Gaussian noise until it becomes pure noise. The reverse process p_θ, typically implemented with a U-Net, iteratively denoises the samples to reconstruct realistic data from noise.

This shows that, given the original data $\boldsymbol{x}_0$ and variance schedule β_t, the noisy sample $\boldsymbol{x}_t$ can be derived in a single step. The reparameterization trick allows sampling from $q\left(\boldsymbol{x}_t \mid \boldsymbol{x}_0\right)$ with $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$ as follows:

$$\begin{aligned} \boldsymbol{x}_t &= \sqrt{(1-\beta_t)} \cdot \boldsymbol{x}_0 + \sqrt{\beta_t} \cdot \boldsymbol{\epsilon}, \\ \boldsymbol{x}_t &= \sqrt{\bar{\alpha}_t} \cdot \boldsymbol{x}_0 + \sqrt{(1-\bar{\alpha}_t)} \cdot \boldsymbol{\epsilon}, \end{aligned} \tag{2.38}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$ is a standard normal variable, enabling the faster sampling of the noisy version from the original sample. In the reverse diffusion process, the goal is to remove the noise added to the training dataset in a structured and controlled manner, thereby reconstructing the original data $\boldsymbol{x}_0$. For example, if the forward diffusion process can be reversed and sample from $q\left(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t\right)$. However, estimating $q\left(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t\right)$ is difficult because it requires the use of the entire dataset. To achieve the reverse diffusion process, a neural network model p_θ is trained to approximate these conditional probabilities. The model learns to reverse this diffusion process during training, generating new data. Starting with pure Gaussian noise $p\left(\boldsymbol{x}_T\right) := \mathcal{N}\left(\boldsymbol{x}_T; 0, I\right)$, the model learns the reverse trajectory or joint distribution $p_\theta\left(\boldsymbol{x}_{0:T}\right)$ as

$$\begin{aligned} p_\theta(\boldsymbol{x}_{0:T}) &= p\left(\boldsymbol{x}_T\right) \prod_{t=1}^{T} p_\theta(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t), \\ \text{with, } p_\theta(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t) &= \mathcal{N}(\boldsymbol{x}_{t-1}; \boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t), \boldsymbol{\Sigma}_\theta(\boldsymbol{x}_t, t)), \end{aligned} \tag{2.39}$$

where $p_\theta\left(\boldsymbol{x}_{0:T}\right)$ denotes the reverse diffusion trajectory. The reverse diffusion kernel $p_\theta\left(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t\right)$ is defined by the mean $\boldsymbol{\mu}_\theta\left(\boldsymbol{x}_t, t\right)$ and the covariance matrix $\boldsymbol{\Sigma}_\theta\left(\boldsymbol{x}_t, t\right)$. Using the Markov chain, $\boldsymbol{x}_0$ is generated by first sampling a noise vector $\boldsymbol{x}_T \sim p\left(\boldsymbol{x}_T\right)$, then iteratively sampling from the learnable reverse diffusion kernel $\boldsymbol{x}_{t-1} \sim p_\theta\left(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t\right)$ until $t = 1$. The sampling process is improved by training the reverse Markov chain to match the forward Markov chain. In other words, the parameter θ has to be adjusted so that the joint distribution of the Markov chain closely approximates that of the forward process. After the necessary parameterization of the reverse diffusion process following Ho et al., [69], the objective formulation to train the diffusion model:

$$\mathcal{L}_{simple} \coloneqq \mathbb{E}_{t,\boldsymbol{x}_0,\boldsymbol{\epsilon}}\big[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}}\,\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}\,\boldsymbol{\epsilon}, t)\|^2\big], \tag{2.40}$$

where $\mathbb{E}$ refers to the expected value and $\boldsymbol{\epsilon}_\theta$ represents the neural network trained to predict the noise in given input $\boldsymbol{x}_t$. The α_t is calculated from $\alpha_t = 1 - \beta_t$ and $\bar{\alpha} = \prod_{s=0}^{t} \alpha_s$. The reparameterization of reverse diffusion, $p_\theta\left(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t\right)$, simplifies the process by fixing the covariance at a constant value. The mean is then defined through variance scheduling, which denotes that it only depends on $\boldsymbol{x}_t$. Consequently, the neural network is trained to predict the noise added at each specific time step, rather than estimating both the mean and covariance [69–71].

The main drawbacks of diffusion models include slow inference and high training cost. Despite their successful implementation, diffusion models have drawbacks, including high computational cost during both training and inference. Each generated sample typically requires thousands of sequential denoising steps. Several approaches have been proposed to mitigate this limitation. Diffusion Denoising Implicit Models (DDIMs) introduce a non-Markovian formulation that enables deterministic sampling and skips certain steps in the diffusion process, thereby accelerating inference while preserving generation quality [72]. Another significant advancement in the diffusion model is the introduction of Latent Diffusion Models (LDMs). In standard DDPMs, the diffusion process occurs directly in the high-dimensional data space, which is computationally demanding. LDMs instead apply diffusion in a compressed latent space, obtained via VAE. By operating on latent representations, the dimensionality is reduced, enabling faster inference and reduced memory requirements while retaining semantic richness [73].

2.6 Loss functions and Evaluation metrics

Loss functions and evaluation metrics are two valuable tools for assessing model performance, but they serve distinct purposes in model development and evaluation. During training, the loss function measures the discrepancy between the predicted outputs and the target outputs. The goal of the loss function is to optimize model parameters by minimizing this gap. In contrast, performance metrics are applied after training to evaluate how well the model works with new data. The evaluation metrics enable comparison across different models and configurations to find the model that performs best for the task.

2.6.1 Loss functions

Mathematically, the loss function $\mathcal{L}(\theta)$ for a model parameterized by θ and trained on dataset $\mathcal{D} = \{(\boldsymbol{x}^i, y^i)\}_{i=1}^N$ can be expressed as

$$\mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N} l(f_\theta(\boldsymbol{x}^i), y^i) \tag{2.41}$$

where $l(\cdot)$ represents the loss per sample and θ represents the model parameters. The mathematical properties of loss functions, such as convexity, differentiability, smoothness, robustness, and monotonicity, are important for the stability of the gradient-based optimization process during model training. Convexity ensures that any local minimum of a loss function is also a global minimum, facilitating the consistent convergence of optimization algorithms to an optimal solution. Differentiability of the loss function enables the calculation of derivatives or gradients, which are crucial for the gradient descent method used in model training. The smoothness of the loss function features a continuous and stable gradient, which prevents sudden changes in the loss landscape and supports steady convergence throughout training. Robustness protects the optimization process from distortions caused by outliers or unusual data points. Monotonicity ensures that the loss value consistently decreases as predicted outputs approach the actual target values, providing a clear direction for optimization. These mathematical properties of the loss function provide a stable and understandable basis for evaluating predictive error, which helps in creating machine learning models with better accuracy and generalization [74, 75]. The following subsection will focus on the loss functions used in the different generative tasks.

Reconstruction Loss

Reconstruction-based losses are employed in generative models to quantify the difference between the original and reconstructed data, thereby forcing the model to capture and reproduce the essential features of the input data. The choice of an appropriate loss function is significant, as it makes an implicit assumption about data distributions that directly affects the learned latent representations. The following subsections reveal the most widely used reconstruction losses in generative models. Depending on the data normalization range, the activation function of the model's output layer, and the specific task, the following reconstruction losses are selected.

Mean Squared Error

The mean squared error (MSE), also known as the L2 loss, is a fundamental loss function that measures the average squared distance between predicted outputs and ground truth values. The squaring of the difference between the predictions and ground truth values results in a higher penalty assigned to larger deviations from the target value. Taking the average normalizes the total loss against the total number of samples in a training dataset. The mathematical equation of MSE is expressed as,

$$\mathcal{L}_{\text{MSE}} = \frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{x}^i - \hat{\boldsymbol{x}}^i)^2. \tag{2.42}$$

Where N denotes the total number of samples in the dataset, $\boldsymbol{x}^i$ is the ground truth value for the i-th sample, and $\hat{\boldsymbol{x}}^i$ is the predicted value for the i-th sample. In generative modeling, the MSE loss is widely used in generative tasks that require pixel-level reconstruction accuracy in models such as autoencoders. However, due to the higher sensitivity of the larger deviations, MSE may over-penalize outliers, resulting in blurred outputs when applied to image generation tasks. This limitation necessitates the use of weighted loss functions, such as perceptual losses, to achieve visually sharp results.

Mean Absolute Error

The mean absolute error (MAE), also referred to as L1 loss, measures the average absolute difference between the predicted and actual values. Unlike MSE, which features a quadratic penalty, MAE loss penalizes errors linearly, making the loss function inherently robust to outliers and encouraging predictions with fine and high-frequency details. The mathematical equation of MAE loss is expressed as,

$$\mathcal{L}_{\text{MAE}} = \frac{1}{N} \sum_{i=1}^{N} \mid \boldsymbol{x}^i - \hat{\boldsymbol{x}}^i \mid . \tag{2.43}$$

Where n denotes the total number of samples in the dataset, $\boldsymbol{x}^i$ is the ground truth value for the i-th sample, and $\hat{\boldsymbol{x}}^i$ is the predicted value for the i-th sample. In image generation, the MAE loss is often preferred for image-to-image translation models, image denoising, and super-resolution tasks.

Binary Cross Entropy

The binary cross-entropy (BCE), also known as log loss, is another widely adopted reconstruction objective in generative modeling when the input data are binary, representing normalized probabilities within the range $[0, 1]$. The BCE loss is formally defined as:

$$\mathcal{L}_{\text{BCE}}(\boldsymbol{x}, \hat{\boldsymbol{x}}) = -\frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{D} \Big[\boldsymbol{x}_j^i \log(\hat{\boldsymbol{x}}_j^i) + (1 - \boldsymbol{x}_j^i) \log(1 - \hat{\boldsymbol{x}}_j^i) \Big], \tag{2.44}$$

where N is the number of samples and D is the dimensionality (e.g., number of pixels). $\boldsymbol{x}_j^i \in \{0,1\}^D$ denotes the ground truth value of the j-th pixel or feature in i-th sample and $\hat{\boldsymbol{x}}_j^i$ represents the corresponding model's predicted probabilities. Similar to the Mean Squared Error (MSE), this loss is averaged over all samples within a mini-batch during training. Minimizing the BCE loss is equivalent to maximizing the log-likelihood of the observed data under the assumption that each feature $\boldsymbol{x}_j^i$ is independently drawn from a Bernoulli distribution parameterized by $\hat{\boldsymbol{x}}_j^i$. This probabilistic interpretation provides a direct measure of how well the reconstructed outputs align with the true data distribution. In the context of generative modeling, BCE reconstruction loss is frequently employed in VAEs, binary image generation, and probabilistic reconstruction tasks such as segmentation mask prediction. In VAE training, the BCE term typically serves as the reconstruction component of the Evidence Lower Bound (ELBO), in conjunction with a Kullback–Leibler (KL) divergence regularization term that enforces latent-space structure. The BCE loss provides a statistically grounded and probabilistically interpretable measure of reconstruction fidelity, making it particularly effective for generative tasks involving binary data.

Categorical Cross Entropy

The categorical cross entropy (CCE) measures the difference between the predicted class distributions and the true class distribution. In a generative context, CCE loss is used when the model's output is discrete, effectively maximizing the likelihood of observing the true data class distributions under the model's parameters. The mathematical equation of CCE loss is expressed as,

$$\mathcal{L}_{\text{CCE}} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} y_c^i \log(\hat{y}_c^i) \tag{2.45}$$

Where C is the number of classes, y_c^i is the true label for the i-th observation or sample and class c, which is a one-hot encoded vector to assign 0 for different classes and 1 for the same class during loss computation, and $\hat{y}_c^i$ is the predicted probability distribution for that class. CCE loss can be sensitive to imbalanced datasets, where the number of samples in each class is not equal, and the requirement for one-hot encoding increases memory usage instead of using real values for the classes of the data. CCE loss functions are used as an objective function in sequence-based generation models, including autoregressive models for language modeling, as well as in text-to-image generation tasks and other image classification tasks.

Adversarial Loss

Adversarial loss establishes a dynamic min-max game between the generator G, which maps noise vectors to synthesized data, and the discriminator D networks, which classify input as real or generated. The generator G is optimized to minimize loss by synthesizing data indistinguishable from real data. At the same time, the discriminator D maximizes the loss by correctly classifying inputs as real or fake. The generator-discriminator competition drives G to learn the true data distribution. Assuming $\boldsymbol{z} \sim p_{\boldsymbol{z}}(\boldsymbol{z})$ is the noise drawn from the latent distribution and $\boldsymbol{x} \sim p_{\text{data}}(\boldsymbol{x})$ is a real data sample from the target distribution, the adversarial loss is expressed as

$$\min_G \max_D \ \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}} [\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}} [\log (1 - D(G(\boldsymbol{z})))] \,. \tag{2.46}$$

Here, $D(\boldsymbol{x})$ outputs the probability that $\boldsymbol{x}$ belongs to the real data distribution, and $G(\boldsymbol{z})$ maps a latent vector $\boldsymbol{z}$ into the data space. This formulation corresponds to the BCE loss between the discriminator predictions and the corresponding real/fake labels. The discriminator is optimized by minimizing

$$\mathcal{L}_D = -\,\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}}[\log D(\boldsymbol{x})] -\,\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}}[\log (1 - D(G(\boldsymbol{z})))] \,. \tag{2.47}$$

while the generator minimizes

$$\mathcal{L}_G = -\,\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}}[\log D(G(\boldsymbol{z}))] \,. \tag{2.48}$$

The BCE-based formulation assumes that the discriminator performs a binary classification task, treating real and generated samples as distinct classes. However, in practice, the BCE formulation can suffer from gradient saturation, where D becomes overly confident early in training, leading to negligible gradients for G. This limitation has motivated alternative loss formulations designed to improve training stability and convergence [64].

Least Squares Adversarial Loss

The least squares adversarial loss modifies the standard GAN objective by replacing the binary cross-entropy loss with a least squares formulation. In conventional GANs, the sigmoid cross-entropy loss can lead to vanishing gradients when generated samples lie on the correct side of the discriminator's decision boundary but remain distant from the real data manifold. The vanishing gradient problem limits the generator's ability to improve further as it generates almost realistic samples. The least-squares GAN (LSGAN) addresses this issue by using a quadratic loss that continues to penalize samples based on their distance from the target, thereby providing smoother, more informative gradients during training.

Formally, let a and b denote the discriminator's target values for fake and real data, respectively. The LSGAN objectives are defined as:

$$\min_D V_{\text{LSGAN}}(D) = \frac{1}{2}\,\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}\left[(D(\boldsymbol{x}) - b)^2\right] + \frac{1}{2}\,\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}\left[(D(G(\boldsymbol{z})) - a)^2\right]. \tag{2.49}$$

$$\min_G V_{\text{LSGAN}}(G) = \frac{1}{2}\,\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}\left[(D(G(\boldsymbol{z})) - c)^2\right]. \tag{2.50}$$

where c represents the value the generator wants the discriminator to output for its generated samples, often set equal to b to encourage the generator to produce realistic data. Unlike the cross-entropy loss, which saturates and yields near-zero gradients for well-classified samples, the least squares loss maintains a non-zero penalty proportional to the prediction error. The least-squares approach ensures that the generator continues to receive meaningful gradient information, even when its outputs are close to realistic, leading to more stable, gradual learning. By penalizing both under- and over-confident discriminator outputs, LSGAN encourages smoother decision boundaries and reduces the likelihood of mode collapse.

Empirically, LSGANs have demonstrated improved stability and visual quality in image generation tasks such as image-to-image translation and super-resolution. Their quadratic objective helps align the generated data distribution more closely with the real distribution while maintaining robust gradient flow [76].

Wasserstein adversarial Loss

The Wasserstein adversarial loss redefines the GAN objective using the Earth Mover's Distance (EMD). The EMD measures the dissimilarity between two multidimensional distributions by calculating the minimum work needed to transform one into the other. Let $p_r(\boldsymbol{x})$ denote the real data distribution and $p_g(\hat{\boldsymbol{x}})$ represent the generator-induced distribution. The Wasserstein distance between $p_r(\boldsymbol{x})$ and $p_g(\hat{\boldsymbol{x}})$ distributions is defined as the minimum cost required to transport the probability mass of one distribution to the other, formulated as the following optimal transport problem:

$$W(p_d, p_g) = \inf_{\gamma \in \Gamma(p_d, p_g)} \int_{\boldsymbol{x}, \hat{\boldsymbol{x}}} || \, \boldsymbol{x} - \hat{\boldsymbol{x}} \, || \, d\gamma(\boldsymbol{x}, \hat{\boldsymbol{x}}), \tag{2.51}$$

where $\Gamma(p_d, p_g)$ is the set of all joint distributions $\gamma(\boldsymbol{x}, \hat{\boldsymbol{x}})$ whose marginals correspond to p_d and p_g, respectively. The term $|| \, \boldsymbol{x} - \hat{\boldsymbol{x}} \, ||$ denotes the Euclidean distance, representing the cost of moving a unit of probability mass from $\hat{\boldsymbol{x}}$ to $\boldsymbol{x}$. The integral thus captures the total minimal transport cost required to morph one distribution into the other, establishing a meaningful geometric distance between them. The Lipschitz continuity constraint is imposed on the discriminator of Wasserstein GAN (WGAN) to ensure that the discriminator behaves as a smooth and bounded function. The constraint prevents the discriminator from becoming overconfident, which provides almost no gradient to guide the generator. The Lipschitz constraint is enforced via a gradient penalty that penalizes deviations in the gradient norm from unity, improving training stability and removing the discrete clamp on weights [77].

Perceptual Loss

Perceptual loss, also known as feature reconstruction losses, differs from traditional pixel-wise loss functions by comparing high-level features extracted from pre-trained convolution neural networks ϕ, such as the Visual Geometry Group (VGG) model. The core idea behind perceptual loss is to use the feature maps from various layers of a convolutional neural network (CNN) trained on a large dataset. By extracting the feature maps from both the target image $\boldsymbol{x}$ and the generated image $\hat{\boldsymbol{x}}$, the difference in the high-level features, such as edges, textures, and patterns that the network has trained to detect, can be quantified. Let $\phi_j(\boldsymbol{x})$ be the activations of the j-th layer of the network ϕ when processing the image $\boldsymbol{x}$. The feature reconstruction loss is then computed as the Euclidean distance between the feature representations.

$$l_{\phi,j}^{\text{feat}}(\hat{\boldsymbol{x}}, \boldsymbol{x}) = \frac{1}{C_j H_j W_j} \mid\mid \phi_j(\hat{\boldsymbol{x}}) - \phi_j(\boldsymbol{x}) \mid\mid_2^2 \tag{2.52}$$

where C_j, H_j, and W_j represent channel, height, and width dimensions of layer j. Minimizing the feature reconstruction loss ensures that the generated outputs $\hat{\boldsymbol{x}}$ are perceptually similar to the target $\boldsymbol{x}$. However, the feature reconstruction loss does not capture the stylistic attributes such as texture, color, and common patterns. To capture the stylistic representations, Gatys et al., [78] proposed the style reconstruction loss, which measures the difference in feature correlations between the generated and target images. For a given layer j, the Gram matrix $G_j^\phi(\boldsymbol{x})$ is defined as

$$G_j^\phi(\boldsymbol{x})_{c,c'} = \frac{1}{C_j H_j W_j} \sum_{h=1}^{H_j} \sum_{w=1}^{W_j} \phi_j(\boldsymbol{x})_{h,w,c} \phi_j(\boldsymbol{x})_{h,w,c'}, \tag{2.53}$$

where $\phi_j(\boldsymbol{x})$ represents the activation tensor of shape $C_j \times H_j \times W_j$. The indices h and w correspond to spatial locations in the feature map, and c and c' denote feature channels. Each product $\phi_j(\boldsymbol{x})_{h,w,c}\phi_j(\boldsymbol{x})_{h,w,c'}$ measures the degree of co-activation between channels c and c' at position (h, w). The complete Gram matrix $G_j^\phi(\boldsymbol{x}) \in \mathbb{R}^{C_j \times C_j}$ encapsulates the texture and style characteristics represented in layer j. The style reconstruction loss measures the discrepancy between the Gram matrices of the generated and target images, formulated as

$$l_{\phi,j}^{\text{style}}(\hat{\boldsymbol{x}}, \boldsymbol{x}) = || \; G_j^\phi(\hat{\boldsymbol{x}}) - G_j^\phi(\boldsymbol{x}) \; ||_F^2 \; . \tag{2.54}$$

The Frobenius norm F calculates the overall difference in feature correlations, capturing stylistic divergence between the two images. Minimizing the style reconstruction loss enables the generated image to reproduce stylistic patterns of the reference image while preserving global content alignment [79].

2.6.2 Evaluation Metrics

Evaluation metrics for generative models are quantitative and qualitative measures that assess how well the selected metrics capture, reproduce, or approximate the underlying real data distribution. The selection of the evaluation metric depends on the objectives of the generative model, such as density estimation, sampling, and latent representation learning. The following subsections describe evaluation metrics used to assess generative model performance across different aspects of the generated sample [75, 80, 81].

Inception Score

The Inception Score (IS) measures the fidelity and diversity of generated images. In practice, IS relies on a pre-trained classifier, commonly the Inception-v3 network, to evaluate the conditional label distribution of generated images [82]. Let $p(y \mid \boldsymbol{x})$ denote the conditional class distribution for a generated image $\boldsymbol{x}$, and $p(y)$ represent the marginal class distribution over all generated samples. The IS is defined as:

$$\mathrm{IS} = \exp\left(\mathbb{E}_{\boldsymbol{x}\sim p_{\text{model}}}\left[D_{\mathrm{KL}}\big(p(y \mid \boldsymbol{x}) \mid\mid p(y)\big)\right]\right), \tag{2.55}$$

where $D_{\mathrm{KL}}(\cdot \mid\mid \cdot)$ represents the Kullback–Leibler divergence (KL). The IS score measures the average KL divergence between the conditional label distribution $p(y \mid \boldsymbol{x})$ of the samples and the marginal distribution $p(y)$ over all samples. A higher IS value indicates that the generated images are both highly classifiable and diverse. However, IS does not directly compare the generated distribution to the real data distribution and is thus sensitive to the choice of classifier.

Fréchet Inception Distance

The Fréchet Inception Distance (FID) quantitatively evaluates the distance between the feature representations of real and generated data extracted from a pre-trained network. FID assumes a Gaussian distribution for the features computed by the inception network for both the real and generated samples. FID employs the Fréchet distance between two multivariate Gaussians, which provides a closed-form expression for the difference between their statistical parameters. For both real and generated samples, Gaussian distributions are fitted to the features obtained from the pool3 layer of the Inception network. Let $(\boldsymbol{\mu}_r, \boldsymbol{\Sigma}_r)$ and $(\boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g)$ denote the empirical mean and covariance matrices of the real and generated feature distributions, respectively. The FID is then computed as

$$\mathrm{FID} = \| \boldsymbol{\mu}_r - \boldsymbol{\mu}_g \|^2 + \mathrm{Tr}\left(\boldsymbol{\Sigma}_r + \boldsymbol{\Sigma}_g - 2(\boldsymbol{\Sigma}_r\boldsymbol{\Sigma}_g)^{1/2}\right), \tag{2.56}$$

where $\mathrm{Tr}(\cdot)$ denotes the trace operator. The performance of the metric depends on features extracted by the inception network. A lower FID value indicates a smaller distance between the real and generated distributions, thereby reflecting high sample fidelity and diversity. Even though the assumption of Gaussian offers computational tractability, it does not represent the true distribution of features, leading to approximation errors [83, 84].

Kernel Inception Distance

The Kernel Inception Distance (KID) provides a non-parametric alternative to the Fréchet Inception Distance (FID) for assessing the similarity between real and generated data distributions. Unlike FID, which assumes Inception features follow a Gaussian distribution, KID uses the maximum mean discrepancy (MMD), making it distribution-agnostic and providing unbiased, more stable estimates, especially on small datasets. MMD measures the distance between the mean embeddings of two distributions in a high-dimensional reproducing kernel Hilbert space. For the real p_r and generated p_g distributions with kernel k, the squared MMD is expressed as

$$\mathrm{MMD}^2(p_r, p_g) = \mathbb{E}_{\boldsymbol{x},\boldsymbol{x}'\sim p_r}\big[k(\boldsymbol{x},\boldsymbol{x}')\big] - 2\,\mathbb{E}_{\boldsymbol{x}\sim p_r,\,\hat{\boldsymbol{x}}\sim p_g}\big[k(\boldsymbol{x},\hat{\boldsymbol{x}})\big] + \mathbb{E}_{\hat{\boldsymbol{x}},\hat{\boldsymbol{x}}'\sim p_g}\big[k(\hat{\boldsymbol{x}},\hat{\boldsymbol{x}}')\big] \tag{2.57}$$

The KID metric is calculated by applying the squared MMD to the feature embeddings of real and generated samples from the final average pooling layer of the inception network.

$$\mathrm{KID} = \mathrm{MMD}^2\big(\phi(\boldsymbol{x}), \phi(\hat{\boldsymbol{x}})\big), \tag{2.58}$$

where $\phi(\cdot)$ denotes the feature embeddings from a pre-trained network. KID utilizes a polynomial kernel and remains unbiased even for finite sample sizes.

Perceptual Metrics

The Learned Perceptual Image Patch Similarity (LPIPS) metric quantifies perceptual similarity between two images using deep feature activations extracted from a pretrained convolutional neural network. LPIPS builds on the observation that deep features encode perceptually relevant information that correlates strongly with human visual judgments.

For a reference image patch $\boldsymbol{x}$ and a distorted patch $\hat{\boldsymbol{x}}$, let $\phi_l(\cdot) \in \mathbb{R}^{H_l \times W_l \times C_l}$ denote the feature activations at layer l. These activations are normalized along the channel dimension to obtain unit-normalized features

$$\hat{\phi}_l(\cdot)_{h,w} = \frac{\phi_l(\cdot)_{h,w}}{\|\phi_l(\cdot)_{h,w}\|_2}.$$

Let $\boldsymbol{w}_l \in \mathbb{R}^{C_l}$ be learned per-channel weights. The LPIPS distance is then defined as

$$\mathrm{LPIPS}(\boldsymbol{x}, \hat{\boldsymbol{x}}) = \sum_l \frac{1}{H_l W_l} \sum_{h,w} \left\| \boldsymbol{w}_l \odot \left(\hat{\phi}_l(\boldsymbol{x})_{h,w} - \hat{\phi}_l(\hat{\boldsymbol{x}})_{h,w} \right) \right\|_2^2, \tag{2.59}$$

where H_l and W_l are the spatial dimensions of the l-th feature map, and $\odot$ denotes element-wise multiplication. Using uniform weights reduces the expression to a cosine distance between deep features.

LPIPS can be computed using features from various pretrained networks, most commonly VGG-16 or VGG-19, which provide deep, perceptually rich representations. Shallower networks such as AlexNet offer faster computation, while compact networks like SqueezeNet are suitable for resource-constrained scenarios. All networks are typically pretrained on ImageNet, and the linear channel weights are learned to calibrate the metric to human perceptual judgments [85].

Peak Signal to Noise Ratio

Peak signal-to-noise ratio (PSNR) is a measure of image quality that compares the maximum possible power of a signal to the power of corrupting noise. The corrupting noise is measured using the MSE between the reference and generated images. The PSNR is then defined as,

$$\mathrm{PSNR}(I, K) = 10 \log_{10} \left(\frac{(\mathrm{MAX}_I)^2}{\mathrm{MSE}(I, K)} \right) \tag{2.60}$$

Here, MAX_I represents the maximum possible pixel value of the images. PSNR is a widely used metric for assessing reconstruction fidelity in grayscale images [84].

Structural Similarity Index Measure

The structural similarity index measure (SSIM) is another perceptual metric that evaluates the similarity between two images by focusing on properties important to human visual perception, such as luminance (I), contrast (C), and structure (S). For two images $\boldsymbol{x}$ and $\hat{\boldsymbol{x}}$, the I, C and S components are defined as follows:

$$\begin{aligned} I(\boldsymbol{x}, \hat{\boldsymbol{x}}) &= \frac{2\mu_{\boldsymbol{x}}\mu_{\hat{\boldsymbol{x}}} + C_1}{\mu_{\boldsymbol{x}}^2 + \mu_{\hat{\boldsymbol{x}}}^2 + C_1}, \\ C(\boldsymbol{x}, \hat{\boldsymbol{x}}) &= \frac{2\sigma_{\boldsymbol{x}}\sigma_{\hat{\boldsymbol{x}}} + C_2}{\sigma_{\boldsymbol{x}}^2 + \sigma_{\hat{\boldsymbol{x}}}^2 + C_2}, \\ S(\boldsymbol{x}, \hat{\boldsymbol{x}}) &= \frac{\sigma_{\boldsymbol{x}\hat{\boldsymbol{x}}} + C_3}{\sigma_{\boldsymbol{x}}\sigma_{\hat{\boldsymbol{x}}} + C_3}. \end{aligned} \tag{2.61}$$

Where $\mu_{\boldsymbol{x}}$ and $\mu_{\hat{\boldsymbol{x}}}$ are the mean intensities of local patches centered at $\boldsymbol{x}$ and $\hat{\boldsymbol{x}}$, $\sigma_{\boldsymbol{x}}$ and $\sigma_{\hat{\boldsymbol{x}}}$ are the standard deviations, and $\sigma_{\boldsymbol{x}\hat{\boldsymbol{x}}}$ is the covariance between corresponding patches. The constants C_1, C_2, and C_3 prevent division by zero and improve numerical stability. The luminance, contrast, and structural terms are combined to compute the SSIM score:

$$\mathrm{SSIM}(\boldsymbol{x}, \hat{\boldsymbol{x}}) = I(\boldsymbol{x}, \hat{\boldsymbol{x}})^{\alpha} C(\boldsymbol{x}, \hat{\boldsymbol{x}})^{\beta} S(\boldsymbol{x}, \hat{\boldsymbol{x}})^{\gamma}, \tag{2.62}$$

where the exponents α, β, and γ control the relative contributions of luminance, contrast, and structural components. SSIM is widely used for image quality assessment because it reflects perceptual differences and captures both intensity and structural information at local image regions [86].

3 Methodological framework of generative deep learning for advanced battery materials

This chapter discusses the methodological frameworks used in this work to implement generative deep learning for the development of advanced battery materials. As shown in the Figure 3.1, the methodological framework of this thesis is classified into the data layer, the generative core, and the application layer. The data layer comprises workflows for data extraction, preprocessing, augmentation, and management. The generative core defines the model architecture type based on the intended application. The application layer defines the possible application modes based on the model architecture.

3.1 Data Layer

This section focuses on the foundational data layer. Data lies at the basis of any data-driven analysis upon which generative models learn, generalize, and generate new data instances. In the context of the development of battery materials, the data span multiple scales, from atomistic, interphase configuration, and microstructural data to macroscale electrochemical material properties and performance metrics. The quality, diversity, and representativeness of the dataset strongly influence the generative model's ability to capture structure-property-performance relationships.

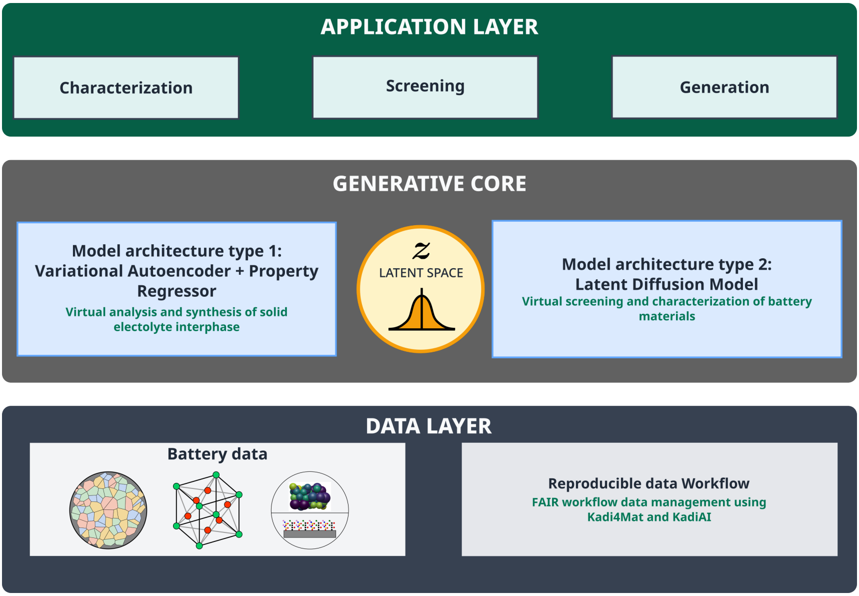

APPLICATION LAYER
Characterization
Screening
Generation
GENERATIVE CORE
Model architecture type 1: Variational Autoencoder + Property Regressor
Virtual analysis and synthesis of solid electolyte interphase
z
LATENT SPACE
Model architecture type 2: Latent Diffusion Model
Virtual screening and characterization of battery materials
DATA LAYER
Battery data
Reproducible data Workflow
FAIR workflow data management using Kadi4Mat and KadiAI

Figure 3.1: Overview of methodological framework of the thesis. The bottom frame constitutes the data preprocessing and management workflow. The middle frame selects a latent-powered generative model architecture for the intended application. The top layer defines the application modes, such as characterization, screening, and generation, based on the selected model architecture.

The following subsection provides an overview of the possible battery data sources along with the preprocessing, normalization, and data representation strategies applied to prepare the data for generative modeling.

3.1.1 Battery data sources

The battery material datasets originate from multiple sources, including experimental measurements, computational simulations, and data-driven methods, which are collected to understand material structure, composition, and performance.

Analyzing the influence of atomic structure, crystallographic domain arrangement, and composition on nano-, micro-, and mesoscale features of electrode materials is essential for improving battery system efficiency. Experimental characterization techniques encompass a wide range of methods for understanding structural, morphological, and electrochemical properties across different length and time scales. X-ray diffraction reveals phase composition and changes in electrode materials. Electron microscopy methods reveal nanoscale morphology and interfacial processes. Spectroscopic techniques, such as X-ray photoelectron spectroscopy, provide detailed insights into the electronic structure, oxidation states, and local chemical environments of electrode and electrolyte materials. Atomic force microscopy and other scanning probe methods enable characterization of morphological, mechanical, and local electrochemical behavior. The probe microscopy techniques also reveal how these factors affect the battery's material properties during charge and discharge cycles [87, 88].

Despite advances in experimental techniques, low throughput remains a challenge for many methods. This limits scale and diversity in available data, resulting in sparse coverage of the vast compositional and structural design space of potential battery materials. Available data is also often biased toward commercially dominant chemistries. To address these challenges, data augmentation and targeted data collection are required. These approaches are designed to mitigate the effects of data sparsity and bias by expanding the diversity and inclusiveness of the dataset. By generating synthetic data through augmentation and focusing data collection on underrepresented areas, the aim is to create a more balanced and comprehensive dataset, enhancing the robustness of the generative deep learning models [89, 90].

Lack of standardized data formats, variations in cell designs, cycling conditions, and measurement protocols complicate data aggregation and interoperability across studies. These limitations highlight the need to complement experimental data with systematically generated computational datasets for scalable generative modeling. In computational simulations of materials, density functional theory (DFT) [91] and molecular dynamics (MD) [6] overcome limitations in experimental data. DFT predicts formation energies, electronic structures, and phase stability of compounds, producing well-labeled, internally consistent datasets for deep learning. Molecular dynamics simulations offer atomistic insights into temperature-dependent transport, interfacial dynamics, and ion-diffusion mechanisms. Machine learning-accelerated potentials enhance these methods, achieving near-DFT accuracy at the molecular dynamics scale [92]. Kinetic Monte Carlo methods bridge quantum and continuum scales, modeling phenomena across length and time scales [93, 94]. Advances in computational techniques have led to the creation of open materials databases such as the Materials Project [95], the Open Quantum Materials Database [96], AFLOW [97], and the Inorganic Crystal Structure Database [98]. In this work, simulation-based databases support the development of generative deep learning frameworks for advanced battery material design and analysis.

3.1.2 Data Preprocessing

Raw data from experimental measurements and simulations are inherently diverse and heterogeneous, often containing noise, missing values, and inconsistencies. Data preprocessing transforms these raw datasets into clean, consistent, and structured formats suitable for data-driven analysis. Preprocessing includes data cleaning, reduction, scaling, and transformation. These steps improve data quality and enhance the performance and interpretability of subsequent analytical tasks [99, 100].

Data cleaning is the first preprocessing step, resolving data issues that may distort model performance by handling missing values, errors, and outliers. Missing values, due to incomplete measurements and simulations, are addressed through imputation, which estimates replacements based on the dataset. Imputation can be univariate, replacing missing values with the column's mean, median, or mode, or multivariate, leveraging features from other datasets. K-Nearest Neighbors (KNN) selects K similar rows to estimate missing values [101–103]. Outlier detection identifies values that deviate from the rest, as outliers bias statistics and model performance. Detection approaches vary by data type: standard deviation flags points more than a set number of standard deviations from the mean (for normal data), the Interquartile Range (IQR) flag points more than 1.5 IQRs from either quartile (for skewed data), and z-scores classify points exceeding a set threshold. For complex, high-dimensional data, techniques such as isolation forests, autoencoder reconstruction, and clustering algorithms can detect anomalies [99, 100].

The data reduction step in preprocessing tackles large, multidimensional datasets. Row-wise reduction selects representative samples via random or stratified sampling, preserving statistical balance between material types or cycling conditions. Clustering further identifies redundant or correlated data segments. Column-wise reduction minimizes the number of input variables by selecting or extracting features. Filter methods rank descriptors by correlation or mutual information; wrapper methods use recursive feature elimination; and embedded methods integrate feature selection into model training via regularization techniques such as LASSO. Feature extraction methods, such as principal component analysis (PCA), compress descriptors into orthogonal components that capture the dominant variance [104]. Nonlinear autoencoders or convolutional encoders capture complex interactions among microstructure, chemistry, and electrochemical response. These techniques efficiently integrate experimental and simulated datasets with redundant descriptors [105, 106].

Data transformation converts data into formats suitable for analysis. During preprocessing, scaling and standardization techniques normalize numerical values to a standard scale or distribution. Commonly used data scaling methods include min-max normalization, which scales each feature to a specified range. Max normalization scales each record by its local maximum to reduce the influence of outliers. Standardization, or z-score normalization, transforms the data so that it fits a Gaussian distribution. This centers observations around the mean with unit variance. For categorical features, label or one-hot encoding converts them into numerical values [107–109].

Advanced imaging techniques used in battery research, such as high-resolution X-ray computed tomography, FIB-SEM, and synchrotron techniques, yield high-resolution and high-dimensional datasets. The imaging datasets require dedicated preprocessing steps, including noise suppression, image alignment, voxel-intensity normalization, and segmentation to determine active material, binder, electrolyte, and pore phases. ML-based segmentation, watershed algorithms, and convolutional neural networks are now used to achieve consistent phase identification. From the segmented imaging data, morphological parameters such as porosity, tortuosity, particle connectivity, and interfacial area can be extracted to simulate their linkage with transport and mechanical properties [110, 111].

3.1.3 Data augmentation

Data augmentation increases the diversity of training datasets by applying realistic, random transformations. Augmentation strategies are tailored to the specific task and dataset dimensionality. For image and microstructural data, augmentation includes geometric and intensity-based transformations to replicate real-world variability. Geometric transformations such as rotation, reflection, translation, cropping, and elastic deformation capture orientation-invariant features. Intensity-based augmentations, including histogram equalization, contrast adjustment, Gaussian noise injection, and Poisson noise replication, reproduce variability introduced by imaging techniques. Advanced augmentation methods, such as patch mixing, style transfer, and texture synthesis, utilize generative models like GANs and VAEs to sample diverse configurations by learning the higher-order structural data information. These generative model-based approaches produce realistic morphologies that maintain pore connectivity and grain morphology in microstructures [112, 113].

For crystal structure-based materials data, the applied data augmentation process must preserve lattice periodicity, stoichiometry, and bonding topology. Conventional augmentation approaches used in this direction include random perturbation, rotation, translation, and axis swapping, which produce new crystal configurations that maintain periodicity and symmetry [114]. In recent approaches, the lattice-scaling method is used to augment the crystal dataset for further data-driven analysis. In lattice scaling, the lattice parameters of the crystal structure are modified isotropically and anisotropically within a narrow range, e.g., simulating the volumetric strain observed in an insertion-based mechanism during charge and discharge cycles [115, 116].

Data extracted from open materials databases, such as the Materials Project [95], is often dominated by widely used materials and their compositions, resulting in imbalanced datasets. Models trained on such datasets may exhibit bias toward majority classes and poor generalization for minority classes. To address this, data augmentation strategies based on sampling techniques are employed. Oversampling duplicates minority-class samples or generates synthetic data via interpolation, such as the Synthetic Minority Oversampling Technique (SMOTE), to preserve statistical properties and prevent overfitting. Undersampling reduces the majority class by removing redundant or non-informative samples, achieving a more balanced dataset at the expense of total data volume [117, 109].

3.1.4 Data management workflow

Incorporating data management into everyday research operations simplifies and organizes the research data lifecycle. Within a machine learning framework, the data layer includes acquisition, preprocessing, and augmentation. An efficient research data management (RDM) tool ensures consistency, reproducibility, and transparency across these stages. An integrated RDM tool promotes FAIR (Findable, Accessible, Interoperable, and Reusable) data principles and supports automation and scalability of complex scientific processes [118].

The growing adoption of data-driven methods in materials science has resulted in the generation of multifaceted datasets from experiments and simulations [119, 120]. Heterogeneous data from these sources must be integrated, standardized, and analyzed to extract meaningful insights. Implementing FAIR data principles in materials research enhances data accessibility and makes it easier for the research community to manage, share, and reuse.

A flexible RDM infrastructure should provide access to both raw and intermediate data generated in ongoing studies, as these datasets contain valuable information for adaptive model refinement and early-stage insight generation. Managing intermediate data enables continuous feedback between experimental and modeling activities, with systematic documentation, versioning, and analysis. Integrating such infrastructure reduces redundant experimentation, enhances reproducibility, and maintains traceable research processes. Established research data platforms, including Zenodo [121], Dataverse [122], DSpace [123], and NOMAD [124], manage and publish finalized datasets. To support daily research operations, additional tools are necessary to handle unprocessed data, automate workflows, and enable cross-platform data integration. These functions are supported by electronic lab notebooks (ELNs), which digitalize experimental and computational records and link recorded data to subsequent analyses.

Commonly used ELNs include Jupyter Notebooks [125], Galaxy [126], Fireworks [95], ElabFTW [127], and Aiida [128]. These ELNs are effective for documenting experiments, conducting simulations, and tracking data provenance. However, most lack interdisciplinary capabilities and require programming expertise to automate research workflows. Therefore, a research data management system capable of supporting interdisciplinary research processes is necessary. Such a system should provide access to intermediate data, which refers to unpublished data yet to be analyzed. Integrating ELNs with repository-based RDM tools that support both user-interface and script-based workflow implementations reduces the effort required for daily research activities such as data retrieval from experimental devices, data sharing, data analysis, and visualization.

To address the aforementioned requirements, Kadi4Mat has been developed as an open-source data platform at the Karlsruhe Institute of Technology [129]. The Kadi ecosystem integrates a data repository, electronic lab notebook (ELN), and workflow management system. The repository organizes heterogeneous data into structured records enriched with metadata, while the ELN component, KadiStudio [130], enables the design and execution of experimental and computational workflows in a reproducible format. Both web-based and programmatic interfaces support the creation of interactive, automated workflows, enabling researchers to smoothly transition between manual exploration and high-throughput computational pipelines.

Beyond data management, Kadi4Mat provides access to KadiAI and CIDS (Computational Intelligence and Data Science) [131–134], which extend the platform capabilities to include machine learning and artificial intelligence (AI) applications. These components enable researchers to define, execute, and monitor AI-driven workflows within the same infrastructure. KadiAI offers interactive dashboards for defining machine learning pipelines and managing model training, while CIDS provides a Python-based library for data preprocessing, feature engineering, model development, hyperparameter tuning, and performance analysis. Together, they enable both graphical and script-based workflow execution, ensuring accessibility for users with varying levels of technical expertise. The functionalities defined in KadiAI and CIDS enable the creation of intelligent data analysis projects and the recording of associated metadata at each step in an ML workflow, describing input sources, preprocessing steps, model configurations, and results [135].

In this work, the Kadi4Mat ecosystem serves as the central RDM framework for implementing reproducible generative deep learning workflows. These workflows establish traceable model development pipelines for each data-driven application, including the virtual analysis of solid electrolyte interphase formation in battery systems, as demonstrated in [135].

3.2 Generative core

This section describes the core components utilized to develop the generative frameworks presented in this thesis. The subsequent subsections outline the principles guiding model selection, architectural customization, and domain-specific modifications for analyzing materials data.

3.2.1 Model selection criteria

Selecting an appropriate generative model for a given application depends on the characteristics of the input data, the model's capacity to capture higher-order structure data and its ability to generate meaningful outputs from learned higher order representations. In materials research, generative models need to reproduce observed data and generate new instances that conform to the physical constraints of the real materials.

The selection of a generative model is influenced by how it learns to estimate the density of the input data distribution. Generative models can be broadly categorized as explicit and implicit density estimators. Explicit density models, such as VAEs, learn a parameterized distribution over the data by optimizing a variational lower bound on the likelihood. VAEs (Figure 3.2a) build on the autoencoder architecture, encoding input data into a reduced latent space that can then be used for generative tasks, such as synthesizing new material structures or images. Unlike standard deterministic autoencoders, which map each input to a single latent vector, VAEs treat the latent space probabilistically, encoding a range of possibilities for each latent variable. This probabilistic encoding allows the model to generate entirely new samples that are unique yet consistent with the original data distribution [57].

To generate meaningful new samples, the learned latent space must satisfy continuity, meaning that nearby points decode to similar outputs, and completeness, meaning that any point sampled from the latent space produces a valid result. These properties are enforced by constraining the latent space to follow a normal distribution. Each latent variable is represented by the mean vector μ, and the standard deviation vector σ, which together define the range of possibilities and expected variance for that variable. The VAE training objective combines a reconstruction loss with an additional KL divergence term that penalizes deviations of the learned latent space from a standard Gaussian distribution. This ensures that sampling from the latent space produces coherent, physically plausible outputs (Section 2.5.1). The balance between reconstruction loss and KL Divergence ensures that the learned latent space remains smooth, interpretable, and suitable for sampling new physically valid configurations. Sampling from the learned latent space enables exploration of the underlying data manifold to generate continuous latent representations in materials science.

Implicit density models, such as GANs (Figure 3.2b), generate data by taking noise as input. Instead, GANs generate samples that mimic the true data distribution through a min-max competition between a generator, which produces synthetic data, and a discriminator, which distinguishes generated from real examples. The adversarial training process allows the generator to capture complex, high-dimensional relationships without assuming an explicit likelihood model. GANs are particularly effective for generating high-fidelity microstructural images or complex morphological features where visual accuracy and fine structural details are essential [64, 136].

Diffusion models (Figure 3.3a) represent a class of generative models that combine the characteristics of both explicit and implicit density estimation. During training, diffusion models behave as explicit density estimators, as they are optimized by maximizing a variational lower bound on the data likelihood. In contrast, during generation, the diffusion model operates as an implicit model by taking noise as input, where generation proceeds through an iterative reverse diffusion process that gradually transforms random noise into structured samples [70, 71, 69].

In addition, hybrid generative models are developed to combine the complementary strengths of different generative models. The VAE-GAN (Figure 3.2c) model integrates the representational stability of VAEs with the perceptual sharpness of GANs. This allows the model to evaluate similarity in a learned feature space rather than at the pixel level, making the reconstruction more invariant to local shifts and noise [137–139]. Extending the diffusion model approach, LDM (Figure 3.3b) integrate autoencoding, adversarial learning, and diffusion mechanisms into a single framework. By performing diffusion in a lower-dimensional latent space, LDMs achieve efficient, high-quality generation while significantly reducing computational cost. This hybridization enables scalable, interpretable, and high-fidelity generative modeling suited for complex materials data [73].

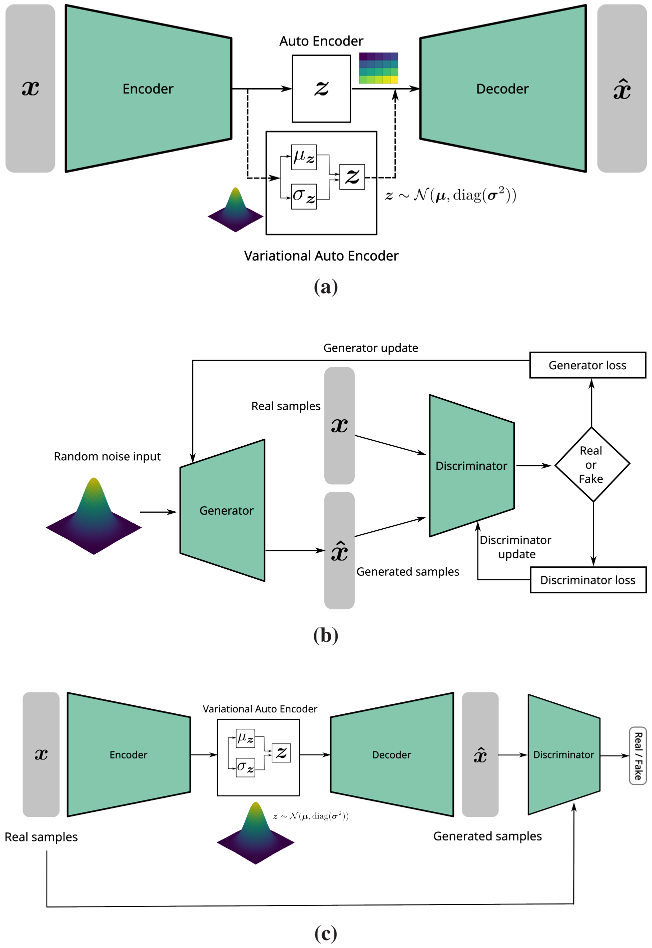
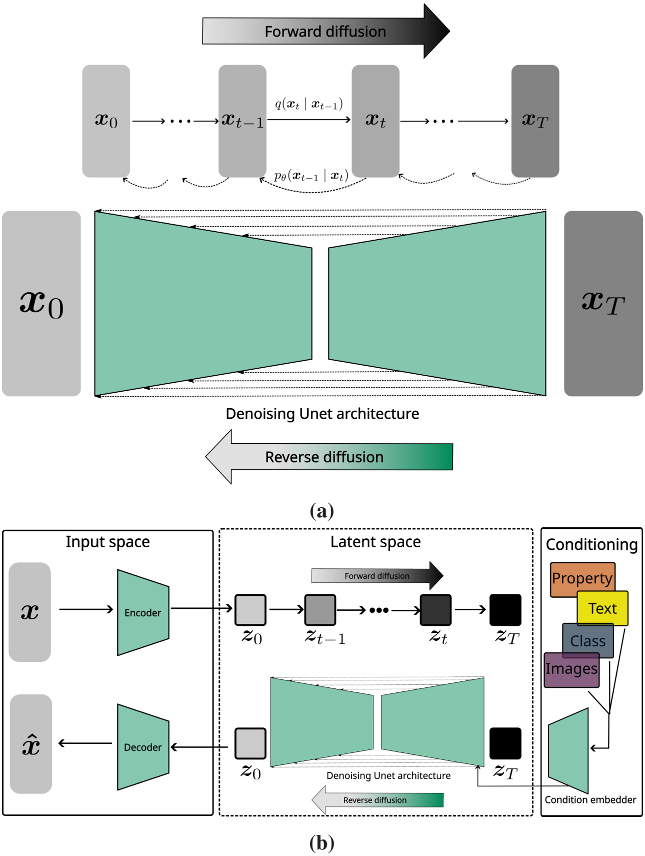


Figure 3.2: Schematic overview of generative deep learning architectures: (a) Variational Autoencoder (VAE), (b) Generative Adversarial Network (GAN), and (c) integrated VAE-GAN model.

In this work, two hybrid generative modeling approaches are implemented to develop latent-space-based frameworks for material characterization, screening, and generation. In the first application, a VAE and a regressor (prVAE) model are jointly trained to enable the virtual analysis and synthesis of the solid electrolyte



(a)

(b)

Figure 3.3: Schematic comparison of diffusion model and latent diffusion model. The diffusion (a) operates in input space, whereas the latent diffusion model (b) employs a two-stage approach using an autoencoder to confine the computationally expensive denoising process to a lower-dimensional latent space.

interphase (SEI) in batteries. The VAE learns a compact latent representation of SEI morphologies, while the regressor maps these representations to physical properties such as thickness, porosity, density, and volume fraction. This joint training organizes the latent space according to relevant material properties, allowing both the characterization of existing SEI configurations and the inverse generation of new interphases with desired properties [108].

In the second application, an LDM integrates a variational autoencoder and a diffusion-based generative process to enable conditional generation and screening of battery materials. The LDM operates in a learned latent space, where high-dimensional material data are efficiently represented, enabling diffusion-based sampling guided by functional properties such as average voltage, capacity, and energy density, as well as the translation of material characterization data into understandable material descriptors [109].

3.3 Application Layer

This section discusses the different application modes of the generative models in materials data analysis. By incorporating additional training objectives beyond reconstruction and statistical accuracy, generative frameworks can be adapted for tasks such as material characterization, screening, and inverse design. The extended objectives guide the models to capture not only statistical similarity but also property correlations in the latent space, enabling meaningful exploration and generation of information-rich samples.

3.3.1 Characterization

In the characterization mode, the explicit generative model is employed to extract and represent latent relationships between structure, composition, and physical properties. The trained model embeds complex high-dimensional material data into a compressed latent representation, providing a reduced representation of the material behavior. The learned latent representation encodes essential features of high-dimensional material data. This mode focuses on identifying physically interpretable patterns and correlations that characterize material behavior and the translation of materials data into interpretable material descriptors.

Information-rich latent space learning

Representation learning enables automatic extraction of generic features to solve other ML tasks, such as regression and classification. Representation learning provides a foundation for understanding complex material data by mapping high-dimensional data to a structured, lower-dimensional latent space [140]. The latent space encodes abstract representations of high-dimensional, complex data that capture the underlying features not directly observable. The desirable properties of the latent space include smoothness and disentanglement. The smoothness of the latent space indicates that it is continuous, which ensures that small variations within result in proportional variations in the output. The disentanglement property ensures that altering a single latent dimension mainly affects the features of a single category, while leaving the features of other categories relatively unchanged [141].

Traditional dimension reduction techniques have been employed to learn meaningful latent representations of high-dimensional data. Linear methods such as principal component analysis (PCA) focus on capturing maximum variance for dimensionality reduction [142, 143]. PCA is an orthogonal linear transformation of a real inner product space that projects the data into a new coordinate system such that the maximum variance lies along the first principal component, with decreasing variance along the remaining principal components. By decomposing the data covariance matrix into eigenvectors and eigenvalues, PCA identifies eigenvectors as directions of maximum variance and eigenvalues as the corresponding magnitudes. Selecting components with the largest eigenvalues forms a compact latent representation that retains most information while reducing redundancy and noise. The linear discriminant analysis (LDA) emphasizes maximizing separation between classes in the low-dimensional space [144, 145]. Kernel PCA is an extension of PCA that applies a nonlinear mapping function to the data before performing any linear transformation, thereby capturing more complex, nonlinear relationships between data points [146, 147].

The advent of deep learning and generative models expanded representation learning beyond linear and nonlinear embeddings. Deep neural networks learn hierarchical representations through multiple layers, progressively extracting abstract features. Convolutional neural networks (CNNs) capture spatial structure in the data and provide translation invariance [148]. Out of the different generative model classes, the VAE provides a tractable latent space for efficient exploration, interpolation, and optimization of a large material design space [149, 150]. However, in most vanilla VAE methods, balancing KL divergence regularization and reconstruction quality is difficult, leading to entangled latent representations. Weighting methods are used in VAEs to balance the contributions of the KL divergence and the reconstruction loss to achieve disentangled latent representations [151]. Different hybrid approaches based on a VAE-discriminative model are used to construct an ordered latent space from discrete labels or continuous properties of the input data. An additional discriminative model, such as a classifier or regressor, helps disentangle the latent space to form an information-rich lower-dimensional representation. The additional discriminative model in the VAE training disentangles and organizes the latent space according to material properties [152, 153].

The above approach is demonstrated in the Section 4.1.3 [108] of this thesis, which focuses on the data-driven virtual characterization and synthesis of SEI to enhance battery performance and safety. In this study, a VAE–regressor hybrid model is trained to jointly learn the structure–property relationships of the SEI. The regressor helps to organize the latent space according to physical properties, enabling virtual exploration and synthesis of interphases with desirable physical properties. The detailed results of this contribution are presented in the Section 4.1.

Translation of characterization data to material descriptors

In translation mode, the generative model converts material data into meaningful material descriptors. This functionality is used to extract information from characterization techniques such as atomic force microscopy, scanning electron microscopy, and transmission electron microscopy [154, 155]. While these methods provide high-resolution morphological and structural information, translating heterogeneous datasets into descriptors that capture structure–property relationships remains challenging. Generative models address this challenge by learning mappings between data modalities, enabling the conversion of raw experimental observations into structured, property-relevant representations.

The Pix2Pix GAN model is a supervised image-to-image translation model that learns direct mappings between the input and output domains. In Pix2Pix GAN model, the generator network transforms input images into target representations, while the discriminator evaluates the realism and structural consistency of the translated outputs. The learning objective combines an adversarial loss that enforces global similarity to the target distribution with a reconstruction loss that ensures spatial correspondence between the target and translated outputs [156]. In this direction, the cycle-consistent GAN introduced unsupervised image-to-image translation by incorporating a cycle-consistency loss that ensures the translation from one domain to another and back preserves the original structure. The cycleGAN is particularly useful when a paired image dataset of identical regions of material is unavailable [157]. The development of such translation models enables multimodal characterization to better understand material behaviour across different scales.

In this work, the VAE-GAN component of LDM, as demonstrated in the Section 4.2.3 [109], is used to translate simulated AFM images of battery electrodes during discharge into meaningful material descriptors. The translation mode of the VAE-GAN framework is activated by incorporating an adversarial objective from Pix2Pix GAN. The results of the translation mode are presented in the Section 4.2.

3.3.2 Screening

Selecting a material configuration for a specific application is often time-consuming and costly. Generative model-based screening extends beyond reproducing or classifying existing data by exploring the learned material space to identify configurations that meet specified target properties. This approach transforms conventional screening into generative exploration guided by the learned distribution of material configurations.

Targeted material screening

The traditional screening process in materials development involves using computational or experimentally cheaper techniques to filter a large dataset to isolate a small subset of candidates with desirable properties, followed by experimental validation. The traditional screening process is employed in sequential steps, including descriptor-based filtering, multi-criteria decision making, regression-based property prediction, and global optimization techniques, e.g., genetic algorithms [158, 159], with subsequent property estimation,n e.g., density functional theory (DFT) and molecular dynamics (MD)[160, 161]. The above mentioned computational approach narrows down candidates by ranking based on specified performance criteria. While effective, this approach relies heavily on preexisting databases, expert intuition, and incremental modifications of known materials, which makes the material development cycle iterative, expensive, and inherently limited to the known material design space.

Generative modeling redefines the material screening workflow by enabling the direct generation of candidate materials from the learned data distribution [162, 163]. Instead of screening what already exists, the generative model can learn the underlying data distribution and sample new candidates directly from the learned continuous latent space, enabling interpolation and extrapolation towards the desired property. In a VAE model, the encoder-decoder architecture extracts the essential features of the material. The learned latent space in VAE encodes material features related to composition, bonding, and stability. The decoder reconstructs the material configurations from this latent space. The property-conditioned variants of VAE frameworks unlock the targeted screening of materials. The VAE-based models seem to perform well with single property constraints since multiple property constraints can cause entanglement of latent representation and thereby limit the target material screening. But the material screening process requires a combination of target properties to decide on the material for an application. In this direction, the conditional LDM is better suited for multi-property screening compared with the vanilla VAE framework. The iterative denoising process in the LDM gradually transforms the random noise in the learned latent representation into a valid material representation over many timesteps. The iterative denoising gradually injects multiple conditioning signals through a cross-attention-based mechanism in the LDM architecture, which influences the update direction of the latent representation during the denoising process [164].

This conditional LDM is demonstrated in [109], which focuses on virtual screening of battery material based on functional properties. The conditional LDM leverages multiple functional properties, such as average voltage, energy density, and capacity, to guide targeted screening of battery materials. The implementation details and results of the conditional LDM will be discussed in Section 4.2.

3.3.3 Generation

The generative mode constitutes the unifying stage of the methodological framework, in which the trained generative model advances beyond characterization and screening to create new, physically meaningful material configurations via the inverse design strategy. In the generation mode, the model synthesizes novel material structures, morphologies, or compositions guided by the learned latent distribution and conditioned target properties. This capability enables inverse mapping from desired material properties to structural or compositional configurations, facilitating virtual design and discovery of new materials [165].

The generative process utilizes latent representations learned during information-rich characterization and targeted screening. Sampling within the latent space enables exploration of previously uncharted regions of the material design space, resulting in novel structures that remain statistically consistent with the studied material distribution. This approach accelerates material discovery, supports reconstruction of incomplete data, augments sparse datasets, and generates hypothetical material configurations for targeted applications.

In this work, the generative mode integrates the generative model architectures developed in Section 4.1 [108] and Section 4.2 [109]. The prVAE model demonstrated in Section 4.1 establishes a structured latent space that supports the virtual synthesis of solid electrolyte interphases (SEIs) with desired physical and chemical properties. Once the latent space is trained, new SEI structures can be generated by sampling latent variables corresponding to target physical properties, such as interphase thickness, density, porosity, and volume fraction. This virtual synthesis capability enables exploration of SEI configurations that may not be easier to observe experimentally but remain realizable within the learned data manifold (Section 4.1.5).

The LDM, as demonstrated in Section 4.2, further extends this capability by introducing an iterative, property-conditioned generative process. In the LDM framework, diffusion is performed within a learned latent representation that encodes structural, compositional, and functional correlations. Through iterative denoising guided by property-conditioning mechanisms, the model generates material candidates that simultaneously satisfy multiple target properties, such as energy density, capacity, and voltage stability. This multi-property conditioning allows for the generative exploration of a high-dimensional design space, bridging the gap between structure-property learning and target property-driven design.

The methodological framework developed in this work integrates the prVAE and latent diffusion model architectures to establish a comprehensive generative approach for materials research. Specifically, the VAE regressor enables the construction of property-structured latent spaces, facilitating interpretable characterization and efficient virtual synthesis of SEI configurations. The LDM enables multi-objective material generation, enabling diverse property constraints during the design process. This unified framework delivers a coherent, traceable workflow that links latent representation learning to the targeted design and synthesis of advanced battery materials.

4 Results of generative deep learning for battery material design and analysis

This chapter presents the results of the methodological framework discussed in the previous chapter. The first demonstration of the framework focuses on data-driven design and analysis of solid electrolyte interphases using property regressor VAE. The second demonstration of the framework focuses on the virtual screening and characterization of battery materials using a latent diffusion model.

4.1 Data-driven design and analysis of solid electrolyte interphases using property regressor VAE

The solid electrolyte interphase (SEI) is a passivation layer formed on electrode surfaces from the decomposition products of electrolytes during the initial charge-discharge cycles of the battery. SEI allows ions to pass while blocking electron transport, preventing further electrolyte decomposition [166]. The formation and stability of the SEI play an important role in the electrochemical performance, safety, and cycle life of batteries. Despite extensive research, the mechanistic understanding of SEI evolution remains incomplete due to its structural complexity and sensitivity to environmental and operational conditions. SEI exhibits a heterogeneous composition of both inorganic and organic compounds, and its distribution and morphology depend on factors such as electrolyte composition, electrode material, temperature, and charge current density.

Different experimental characterization techniques are used to understand the chemistry, physical properties, and formation process of SEI. Ex situ methods, such as X-ray photoelectron spectroscopy (XPS), are used to reveal chemical information about the SEI [167–170]. In situ and operando methods, such as atomic force microscopy (AFM), transmission electron microscopy (TEM), and scanning electron microscopy (SEM), are used to study the morphological evolution of the SEI, including volume expansion and crack formation. However, these techniques are limited by resolution, the higher sensitivity of SEI, and the difficulty of accessing buried interfaces. As a result, a complete understanding of SEI growth and transformation mechanisms has yet to be achieved [171].

Computational modeling provides a complementary approach to probe SEI formation across scales that are experimentally inaccessible. Quantum chemical [166, 172] and molecular dynamics [173, 174] simulations are widely used to investigate the early stages of electrolyte reduction and SEI nucleation at the atomic level. Although computational methods offer detailed mechanistic insight, the limited temporal and spatial scales prevent simulating SEI growth beyond initial formation. To bridge the scales, the kinetic Monte Carlo (KMC) method provides an intermediate framework that captures both molecular and mesoscale evolution by using reaction rates derived from quantum-chemical calculations. KMC captures the spatiotemporal evolution of SEI, governed by a series of chemical reactions, diffusion, and aggregation, based on kinetic information computed for the given electrolyte-anode chemistries [94].

Conventional approaches to optimizing battery interphases rely on trial-and-error, progress from known structures, and require substantial expert effort to progress further. Inverse design, in contrast, based on a data-driven approach, starts with target performance or properties and generates the structures that realize them, addressing the complexity of battery systems across multiple scales and phases [4]. Generative models offer a principled approach to modeling complex SEI configurations with target observable properties such as porosity, volume fraction, density, and thickness [163]. In this work, a VAE with a property regressor (prVAE) is jointly trained to learn the key features of 2D SEI configurations from 50000 samples obtained at the end of each KMC simulation for a given set of possible reaction barriers in lithium-ion battery chemistry. The key features extracted from the SEI configuration are studied at the VAE bottleneck to understand how the SEI's observable properties influence the learned data-driven properties. To further improve the classification of 2D SEI configurations based on their influential properties, the variational autoencoder model was conditioned on a reaction barrier set tailored to specific SEI conditions. Therefore, this data-driven strategy generates SEI configurations with tailored physical properties for given sets of reaction barriers.

4.1.1 Data generation and preprocessing

The SEI configuration used for the study is simulated using a 2D KMC scheme developed by Esmaeilpour et al. 2023 [94], which implements a rejection-free BKL algorithm to simulate SEI growth at molecular level resolution. [175]. The implemented KMC scheme follows a bottom-up multiscale modeling strategy that captures the evolution of mesoscale processes. In this study, simulation is applied to lithium-ion battery (LIB) chemistry to model SEI growth. The results of the simulation are represented in a 2D square lattice where each site corresponds to individual molecular level resolution $\approx 1nm$ of components of SEI in LIBs such as Li_2EDC, Li_2CO_3, and $C_2H_4OCOOLi$. The simulations are performed on a 50 x 50 lattice, with the bottom boundary representing the electrode surface and the opposite boundary modeled as an open absorbing boundary condition. Open-absorbing conditions allow components to diffuse through this boundary and leave the system irreversibly. Each lattice site can represent either a solvent or a product of the reaction between neighboring sites. The rejection-free KMC algorithm selects one reaction from the set of possible reactions to transition from one state to the next, based on transition-state theory [176]. SEI formation proceeds through a sequence of chemical reactions, beginning with electron reduction that produces both inorganic and organic species. Inorganic components tend to cluster, forming the inorganic domains, while organic molecules diffuse and aggregate into the organic portion of the SEI. To model the diversity of SEI growth, reaction rates are drawn from the literature [177–179]and sampled systematically using Latin hypercube sampling. In total, fifteen distinct reaction rates govern the process, and 50,000 unique, 15D reaction rate vectors are generated for the simulations. Each vector leads to a distinct spatiotemporal configuration of the SEI. The resulting simulation outputs are color-coded to visually distinguish inorganic, organic, and intermediate precursor species in the final SEI snapshot. Key structural observables of SEI calculated in this study include volume fraction, thickness, density, and porosity of the SEI layer.

After generating the SEI dataset using the KMC algorithm, the resulting simulation outputs are preprocessed for a data-driven study using prVAE. The source dataset from the simulation consists of 50000 2D SEI configurations obtained at the last step of KMC simulations, along with their observables, such as volume fraction, thickness, density, and porosity, and the corresponding reaction barrier for the 15-dimensional vector. The 2D SEI configurations are encoded categorically, followed by one-hot encoding based on the color codes assigned to each reaction product in the considered configuration. The observables of the SEI configuration and reaction barrier set are preprocessed to a normalized range to improve the performance and stability of the implemented generative framework. The preprocessed data for each SEI configuration is written to a TFRecord file for efficient serialization of structured data and to prepare the data for further machine learning study.

In this study, the machine learning workflow is managed using Kadi4Mat, an open-source data management platform discussed in detail in the methodology chapter. The use of Kadi4Mat enables the systematic collection and organization of simulation data and associated metadata [129]. The electronic laboratory notebook component of Kadi4Mat offers a variety of tools for data handling, preprocessing, and analysis [130]. To support data-driven research, KadiAI and CIDS extensions within the Kadi4Mat framework are used, which provide specialized functions for developing and running machine learning tasks.

4.1.2 Property regressor VAE architecture and Training

The proposed prVAE in this study uses a complementary approach of generative and discriminative modeling. As discussed in the Chapter 2, the vanilla VAE learns the continuous latent space focusing only on the extracted essential spatial features of the SEI configurations. To enable effective inverse design of SEI configurations, it is necessary to understand the structure-properties relationship of the SEI formation mechanism for a given reaction barrier set.

To enable data-driven characterization, VAE can be trained jointly with a property regressor network. The property regressor implemented as a dense neural network predicts the desired SEI observable properties ($\hat{\mathbf{P}}$) from the latent representation and backpropagates the learned physical information into the latent space during training. This architectural setup incorporates a regression loss alongside the standard VAE objectives, guiding the model to learn both spatial features and key physical properties simultaneously. The convolutional neural network is used to define the encoder and decoder components of VAE. The property regressor is defined using fully connected layers to map the latent space to property predictions. During training, the decoder reconstructs SEI configurations from latent vectors sampled from the approximate posterior, whereas the property regressor receives only the latent mean as input. Figure 4.1a presents the prVAE model architecture used for training.

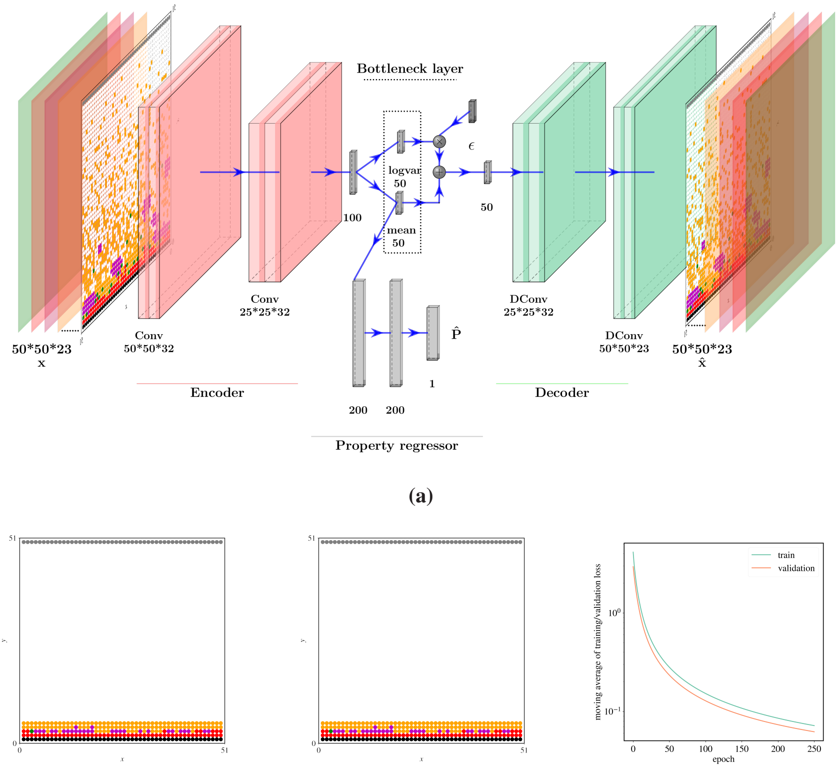
The following hyperparameters for the prVAE model are selected through manual hyperparameter tuning. The encoder is built from a sequence of 2D convolutional layers with filters of sizes 32, 64, and 128, followed by a dense layer with 100 units. The decoder reconstructs data via a series of 2D deconvolutional layers with 128, 64, 32, and 23 filters. To enhance stability and boost performance, batch normalization is incorporated after each convolutional layer. The latent space dimension is set to 50 based on empirical testing. Stride values of two and one are applied within the convolutional architecture, and rectified linear units are used as the activation function throughout. For property prediction, the regressor component uses two fully connected layers with 200 neurons each. For training, the preprocessed dataset is split into 0.7 for training, 0.15 for validation, and 0.15 for testing. The model is trained for 250 epochs with a learning rate of $3e^{-6}$. The ADAM optimizer minimizes the loss during training by updating model parameters.

Figure 4.1b and Figure 4.1c compare the reconstructed SEI configurations predicted by the prVAE to the actual ground truth of test samples outside the training dataset. The results show that the model accurately captures the essential features of SEI configurations and performs well on unseen data. Figure 4.1d further confirms that the loss curves for the training and validation sets are closely aligned, indicating that the prVAE maintains strong generalization and does not overfit to the training data.

4.1.3 Latent space based characterization of SEI

The learned latent space of prVAE is used to understand the structure-property relationship in the SEI configuration. The jointly trained regressor incorporates physical property information into the latent space via backpropagation during training. The integration of physical information in the learned data-driven properties of SEI configuration improves the expressiveness of the learned latent space. The obtained information-rich latent space enables characterization of SEI configurations with target physical properties.

The randomly selected test dataset of 7500 samples is used to evaluate the performance of prVAE in encoding complex structural configurations and physical properties of SEI in a reduced lower-dimensional latent space. To improve the visualization and interpretability of the 50-dimensional learned latent space, PCA is used to reduce its dimension by selecting principal components. The performance of the prVAE model is compared with that of the vanilla VAE in learning a continuous, information-rich latent space. Figure 4.2 shows that the prVAE model performs better than the vanilla VAE in capturing the physical information associated with the latent feature of SEI.



(a)

(b)

(c)

(d)

Figure 4.1: a) Schematic of the prVAE architecture, highlighting mean and variance as parameters of the encoded distribution, with ϵ indicating the stochastic term used for model reparameterization during training. b) Example SEI configuration selected from the test set. c) Predicted SEI configuration for the same test sample, demonstrating that the prVAE reliably reconstructs SEI features. d) Training loss curves, with $\hat{\mathbf{P}}$ representing model predictions for physical properties of SEI, such as volume fraction, thickness, porosity, and density. Reproduced from [108] under the terms of the Creative Commons Attribution–NonCommercial–NoDerivatives (CC BY-NC-ND) license.

The principal components of the learned latent space indicate that it is smooth, property-informed, and interpretable, enabling systematic characterization, optimization, and interpolation of SEI configurations. The prVAE encodes property information into the latent space, where each latent-space point decodes into a valid SEI configuration.

4.1.4 Prediction of SEI properties

The prVAE model enables targeted design of SEI configurations and property-informed sampling of SEI configurations with a target property. The VAE in prVAE serves as a generative framework that can reconstruct new SEI configurations by decoding sampled latent points. The property regressor in prVAE introduces an auxiliary neural network that predicts physical properties directly from the encoded latent representation of each SEI configuration. The performance of the property regressor is evaluated using R^2 score between the actual and predicted property on the test dataset. As shown in Figure 4.3, the property regressor achieves a higher coefficient of determination (R^2) between actual and predicted properties of SEI configurations from the test dataset. The high coefficient of determination shows that the property regressor captured almost most of the physical property information to encode in the latent space. The slightly lower values of R^2 score observed for porosity compared to other properties can be attributed to its low variance across the dataset.

Figure 4.4 shows that porosity exhibits weak correlation with the other physical properties of the SEI, indicating that it depends less on the encoded structural features that predominantly influence the other physical properties.

4.1.5 Guided generation of SEI

The SEI configurations generated from the latent space can be guided by introducing conditional inputs to the prVAE decoder. In this study, the reaction barrier space is used as an additional input alongside the sampled latent vectors from the encoded low-dimensional distribution. Through this conditioning, the prVAE learns to generate SEI configurations that correspond to specific reaction

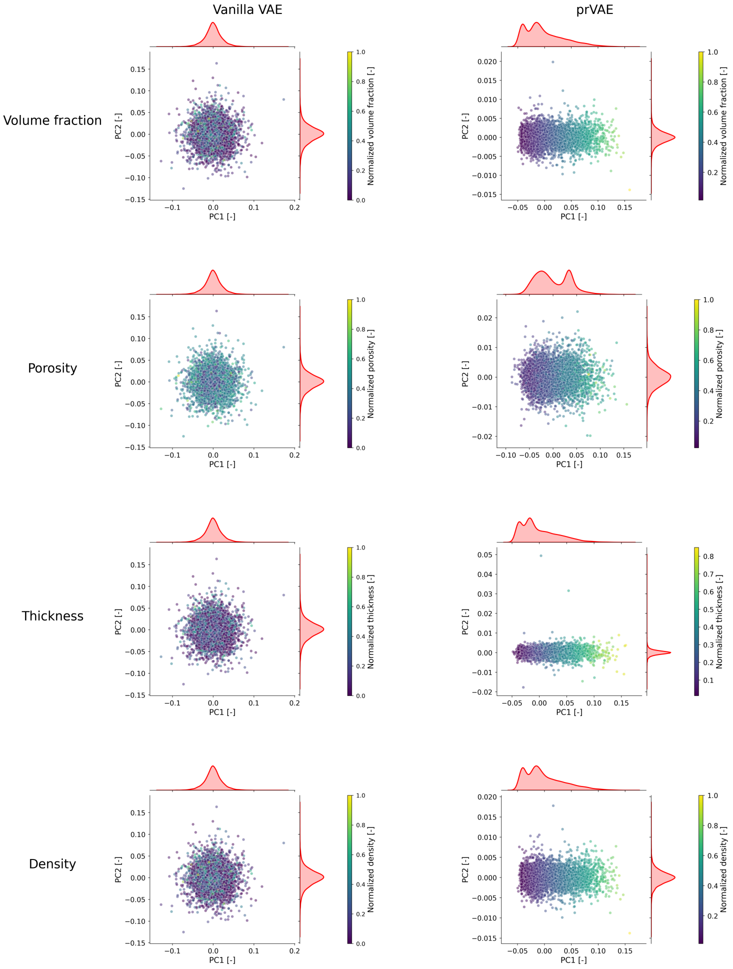


Figure 4.2: PCA of learned latent spaces for the vanilla VAE (left) and prVAE (right), showing that the prVAE produces a more structured, property-aware latent representation linked to SEI volume fraction, porosity, thickness, and density. Reproduced from [108] under the terms of the Creative Commons Attribution–NonCommercial–NoDerivatives (CC BY-NC-ND) license.

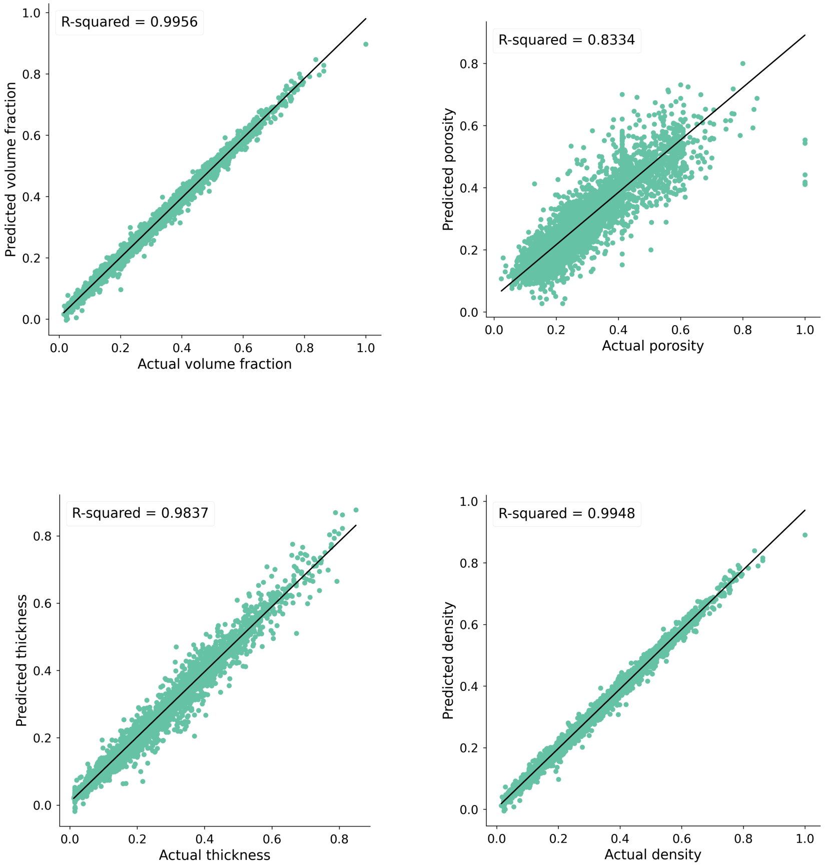


Figure 4.3: Prediction of selected properties of the SEI by the property regressor. The property regressor trained alongside the VAE achieved a high R^2 score for each selected property and high prediction accuracy. Reproduced from [108] under the terms of the Creative Commons Attribution–NonCommercial–NoDerivatives (CC BY-NC-ND) license.

barrier environments, allowing the reconstruction of reaction-dependent SEI configurations.

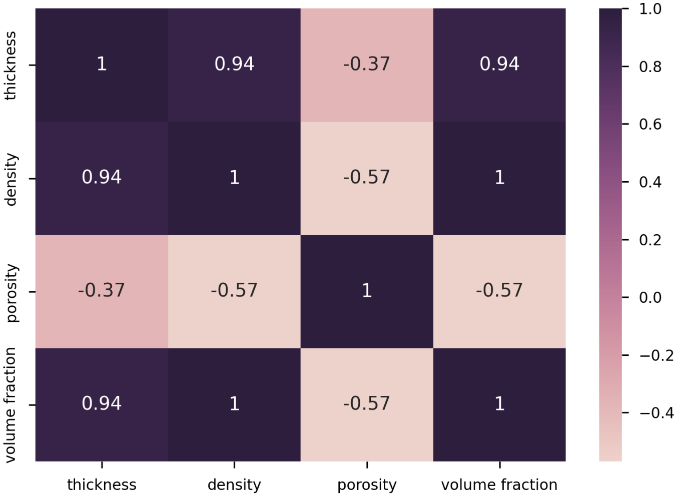


Figure 4.4: Heatmap showing the correlation between different target physical properties. The low variance and weak correlation of porosity with other properties explain the low R^2 score in predicting the porosity information of SEI configurations. Reproduced from [108] under the terms of the Creative Commons Attribution–NonCommercial–NoDerivatives (CC BY-NC-ND) license.

As presented in previous sections, the prVAE organizes its latent space according to gradients of physical properties. When extended with conditional inputs, this organization becomes sensitive to both structural and kinetic parameters. In this work, the conditional prVAE is trained with thickness as the guiding property, enabling the latent dimensions to capture the distribution of encoded features in an ordered manner. PCA is used to evaluate the effectiveness of the learned latent space of conditional prVAE in describing the SEI configurations and property information. Figure 4.5 shows that the first principal component accounts for the major variations related to SEI thickness, while the second principal component and third principal component, as shown in Figure 4.6 and Figure 4.7 remain nearly constant, despite changes in thickness.

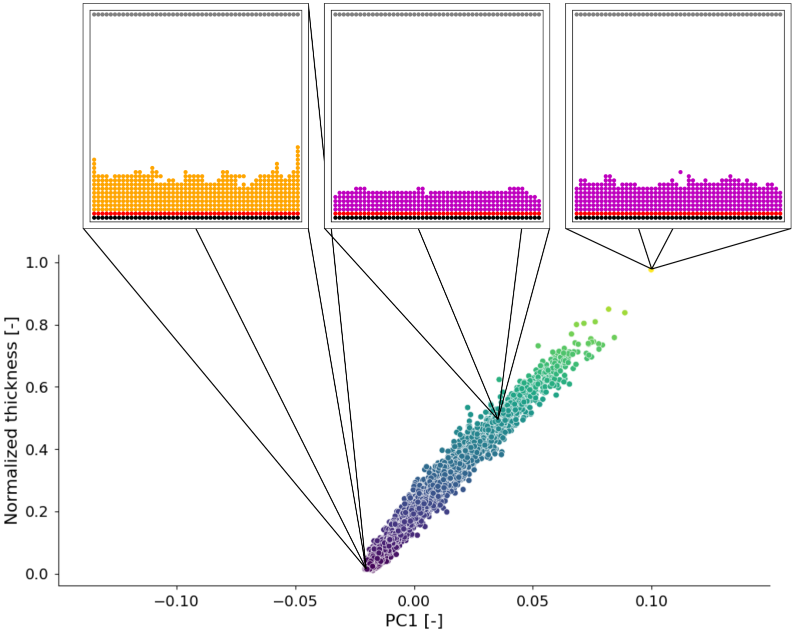


Figure 4.5: Latent space exploration along the first principal component: The first principal component holds the majority of the variational information of the property thickness and is conditioned by the reaction barrier set. The decoding of minimum (left), mean (middle), and maximum (right) values of the first principal component to the corresponding SEI configuration is shown here. Reproduced from [108] under the terms of the Creative Commons Attribution–NonCommercial–NoDerivatives (CC BY-NC-ND) license.

The increase in purple and red regions in the SEI configuration reflects the expanding extent of organic and inorganic layers, respectively, while the orange areas indicate intermediate reaction products that contribute to organic SEI formation. To explore how property information and reaction barriers influence the learned representation, the latent variables and reaction barrier values are interpolated within their minimum and maximum ranges.

Walking the latent space along these interpolation paths generates SEI configurations corresponding to different reaction barrier conditions. Figure 4.8b

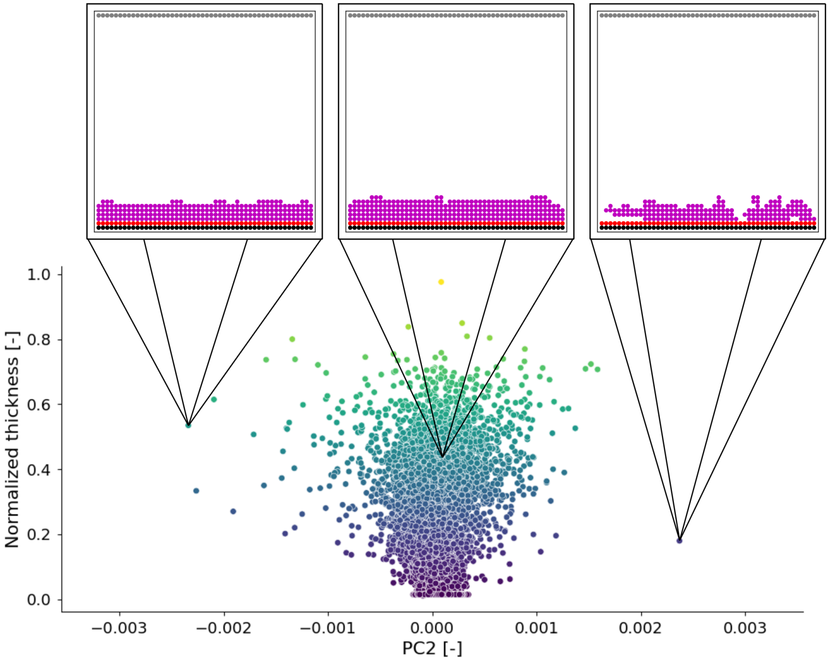


Figure 4.6: Latent space exploration along the second principal component: The second principal component does not carry any variational information about the thickness. The explainability of this principal component in describing the SEI configuration is insignificant compared to that of the first principal component. The decoding of minimum (left), mean (middle), and maximum (right) values of the second principal component to the corresponding SEI configuration is shown here. Reproduced from [108] under the terms of the Creative Commons Attribution–NonCommercial–NoDerivatives (CC BY-NC-ND) license.

demonstrates how, by varying either the reaction barrier input or the latent variables while keeping the other constant, the prVAE produces SEI configurations with controlled thickness and composition. Conditioning the decoder with reaction barrier inputs thus enables the generation of SEI configurations that align with specified reaction conditions, supporting the targeted design of interphases with tunable physical properties.

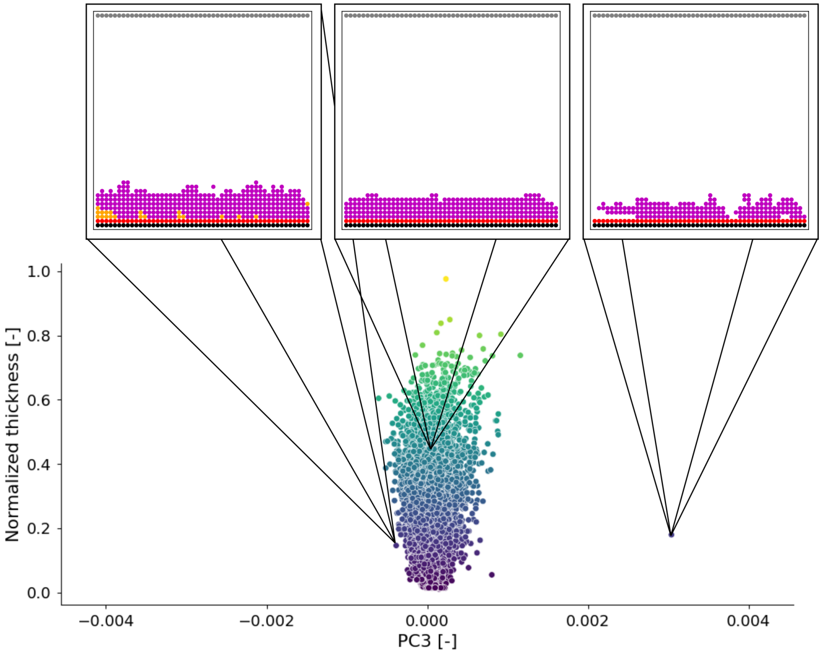
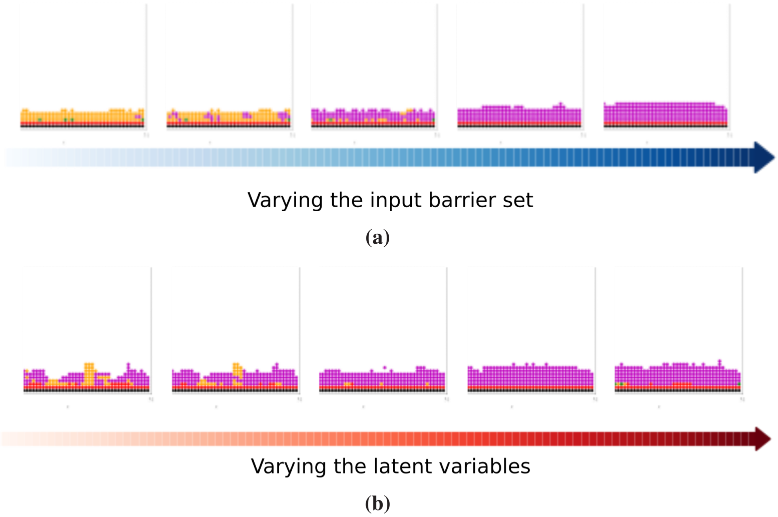


Figure 4.7: Latent space exploration along the third principal component: The third principal component also shows a similar trend to the second principal component. The decoding of minimum (left), mean (middle), and maximum (right) values of the third principal component to the corresponding SEI configuration is shown here. Reproduced from [108] under the terms of the Creative Commons Attribution–NonCommercial–NoDerivatives (CC BY-NC-ND) license.

4.1.6 Discussion

This study demonstrates characterization of structure-property relationship of SEI configurations in LIBs using information-rich latent space learning. The proposed prVAE model, captures essential data-driven features of SEI configurations required for efficient characterizations. The integration of a property regressor into the VAE architecture allows the prVAE model to encode a continuous, information-rich latent space. The property regressor also shows high accuracy



(a)

(b)

Figure 4.8: Latent space exploration of the SEI configuration: a) Guided SEI generation from the sampled mean of the latent variables while varying the conditional input barrier; b) SEI generation from the sampled mean of the input barrier while varying the latent variables. Reproduced from [108] under the terms of the Creative Commons Attribution–NonCommercial–NoDerivatives (CC BY-NC-ND) license.

in predicting the physical properties. The learned, information-rich latent space consists of data-driven properties organized by physical properties, enabling systematic exploration and optimization of SEI configurations. The prVAE model demonstrates exceptional performance in predicting key physical properties of SEI structures, achieving R^2 scores exceeding 0.98 for volume fraction, thickness, and density, while porosity yields R^2 of 0.8334. The high correlation between predicted and target properties indicates that the latent space effectively encodes the essential physical information governing SEI stability. The continuous information-rich latent space enables smoother interpolation between known SEI configurations, allowing exploration of intermediate SEI configurations that are not present in the original dataset. Conditioning the decoder input with a reaction barrier further extends the generative capability of prVAE. The conditioning mechanism allows the prVAE to synthesize SEI configurations corresponding to specific reaction environments and specific properties. The prVAE framework can be adapted for a range of material design strategies, including exploration, interpolation, and optimization. The continuous latent space obtained serves as a valuable input to optimization models, enabling the identification of SEI configurations and property combinations that enhance battery performance and safety.

4.2 Virtual screening and characterization of battery materials using latent diffusion model

Material characterization is an important aspect of the materials development cycle, providing a means to probe and measure structural, chemical, and physical properties. Through characterization techniques, insights into the relationship between structural and functional properties can be obtained. Computational material screening serves as an early-stage ranking of materials against defined performance metrics. The integration of characterization and screening processes in the material development cycle accelerates the identification of materials that meet specific functional requirements of the application while reducing time and cost. [180, 181]

Comprehensive characterization across multiple length scales is vital for understanding how atomic structure, crystallographic domains, and chemical composition govern the electrochemical performance of battery electrodes. Selecting suitable characterization techniques to capture these features is complex, as each method offers distinct strengths and limitations [87]. Atomic force microscopy (AFM) is one of the multifunctional techniques used to study and characterize batteries with nanometric resolution [182, 183]. AFM can be operated in multiple modes to image the topographic evolution of electrodes during charge-discharge. AFM can also be used to study the local electrochemical activity, current distributions, surface potential, ion transfer, and other surface properties of the electrode surface with high resolution [184, 185]. The current AFM machines employ an automated tip preparation method that uses ML techniques to provide the operator with information on the required tip functionalization and tip quality, thereby reducing the time spent on tip preparation. As a result, the automated tip functionalization enables the investigation of thousands of samples daily. But AFM experiments produce unlabeled data, making interpretation tedious and challenging even for experts [186–191].

Interpreting the contrast patterns observed in high-resolution atomic force microscopy images would require a detailed understanding of the various factors that give rise to these variations [192]. The contrast features captured in AFM data are essential for understanding the distinct nature of image contrasts and for discerning intermolecular interactions in AFM images. However, interpreting experimental AFM images, particularly for molecular systems, is challenging due to the complexity of structural corrugations, chemical heterogeneity, and instrumental parameters [193]. As a result, analyzing AFM images is often time-consuming and dependent on expert interpretation, underscoring the need for a more efficient, systematic approach.

The Probe-Particle AFM (PPAFM) developed by Hapala et al. [194–196] is an open-source software that simulates high-resolution AFM and various scanning probe microscopy techniques with sub-molecular resolution, encompassing all essential elements of AFM imaging. PPAFM uses a simple mechanical model to simulate the interaction between the probe and the sample. PPAFM is an open-source software that simulates high-resolution AFM captured using a diverse range of modified tips across different tip-sample distances. The PPAFM model treats the tip apex as a flexible particle, typically functionalized with a carbon monoxide (CO) molecule, and also with other molecules such as Xe, Cl^-, and H_2O to capture tip-sample interactions. The functionalized tip deflects in response to local force fields encompassing Pauli repulsion, Van der Waals attraction, and electrostatic interactions, which capture the mechanisms governing contrast formation while maintaining computational efficiency. High-resolution AFM distinguishes atoms within a molecule by inferring the subtle variation in tip-sample interactions [197, 198]. While 2D image features, such as sharp vertices, are easily identifiable visually, extracting three-dimensional information to differentiate atoms in a similar environment remains a significant challenge [199].

Advances in deep learning have enabled data-driven strategies capable of addressing the challenges of interpreting complex high-dimensional data. Deep learning models enable the automatic extraction of important features from experimental and simulation data, supporting applications such as phase segmentation, the synthesis of artificial datasets, and material optimization [62, 110, 200, 63, 111]. Machine learning methods have enabled the prediction of molecular structures and chemical compositions by analyzing 3D stacks of constant-height AFM images. This progress opens new possibilities for battery material research to predict the material behavior at the molecular level. The previous data-driven approaches in AFM analysis have focused on deterministic convolutional neural networks to map force maps to individual material descriptors. These approaches have demonstrated effectiveness in reconstructing particular features and facilitating nanoscale characterization. However, they are limited in their capacity to perform generative modeling tasks, such as sampling, interpolation, and generating novel material configurations. [201–203]

The functional properties of materials play a significant role in screening and designing battery materials tailored to specific applications. The average voltage of a material plays a crucial role in defining its operating voltage, which can enhance energy production and inspire new system designs. Gravimetric and volumetric capacities evaluate the total charge a material can store and the energy it holds per unit weight and per unit volume. Furthermore, volume change refers to the expansion and contraction of electrode materials during charge and discharge cycles, which is crucial for achieving long-term stability and high performance. Finding the right balance in these properties enhances battery performance, increases energy storage, and reduces size for various applications. [204] To develop battery materials that improve performance and safety, researchers often conduct numerous trial-and-error experiments based on characterization and screening. ML-assisted methods can accelerate the effective translation of materials informatics from characterization techniques to the inverse design of battery materials tailored to the functional properties required for a specific application.

This study proposes the BattGen model to virtually characterize and screen battery materials based on their functional properties, such as average voltage, volume change, gravimetric and volumetric capacity, and gravimetric and volumetric energy. The proposed model implements data on battery materials from the Materials Project database [95], which provides information on electrode compositions and their associated functional properties for various working ions [205–207]. For this study, the chemical formulas of approximately 3,154 electrode materials at the end of their discharge cycles are used to generate inputs for virtual atomic force microscopy (AFM) simulations using the PPAFM model, thereby constructing a comprehensive material database. In this proof-of-concept, virtual 3D AFM image stacks serve as input data, leveraging the compatibility of image-like data with convolutional neural network-based latent diffusion models. This process involves converting 3D AFM stacks into 2D image descriptors that capture the essential features of the battery electrodes under investigation. The resulting lower-dimensional latent space, which integrates both the latent features of the 3D AFM data and corresponding battery functional properties, enables efficient screening and identification of target materials. The developed framework merges atomic-level structural information with compositional data to support early-stage virtual screening of battery candidates. While the implemented approach does not encompass all experimental and device-level considerations, it focuses on critical material characteristics to streamline the identification of promising options for subsequent experimental validation.

4.2.1 Data generation and preprocessing

The source data for this study are obtained from the Materials Project (MP) database, which provides programmatic and web-based access to a wide range of material properties and computational results. The Battery Explorer module of MP is used to retrieve battery-specific information, including electrode compositions and their corresponding functional properties such as average voltage, gravimetric and volumetric capacities, energy density, and volume change during charge and discharge cycles. The battery materials extracted from the MP database are defined based on the following battery systems: Lithium (Li), Magnesium (Mg), Zinc (Zn), Calcium (Ca), Potassium (K), Sodium (Na), Aluminum (Al), Yttrium (Y), Rubidium (Rb), and Caesium (Cs). The extracted materials are filtered to remove duplicate entries in the battery composition, yielding 3,167 unique battery material configurations.

For each electrode material, the atomic coordinates of the discharge-state structure are extracted from the database and used as input for virtual AFM simulations generated through the PPAFM model. Simulations are performed at multiple tip–sample separations to produce three-dimensional AFM stacks that capture variations in force response with respect to tip height. Each 3D stack is further processed into a set of 2D image-based descriptors that capture the material's structural and chemical characteristics. For the AFM simulation, the maximum scan dimension is set to 128×128, and each augmented molecule is scanned for ten different sample distances. The auxiliary map function of the PPAFM model allows the generation of corresponding material descriptors, such as height maps, Van-der-Waals sphere, atomic discs, and multimap sphere elements (type map). [201–203]. To increase representational diversity, each atomic structure is rotated through a series of quasi-random orientations generated using Fibonacci spiral sampling in tangent sphere space. The most informative configurations are then selected based on entropy-driven variation in depth profiles to maintain structural diversity across the dataset. Each AFM image is evaluated using Minimum distance to edge $= \min(x, \text{width} - 1 - x, y, \text{height} - 1 - y)$ to confirm that the entire molecular structure lies within the simulated scan window. Here, (x, y) denotes the coordinates of the pixel with maximum intensity, while the image dimensions are 128 pixels each. AFM images with a minimum distance to the edge equal to zero are considered invalid, indicating that the main feature lies outside the defined scan dimensions used to simulate the entire dataset. During the AFM simulations, 13 material configurations, along with the 390 corresponding rotations that consistently produced invalid AFM images, are removed to prevent the inclusion of data lacking learnable information for training.

The resulting dataset contains $3,154$ distinct battery electrode compositions and approximately $94,640$ valid AFM image samples. The force-map data are normalized to a range of $[-1,1]$, while functional properties are scaled to $[0,1]$ to improve numerical stability during model training. Each atomic configuration is also encoded into a fixed-length material composition vector aligned with the periodic table, where each index records the atomic count of a corresponding element up to atomic number 112. This representation enables the latent diffusion model to associate learned image features with the underlying chemical composition and functional properties. The dataset is divided into training, validation, and test sets in a $70:15:15$ ratio, ensuring that identical material compositions do not appear across subsets. Sample weights are applied inversely to class frequency to reduce bias toward more common working ions. The data generation, preprocessing, and storage processes are documented and managed using the Kadi4Mat infrastructure [129, 130]. This study uses CIDS and KadiAI [134] to set up and run ML workflows. Figure 4.9 illustrates the data generation pipeline of the study.

4.2.2 BattGen architecture and training

The proposed BattGen model, based on LDM, enables characterization and screening of battery materials conditioned on functional properties. The BattGen model consists of three primary components: a VAE with a patch discriminator as the translation model, a conditional LDM as the screening model, and an atom regressor as the material composition predictor. Figure 4.10 shows the schematic illustration of the BattGen model.

The translation model automates the characterization of material structure in AFM images into meaningful material descriptors. The translation model is a VAE with a patch discriminator that integrates an image-conditioned patch-based adversarial objective from the Pix2Pix GAN. The image-conditioned patch-based

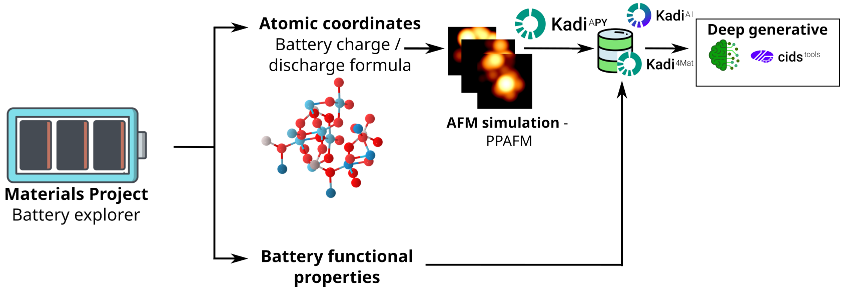


Figure 4.9: Training data generation pipeline. The source data for this study is obtained from the Battery Explorer feature of the Materials Project database. Atomic coordinates of the battery formulas and their functional properties are extracted using the Materials Project Application Programming Interface (API). The extracted data is organized and managed through the Kadi4Mat programming interface. KadiAI and CIDS enable the machine learning process in this study. Reproduced from [109] under the terms of the Creative Commons Attribution (CC BY) license.

adversarial objective allows a supervised image-image translation task. The encoder component of the VAE receives a three-dimensional AFM stack as input and learns a meaningful latent representation by extracting essential features. The decoder component of VAE decodes the compressed latent representation into material descriptors. The implemented patch discriminator acts as a critic by evaluating the local spatial patch between the target and generated material descriptors. The translation model consists mainly of convolution layers along with residual and attention layers. The residual block of layers allows the model to learn deep hierarchical features while preserving essential information between layers. The attention layers help the model to focus on the global features of the input.

The learned compressed latent representation from the translation model is used to train the screening model. The screening model is a conditional LDM that enables targeted screening of battery materials based on functional properties. As discussed in the Section 2.5.3, the DM can be seen as a composition of forward and reverse diffusion processes. The forward diffusion process gradually injects Gaussian noise via a series of variance-scheduled noise steps. The role of the implemented denoising Unet here is to learn the reverse diffusion to predict the noise at each step of the forward diffusion process. In LDM, the diffusion process is applied to the compressed lower-dimensional representation of data. In this study, the conditional LDM uses a denoising U-Net architecture to learn a controlled reverse diffusion process on the latent representation conditioned on the given conditions.

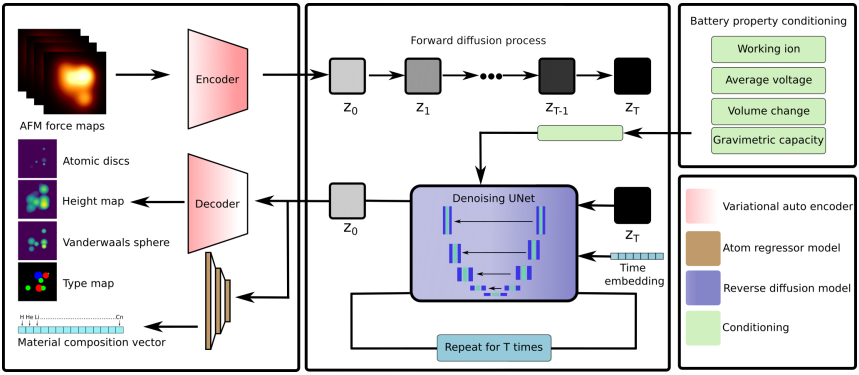


Figure 4.10: Overview of the BattGen model. The first frame shows the encoder and decoder of the translational model, which maps AFM force maps to corresponding battery material descriptors. The middle frame highlights the diffusion model in the learned latent space, guided by battery functional properties for the virtual screening of target battery materials. Reproduced from [109] under the terms of the Creative Commons Attribution (CC BY) license.

The implemented U-Net is a symmetric architecture with convolution, attention, and residual block layers. The U-Net consists of an encoding-decoding architecture with bridge networks. The overall architecture resembles a letter U, hence the name U-Net. The layers in the encoder (downsampling) part of the U-Net architecture are skip connected and concatenated with layers in the decoder (upsampling) part. These skip connections enable the U-Net to leverage fine-grained details learned in the encoder to construct an image in the decoder. The conditioning of the U-Net on battery functional properties is achieved through the symmetric concatenation of these properties at the start of each down, middle, and up blocks in the U-Net. This symmetric conditioning along the encoding, bridging, and decoding paths of the U-net moves the generated compressed latent representation towards conditioned functional properties. The Unet is trained using the MSE loss function. For the forward diffusion process, the number of time steps T is set to 1000, as used in the original implementation by Ho et al. [69]. The number of diffusion timesteps of 1000 is found empirically sufficient to disrupt the original data structure, allowing gradual noise injection necessary for stable training of the reverse diffusion process. The sequence $\beta_1, \ldots, \beta_T$ determines the variance of the Gaussian noise added at each step of the forward process, increasing linearly from 10^{-4} to 0.02, which are small relative to data normalized to the range $[-1, 1]$.

In addition to translating and screening battery materials based on material descriptors, an atom regressor model is used to decode the molecular composition vector from the learned latent space. The atom regressor model constitutes a balanced combination of convolution and dense layers. The last layer of the atom regressor model uses softplus activation to avoid negative values in the predicted molecule composition vector. The molecular composition vector consists of a vector length up to atomic number 112. Training the atom regressor model is optimized using the MSE loss function.

4.2.3 Automated translation of characterization data to material descriptors

The trained VAE component of the BattGen model is used to translate a 3D stack of simulated AFM images with fore maps at varying tip-sample distances into corresponding material descriptors, such as a height map, Van der Waals sphere, atomic disc, and type map. The patch discriminator trained adversarially with the VAE consists of two convolutional branches to handle grayscale descriptors, such as height maps, van der Waals spheres, and atomic discs, and RGB-based descriptors, such as type maps, which encode red, green, and blue colors to represent different atoms based on van der Waals radius. In the type map descriptor, the red, green, and blue channels are used to classify the different types of atomic species relevant to the study. The blue channel signifies the working ion species, the green channel represents gas phase elements, and the red channel corresponds to all other elements up to atomic number 112 in the periodic table.

The visual representations of the studied material descriptors are presented in Figure 4.10. Modifying the patch discriminator's architecture to process patches from both grayscale and RGB images improved the VAE's reconstruction performance. This adjustment reduced color prediction errors, particularly in type map generation, and enabled better distinction of local features in force maps through the receptive fields established at each convolutional layer of the discriminator. As shown in Figure 4.11, the model accurately translates image descriptors from test samples corresponding to different battery ion types. The translation framework achieves an FID score of 2.301 when converting 3D AFM scans into grayscale descriptors, indicating high reconstruction fidelity. For type map translation, an FID score of 26.845 is achieved, indicating moderate fidelity with satisfactory visual quality. Overall, the low FID values demonstrate that the model's generated outputs closely approximate the real data distribution.

4.2.4 Application targeted screening of battery materials

The virtual screening of battery materials is achieved by conditioning the learned latent space of the translational model on functional properties. The conditioned functional properties include average voltage, gravimetric and volumetric capacity, gravimetric and volumetric energy density, volume change, and working ion. The functional properties of batteries as conditionals, along with the time embedding and learned latent representation from the translation model, are provided as inputs to the U-Net component of the LDM. The time embedding guides the model in estimating the level of noise added at each step of the forward diffusion process. Incorporating these functional properties as conditioning inputs enables the application targeted screening of electrode materials, considering potential working ions such as Lithium (Li), Magnesium (Mg), Zinc (Zn), Calcium (Ca), Potassium (K), Sodium (Na), Aluminum (Al), Yttrium (Y), Rubidium (Rb), and

Caesium (Cs). Demonstrating the implemented virtual screening tool involves providing pure random Gaussian noise, along with working ion and battery properties, as inputs to the trained U-Net model. The model then screens various material configurations based on the specified battery properties. In this case, the initial condition of the Gaussian noise is crucial for determining the types of material descriptors that are generated. Figure 4.12 shows the predictions of the LDM under certain seeding conditions and rotations on test samples outside the training dataset for different working-ion and battery functional properties. To assess the impact of each conditioning variable on the generated outputs, a sensitivity analysis is performed using the L2 distance metric. This approach measures variations in the output by perturbing each conditioning variable by ± 0.05 while maintaining the remaining variables constant. As all continuous variables are min–max normalized to the $[0, 1]$ range, the value ± 0.05 corresponds to a 5% relative change and allows consistent comparison across given conditionals. Applying both positive and negative perturbations allows the identification of potential asymmetries in the model's response. For categorical variables, such as the working ion, a discrete substitution strategy is employed, in which the original ion is replaced with each valid alternative, and the resulting L2 distances are averaged to evaluate their overall influence. Figure 4.13 shows that the gravimetric and volumetric capacities demonstrate the strongest influence, followed by gravimetric energy density and average voltage. The volumetric energy density, volume change, and working ion contribute less, indicating reduced influence on generation. These results highlight the dominant role of gravimetric and volumetric properties in the screening of battery materials.

4.2.5 Composition vector prediction

The atom regressor of the BattGen model is used to predict chemical information from scanned AFM images. By leveraging the latent space learned, the regressor predicts the composition vector, revealing chemical information associated with each AFM image. The model demonstrates strong predictive capability, estimating material composition vectors for atomic numbers up to 112 and effectively mapping

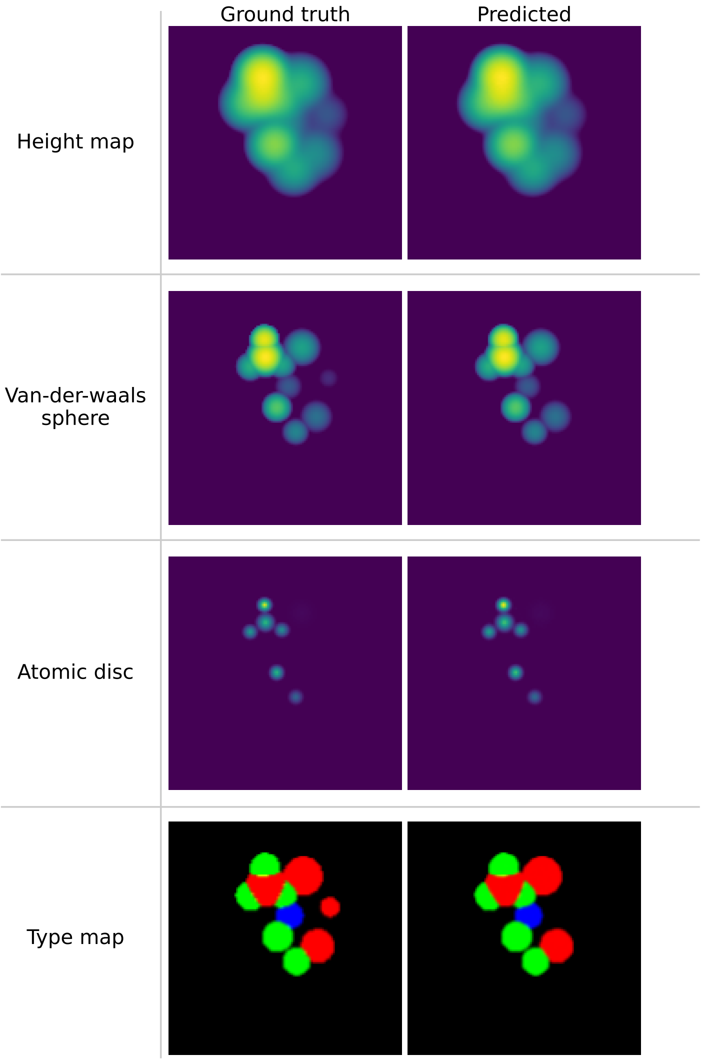


Figure 4.11: Comparison plot between image-based material descriptor ground truth and corresponding predictions using the implemented translation model. Reproduced from [109] under the terms of the Creative Commons Attribution (CC BY) license.

image features to their underlying elemental distributions. As shown in Figure 4.14, the predicted atomic counts align closely with the true values for most working ions, with minor deviations observed for Aluminum (Al), Rubidium (Rb), Yttrium (Y), and Cesium (Cs). Quantitative evaluation results presented in Table 4.1 and visualized in Figure 4.15 indicate high predictive accuracy across several working ions. The model demonstrates high predictive accuracy, achieving a coefficient of determination (R2) score of 0.9996 for Magnesium (Mg), 0.9989 for Lithium (Li) and Zinc (Zn), and 0.9931 for Calcium (Ca). However, the model performance for Cesium (Cs) and Rubidium (Rb), with R2 scores of 0.5087 and 0.7284, respectively, indicates challenges in capturing their underlying patterns due to their lower frequency in the training samples. For Aluminum (Al) and Potassium (K), the model exhibits intermediate performance, with R2 scores of 0.8097 and 0.849, respectively. The model generalizes well for most elements, but certain atoms will need additional samples for improved accuracy.

Table 4.1: Evaluation metrics of the atom regression model

Atom		Metrics	
	MAE	MSE	R2 score
Li	0.021	0.0236	0.9989
Na	0.0181	0.1593	0.9707
Mg	0.0138	0.03	0.9996
Al	0.0475	0.2687	0.8097
K	0.0211	0.1054	0.849
Ca	0.0089	0.011	0.9931
Zn	0.0072	0.0073	0.9989
Rb	0.0169	0.0738	0.7284
Y	0.0158	0.0411	0.9015
Cs	0.0115	0.0705	0.5087

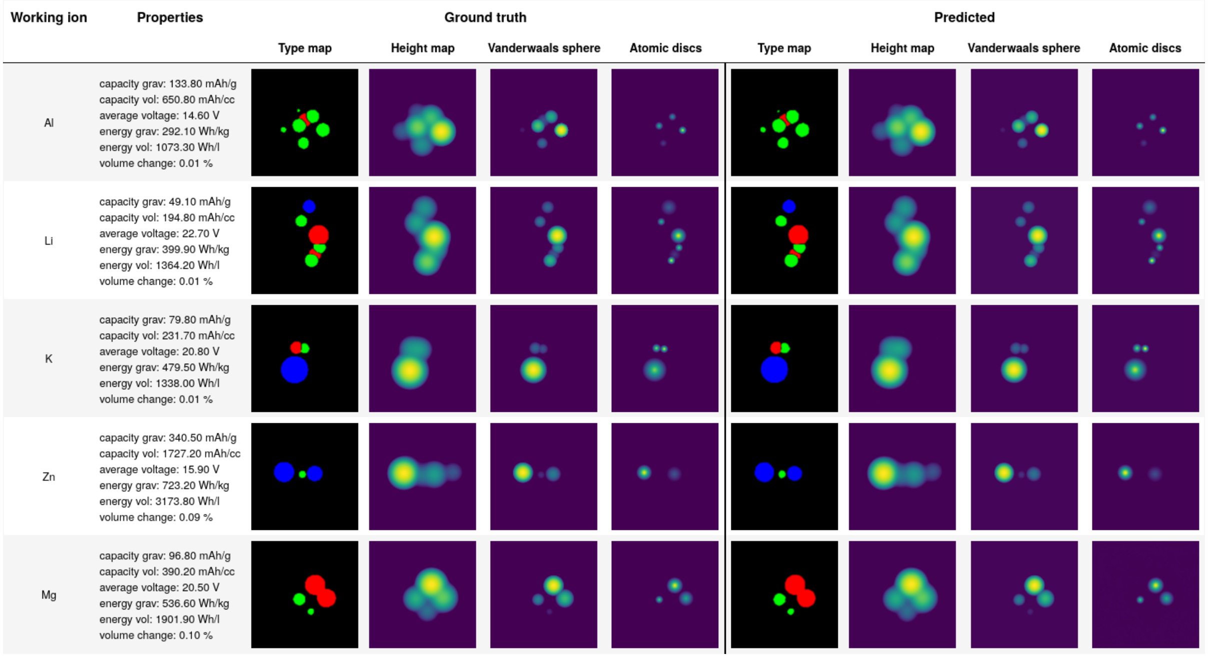


Figure 4.12: Demonstration of the implemented virtual screening tool to predict the image descriptors of the materials from the test dataset. Here, the latent space, configured with battery contextual properties such as working ion, gravimetric capacity, volumetric capacity, average voltage, gravimetric energy, volumetric energy, and volume change, is used to input the U-net part of LDM to predict the material image descriptors. The predicted image descriptors match the corresponding ground truths exactly. Reproduced from [109] under the terms of the Creative Commons Attribution (CC BY) license.

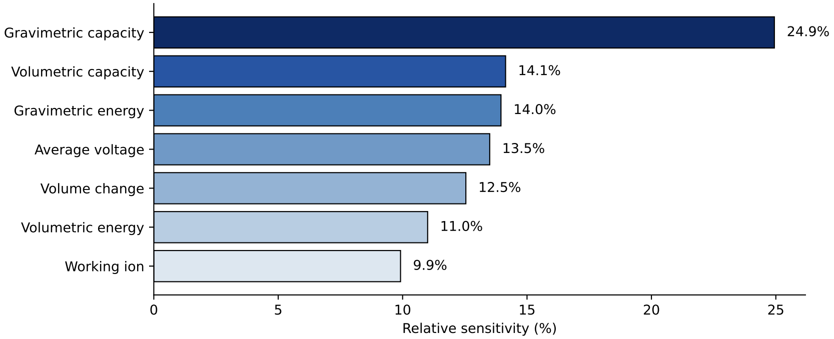


Figure 4.13: Relative sensitivity of conditioning variables in the conditional latent diffusion model. The influence of each input condition on the generated outputs is quantified using a normalized L2 distance metric. The sensitivity analysis indicates that gravimetric capacity and volumetric capacity have a significant impact on the model's performance in screening battery materials. Reproduced from [109] under the Creative Commons CC-BY license.

4.2.6 Discussion

The proposed BattGen model demonstrates automated structure characterization of AFM scans and application-targeted screening of battery materials. By transforming high-resolution AFM images into meaningful latent representations, the model enables a direct connection between structural features and the underlying chemical composition. The integration of the LDM for virtual screening allows the generation of material descriptors conditioned on specific battery functional properties, providing a systematic approach to identify promising electrode materials for defined applications.

The translation component of BattGen demonstrates strong performance in reconstructing material descriptors and predicting elemental composition for most working ions. The atom regressor effectively predicts atomic distributions, with high accuracy for commonly represented elements such as Magnesium, Lithium, Zinc, and Calcium. Lower predictive performance for elements such as Cesium and Rubidium reflects the limited representation of these ions in the training dataset, indicating the need for additional data to improve generalization. Intermediate accuracy for Aluminum and Potassium suggests that moderate dataset representation is sufficient for reliable prediction, but further improvement could be achieved through targeted data augmentation or additional experimental samples. The translation component performs well on simulated AFM scans of simplified cathode compositions; the extension to experimental AFM data requires adaptation. Experimental scans often exhibit complex surface features, mixed compositions, imaging noise, and variable conditions not captured in simulations. Addressing these challenges through preprocessing, augmentation, and transfer learning on a limited set of labeled experimental images is expected to enhance the model's generalization and applicability to real-world material discovery.

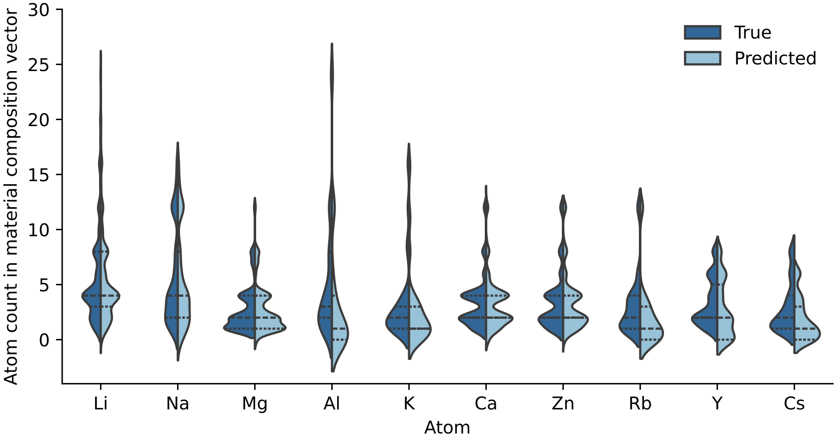


Figure 4.14: Graphical visualization of the performance of the atom regression model in the prediction of atoms based on the working ion of the battery in the material composition vector. The distribution of true and predicted atom counts aligns well, with notable deviations observed for aluminum (Al), rubidium (Rb), yttrium (Y), and cesium (Cs). The width of each violin represents the kernel density estimation of the distribution. The thick dashed line in each violin represents the median, while the thin dashed lines denote the 25th and 75th percentiles. Reproduced from [109] under the terms of the Creative Commons Attribution (CC BY) license.

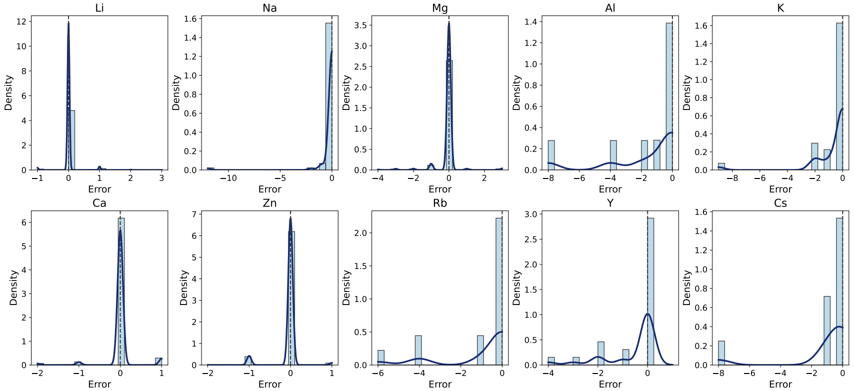


Figure 4.15: Prediction error distributions for individual atomic species in the atom regression model. Each subplot presents a histogram and kernel density estimate (KDE) of the errors, with a dashed line indicating zero error. While most atoms are predicted accurately, larger deviations are observed for aluminum (Al), rubidium (Rb), yttrium (Y), and cesium (Cs). Reproduced from [109] under the terms of the Creative Commons Attribution (CC BY) license.

The screening component of BattGen demonstrates the selection of battery materials based on functional metrics. The implemented screening approach prioritizes materials that are promising for a specific use case, thereby narrowing the candidate list for further computational or experimental validation. The implemented application-driven screening approach enables application-targeted exploration of the materials space, especially where extensive DFT datasets are not available. Future work will focus on incorporating DFT-based predictions of battery properties [208–211] to guide the second-stage screening of optimized battery materials for a specific use case.

5 Conclusion

5.1 Summary and Concluding Remarks

This thesis presents a comprehensive generative deep learning framework for the advanced design of battery materials. Based on the contributions [108, 109, 135, 15], the theory, methodology, and demonstration of results of generative deep learning frameworks in characterization, screening, and guided generation of battery materials are discussed. To solve the complex physicochemical evolution of battery materials across different scales and time, the generative model, in tandem with the discriminative model, is used as a methodology framework of the thesis to accelerate the different battery material design stages. The integration of a generative model and a complementary discriminative model enables us to access the expansion and contraction design pathways of battery materials.

VAE is the primary foundation of the implemented generative framework. The VAE architecture enables multiscale material design by extracting essential features from high-dimensional material data and encoding them into a compact latent representation. The learned latent representation captures the underlying distribution of data-driven features and enables tractable exploration of material design space. Integrating discriminative models and conditional information into latent space learning enhances the latent space's expressiveness by infusing key property information that influences material design and performance. The results confirm that combined generative and discriminative modeling of material data yields latent representations that are both interpretable and predictive. The information-rich latent space enables interpolation, optimization, and exploration of material design space.

For SEI analysis, the implemented prVAE model accurately captures key structural and physical properties of SEI configurations by integrating generative and discriminative modeling. The jointly trained property regressor with VAE organizes the learned latent space according to observable properties of SEI configurations. The resulting information-rich latent space shows that the learned representation captures the essential multiscale physics governing simulated SEI configurations and allows controlled exploration and generation of SEI configurations with target physical properties. The analysis reveals the broader potential of latent generative models as a foundation for inverse material design. This stage of the framework relies on a single decoding step to achieve target material configurations. The single decoding step in the prVAE model limits its ability to generate diverse material configurations for given target properties.

Integrating LDM in the framework extends beyond one-shot generation by using a series of denoising steps. The number of denoising steps in DM allows the model to gradually refine the latent representation toward a desirable configuration, yielding multiple decoding trajectories rather than a single decoding step. The multi-step decoding process enhances the diversity and fidelity of generated material configurations. Integrating patch discriminators with LDM further improves the generated configurations, preserving global coherence and fine structural details. The generation process in LDM can be guided by gradually conditioning the denoising process. This allows LDM to define different application modes, such as translation and screening. The translation application uses a VAE and a patch discriminator to convert high-dimensional characterization data into meaningful material descriptors and composition vectors. The screening process uses a conditional denoising U-Net architecture for targeted, diverse screening of battery materials based on specified functional properties.

The developed generative-discriminative framework is a step toward data-driven, property-aware materials design. The flexibility of the framework supports a wide range of applications, from screening to inverse design, by defining an information-rich latent space. The latent-powered generative knowledge, combined with discriminative meaningful assessment, provides a cohesive approach to accelerate data-driven design of advanced battery materials.

Despite these advancements, seamlessly integrating data-driven modeling into battery research poses four critical challenges for the development of next-generation battery materials. First, the scarcity of high-quality, diverse, large-scale datasets tailored to battery materials. Experimental battery data are often sparse, heterogeneous, and expensive to obtain, leading to overfitting, limited generalization, and reduced robustness in generative models. The complexity of battery systems, with intricate electrochemical processes, requires comprehensive datasets that capture multiscale interactions and dynamic behaviors.

Second, generative models often function as black boxes, making it difficult to interpret their decision-making processes and the physical and chemical principles behind material generation. This lack of interpretability is a barrier for experimentalists and engineers seeking to use these models effectively.

Third, ensuring that generated materials are novel, synthesizable, and stable under real-world conditions remains a significant challenge. Bridging the gap between computational predictions and experimental validation is essential for translating generative model outputs into practical battery materials. While models may propose promising candidates, their utility depends on efficient and accurate experimental synthesis and characterization. The development of closed-loop autonomous experimentation platforms that integrate generative models, robotic synthesis systems, and advanced characterization techniques is still in its early stages.

Finally, training and deploying generative models, especially those with large latent spaces and complex architectures, requires substantial computational resources and energy, which can be prohibitive for researchers without access to high-performance computing infrastructure. As a result, battery performance is influenced by interactions across multiple length scales involving coupled physical phenomena, making it highly complex to integrate diverse data types and physical constraints into a unified generative framework.

5.2 Outlook

The implemented framework can be extended to act as a surrogate model to understand the range of simulation methods for battery materials. Integrating automated AI model generation based on input-output pairs enables access to multiscale design pathways for battery materials and delivers faster insights without requiring simulation expertise. This will help battery researchers to make the computational design process faster, and can reduce the cost of setting up the simulation environment.

To support this, research should prioritize developing techniques for generating synthetic data, using active learning, and enhancing transfer learning to address data scarcity. In addition, establishing robust data standardization and sharing practices within the research community will foster collaboration and enhance data utility. Building on these foundations, integrating core electrochemical principles into generative model design and operation can improve interpretability, predictive accuracy, and the generation of realistic, stable battery materials. Furthermore, future research should focus on creating automated feedback loops between computational design and experimental validation to increase process accuracy and efficiency. The implementation of autonomous laboratories equipped with robotic synthesis and characterization systems guided by generative models will further streamline the research process. Collectively, these advancements will enable rapid iteration through cycles of material design, synthesis, and testing, thereby accelerating the discovery of new battery materials.

The development of explainable artificial intelligence techniques specifically designed for generative models improves understanding of how materials are generated and the influence of material features on achieving desired properties. By elucidating the mechanisms underlying model predictions, researchers can gain critical insights into prediction accuracy. Advancing generative models in materials science necessitates processing and generating information across multiple modalities, such as chemical formulas, crystal structures, spectroscopic data, and performance curves, while simultaneously performing tasks including material generation, property prediction, and synthesis pathway recommendation. Implementing robust uncertainty quantification within generative models can guide experimental efforts by enabling researchers to prioritize materials based on the confidence in predicted properties, thereby improving the interpretability of model outputs.

Bibliography

[1] Palacin, M. R. (2021). Battery materials design essentials. *Accounts of Materials Research 2*(5), 319–326.

[2] Kittner, N., F. Lill, and D. M. Kammen (2017). Energy storage deployment and innovation for the clean energy transition. *Nature Energy 2*(9), 1–6.

[3] Manna, S., P. Paul, S. S. Manna, S. Das, and B. Pathak (2025). Utilizing machine learning to advance battery materials design: Challenges and prospects. *Chemistry of Materials 37*(5), 1759–1787.

[4] Bhowmik, A., I. E. Castelli, J. M. Garcia-Lastra, P. B. Jørgensen, O. Winther, and T. Vegge (2019). A perspective on inverse design of battery interphases using multi-scale modelling, experiments and generative deep learning. *Energy Storage Materials 21*, 446–456.

[5] Spotte-Smith, E. W. C., S. M. Blau, X. Xie, H. D. Patel, M. Wen, B. Wood, S. Dwaraknath, and K. A. Persson (2021). Quantum chemical calculations of lithium-ion battery electrolyte and interphase species. *Scientific Data 8*(1), 203.

[6] Yao, N., X. Chen, Z.-H. Fu, and Q. Zhang (2022). Applying classical, ab initio, and machine-learning molecular dynamics simulations to the liquid electrolyte for rechargeable batteries. *Chemical Reviews 122*(12), 10970–11021.

[7] Yoon, G., S. Moon, G. Ceder, and K. Kang (2018). Deposition and stripping behavior of lithium metal in electrochemical system: continuum mechanics study. *Chemistry of Materials 30*(19), 6769–6776.

[8] Miranda, D., C. Costa, and S. Lanceros-Mendez (2015). Lithium ion rechargeable batteries: State of the art and future needs of microscopic theoretical models and simulations. *Journal of Electroanalytical Chemistry 739*, 97–110.

[9] Peng, J., D. Schwalbe-Koda, K. Akkiraju, T. Xie, L. Giordano, Y. Yu, C. J. Eom, J. R. Lunger, D. J. Zheng, R. R. Rao, et al. (2022). Human-and machine-centred designs of molecules and materials for sustainability and decarbonization. *Nature Reviews Materials 7*(12), 991–1009.

[10] George, J. and G. Hautier (2021). Chemist versus machine: Traditional knowledge versus machine learning techniques. *Trends in chemistry 3*(2), 86–95.

[11] Walsh, A. (2015). The quest for new functionality. *Nature chemistry 7*(4), 274–275.

[12] Choi, R. Y., A. S. Coyner, J. Kalpathy-Cramer, M. F. Chiang, and J. P. Campbell (2020). Introduction to machine learning, neural networks, and deep learning. *Translational vision science & technology 9*(2), 14–14.

[13] Wang, A. Y.-T., R. J. Murdock, S. K. Kauwe, A. O. Oliynyk, A. Gurlo, J. Brgoch, K. A. Persson, and T. D. Sparks (2020). Machine learning for materials scientists: an introductory guide toward best practices. *Chemistry of Materials 32*(12), 4954–4965.

[14] Lombardo, T., M. Duquesnoy, H. El-Bouysidy, F. Årén, A. Gallo-Bueno, P. B. Jørgensen, A. Bhowmik, A. Demortière, E. Ayerbe, F. Alcaide, M. Reynaud, J. Carrasco, A. Grimaud, C. Zhang, T. Vegge, P. Johansson, and A. A. Franco (2021). Artificial intelligence applied to battery research: hype or reality? *Chemical reviews 122*(12), 10899–10969.

[15] Rajagopal, D., A. Cierpka, B. Nestler, and A. Koeppe (2025). Generative deep learning for advanced battery materials. *Batteries & Supercaps 0*, e202500494.

[16] Noh, J., J. Kim, H. S. Stein, B. Sanchez-Lengeling, J. M. Gregoire, A. Aspuru-Guzik, and Y. Jung (2019). Inverse design of solid-state materials via a continuous representation. *Matter 1*(5), 1370–1384.

[17] Nouira, A., N. Sokolovska, and J.-C. Crivello (2018). Crystalgan: learning to discover crystallographic structures with generative adversarial networks. *arXiv preprint arXiv:1810.11203 1*, 1–9.

[18] Zhao, Y., M. Al-Fahdi, M. Hu, E. M. Siriwardane, Y. Song, A. Nasiri, and J. Hu (2021). High-throughput discovery of novel cubic crystal materials using deep generative neural networks. *Advanced Science 8*(20), 2100566.

[19] Xie, T., X. Fu, O.-E. Ganea, R. Barzilay, and T. Jaakkola (2021). Crystal diffusion variational autoencoder for periodic material generation. *arXiv preprint arXiv:2110.06197 1*, 1–20.

[20] Yang, S., K. Cho, A. Merchant, P. Abbeel, D. Schuurmans, I. Mordatch, and E. D. Cubuk (2023). Scalable diffusion for materials generation. *arXiv preprint arXiv:2311.09235 1*, 1–18.

[21] Zeni, C., R. Pinsler, D. Zügner, A. Fowler, M. Horton, X. Fu, Z. Wang, A. Shysheya, J. Crabbé, S. Ueda, R. Sordillo, L. Sun, J. Smith, B. Nguyen, H. Schulz, S. Lewis, C.-W. Huang, Z. Lu, Y. Zhou, H. Yang, H. Hao, J. Li, C. Yang, W. Li, R. Tomioka, and T. Xie (2025). A generative model for inorganic materials design. *Nature 639*(8055), 624–632.

[22] Yoon, D., J. Lee, and S. Lee (2025). Deep learning-driven molecular generation and electrochemical property prediction for optimal electrolyte additive design. *Applied Sciences 15*(7), 3640.

[23] Khajeh, A., X. Lei, W. Ye, Z. Yang, L. Hung, D. Schweigert, and H.-K. Kwon (2025). A materials discovery framework based on conditional generative models applied to the design of polymer electrolytes. *Digital Discovery 4*(1), 11–20.

[24] Yang, Z., W. Ye, X. Lei, D. Schweigert, H.-K. Kwon, and A. Khajeh (2024). De novo design of polymer electrolytes using gpt-based and diffusion-based generative models. *npj Computational Materials 10*(1), 296.

[25] Gayon-Lombardo, A., L. Mosser, N. P. Brandon, and S. J. Cooper (2020). Pores for thought: generative adversarial networks for stochastic reconstruction of 3d multi-phase electrode microstructures with periodic boundaries. *npj Computational Materials 6*(1), 82.

[26] Kench, S. and S. J. Cooper (2021). Generating three-dimensional structures from a two-dimensional slice with generative adversarial network-based dimensionality expansion. *Nature Machine Intelligence 3*(4), 299–305.

[27] Dahari, A., S. Kench, I. Squires, and S. J. Cooper (2023). Fusion of complementary 2d and 3d mesostructural datasets using generative adversarial networks. *Advanced Energy Materials 13*(2), 2202407.

[28] Wang, Y., J. Wang, H. Zhang, and J. Song (2025). Bridging prediction and decision: Advances and challenges in data-driven optimization. *Nexus 2*, 100057.

[29] Habib, M. K., S. A. Ayankoso, and F. Nagata (2021). Data-driven modeling: concept, techniques, challenges and a case study. In *2021 IEEE international conference on mechatronics and automation (ICMA)*, pp. 1000–1007. IEEE.

[30] Mohri, M., A. Rostamizadeh, and A. Talwalkar (2018). *Foundations of machine learning*. MIT press.

[31] Hastie, T. (2009). The elements of statistical learning: data mining, inference, and prediction.

[32] Hestenes, M. R., E. Stiefel, et al. (1952). Methods of conjugate gradients for solving linear systems. *Journal of research of the National Bureau of Standards 49*(6), 409–436.

[33] Robbins, H. and S. Monro (1951). A stochastic approximation method. *The annals of mathematical statistics 22*, 400–407.

[34] Kingma, D. P. and J. Ba (2017). Adam: A method for stochastic optimization.

[35] Duchi, J., E. Hazan, and Y. Singer (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research 12*(7), 2121–2159.

[36] Tieleman, T. (2012). Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning 4*(2), 26.

[37] Hajji, M., B. Benhala, and I. Hamdi (2024). Survey of gradient descent variants and evaluation criteria. In *2024 4th International Conference on Innovative Research in Applied Science, Engineering and Technology (IRASET)*, pp. 01–07. IEEE.

[38] Ladha, L. and T. Deepa (2011). Feature selection methods and algorithms. *International journal on computer science and engineering 3*(5), 1787–1797.

[39] Rangarajan, L. et al. (2010). Bi-level dimensionality reduction methods using feature selection and feature extraction. *International Journal of Computer Applications 4*(2), 33–38.

[40] Howley, T., M. G. Madden, M.-L. O'Connell, and A. G. Ryder (2005). The effect of principal component analysis on machine learning accuracy with high dimensional spectral data. In *International Conference on Innovative Techniques and Applications of Artificial Intelligence*, pp. 209–222. Springer.

[41] Guyon, I. and A. Elisseeff (2003). An introduction to variable and feature selection. *Journal of machine learning research 3*(Mar), 1157–1182.

[42] Dargan, S., M. Kumar, M. R. Ayyagari, and G. Kumar (2020). A survey of deep learning and its applications: a new paradigm to machine learning. *Archives of computational methods in engineering 27*(4), 1071–1092.

[43] LeCun, Y., Y. Bengio, and G. Hinton (2015). Deep learning. *nature 521*(7553), 436–444.

[44] Bishop, C. M. (1994). Neural networks and their applications. *Review of scientific instruments 65*(6), 1803–1832.

[45] Yang, G. R. and X.-J. Wang (2020). Artificial neural networks for neuroscientists: a primer. *Neuron 107*(6), 1048–1070.

[46] Argatov, I. (2019). Artificial neural networks (anns) as a novel modeling technique in tribology. *Frontiers in Mechanical Engineering 5*, 30.

[47] Rasamoelina, A. D., F. Adjailia, and P. Sinčák (2020). A review of activation function for artificial neural network. In *2020 IEEE 18th world symposium on applied machine intelligence and informatics (SAMI)*, pp. 281–286. IEEE.

[48] Ramachandran, P., B. Zoph, and Q. V. Le (2017). Searching for activation functions.

[49] Foster, D. (2022). Generative deep learning. *O'Reilly Media, Inc. 1*(1), 1–450.

[50] Goodfellow, I., Y. Bengio, A. Courville, and Y. Bengio (2016). Deep learning. *MIT Press 1*(2), 1–775.

[51] van den Oord, A., N. Kalchbrenner, and K. Kavukcuoglu (2016). Pixel recurrent neural networks. *arXiv preprint arXiv:1601.06759 1*, 1–11.

[52] van den Oord, A., N. Kalchbrenner, O. Vinyals, L. Espeholt, A. Graves, and K. Kavukcuoglu (2016). Conditional image generation with pixelcnn decoders. *arXiv preprint arXiv:1606.05328 1*, 1–9.

[53] Xiong, J., G. Liu, L. Huang, C. Wu, T. Wu, Y. Mu, Y. Yao, H. Shen, Z. Wan, J. Huang, C. Tao, S. Yan, H. Yao, L. Kong, H. Yang, M. Zhang, G. Sapiro, J. Luo, P. Luo, and N. Wong (2025). Autoregressive models in vision: A survey.

[54] Bengio, Y., R. Ducharme, P. Vincent, and C. Jauvin (2003). A neural probabilistic language model. *Journal of machine learning research 3*(Feb), 1137–1155.

[55] Dalal, M., A. C. Li, and R. Taori (2019). Autoregressive models: What are they good for? *arXiv preprint arXiv:1910.07737 1*, 1–12.

[56] Bond-Taylor, S., A. Leach, Y. Long, and C. G. Willcocks (2021). Deep generative modelling: A comparative review of vaes, gans, normalizing flows, energy-based and autoregressive models. *IEEE transactions on pattern analysis and machine intelligence 44*(11), 7327–7347.

[57] Kingma, D. P. and M. Welling (2022). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114 1*, 1–14.

[58] Goodfellow, I., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio (2020). Generative adversarial networks. *Communications of the ACM 63*(11), 139–144.

[59] Kingma, D. P. and M. Welling (2019). An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning 12*(4), 307–392.

[60] Bank, D., N. Koenigstein, and R. Giryes (2023). *Autoencoders*, pp. 353–374. Cham: Springer International Publishing.

[61] Li, P., Y. Pei, and J. Li (2023). A comprehensive survey on design and application of autoencoder in deep learning. *Applied Soft Computing 138*, 110176.

[62] Zhao, Y., P. Altschuh, J. Santoki, L. Griem, G. Tosato, M. Selzer, A. Koeppe, and B. Nestler (2023). Characterization of porous membranes using artificial neural networks. *Acta Materialia 253*, 118922.

[63] Ji, Y., A. Koeppe, P. Altschuh, D. Rajagopal, Y. Zhao, W. Chen, Y. Zhang, Y. Zheng, and B. Nestler (2024). Towards automatic feature extraction and sample generation of grain structure by variational autoencoder. *Computational Materials Science 232*, 112628.

[64] Goodfellow, I. (2016). Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160 1*, 1–57.

[65] Ghojogh, B., A. Ghodsi, F. Karray, and M. Crowley (2021). Generative adversarial networks and adversarial autoencoders: Tutorial and survey. *CoRR abs/2111.13282*, 1–37.

[66] Wang, K., C. Gou, Y. Duan, Y. Lin, X. Zheng, and F.-Y. Wang (2017). Generative adversarial networks: introduction and outlook. *IEEE/CAA Journal of Automatica Sinica 4*(4), 588–598.

[67] Gonog, L. and Y. Zhou (2019). A review: generative adversarial networks. *14th IEEE conference on industrial electronics and applications (ICIEA) 1*, 505–510.

[68] Sohl-Dickstein, J., E. Weiss, N. Maheswaranathan, and S. Ganguli (2015). Deep unsupervised learning using nonequilibrium thermodynamics. *International conference on machine learning 37*, 2256–2265.

[69] Ho, J., A. Jain, and P. Abbeel (2020). Denoising diffusion probabilistic models. *Advances in neural information processing systems 33*, 6840–6851.

[70] Croitoru, F.-A., V. Hondru, R. T. Ionescu, and M. Shah (2023). Diffusion models in vision: A survey. *IEEE transactions on pattern analysis and machine intelligence 45*(9), 10850–10869.

[71] Yang, L., Z. Zhang, Y. Song, S. Hong, R. Xu, Y. Zhao, W. Zhang, B. Cui, and M.-H. Yang (2023). Diffusion models: A comprehensive survey of methods and applications. *ACM Computing Surveys 56*(4), 1–39.

[72] Song, J., C. Meng, and S. Ermon (2020). Denoising diffusion implicit models. *CoRR abs/2010.02502*, 1–22.

[73] Rombach, R., A. Blattmann, D. Lorenz, P. Esser, and B. Ommer (2022). High-resolution image synthesis with latent diffusion models. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition 1*, 10684–10695.

[74] Wang, Q., Y. Ma, K. Zhao, and Y. Tian (2022). A comprehensive survey of loss functions in machine learning. *Annals of Data Science 9*(2), 187–212.

[75] Terven, J., D.-M. Cordova-Esparza, J.-A. Romero-González, A. Ramírez-Pedraza, and E. Chávez-Urbiola (2025). A comprehensive survey of loss functions and metrics in deep learning. *Artificial Intelligence Review 58*(7), 195.

[76] Mao, X., Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley (2017). Least squares generative adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 2794–2802.

[77] Arjovsky, M., S. Chintala, and L. Bottou (2017). Wasserstein generative adversarial networks. In *International conference on machine learning*, pp. 214–223. PMLR.

[78] Gatys, L. A., A. S. Ecker, and M. Bethge (2016). Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2414–2423.

[79] Johnson, J., A. Alahi, and L. Fei-Fei (2016). Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pp. 694–711. Springer.

[80] Salimans, T., I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen (2016). Improved techniques for training gans. *Advances in neural information processing systems 29*, 1–9.

[81] Betzalel, E., C. Penso, and E. Fetaya (2024). Evaluation metrics for generative models: An empirical study. *Machine Learning and Knowledge Extraction 6*(3), 1531–1544.

[82] Barratt, S. and R. Sharma (2018). A note on the inception score.

[83] Heusel, M., H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter (2017). Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems 30*, 1–12.

[84] Alqahtani, H., M. Kavakli-Thorne, G. Kumar, and F. SBSSTC (2019). An analysis of evaluation metrics of gans. *International Conference on Information Technology and Applications (ICITA) 7*, 2.

[85] Zhang, R., P. Isola, A. A. Efros, E. Shechtman, and O. Wang (2018). The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 586–595.

[86] Wang, Z., A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli (2004). Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing 13*(4), 600–612.

[87] Finegan, D. P., I. Squires, A. Dahari, S. Kench, K. L. Jungjohann, and S. J. Cooper (2022). Machine-learning-driven advanced characterization of battery electrodes. *ACS Energy Letters 7*(12), 4368–4378.

[88] Ning, Y., F. Yang, Y. Zhang, Z. Qiang, G. Yin, J. Wang, and S. Lou (2024). Bridging multimodal data and battery science with machine learning. *Matter 7*(6), 2011–2032.

[89] Xue, P., R. Qiu, C. Peng, Z. Peng, K. Ding, R. Long, L. Ma, and Q. Zheng (2024). Solutions for lithium battery materials data issues in machine learning: Overview and future outlook. *Advanced Science 11*(48), 2410065.

[90] Ling, C. (2022). A review of the recent progress in battery informatics. *npj Computational Materials 8*(1), 33.

[91] He, Q., B. Yu, Z. Li, and Y. Zhao (2019). Density functional theory for battery materials. *Energy & Environmental Materials 2*(4), 264–279.

[92] Tong, Q., P. Gao, H. Liu, Y. Xie, J. Lv, Y. Wang, and J. Zhao (2020). Combining machine learning potential and structure prediction for accelerated materials design and discovery. *The journal of physical chemistry letters 11*(20), 8710–8720.

[93] Gavilán-Arriazu, E. M., M. Mercer, D. Barraco, H. E. Hoster, and E. P. M. Leiva (2021). Kinetic monte carlo simulations applied to li-ion and post li-ion batteries: a key link in the multi-scale chain. *Progress in Energy 3*(4), 042001.

[94] Esmaeilpour, M., S. Jana, H. Li, M. Soleymanibrojeni, and W. Wenzel (2023). A solution-mediated pathway for the growth of the solid electrolyte interphase in lithium-ion batteries. *Advanced Energy Materials 13*(14), 2203966.

[95] Jain, A., S. P. Ong, G. Hautier, W. Chen, W. D. Richards, S. Dacek, S. Cholia, D. Gunter, D. Skinner, G. Ceder, and K. A. Persson (2013). Commentary: The materials project: A materials genome approach to accelerating materials innovation. *APL materials 1*(1), 011002.

[96] Kirklin, S., J. E. Saal, B. Meredig, A. Thompson, J. W. Doak, M. Aykol, S. Rühl, and C. Wolverton (2015). The open quantum materials database (oqmd): assessing the accuracy of dft formation energies. *npj Computational Materials 1*(1), 1–15.

[97] Curtarolo, S., W. Setyawan, G. L. Hart, M. Jahnatek, R. V. Chepulskii, R. H. Taylor, S. Wang, J. Xue, K. Yang, O. Levy, M. J. Mehl, H. T. Stokes, D. O. Demchenko, and D. Morgan (2012). Aflow: An automatic framework for high-throughput materials discovery. *Computational Materials Science 58*, 218–226.

[98] Zagorac, D., H. Müller, S. Ruehl, J. Zagorac, and S. Rehme (2019). Recent developments in the inorganic crystal structure database: theoretical crystal structure data and related features. *Applied Crystallography 52*(5), 918–925.

[99] Fan, C., M. Chen, X. Wang, J. Wang, and B. Huang (2021). A review on data preprocessing techniques toward efficient and reliable knowledge discovery from building operational data. *Frontiers in energy research 9*, 652801.

[100] García, S., S. Ramírez-Gallego, J. Luengo, J. M. Benítez, and F. Herrera (2016). Big data preprocessing: methods and prospects. *Big data analytics 1*(1), 9.

[101] Fan, C., F. Xiao, and C. Yan (2015). A framework for knowledge discovery in massive building automation data and its application in building diagnostics. *Automation in Construction 50*, 81–90.

[102] Kang, P. (2013). Locally linear reconstruction based missing value imputation for supervised learning. *Neurocomputing 118*, 65–78.

[103] Cui, Z., Y. Cao, G. Wu, H. Liu, Z. Qiu, and C. Chen (2018). Research on preprocessing technology of building energy consumption monitoring data based on machine learning algorithm. *Build. Sci 34*(2), 94–99.

[104] Dunteman, G. H. (1989). *Principal components analysis*, Volume 69. Sage.

[105] Wang, Y., H. Yao, and S. Zhao (2016). Auto-encoder based dimensionality reduction. *Neurocomputing 184*, 232–242.

[106] Wang, S., Z. Ding, and Y. Fu (2017). Feature selection guided auto-encoder. In *Proceedings of the AAAI conference on artificial intelligence*, Volume 31, pp. 1–7.

[107] Sujon, K. M., R. B. Hassan, Z. T. Towshi, M. A. Othman, M. A. Samad, and K. Choi (2024). When to use standardization and normalization: empirical evidence from machine learning models and xai. *IEEE access 12*, 135300–135314.

[108] Rajagopal, D., A. Koeppe, M. Esmaeilpour, M. Selzer, W. Wenzel, H. Stein, and B. Nestler (2023). Data-driven virtual material analysis and synthesis for solid electrolyte interphases. *Advanced Energy Materials 13*(40), 2301985.

[109] Rajagopal, D., A. Koeppe, A. Cierpka, and B. Nestler (2025). Latent diffusion models for virtual battery material screening and characterization. *Batteries & Supercaps 0*, 202500075.

[110] Griem, L., A. Koeppe, A. Greß, T. Feser, and B. Nestler (2025). Synthetic training data for ct image segmentation of microstructures. *Acta Materialia 296*, 121220.

[111] Ji, Y., A. Koeppe, P. Altschuh, L. Griem, D. Rajagopal, and B. Nestler (2024). A u-net-based self-stitching method for generating periodic grain structures. *Physica Scripta 99*(7), 076010.

[112] Wang, Z., P. Wang, K. Liu, P. Wang, Y. Fu, C.-T. Lu, C. C. Aggarwal, J. Pei, and Y. Zhou (2025). A comprehensive survey on data augmentation. *IEEE Transactions on Knowledge and Data Engineering 1*, 1–20.

[113] Khosla, C. and B. S. Saini (2020). Enhancing performance of deep learning models with different data augmentation techniques: A survey. In *2020 international conference on intelligent engineering and management (ICIEM)*, pp. 79–85. IEEE.

[114] Magar, R., Y. Wang, C. Lorsung, C. Liang, H. Ramasubramanian, P. Li, and A. B. Farimani (2022). Auglichem: data augmentation library of chemical structures for machine learning. *Machine Learning: Science and Technology 3*(4), 045015.

[115] Dinic, F., Z. Wang, I. Neporozhnii, U. B. Salim, R. Bajpai, N. Rajiv, V. Chavda, V. Radhakrishnan, and O. Voznyy (2023). Strain data augmentation enables machine learning of inorganic crystal geometry optimization. *Patterns 4*(2), 1–8.

[116] Abenza, E., C. Alonso, I. Sobrados, J. M. Amarilla, J. L. Rodríguez, J. A. Alonso, R. G. E. Martín, and M. C. Asensio (2023). Data augmentation for battery materials using lattice scaling.

[117] Wongvorachan, T., S. He, and O. Bulut (2023). A comparison of undersampling, oversampling, and smote methods for dealing with imbalanced classification in educational data mining. *Information 14*(1), 54.

[118] Scheffler, M., M. Aeschlimann, M. Albrecht, T. Bereau, H.-J. Bungartz, C. Felser, M. Greiner, A. Groß, C. T. Koch, K. Kremer, et al. (2022). Fair data enabling new horizons for materials research. *Nature 604*(7907), 635–642.

[119] Ludwig, A. (2019). Discovery of new materials using combinatorial synthesis and high-throughput characterization of thin-film materials libraries combined with computational methods. *npj computational materials 5*(1), 70.

[120] Draxl, C. and M. Scheffler (2020). *Big Data-Driven Materials Science and Its FAIR Data Infrastructure*, pp. 49–73. Cham: Springer International Publishing.

[121] Peters, I., P. Kraker, E. Lex, C. Gumpenberger, and J. I. Gorraiz (2017). Zenodo in the spotlight of traditional and new metrics. *Frontiers in research metrics and analytics 2*, 13.

[122] Magazine, D.-L. (2011). The dataverse network®: an open-source application for sharing, discovering and preserving data. *D-lib Magazine 17*(1/2), 1–2.

[123] Smith, M., M. Barton, M. Bass, M. Branschofsky, G. McClellan, D. Stuve, R. Tansley, and J. H. Walker (2003). An open source dynamic digital repository. *D-Lib Magazine 9*(1), 1082–9873.

[124] Scheidgen, M., L. Himanen, A. N. Ladines, D. Sikter, M. Nakhaee, Á. Fekete, T. Chang, A. Golparvar, J. A. Márquez, S. Brockhauser, et al. (2023). Nomad: A distributed web-based platform for managing materials science research data. *Journal of Open Source Software 8*(90), 5388.

[125] Kluyver, T., B. Ragan-Kelley, F. Pérez, B. Granger, M. Bussonnier, J. Frederic, K. Kelley, J. Hamrick, J. Grout, S. Corlay, et al. (2016). Jupyter notebooks–a publishing format for reproducible computational workflows. In *Positioning and power in academic publishing: Players, agents and agendas*, pp. 87–90. IOS press.

[126] Afgan, E., D. Baker, B. Batut, M. Van Den Beek, D. Bouvier, M. Čech, J. Chilton, D. Clements, N. Coraor, B. A. Grüning, et al. (2018). The galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2018 update. *Nucleic acids research 46*(W1), W537–W544.

[127] CARP, N., A. Minges, and M. Piel (2017). elabftw: An open source laboratory notebook for research labs. *J. Open Source Softw. 2*(12), 146.

[128] Pizzi, G., A. Cepellotti, R. Sabatini, N. Marzari, and B. Kozinsky (2016). Aiida: automated interactive infrastructure and database for computational science. *Computational Materials Science 111*, 218–230.

[129] Brandt, N., L. Griem, C. Herrmann, E. Schoof, G. Tosato, Y. Zhao, P. Zschumme, and M. Selzer (2021). Kadi4mat: A research data infrastructure for materials science. *Data Science Journal 20*, 8–8.

[130] Griem, L., P. Zschumme, M. Laqua, N. Brandt, E. Schoof, P. Altschuh, and M. Selzer (2022). Kadistudio: Fair modelling of scientific research processes. *Data Science Journal 21*, 16–16.

[131] Koeppe, A., F. Bamer, and B. Markert (2019). An efficient Monte Carlo strategy for elasto-plastic structures based on recurrent neural networks. *Acta Mechanica 230*, 3279–3293.

[132] Koeppe, A., F. Bamer, M. Selzer, B. Nestler, and B. Markert (2022). Explainable artificial intelligence for mechanics: physics-explaining neural networks for constitutive models. *Frontiers in Materials 8*, 636.

[133] Mundt, M., A. Koeppe, S. David, T. Witter, F. Bamer, W. Potthast, and B. Markert (2020). Estimation of gait mechanics based on simulated and measured IMU data using an artificial neural network. *Frontiers in bioengineering and biotechnology 8*, 41.

[134] Koeppe, A. and the CIDS team (2025). Cids and kadiai, v3.2a. `https://gitlab.com/intelligent-analysis/cids/-/tree/v3.2a`.

[135] Rajagopal, D., A. Koeppe, M. Esmaeilpour, M. Selzer, H. Stein, and B. Nestler (2023, Nov.). *A Reproducible Machine Learning Workflow to Characterize the Solid Electrolyte Interphase*, pp. 19–31. heiBOOKS.

[136] Mohamed, S. and B. Lakshminarayanan (2017). Learning in implicit generative models.

[137] Razghandi, M., H. Zhou, M. Erol-Kantarci, and D. Turgut (2022). Variational autoencoder generative adversarial network for synthetic data

generation in smart home. In *ICC 2022-IEEE International Conference on Communications*, pp. 4781–4786. IEEE.

[138] Mukesh, K., S. Ippatapu Venkata, S. Chereddy, E. Anbazhagan, and I. Oviya (2022). A variational autoencoder—general adversarial networks (vae-gan) based model for ligand designing. In *International Conference on Innovative Computing and Communications: Proceedings of ICICC 2022, Volume 1*, pp. 761–768. Springer.

[139] Park, S., H. Yoon, D. Lee, J. Choi, H. Kwon, and C. Won (2023). Topological magnetic structure generation using vae-gan hybrid model and discriminator-driven latent sampling. *Scientific Reports 13*(1), 20377.

[140] Bengio, Y., A. Courville, and P. Vincent (2013). Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence 35*(8), 1798–1828.

[141] Tschannen, M., O. Bachem, and M. Lucic (2018). Recent advances in autoencoder-based representation learning.

[142] Greenacre, M., P. J. Groenen, T. Hastie, A. I. d'Enza, A. Markos, and E. Tuzhilina (2022). Principal component analysis. *Nature Reviews Methods Primers 2*(1), 100.

[143] Abdi, H. and L. J. Williams (2010). Principal component analysis. *Wiley interdisciplinary reviews: computational statistics 2*(4), 433–459.

[144] Zhao, H., Z. Wang, and F. Nie (2018). A new formulation of linear discriminant analysis for robust dimensionality reduction. *IEEE Transactions on Knowledge and data engineering 31*(4), 629–640.

[145] Zhao, S., B. Zhang, J. Yang, J. Zhou, and Y. Xu (2024). Linear discriminant analysis. *Nature Reviews Methods Primers 4*(1), 70.

[146] Schölkopf, B., A. Smola, and K.-R. Müller (1997). Kernel principal component analysis. In *International conference on artificial neural networks*, pp. 583–588. Springer.

[147] Jade, A., B. Srikanth, V. Jayaraman, B. Kulkarni, J. Jog, and L. Priya (2003). Feature extraction and denoising using kernel pca. *Chemical Engineering Science 58*(19), 4441–4448.

[148] Zhong, G., L.-N. Wang, X. Ling, and J. Dong (2016). An overview on data representation learning: From traditional feature learning to recent deep learning. *The Journal of Finance and Data Science 2*(4), 265–278.

[149] Wei, R. and A. Mahmood (2020). Recent advances in variational autoencoders with representation learning for biomedical informatics: A survey. *Ieee Access 9*, 4939–4956.

[150] Grossutti, M., J. D'Amico, J. Quintal, H. MacFarlane, A. Quirk, and J. R. Dutcher (2022). Deep learning and infrared spectroscopy: representation learning with a β-variational autoencoder. *The Journal of Physical Chemistry Letters 13*(25), 5787–5793.

[151] Burgess, C. P., I. Higgins, A. Pal, L. Matthey, N. Watters, G. Desjardins, and A. Lerchner (2018). Understanding disentangling in β-vae.

[152] Gómez-Bombarelli, R., J. N. Wei, D. Duvenaud, J. M. Hernández-Lobato, B. Sánchez-Lengeling, D. Sheberla, J. Aguilera-Iparraguirre, T. D. Hirzel, R. P. Adams, and A. Aspuru-Guzik (2018). Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science 4*(2), 268–276.

[153] Wang, L., Y.-C. Chan, F. Ahmed, Z. Liu, P. Zhu, and W. Chen (2020). Deep generative modeling for mechanistic-based learning and design of metamaterial systems. *Computer Methods in Applied Mechanics and Engineering 372*, 113377.

[154] Ziesche, R. F., N. Kardjilov, W. Kockelmann, D. J. Brett, and P. R. Shearing (2022). Neutron imaging of lithium batteries. *Joule 6*(1), 35–52.

[155] Liu, X., L. Zhang, H. Yu, J. Wang, J. Li, K. Yang, Y. Zhao, H. Wang, B. Wu, N. P. Brandon, et al. (2022). Bridging multiscale characterization

technologies and digital modeling to evaluate lithium battery full lifecycle. *Advanced energy materials 12*(33), 2200889.

[156] Aljohani, A. and N. Alharbe (2022). Generating synthetic images for healthcare with novel deep pix2pix gan. *Electronics 11*(21), 3470.

[157] Zhu, J.-Y., T. Park, P. Isola, and A. A. Efros (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 2223–2232.

[158] Supady, A., V. Blum, and C. Baldauf (2015). First-principles molecular structure search with a genetic algorithm. *Journal of Chemical Information and Modeling 55*(11), 2338–2348.

[159] Pyzer-Knapp, E. O., C. Suh, R. Gómez-Bombarelli, J. Aguilera-Iparraguirre, and A. Aspuru-Guzik (2015). What is high-throughput virtual screening? a perspective from organic materials discovery. *Annual Review of Materials Research 45*(1), 195–216.

[160] Jahan, A., M. Y. Ismail, S. Sapuan, and F. Mustapha (2010). Material screening and choosing methods–a review. *Materials & Design 31*(2), 696–705.

[161] Rahim, A. A., S. N. Musa, S. Ramesh, and M. K. Lim (2020). A systematic review on material selection methods. *Proceedings of the Institution of Mechanical Engineers, Part L: Journal of Materials: Design and Applications 234*(7), 1032–1059.

[162] Kim, E., K. Huang, S. Jegelka, and E. Olivetti (2017). Virtual screening of inorganic materials synthesis parameters with deep learning. *npj Computational Materials 3*(1), 53.

[163] Sanchez-Lengeling, B. and A. Aspuru-Guzik (2018). Inverse molecular design using machine learning: Generative models for matter engineering. *Science 361*(6400), 360–365.

[164] Park, J., Y. Lee, and J. Kim (2025). Multi-modal conditional diffusion model using signed distance functions for metal-organic frameworks generation. *Nature Communications 16*(1), 34.

[165] Fuhr, A. S. and B. G. Sumpter (2022). Deep generative models for materials discovery and machine learning-accelerated innovation. *Frontiers in Materials 9*, 865270.

[166] Wang, A., S. Kadam, H. Li, S. Shi, and Y. Qi (2018). Review on modeling of the anode solid electrolyte interphase (sei) for lithium-ion batteries. *NPJ Computational materials 4*(1), 15.

[167] Nie, M., D. P. Abraham, D. M. Seo, Y. Chen, A. Bose, and B. L. Lucht (2013). Role of solution structure in solid electrolyte interphase formation on graphite with lipf6 in propylene carbonate. *The Journal of Physical Chemistry C 117*(48), 25381–25389.

[168] Shkrob, I. A., Y. Zhu, T. W. Marin, and D. Abraham (2013). Reduction of carbonate electrolytes and the formation of solid-electrolyte interface (sei) in lithium-ion batteries. 1. spectroscopic observations of radical intermediates generated in one-electron reduction of carbonates. *The Journal of Physical Chemistry C 117*(38), 19255–19269.

[169] Lu, P. and S. J. Harris (2011). Lithium transport within the solid electrolyte interphase. *Electrochemistry Communications 13*(10), 1035–1037.

[170] Lu, P., C. Li, E. W. Schneider, and S. J. Harris (2014). Chemistry, impedance, and morphology evolution in solid electrolyte interphase films during formation in lithium ion batteries. *The Journal of Physical Chemistry C 118*(2), 896–903.

[171] Liu, D., Z. Shadike, R. Lin, K. Qian, H. Li, K. Li, S. Wang, Q. Yu, M. Liu, S. Ganapathy, et al. (2019). Review of recent development of in situ/operando characterization techniques for lithium battery research. *Advanced Materials 31*(28), 1806620.

[172] Wu, J., M. Ihsan-Ul-Haq, Y. Chen, and J.-K. Kim (2021). Understanding solid electrolyte interphases: Advanced characterization techniques and theoretical simulations. *Nano Energy 89*, 106489.

[173] Alzate-Vargas, L., S. M. Blau, E. W. C. Spotte-Smith, S. Allu, K. A. Persson, and J.-L. Fattebert (2021). Insight into sei growth in li-ion batteries using molecular dynamics and accelerated chemical reactions. *The Journal of Physical Chemistry C 125*(34), 18588–18596.

[174] Bertolini, S. and P. B. Balbuena (2018). Buildup of the solid electrolyte interphase on lithium-metal anodes: reactive molecular dynamics study. *The Journal of Physical Chemistry C 122*(20), 10783–10791.

[175] Bortz, A. B., M. H. Kalos, and J. L. Lebowitz (1975). A new algorithm for monte carlo simulation of ising spin systems. *Journal of Computational physics 17*(1), 10–18.

[176] Bittner, E. R. (2006). Chemical dynamics in the condensed phases:relaxation, transfer, and reactions in condensed molecular systems by abraham nitzan (tel aviv university). oxford university press: Oxford, new york. 2006. xxii + 720 pp. $89.50. isbn 0-19-852979-1. *Journal of the American Chemical Society 128*(51), 17156–17157.

[177] Ushirogata, K., K. Sodeyama, Y. Okuno, and Y. Tateyama (2013). Additive effect on reductive decomposition and binding of carbonate-based solvent toward solid electrolyte interphase formation in lithium-ion battery. *Journal of the American Chemical Society 135*(32), 11967–11974.

[178] Wang, Y., S. Nakamura, M. Ue, and P. B. Balbuena (2001). Theoretical studies to understand surface chemistry on carbon anodes for lithium-ion batteries: reduction mechanisms of ethylene carbonate. *Journal of the American Chemical Society 123*(47), 11708–11718.

[179] Miyabe, K. and R. Isogai (2011). Estimation of molecular diffusivity in liquid phase systems by the wilke–chang equation. *Journal of Chromatography A 1218*(38), 6639–6645.

[180] Ren, E., P. Guilbaud, and F.-X. Coudert (2022). High-throughput computational screening of nanoporous materials in targeted applications. *Digital Discovery 1*(4), 355–374.

[181] Cross, J. O., R. L. Opila, I. W. Boyd, and E. N. Kaufmann (2015). Materials characterization and the evolution of materials. *Mrs Bulletin 40*(12), 1019–1034.

[182] Binnig, G., C. F. Quate, and C. Gerber (1986). Atomic force microscope. *Physical review letters 56*(9), 930.

[183] Loos, J. (2005). The art of spm: Scanning probe microscopy in materials science. *Advanced Materials 17*(15), 1821–1833.

[184] Tennyson, E. M., C. Gong, and M. S. Leite (2017). Imaging energy harvesting and storage systems at the nanoscale. *ACS Energy Letters 2*(12), 2761–2777.

[185] Krieg, M., G. Fläschner, D. Alsteens, B. M. Gaub, W. H. Roos, G. J. Wuite, H. E. Gaub, C. Gerber, Y. F. Dufrêne, and D. J. Müller (2019). Atomic force microscopy-based mechanobiology. *Nature Reviews Physics 1*(1), 41–57.

[186] Huang, B., Z. Li, and J. Li (2018). An artificial intelligence atomic force microscope enabled by machine learning. *Nanoscale 10*(45), 21320–21326.

[187] Alldritt, B., F. Urtev, N. Oinonen, M. Aapro, J. Kannala, P. Liljeroth, and A. S. Foster (2022). Automated tip functionalization via machine learning in scanning probe microscopy. *Computer Physics Communications 273*, 108258.

[188] Zhang, J., R. Wang, X. Yang, W. Lu, X. Wu, X. Wang, H. Li, and L. Chen (2012). Direct observation of inhomogeneous solid electrolyte interphase on mno anode with atomic force microscopy and spectroscopy. *Nano letters 12*(4), 2153–2157.

[189] Zhang, Z., S. Said, K. Smith, R. Jervis, C. A. Howard, P. R. Shearing, D. J. Brett, and T. S. Miller (2021). Characterizing batteries by in situ

electrochemical atomic force microscopy: a critical review. *Advanced Energy Materials 11*(38), 2101518.

[190] Arruda, T. M., M. Heon, V. Presser, P. C. Hillesheim, S. Dai, Y. Gogotsi, S. V. Kalinin, and N. Balke (2013). In situ tracking of the nanoscale expansion of porous carbon electrodes. *Energy & Environmental Science 6*(1), 225–231.

[191] Deng, Z., X. Lin, Z. Huang, J. Meng, Y. Zhong, G. Ma, Y. Zhou, Y. Shen, H. Ding, and Y. Huang (2021). Recent progress on advanced imaging techniques for lithium-ion batteries. *Advanced Energy Materials 11*(2), 2000806.

[192] Gross, L., F. Mohn, N. Moll, P. Liljeroth, and G. Meyer (2009). The chemical structure of a molecule resolved by atomic force microscopy. *Science 325*(5944), 1110–1114.

[193] Carracedo-Cosme, J., P. Hapala, and R. Pérez (2024). Atomic force microscopy simulations for co-functionalized tips with deep learning. *Machine Learning: Science and Technology 5*(2), 025025.

[194] Oinonen, N., A. V. Yakutovich, A. Gallardo, M. Ondráček, P. Hapala, and O. Krejčí (2024). Advancing scanning probe microscopy simulations: A decade of development in probe-particle models. *Computer Physics Communications 305*, 109341.

[195] Hapala, P., R. Temirov, F. S. Tautz, and P. Jelínek (2014). Origin of high-resolution iets-stm images of organic molecules with functionalized tips. *Physical review letters 113*(22), 226101.

[196] Hapala, P., G. Kichin, C. Wagner, F. S. Tautz, R. Temirov, and P. Jelínek (2014). Mechanism of high-resolution stm/afm imaging with functionalized tips. *Physical Review B 90*(8), 085421.

[197] Guo, C.-S., M. A. Van Hove, R.-Q. Zhang, and C. Minot (2010). Prospects for resolving chemical structure by atomic force microscopy: A first-principles study. *Langmuir 26*(21), 16271–16277.

[198] Kaiser, K., F. Schulz, J. F. Maillard, F. Hermann, I. Pozo, D. Peña, H. J. Cleaves II, A. S. Burton, G. Danger, C. Afonso, S. Sandford, and L. Gross (2022). Visualization and identification of single meteoritic organic molecules by atomic force microscopy. *Meteoritics & Planetary Science 57*(3), 644–656.

[199] Ellner, M., P. Pou, and R. Pérez (2019). Molecular identification, bond order discrimination, and apparent intermolecular features in atomic force microscopy studied with a charge density based method. *ACS nano 13*(1), 786–795.

[200] Kulagin, R., P. Reiser, K. Truskovskyi, A. Koeppe, Y. Beygelzimer, Y. Estrin, P. Friederich, and P. Gumbsch (2023). Lattice metamaterials with mesoscale motifs: exploration of property charts by bayesian optimization. *Advanced Engineering Materials 25*(13), 2300048.

[201] Alldritt, B., P. Hapala, N. Oinonen, F. Urtev, O. Krejci, F. Federici Canova, J. Kannala, F. Schulz, P. Liljeroth, and A. S. Foster (2020). Automated structure discovery in atomic force microscopy. *Science advances 6*(9), eaay6913.

[202] Priante, F., N. Oinonen, Y. Tian, D. Guan, C. Xu, S. Cai, P. Liljeroth, Y. Jiang, and A. S. Foster (2024). Structure discovery in atomic force microscopy imaging of ice. *ACS nano 18*(7), 5546–5555.

[203] Oinonen, N., C. Xu, B. Alldritt, F. F. Canova, F. Urtev, S. Cai, O. Krejčí, J. Kannala, P. Liljeroth, and A. S. Foster (2021). Electrostatic discovery atomic force microscopy. *ACS nano 16*(1), 89–97.

[204] Yu, T., C. Wang, H. Yang, and F. Li (2024). Machine learning in metal-ion battery research: Advancing material prediction, characterization, and status evaluation. *Journal of Energy Chemistry 90*, 191–204.

[205] Zhou, F., M. Cococcioni, C. A. Marianetti, D. Morgan, and G. Ceder (2004). First-principles prediction of redox potentials in transition-metal compounds with lda+ u. *Physical Review B—Condensed Matter and Materials Physics 70*(23), 235121.

[206] Wang, L., T. Maxisch, and G. Ceder (2007). A first-principles approach to studying the thermal stability of oxide cathode materials. *Chemistry of materials 19*(3), 543–552.

[207] Adams, S. and R. P. Rao (2011). High power lithium ion battery materials by computational design. *physica status solidi (a) 208*(8), 1746–1753.

[208] Bölle, F. T., N. R. Mathiesen, A. J. Nielsen, T. Vegge, J. M. Garcia-Lastra, and I. E. Castelli (2020). Autonomous discovery of materials for intercalation electrodes. *Batteries & Supercaps 3*(6), 488–498.

[209] Mishra, N., R. Boral, and T. Paul (2025). Designing layered oxides as cathodes for sodium-ion batteries: Machine learning and density functional theory based modeling. *Materials Today Physics 51*, 101634.

[210] Azambou, C. I., F. H. K. Djioko, O. O. Obiukwu, P. K. Tsobnang, E. E. Kalu, I. T. Kenfack, and E. E. Oguzie (2023). Structural, electronic, mechanical and thermodynamic properties of lithium-rich layered oxides cathode materials for lithium-ion battery: Computational study. *Materials Today Communications 35*, 105738.

[211] Hu, W., H. Wang, W. Luo, B. Xu, and C. Ouyang (2020). Formation and thermodynamic stability of oxygen vacancies in typical cathode materials for li-ion batteries: Density functional theory study. *Solid State Ionics 347*, 115257.

List of Figures

List of Tables

List of Symbols and Abbreviations

Symbols

$\mathcal{D}$	Dataset
$\boldsymbol{\theta}$	Model parameters
η	Learning rate
$\mathcal{L}$	Loss function
$\mathscr{L}$	Likelihood function
ℓ	Log likelihood function
$\mathcal{X}$	Input space
$\mathcal{Z}$	Latent space
$\boldsymbol{x}$	Input sample
$\hat{\boldsymbol{x}}$	Reconstructed or generated sample
$\boldsymbol{z}$	Latent sample (sampled from latent space $\mathcal{Z}$)
$p(\boldsymbol{x})$	Probability distribution of variable $\boldsymbol{x}$
$p(\boldsymbol{z} \mid \boldsymbol{x})$	Conditional probability distribution of $\boldsymbol{z}$ given $\boldsymbol{x}$
$f(\boldsymbol{x};\boldsymbol{\theta})$	A function of $\boldsymbol{x}$ parametrized with $\boldsymbol{\theta}$
$\mathcal{N}(\boldsymbol{x};\boldsymbol{\mu},\boldsymbol{\Sigma})$	Gaussian distribution over $\boldsymbol{x}$ with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$
$\mathbb{E}_{\boldsymbol{x}\sim p}[f(\boldsymbol{x})]$	Expectation of $f(\boldsymbol{x})$ with respect to $p(\boldsymbol{x})$
$\mathcal{R}$	Expected risk
$\boldsymbol{\theta}$	Model parameters
$\boldsymbol{w}$	Weight vector

b	Bias vector

Abbreviations

AFM	Atomic Force Microscopy
AI	Artificial Intelligence
BCE	Binary Cross Entropy
CCE	Categorical Cross Entropy
CNN	Convolutional Neural Network
DDIM	Denoising Diffusion Implicit Model
DDPM	Denoising Diffusion Probabilistic Model
DFT	Density Functional Theory
DL	Deep Learning
DM	Diffusion Model
ELN	Electronic Lab Notebook
FID	Fréchet Inception Distance
GAN	Generative Adversarial Network
IS	Inception Score
KID	Kernel Inception Distance
KMC	Kinetic Monte Carlo
KNN	K Nearest Neighbors
LDA	Linear Discriminant Analysis
LDM	Latent Diffusion Model
MAE	Mean Absolute Error
MD	Molecular Dynamics
ML	Machine Learning
MSE	Mean Squared Error
NN	Neural Network
PCA	Principal Component Analysis

PPAFM	Probe Particle Atomic Force Microscopy
PSNR	Peak Signal Noise Ratio
RDM	Research Data Management
SEI	Solid Electrolyte Interphase
SEM	Scanning Electron Microscopy
SGD	Stochastic Gradient Descent
SSIM	Structural Similarity Index Metric
TEM	Transmission Electron Microscopy
VAE	Variational AutoEncoder
XPS	X-ray Photoelectron spectroscopy

A List of Contributions

Peer-reviewed journal publications

[1] **D.Rajagopal**, A. Koeppe, M. Esmaeilpour, M. Selzer, W. Wenzel, H. Stein, and B. Nestler, Data-Driven Virtual Material Analysis and Synthesis for Solid Electrolyte Interphases. Adv. Energy Mater. 2023, 13, 2301985. https://doi.org/10.1002/aenm.202301985.

[2] **D. Rajagopal**, A. Koeppe, A.Cierpka and B.Nestler, 2025. Latent Diffusion Models for Virtual Battery Material Screening and Characterization. Batteries & Supercaps, p.202500075. doi: https://doi.org/10.1002/batt.202500075

[3] **D. Rajagopal**, A.Cierpka, B.Nestler, and A.Koeppe, 2025. Generative Deep Learning for Advanced Battery Materials. Batteries & Supercaps, p.e202500494. doi: https://doi.org/10.1002/batt.202500494

[4] Y. Ji, A. Koeppe, P. Altschuh, **D. Rajagopal**, Y.Zhao, W. Chen, Y.Zhang, Y. Zheng and B. Nestler, “Towards automatic feature extraction and sample generation of grain structure by variational autoencoder,” Computational Materials Science, vol. 232, p. 112628, Nov. 2023, doi: https://doi.org/10.1016/j.commatsci.2023.112628.

[5] Y. Ji, A. Koeppe, P. Altschuh, L. Griem, **D. Rajagopal**, and B. Nestler, “A U-Net-Based Self-Stitching Method for Generating Periodic Grain Structures,” Physica Scripta, vol. 99, no. 7, pp. 076010–076010, May 2024, doi: https://doi.org/10.1088/1402-4896/ad52cf.

[6] P. Stüble, C. Müller, N. Bohn, M. Müller, A. Hofmann, T. Akçay, J. Klemens, A. Koeppe, S. Kolli, **D. Rajagopal**, H. Geßwein, W. Schabel, P. Scharfer, M. Selzer, J. R. Binder, A. Smith, Batteries & Supercaps 2024, 7, e202400406. doi: `https://doi.org/10.1002/batt.202400406`

Peer-reviewed conference proceedings

[8] **D. Rajagopal**, A. Koeppe, M. Esmaeilpour, M. Selzer, H. Stein, and B. Nestler, "A Reproducible Machine Learning Workflow to Characterize the Solid Electrolyte Interphase", in E-Science-Tage 2023: Empower Your Research – Preserve Your Data, V. Heuveline, N. Bisheh, and P. Kling, Hrsg. Heidelberg: heiBOOKS, 2023, S. 19–31. doi: `https://doi.org/10.11588/heibooks.1288.c18061`.

Conference contributions

[9] **D.Rajagopal**, A. Koeppe, M. Esmaeilpour, M. Selzer, W. Wenzel, H. Stein, B. Nestler, "Characterization of Solid electrolyte interphase using deep generative model", MSE (2022), Materials Science and Engineering, Darmstadt, Germany, Poster.

[10] **D. Rajagopal**, A. Koeppe, M. Esmaeilpour, M. Selzer, H. Stein, und B. Nestler, "A Reproducible Machine Learning Workflow to Characterize the Solid Electrolyte Interphase", in E-Science-Tage 2023, Heidelberg, Germany, Presentation.

[11] **D. Rajagopal**, A. Koeppe, G.Tosato, H.Noman, M.Jayavarapu, and M. Selzer, "Integrated data analysis with Kadi4Mat". HeFDiWorkshop 2024, online, Presentation.