Karlsruhe Institute of Technology

Estimating Dependency, Monitoring and Knowledge Discovery in High-Dimensional Data Streams

zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften

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

genehmigte

Dissertation

von

Edouard Fouché

Tag der mündlichen Prüfung:	15. Juli 2020
Erster Gutachter:	Prof. Dr.-Ing. Klemens Böhm
Zweiter Gutachter:	Asst. Prof. Dr. Junpei Komiyama

Acknowledgements

The journey of an aspiring doctor is long and tedious. At the same time, it is an incredibly fulfilling and transcending experience. I consider this work as my most outstanding achievement so far, but also as a product of my interactions with the people I have met on the way. In this short essay, I want to acknowledge their contributions.

My first thank goes to my Doktorvater, Prof. Klemens Böhm. Thanks to his tireless feedback, I have tremendously improved the quality of my writing and thinking. He gave me all the freedom I needed to thrive and explore. Yet, he kept a sharp eye on my work and continuously challenged my ideas. In the beginning, I blindly trusted Prof. Böhm about my capacity to go through the process of becoming a doctor. I see him as a mentor, and I hope that our relationship will continue to grow.

The quality of this work would not be the same without the support of Junpei Komiyama. He is my first external co-author and my second reviewer. Although we did not know each other before, Junpei came to my aid as I asked him questions about Multi-Armed Bandits. I am proud of the results of our joint work, featured in this dissertation. Junpei invited me to visit him for a short research stay in Tokyo, and I am grateful for this experience. I now consider him a friend, and I wish him all the best with his academic endeavours.

I was also fortunate to visit Prof. Jiawei Han in his lab (the Data Mining Group), at UIUC. During my studies, I learned the fundamentals of Data Mining from his books, and the idea of working with him was like a childhood dream. Thank you, Prof. Han, for sharing with me your wisdom and optimism! Thank you for your gifts and your generosity!

About six years ago, I was an exchange student (via the Erasmus program) at KIT, with the ambition to become fully-enrolled. Prof. Achim Streit, who I met early in my stay, strongly supported my application. Without him, I might never have had the chance to write this dissertation. It was an honour and a great pleasure to have him as an examiner for my defence. I felt very proud to show him how right he was to support me.

Of course, I shall not forget where I come from. Before studying at KIT, I graduated as an engineer from ESIEE Paris. The privileged and practical training I received from this school was key to my development. I am grateful to my professors and teachers, and in particular: Jean-Francois Bercher, Denis Bureau, Michel Couprie, Jean Cousty, Elia Habib. Their early recognition of my potential gave me the confidence to go as far as I am now.

I am also thankful to all my students, and I want to address a special thank to Daniel, Rosina, Lucas, Alan, Marco and Florian. I have learned a lot by interacting with you. Thank you for the fruitful exchanges and our long-lasting collaborations!

Working at IPD would not be the same without the constant help of our secretaries Barbara and Bettina. I would not have been able to focus as much on my research without the support of our system administrators Christian and Herma. I also want to thank Jutta for the orchestration of our teaching activities. Finally, I want to acknowledge the contribution of my colleagues: Vadim, Georg, Adrian, Holger, Michael. Although we

may not have always agreed on fundamental questions, our exchanges have always been pleasant and stimulating! I also want to thank our Alumnus, Fabian Keller. Thanks for your insights and letting me stand on your shoulders! Let me cross-reference here the dissertations of my new confrères: [Ste20; Vol20; Tri20; Kel15].

I must also acknowledge my financial sponsors: The DFG Research Training Group 2153: 'Energy Status Data – Informatics Methods for its Collection, Analysis and Exploitation' and the German Federal Ministry of Education and Research (BMBF) via Software Campus (01IS17042). Thanks to them, I obtained the resources I needed to conduct my research.

I want to thank the fantastic people I have met from the Roller Derby community. Skating together on the track or the ramp was the best evasion I could imagine. Thanks to all the members from Roller Derby Karlsruhe, and in particular: Track Gyver, Effi Biest and Scarylin Manson. Thanks to the skaters of South German Men's Roller Derby (SGMRD) for giving me my first tournament experience. Even outside of Germany, the Roller Derby community provided a home to me. I want to thank the Twin City Derby Girls and the Prairieland Punishers for skating and hanging out with me! Thank you, Tommy, Crash, Beasly, Splatsy, Funder, Killpop, Naughty, Veligerent, Nikita and Steam. I admire your kindness and courage, and my stay in Illinois would not have been the same without you.

I would not have done it without the support of my friends. I want to thank Vojtech, Anguel, Bojan, Maxim, Cédric, Matthias, Daniel, Sancho, Jano, Cassandra, Markus, Noémie and Pauline. Thank you so much, Alex, for checking my German abstract, I owe you one!

Also, I want to thank my family, especially my mother, Béatrice, my father, Emmanuel, and my little sister, Clémence. Despite our occasional disagreements, you have been by my side all this time. I want to thank my grandfather, Florent, for showing me the beauty of nature, and my grandmother, Marie-Claire, for knowing me so well. Thank you, Colette and Jean-Pierre, for your encouragements. I also want to thank those who did not think I should pursue this degree – You gave me the stamina to go for the extra mile.

My dearest thanks go to Charlotte. Thank you for being my sparring partner and always discussing my ideas, even at their most primitive stages. Thanks for sharing your life with me and for believing in me. Also, thank you, Chantal and Franck, for your constant support and understanding.

May you, dear reader, have as much pleasure as I had when I wrote this dissertation!

Edouard Fouché, 29th November 2020, Karlsruhe.

Suggested soundtrack: *Rendez-vous*, Jean-Michel Jarre, 1986 (side one).

Digital version: `https://doi.org/10.5445/IR/1000127232`

Abstract

Data Mining – known as the process of extracting knowledge from massive data sets – leads to phenomenal impacts on our society, and now affects nearly every aspect of our lives: from the layout in our local grocery store, to the ads and product recommendations we receive, the availability of treatments for common diseases, the prevention of crime, or the efficiency of industrial production processes.

However, Data Mining remains difficult when (1) data is high-dimensional, i.e., has many attributes, and when (2) data comes as a stream. Extracting knowledge from high-dimensional data streams is impractical because one must cope with two orthogonal sets of challenges. On the one hand, the effects of the so-called *curse of dimensionality* bog down the performance of statistical methods and yield to increasingly complex Data Mining problems. On the other hand, the statistical properties of data streams may evolve in unexpected ways, a phenomenon known in the community as *concept drift*. Thus, one needs to update their knowledge about data over time, i.e., to monitor the stream.

While previous work addresses high-dimensional data sets and data streams to some extent, the intersection of both has received much less attention. Nevertheless, extracting knowledge in this setting is advantageous for many industrial applications: identifying patterns from high-dimensional data streams in real-time may lead to larger production volumes, or reduce operational costs. The goal of this dissertation is to bridge this gap.

We first focus on dependency estimation, a fundamental task of Data Mining. Typically, one estimates dependency by quantifying the strength of statistical relationships. We identify the requirements for dependency estimation in high-dimensional data streams and propose a new estimation framework, Monte Carlo Dependency Estimation (MCDE), that fulfils them all. We show that MCDE leads to efficient dependency monitoring.

Then, we generalise the task of monitoring by introducing the Scaling Multi-Armed Bandit (S-MAB) algorithms, extending the Multi-Armed Bandit (MAB) model. We show that our algorithms can efficiently monitor statistics by leveraging user-specific criteria.

Finally, we describe applications of our contributions to Knowledge Discovery. We propose an algorithm, Streaming Greedy Maximum Random Deviation (SGMRD), which exploits our new methods to extract patterns, e.g., outliers, in high-dimensional data streams. Also, we present a new approach, that we name kj-Nearest Neighbours (kj-NN), to detect outlying documents within massive text corpora.

We support our algorithmic contributions with theoretical guarantees, as well as extensive experiments against both synthetic and real-world data. We demonstrate the benefits of our methods against real-world use cases. Overall, this dissertation establishes fundamental tools for Knowledge Discovery in high-dimensional data streams, which help with many applications in the industry, e.g., anomaly detection, or predictive maintenance.

To facilitate the application of our results and future research, we publicly release our implementations, experiments, and benchmark data via open-source platforms.

Zusammenfassung

Die Forschung im Bereich Data-Mining — gemeinhin bekannt als der Prozess der Extraktion von Wissen aus riesigen Datensätzen — hat phänomenale Auswirkungen auf unsere Gesellschaft. Data-Mining beeinflusst fast alle Aspekte unseres Lebens, sei es die Produktanordnung im örtlichen Lebensmittelgeschäft, die uns angezeigten Kaufempfehlungen, das Finden passender Therapien für neue Krankheiten, die Verbrechensprävention oder die Effizienz industrieller Produktionsprozesse.

Data-Mining bleibt jedoch schwierig, insbesondere wenn (1) die Daten hochdimensional sind, also viele Attribute haben, und wenn (2) die Daten kontinuierlich eintreffen. Daten, die kontinuierlich eintreffen, werden als Datenstrom bezeichnet. Das Extrahieren von Wissen aus hochdimensionalen Datenströmen ist kompliziert, weil man zwei orthogonale Herausforderungen bewältigen muss: Einerseits verringert der sogenannte *Fluch der Dimensionalität* die Leistungsfähigkeit der statistischen Methoden, was zu immer komplexeren Data-Mining-Aufgaben führt. Andererseits können sich die statistischen Eigenschaften von Datenströmen auf unberechenbare Weise ändern. Dieses Phänomen ist in der Fachwelt als *Konzeptdrift* bekannt. Daher muss unser Wissen über Daten im Laufe der Zeit aktualisiert werden, was auch bedeutet, den Datenstrom ständig zu überwachen.

Während sich die bestehende Literatur bis zu einem gewissen Grad mit hochdimensionalen Datensätzen und Datenströmen auseinandersetzt, ist die Schnittmenge der hier beschriebenen Herausforderungen in der Forschung unterrepräsentiert. Dennoch ist die Extraktion von Wissen in diesem Umfeld für viele industrielle Anwendungen wichtig: Die Echtzeit-Identifizierung von Mustern aus hochdimensionalen Datenströmen kann die Betriebskosten senken oder das Produktionsvolumen erhöhen. Das Ziel dieser Dissertation ist es daher, diese Lücke zu schließen.

Zunächst beschäftigen wir uns mit Verfahren zur Schätzung von Abhängigkeit, einer grundlegenden Aufgabe des Data-Mining. Typischerweise schätzt man dabei die Abhängigkeit durch Quantifizierung der Stärke der statistischen Beziehung zwischen Attributen. Diese Schätzung basiert auf den verfügbaren Beobachtungen. In dieser Arbeit identifizieren wir die wünschenswerten Merkmale für solche Verfahren in hochdimensionalen Datenströmen. Anschließend schlagen wir ein neues Verfahren mit dem Namen Monte-Carlo-Abhängigkeitsschätzung (Monte Carlo Dependency Estimation (MCDE)) vor, das all diese Merkmale erfüllt. Wir zeigen, dass MCDE eine effiziente Überwachung der Abhängigkeiten ermöglicht.

Im nächsten Schritt verallgemeinern wir die Aufgabe der Überwachung von Statistiken. Wir führen die "Skalierenden Mehrarmigen Banditen"-Algorithmen (Scaling Multi-Armed Bandit (S-MAB)) als Erweiterung des sogenannten "Mehrarmigen Banditen"-Modell (Multi-Armed Bandit (MAB)) ein. Hierbei zeigen wir, dass unsere Algorithmen viele Statistiken effizient überwachen können, indem sie benutzerspezifische Kriterien berücksichtigen.

Schließlich beschreiben wir Anwendungen unserer Beiträge zur Entdeckung von Wissen. Wir schlagen einen Algorithmus mit dem Namen Streaming Greedy Maximum Random Deviation (SGMRD) vor, der unsere neu entwickelten Methoden zur Erleichterung der Unterraumsuche in hochdimensionalen Datenströmen nutzt. Außerdem hilft SGMRD, Muster, wie zum Beispiel Ausreißer, zu extrahieren. Abschließend stellen wir eine neue Methode namens kj-Nächste Nachbarn (kj-Nearest Neighbours (kj-NN)) vor, um ungewöhnliche Dokumente innerhalb großer Textkorpora zu erkennen.

Für die von uns eingeführten Algorithmen entwickeln wir zudem theoretische Garantien. Zusätzlich führen wir umfangreiche Experimente mit synthetischen und realen Daten durch. Wir demonstrieren die Vorteile unserer Methoden mit Hilfe von realer Anwendungsfälle. Insgesamt werden in dieser Arbeit grundlegende Werkzeuge für die Wissensentdeckung in hochdimensionalen Datenströmen eingeführt, die bei einem breiten Spektrum von Anwendungen in der Industrie helfen, um beispielsweise Anomalien zu erkennen oder vorausschauender zu warten.

Um die Anwendung unserer Methoden und zukünftige Forschung zu erleichtern, veröffentlichen wir unsere Implementierungen, Experimente und Benchmark-Daten über Open-Source-Plattformen.

Table of Contents

Part I.

Introduction

1. Overview

1.1. High-Dimensional Data Stream Mining

1.1.1. Knowledge Discovery from Data

Data Mining, also referred to as Knowledge Discovery from Data (KDD), is known in the community as the process of extracting useful patterns from massive data repositories, such as large databases, data warehouses, or data streams [HKP11]. The KDD process usually serves as a reference for the required steps of any data analysis task and draws the plan of most textbooks in the field [MR10; HKP11; Kan19; Agg15]. In Figure 1.1, we represent this process, which consists of seven steps, with multiple feedback loops:

1. **Data Cleaning**: Removing noise and inconsistency in the data.
2. **Data Integration**: Combining multiple data sources into a Data Warehouse.
3. **Data Selection**: Selecting the observations/attributes relevant to the task at hand.
4. **Data Transformation**: Transforming the data into a format adequate for further analysis (e.g., deriving new features and aggregates from the existing attributes).
5. **Data Mining**: Using specific algorithms to extract patterns from data. Examples of potentially interesting patterns are groups of frequent items, clusters or outliers.
6. **Pattern Evaluation**: Filtering the patterns found in the previous step by identifying the most valuable ones for a given task, e.g., based on a set of quality criteria.
7. **Knowledge Representation**: Representing the patterns via representation and visualisation methods to make them interpretable for the users.

The KDD process typically is treated as a cyclic pipeline, in which the knowledge gained over each iteration helps to improve the execution of each step.

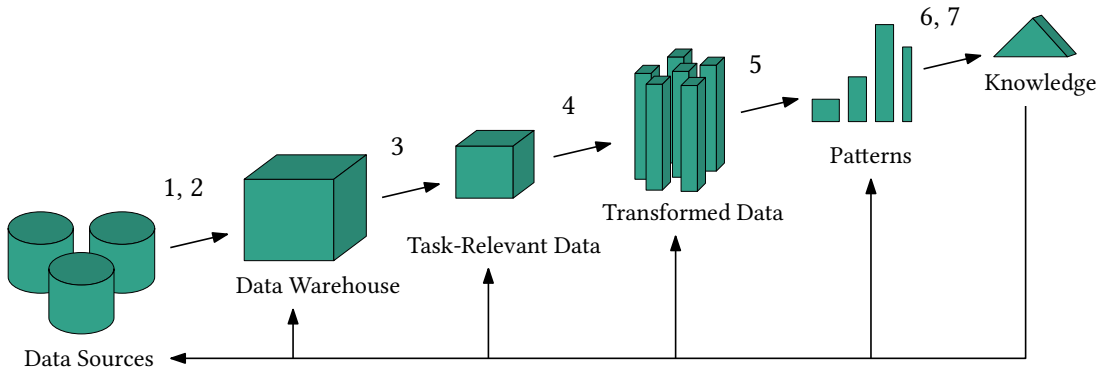


Figure 1.1.: Knowledge Discovery from Data (KDD), according to [HKP11].

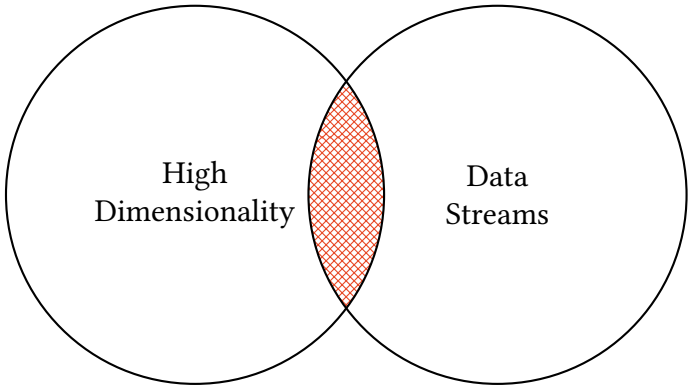


Figure 1.2.: A gap in the Data Mining research landscape.

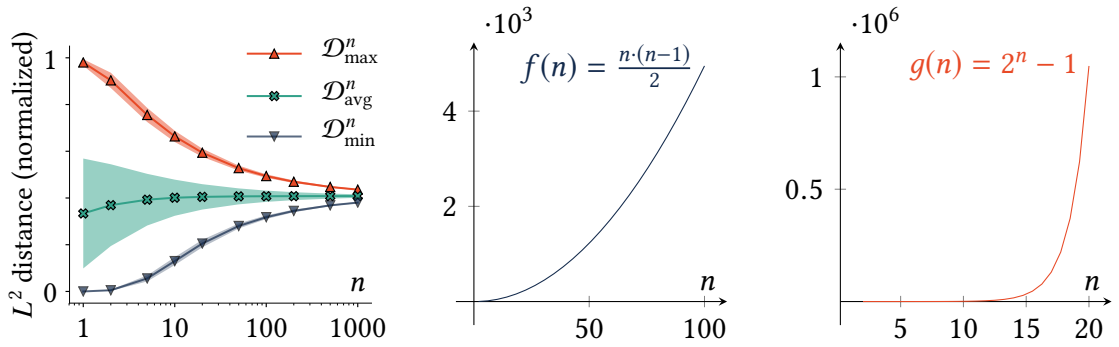
As we can see, Data Mining also refers to the core step of this process (Step 5). The literature uses the term Data Mining interchangeably with KDD, perhaps for conciseness [HKP11]. In turn, the community usually describes Data Mining as a discipline at the frontier of multiple fields: Database Management, Artificial Intelligence, Machine Learning, Pattern Recognition, Data Visualisation. In this dissertation, we refer to Data Mining or Knowledge Discovery as the whole process and leave the debate on the semantic differences w.r.t. Machine Learning, Data Science, etc., to the literature [TSK19; Fri97].

Researchers have examined every step of the process, and a wide range of methods is now available to extract patterns from data. However, Data Mining remains challenging under certain circumstances. In particular, when (1) the data is high-dimensional, i.e., it has many attributes, and when (2) the data comes as a stream. These two problems have raised the interest of the research community for years [ZSK12; Agg13; Ram+17; Gam12]. While the literature separately addressed them to some extent, the intersection of both has received less attention. For example, recent clustering [Sil+13; Gam10; ATS14; WLH12] or prediction [TTL11; Wan+17; ZSK12; Ass+12; SA16; Che+17] algorithms tend to tackle at most one of those aspects: either high-dimensionality restricted to static data, or the streaming scenario, limited to few variables. The difficulty is that both the high-dimensionality and the streaming setting come with distinct sets of challenges.

Data Mining at the intersection of both problems remains mostly unaddressed [SR18], i.e., there is a gap in the current research landscape (Figure 1.2). While one option is to extend or invent specific methods for high-dimensional data streams, as in [ZGW08; Nto+12], a perhaps more general – but not less effective – contribution is to develop fundamental tools for Knowledge Discovery in this particularly challenging setting.

This observation forms the starting point of our dissertation. We bridge this gap by proposing algorithms to address both sets of challenges. We start with a fundamental task: dependency estimation. We show that, combined with efficient monitoring techniques, our algorithms support further Knowledge Discovery in high-dimensional data streams.

In the remaining of this chapter, we detail the challenges associated with high-dimensionality and data streams. Then, we show that the task of dependency estimation is critical to most Data Mining applications and present our running use case: the Bioliq power plant. Finally, we detail our contributions and the outline of this dissertation.



(a) The *distance concentration* effect (b) Number of pairs (c) Number of subspaces

Figure 1.3.: Illustration of the effects of the *curse of dimensionality*.

1.1.2. Challenges of High-Dimensionality

When the data is high-dimensional, several effects, summarised as the *curse of dimensionality* [Bel57], bog down the performance of traditional statistical approaches [Bey+99].

A linear increase in the number of dimensions in the euclidean space leads to an exponential increase in volume. The space becomes extremely sparse, and the distances between each pair of points converge to the same value. This effect is known as *distance concentration*. [Bey+99] showed that, under broad conditions, the probability that the minimum $\mathcal{D}^n_{\min}$ and the maximum distance $\mathcal{D}^n_{\max}$ differ by a factor smaller than $1 + \epsilon$ converges to 1 as the number of dimensions n increases, i.e.,

$$\lim_{n\to\infty} \Pr\left[\mathcal{D}^n_{\max} \leq (1+\epsilon)\cdot\mathcal{D}^n_{\min}\right] = 1, \quad \forall\epsilon > 0. \tag{1.1}$$

Figure 1.3(a) illustrates this via simulation. We report the maximum, average and minimum euclidean pairwise normalised distances within 100 *i.i.d.* observations in $\mathcal{U}[0,1]$. We average the results over 1000 independent trials; The coloured areas show the standard deviation. As we can see, the distances converge as n increases. Thus, in high-dimensional spaces, observations tend to be equally far from each other. The performance of most Data Mining algorithms, which rely on local neighbourhoods, drastically decreases.

A common workaround consists in restricting the analysis to small sets of attributes (subspaces) of interest, e.g., via *dimensionality reduction* methods. However, *dimensionality reduction* also is challenging: the number of pairs for a number d of attributes is $\frac{n\cdot(n-1)}{2}$ for symmetric measures (Figure 1.3(b)), while the number of subspaces is $2^n - 1$ (Figure 1.3(c)). As a result, they grow quadratically/exponentially with n, so that, even for a moderate amount of attributes, one cannot afford to investigate each attribute pair or subspace.

For example, with 20 attributes, the number of subspaces is already more than one million (Figure 1.3(c)). [Agg13] famously compared the task of finding a pattern (e.g., an outlier) in high-dimensional spaces to that of searching for a needle in a haystack, while the haystack is one from an exponential number of haystacks. Subspace search methods, such as [Ngu+13; NMB13; KMB12; TB19], efficiently solve this problem to some extent.

We refer to [HTF09] for further illustrations of the effect of the *curse of dimensionality* and to [ZSK12] for a discussion centred on outlier detection.

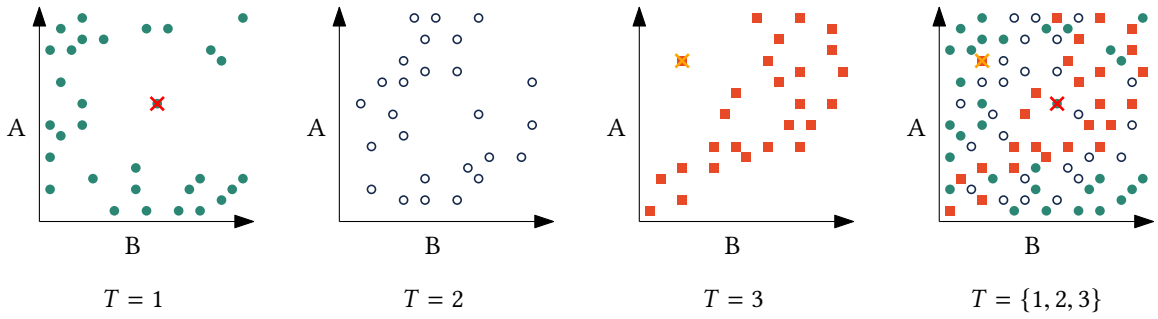


Figure 1.4.: Concept drift – Patterns are only visible in their local context.

1.1.3. Challenges of Data Streams

In the real world, data often is a stream, i.e., data is collected online and may evolve in unforeseeable ways. In this setting, static approaches are ineffective [Gam12] because of changes in the data generation, characterised by a phenomenon known as *concept drift*.

Thus, patterns are meaningful only in the right time context. For example, in Figure 1.4, an observed outlier at time $T = 1$ might not be relevant any more at time $T = 2$ and vice-versa. Next, the scale of this time context also is unknown. In broader time contexts, e.g., $T = \{1, 2, 3\}$, the outliers may not be visible as such.

Data streams are subject to various types of *concept drifts* [BGE15], which can vary in speed and intensity and may also have seasonal components. The streaming nature of the data constrains system design in several ways, laid out by [DH03] as follows:

- **Efficiency (C1):** The system must spend a short constant time and a constant amount of memory to process each incoming record.
- **Single Scan (C2):** The system may perform at most one scan over the data since access to past observations is often unavailable or impractical.
- **Adaptation (C3):** Whenever the data distribution changes over time, the system must adapt, e.g., by forgetting outdated information.
- **Anytime (C4):** The system must be available at any point in time, with an output ideally equivalent (or nearly identical) to the one of a non-streaming system, operating without the streaming constraints.

The streaming scenario is particularly challenging because one needs to monitor the evolution of data. Data Mining algorithms must integrate an adequate forgetting mechanism to discard obsolete information and monitor the stream.

Furthermore, streams often consist of observations with various types, e.g., numerical, ordinal, or categorical. Such data sources are known as *Heterogeneous Data Streams* [YZ06]. [Dit+15] identified mining from heterogeneous data as one of the most challenging problems of Data Mining. Thus, *heterogeneity* adds up to the list of constraints above:

- **Heterogeneity (C5)**: The system must handle not only numerical types but ideally all data types such as strings, categories, ordinal values.

The main challenge is to provide Data Mining techniques that can cope both with the streaming constraints and high-dimensionality – our goal in this dissertation.

1.1.4. The Central Role of Estimating Dependency

Dependency estimation is a fundamental technique in Data Mining [CHY96] and consists of quantifying the strength of the statistical relationship between attributes, based on the available observations. It helps to find the relevant variables for the task at hand, which leads to a better understanding of data and improves both the runtime and the outcomes of downstream mining tasks, e.g., classification, outlier detection, or clustering.

To this end, dependency estimators such as the *Pearson correlation coefficient*, or *mutual information* [Sha48] as a non-linear counterpart, are perhaps the most well-known examples. Dependency estimation techniques are a building block of many Data Mining approaches, and play a role at every step of the process (see Figure 1.1), for example:

- Exploratory Data Mining (EDM) [DJ03; Ass+07; Tat+12] is fundamental for **Data Cleaning** and **Data Integration**. EDM reveals structures in the data, e.g., artefacts and inconsistencies that one should clean, or relevant attributes to integrate before subsequent analysis. Dependency estimators [PHL04; ZS02; MNP10] are a characteristic tool of EDM to discover such structures.
- Filtering out the attributes irrelevant for the task at hand is key to deal with high-dimensionality. *Feature Selection* [GE03] methods are critical for **Data Selection** and heavily rely on dependency estimators [PLD05; HH03; BHS15; Liu+09].
- **Data Transformation**, also known as *Feature Engineering*, consists in transforming the data into an adequate format, and possibly enhancing the original data with additional features. Dependency estimates can be used as feature to improve the results of downstream analysis [ZS02; Tor03; YH09; Vol+19a].
- Numerous **Data Mining** algorithms, e.g., clustering [Ngu+13; NMB13] or outlier detection [KMB12; TB19] algorithms, intrinsically rely on dependency estimation.
- Finally, dependency estimation also helps for **Pattern Evaluation** and **Knowledge Representation**. For example, [Geb+14; Geb+18; Vol+19b] leverage *mutual information* to evaluate and represent rules learned from data.

Also, dependency estimation is essential in high-dimensional data streams, because most tasks are *unsupervised*, i.e., the nature of objects (e.g., inlier/outlier) may be unknown or revealed only with some latency [Dit+15]. Such tasks typically boil down to discovering structures in data, which in turns often relies on dependency estimation.

However, there are – to our knowledge – no dependency estimation technique suitable for high-dimensional data streams. The existing methods are impractical, because they either are inefficient or restricted to the bivariate case.

Next, there exists no efficient solution to maintain an overview of statistics (e.g., dependency estimates) concerning numerous subspaces. One often has no choice but to recompute the measures of interest periodically, which can be very expensive. While there exist a few monitoring techniques [ZS02; BG07], they can either (1) only monitor a single statistic or (2) only support a specific estimator, and do not generalise beyond.

Data Mining algorithms currently are unable to leverage dependency estimation in high-dimensional data streams. Therefore, developing new methods for dependency estimation and monitoring in this setting is much needed.

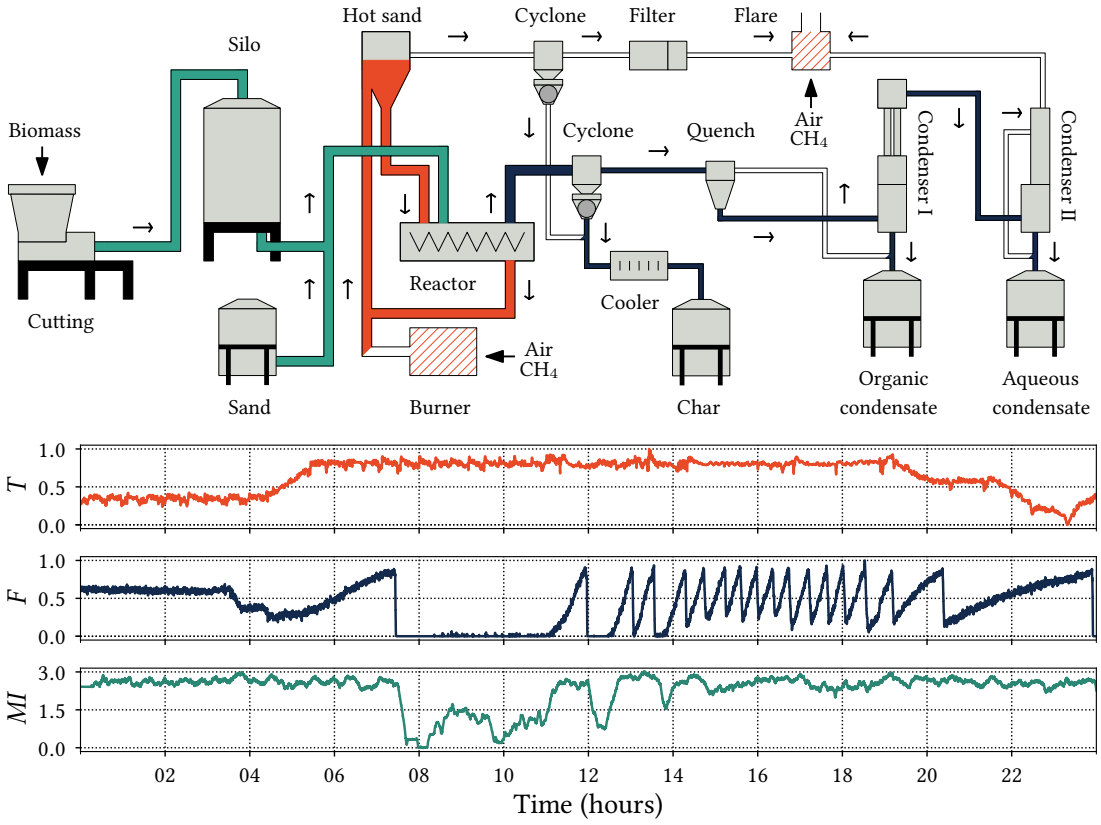


Figure 1.5.: Schematics of the Bioliq fast pyrolysis and monitoring example.

1.1.5. The Bioliq® Power Plant: A Real-World Use Case

Throughout this dissertation, we illustrate our results against a real-world use case: the Bioliq power plant. In a nutshell, the goal of the Biomass-to-Liquid (Bioliq) project is to develop the process chain for producing fuels from biomass at an industrial scale. The Bioliq plant provides real-world data that we use to motivate and validate our research.

We focus in particular on the first part of the Bioliq process: the fast pyrolysis, which takes place in the 'Bioliq I' pilot plant in the surroundings of Karlsruhe [Pfi+16]. We show in Figure 1.5 a simplified representation of the process. The following example illustrates the importance of monitoring statistics, such as dependency estimates, at Bioliq:

Example 1.1 (Monitoring at Bioliq)**.** *Let us observe a 24-hours measurement period of two sensors at Bioliq: T is the temperature in the reactor, and F is the filling level of the flue gas cyclone connected to its output. We report the Mutual Information (MI) between T and F at any time over the last 15 minutes. In the beginning, the reactor heats up to its operational temperature. The material introduced leads to the exhaustion of flue gas, stored temporarily in the cyclone for further processing. The MI between the two streams suddenly drops from 2.5 bits to 0 at 7:45. The cyclone does not seem to operate as it should, i.e., as in the later timespan between 12:00 and 20:00. The data indicates an interruption in production. Such interruptions can become very costly if unnoticed. Thus, the careful monitoring of plant elements is essential, as drifting dependencies might indicate abnormal events [Has11].*

1.2. Contributions

The goal of this dissertation is to address the challenges of Data Mining w.r.t. high-dimensionality and streaming data. To this end, we first focus on fundamental problems of Data Mining: estimating dependency and monitoring. Then, we show how our new methods help to address Knowledge Discovery in high-dimensional data streams. We organise our contributions around the three following research questions:

- **Q1 (Estimating Dependency):** The increasing number of dimensions and observations challenge dependency estimation. How to estimate dependency in high-dimensional data streams efficiently and effectively?
- **Q2 (Monitoring):** By nature, data streams are infinite and may evolve, so the relationship between variables might change. How can we keep track of the evolution of statistics (e.g., dependency estimates) in streams with high-dimensionality?
- **Q3 (Knowledge Discovery):** We see the answer to **Q1** and **Q2** as fundamental to extract knowledge from high-dimensional streams. Thus, we ask: how to leverage our contributions to mine patterns (e.g., anomalies) in this setting?

To deal with **Q1**, we propose a list of desirable features an estimator must ideally fulfil for high-dimensional data streams. Then, we introduce Monte Carlo Dependency Estimation (MCDE), a framework that quantifies multivariate dependency as the average statistical discrepancy between marginal and conditional distributions, via Monte Carlo (MC) simulations. MCDE handles heterogeneity by leveraging three statistical tests: the Mann-Withney U, the Kolmogorov-Smirnov and the Chi-Squared test. We determine a lower bound for the quality of our estimates, which only depends on the number of MC simulations. Such bound allows users to trade estimation accuracy for a computational advantage. We demonstrate that MCDE goes beyond the current state of the art regarding dependency estimation by meeting all the features defined previously. Finally, we show against our real-word use case (Bioliq) that MCDE can discover useful patterns in high-dimensional data streams. We recently published those results in [FB19; Fou+20a].

Concerning **Q2**, we propose to formulate the problem of monitoring statistics in high-dimensional data streams as a Multi-Armed Bandit (MAB) problem. We find that our setting requires to extend the existing models, and call our extension the Scaling Multi-Armed Bandit (S-MAB). We propose a new algorithm based on Thompson Sampling [Tho33] (TS), with strong theoretical guarantees and excellent empirical performance. Furthermore, we combine our algorithm with Adaptive Windowing [BG07] (ADWIN), a state-of-the-art change detector, to deal with non-static environments. We illustrate the benefits of our contribution using synthetic data, as well as data from our real-world use case. We published our findings in [FKB19].

Finally, we address **Q3** by exploiting synergies between our previous contributions. We show that, by combining the ideas behind MCDE and S-MAB, we can facilitate the search for subspaces in data streams. We introduce a new algorithm, Streaming Greedy Maximum Random Deviation (SGMRD), and show that SGMRD leads to state-of-the-art performance for Knowledge Discovery tasks, such as outlier detection. Then, we propose a new method, kj-Nearest Neighbours (kj-NN), to detect outlying documents within large text corpora. Those contributions are featured here [Fou+20b; FKB20].

1.3. Outline

We structure the rest of this dissertation as follows:

- Chapter 2 is a joint introduction to our contributions w.r.t. **Q1**, **Q2** and **Q3**. We introduce a set of desirable features for dependency estimators in high-dimensional data streams and formulate the problem of monitoring in high-dimensional data streams as a Multi-Armed Bandit (MAB) problem. Then, we discuss our applications to Knowledge Discovery in high-dimensional data streams.
- Chapter 3 presents the related work concerning each of our contributions.
- Part II focuses on **Q1**. In Chapter 4, we present a new framework for dependency estimation: Monte Carlo Dependency Estimation (MCDE).
- Part III answers **Q2**. Chapter 5 introduces the Scaling Multi-Armed Bandit (S-MAB) algorithms, our solution to monitor high-dimensional data streams.
- Part IV deals with **Q3**. Chapter 6 introduces a general method for Subspace Search in Data Streams, named Streaming Greedy Maximum Random Deviation (SGMRD). Chapter 7 presents a new algorithm, kj-Nearest Neighbours (kj-NN), to detect outlying documents within large text corpora.
- Part V concludes by summarising the outcome of this dissertation in Chapter 8, while Chapter 9 provides an outlook on future work and research directions.

The content relevant to this dissertation extends beyond the scope of this document. Readers may find complementary information about Bioliq in the literature [Pfi+16] and the official website of the Bioliq project:

- Bioliq: `https://www.bioliq.de`

Furthermore, we systematically release our source code, data and experiments via open-source platforms, under the GNU Affero General Public License version 3 (AGPLv3):

- MCDE: `https://github.com/edouardfouche/MCDE-experiments`; `https://github.com/edouardfouche/MCDE`; `https://github.com/edouardfouche/MCDE-extended`
- S-MAB: `https://github.com/edouardfouche/S-MAB`
- SGMRD: `https://github.com/edouardfouche/SGMRD`
- kj-NN: `https://github.com/edouardfouche/MiningTextOutliers`

See also the related entries on the author's website (`https://edouardfouche.com`) and the sources of this document (`https://github.com/edouardfouche/phd-thesis`), under the Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0).

2. Preliminaries

This chapter is a joint introduction to our core contributions. We describe the underlying motivations and the notation for our work.

2.1. Estimating Dependency

2.1.1. Motivations

The discovery of relationships between attributes is fundamental to many Data Mining applications, e.g., Feature Selection [PLD05], Clustering [Ngu+13] or Outlier Detection [KMB12], and it is a prominent topic in the database community [CHY96; ZS02; HH03].

One typically estimates the strength of a relationship by estimating the 'dependence' among the attributes of a subspace. To do so, one often leverages well-known 'dependency estimators' such as the Pearson correlation coefficient (also known as *correlation*), or Mutual Information (MI) [Sha48] as a non-linear counterpart.

Dependency estimation not only plays a central role in Data Mining – as we discussed in Section 1.1.4 – but also delivers useful information per se: knowing the relationship between attributes helps to predict and understand certain outcomes.

For instance, knowing that weight and arterial pressure correlate with the odds of contracting certain diseases may guide physicians, when predicting whether a patient will become sick within a year or not. Dependency may also reflect natural physio-chemical relationships, say, between the temperature and pressure of a fluid in a pipe. When dependency changes, it either means that the system's state is transitioning, e.g., the fluid solidifies, or that equipment deteriorates, e.g., there is a leak. Fluctuations of dependency often reflect changes in the stream, i.e., the dependency matrix delivers information about the state of a system. This applies, for example, to the Bioliq plant (Figure 1.5).

However, concerning real-world settings, dependency estimation remains mostly unaddressed: data often comes as an open-ended, ever-evolving stream of sensor signals. The signals can be noisy, redundant or generated at a varying speed. In this setting, the timely detection of changes in the stream is crucial; the early discovery of anomalies can, say, facilitate predictive maintenance and yield tremendous cost savings.

Also, most dependency estimators only deal with numerical data, while the stream often consists of measurements or indicators of various types, e.g., numerical, ordinal, or categorical observations. Such data sources are known as *Heterogeneous Data Streams* [YZ06]. In Section 1.1.3, we reviewed the constraints associated with data streams. Naturally, addressing those constraints is a prerequisite for any algorithm operating on streams.

With this in mind, and orthogonally to the constraints of the streaming setting, modern dependency estimators also have their own set of requirements – or desirable features – that they ideally must fulfil. We describe those requirements hereafter.

2.1.2. Requirements

Based on our observation of the current state of the art, our first contribution is to establish the following set of requirements, that any dependency estimator must ideally satisfy:

- **Multivariate (R1):** Bivariate measures only apply to two entities (i.e., variables, vectors). Estimating the dependency between more than two entities is useful as well, but existing attempts to generalise bivariate measures lack efficiency or effectiveness.
- **General-purpose (R2):** Estimators should not restrict to specific types of dependencies. Otherwise, they may miss relevant attribute relationships. Existing multivariate estimators are typically limited to, say, monotonous or functional dependencies.
- **Intuitive (R3):** A method is intuitive if its parameters are easy to set, i.e., users understand their impact on the estimation. Existing solutions tend to have unintuitive parameters, and the suggestion of 'good' parameter values happens (or does not happen) at the discretion of the inventors.
- **Non-parametric (R4):** Since real data can exhibit virtually any kind of distribution, it is not reasonable to use measures relying on parametric assumptions. The risk is to miss relevant effects systematically.
- **Interpretable (R5):** The results of dependency estimators should be interpretable. In particular, the returned estimate should have a maximum and a minimum, so that one can interpret and compare two given estimates.
- **Sensitive (R6):** Dependency estimation is not only about detecting the existence of a relationship, but also about quantifying its strength. Data points generally are observations sampled from a potentially noisy process. The same dependency should get a higher score when observed with more observations, as the size of the observed effect – the 'effect size' – is larger.
- **Robust (R7):** Real-world data may be of poor quality. Measuring devices often have limited precision, so that values are rounded or trimmed, leading to points with the same values. It is also common to discretise attributes, for a more compact representation. Such artefacts can have a negative influence on the estimation. Thus, estimators need to be robust against duplicates and imprecision.

To the best of our knowledge, any existing solution only fulfils some of those requirements at best. For example, the Pearson correlation coefficient is parametric (**R4**), targets at linear dependencies (**R2**) and is only applicable to numerical data (**C5**).

There exist alternative criteria to compare dependency estimators. For example, [Rén59] propose a set of rules that dependency estimators should satisfy. However, there is no standard benchmark and, for most estimators, whether they meet those rules or not is unknown. [Res+11] propose to evaluate measures against a notion of 'equitability'. However, there is so far no commonly accepted formalisation of this notion [KA14; MMM14]. In contrast, we introduce a set of pragmatic characteristics, focusing on the real-world requirements of dependency estimators in high-dimensional data streams.

In this dissertation, we propose a framework, Monte Carlo Dependency Estimation (MCDE), which features these characteristics and compare MCDE via systematic experiments with the existing competitors. In the next section, we introduce our notation.

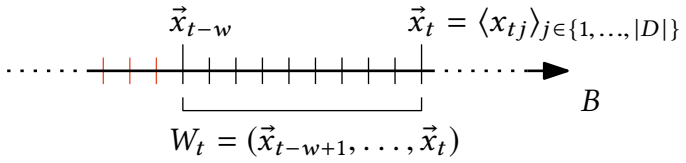


Figure 2.1.: The sliding window contains the w latest observations (here, $w = 10$).

2.1.3. Notation

A data stream is a set of attributes $D = \{s_1, \ldots, s_{|D|}\}$ and an open list of observations $B = (\vec{x}_1, \vec{x}_2, \ldots)$, where $\vec{x}_i = \langle x_{ij} \rangle_{j \in \{1,\ldots,|D|\}}$ is a vector of values with $|D|$ attributes, and we see an attribute $s_i = (x_{1i}, x_{2i}, \ldots)$, with $i \in \{1, \ldots, |D|\}$, as an open list of values.

Since the stream is virtually infinite, we use the sliding window model: At any time $t > 1$, we only keep the w latest observations, $W_t = (\vec{x}_{t-w+1}, \ldots, \vec{x}_t)$. We assume, without loss of generality, that observations arrive at equidistant time steps. Note that one could easily adapt our methods to accommodate other summarisation techniques, such as the landmark window or reservoir sampling [Gam12]. Figure 2.1 illustrates our notation.

We call a subspace S a projection of the window W_t on $|S|$ attributes, with $S \subseteq D$ and $|S| \leq |D|$. We treat an attribute $s_i \in D$ as a random variable X_{s_i}. To address *heterogeneity*, we also make the distinction between numerical, ordinal and categorical attributes:

- We say that s_i is of numerical type ($s_i \in Num$) if one can see X_{s_i} as a continuous variable on a given interval.
- We say that s_i is of ordinal type ($s_i \in Ord$) if one can see X_{s_i} as a discrete variable, i.e., it can take a finite number of ordered values.
- We say that s_i is of categorical type ($s_i \in Cat$) if one can see X_{s_i} as a categorical variable, with a fixed amount of nominal categories.

Naturally, knowing whether a given attribute is numerical, ordinal or categorical requires domain knowledge. Typically, ordinal attributes have many tying values, while values from a numerical attribute are unique, given enough precision. On the other hand, values from categorical attributes might not be numeric and do not have any meaningfully ordering.

Then, $p(X)$ is the joint probability distribution function (*pdf*) of a random vector $X = \langle X_{s_i} \rangle_{s_i \in S}$, and $\hat{p}(X)$ denotes the empirical estimation of this distribution. We use $p_{s_i}(X)$ and $\hat{p}_{s_i}(X)$ for the marginal *pdf* and its estimation for each variable s_i. $\mathcal{P}(S)$ is the power set of S, i.e., the set of all attribute subsets. For any subset $S' \in \mathcal{P}(S)$, its random vector is $X_{S'} = \langle X_{s_i} \rangle_{s_i \in S'}$ and its complement $\overline{X_{S'}} = X_{S \setminus S'} = \langle X_{s_i} \rangle_{s_i \in S \setminus S'}$. In our algorithms, '$\oplus$' and '$\wedge$' stand for concatenation and element-wise logical conjunction.

Dependency estimation determines to which extent a relationship differs from randomness. In this spirit, MCDE quantifies a dependency, i.e., a degree of independence violation, based on marginal and conditional distributions. In Part II, we extensively describe MCDE.

Nonetheless, MCDE only estimates the dependency within a given subspace. In the next section, we abstract from the underlying dependency estimator and consider the problem of monitoring numerous estimates in high-dimensional data streams.

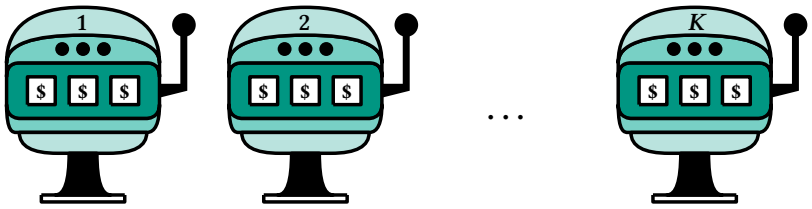


Figure 2.2.: The MAB problem with K arms (one-armed bandits).

2.2. Monitoring

2.2.1. Motivations

Monitoring, i.e., the real-time computation of statistics (such as dependency estimates), is crucial for Knowledge Discovery in data streams. Transient statistics may reveal the current state of manufacturing machines or the ongoing behaviour of financial markets.

However, monitoring is computationally demanding, especially when the number of dimensions and the rate of incoming data increase (cf. Sections 1.1.2 and 1.1.3). In this setting, recomputing the statistics at every time step does not scale. Thus, stream statistics monitoring – for say, high-frequency trading or the monitoring of large factories – is impractical with current methods. At the same time, it can be advantageous. Think for example of monitoring dependency in the Bioliq plant:

Example 2.1 (Dependency Monitoring)**.** *Dependency often results from physical relationships between, say, the temperature and pressure of a fluid. When monitoring the pyrolysis process at Bioliq, it is useful to maintain an overview of dependencies to keep operation costs down. However, continuously updating the complete dependency matrix is impractical with current methods since the data is typically high-dimensional and ever-evolving.*

From a practical point of view, all statistics are not equally interesting. For example, for Knowledge Discovery, one is often interested in high dependency. Our idea is to update only a few elements of the matrix, based on a notion of 'utility', e.g., high dependency values. The system must minimise the cost of monitoring while maximising the total utility. In other words, the system faces a trade-off between exploitation and exploration: Should the system keep updating estimates known to deliver large utility (gain) or use resources to update other estimates, for which the potential utility is not yet well-known?

In this dissertation, we see the problem of monitoring as a generalisation of the Multi-Armed Bandit (MAB) problem. The MAB problem captures the dilemma between exploration and exploitation in sequential decision-making. At every time step, a forecaster selects a set of arms and observes a reward from each arm. The name of the MAB problem finds its origin in the typical trade-off a gambler faces in a casino: given a set of slot machines (see Figure 2.2), a.k.a. 'one-armed bandits', which machine should the gambler play to maximise their expected gains? Should they try different machines (exploration) or keep playing the machine that they believe to be the best so far (exploitation)?

However, the existing MAB formulations do not quite match our problem, as we will see. In what follows, we propose to cast data stream monitoring as a new bandit problem, that we call the Scaling Multi-Armed Bandit (S-MAB).

2.2.2. Monitoring as a Bandit Problem

In the classical MAB problem, a forecaster must choose one of K arms at each round, and playing it yields a reward. Their goal is to maximise the cumulative reward over time. In a variant of the problem, known as the Multiple-Play Multi-Armed Bandit (MP-MAB) [AVW87; KHN15], the forecaster must choose L distinct arms per round, where L is an exogenous parameter. However, while it is relevant for some applications (e.g., web content optimisation), adequately setting L is not easy when monitoring streams.

Thus, we consider a new variant of the MP-MAB where the forecaster not only must choose the best arms but also must 'scale' L, i.e., change the number of plays, to maximise the rewards and minimise the costs. By doing so, the forecaster controls the efficiency of playing. We name this setting the Scaling Multi-Armed Bandit (S-MAB) problem.

Think of a new casino game, which we call the 'blind roulette': the player places bets on distinct numbers, and each number has an independent but unknown probability of being drawn. Bets correspond to a fixed amount, e.g., one can only bet 1$ on a number or nothing. In each round, the player must decide how many bets to place, and on which numbers. The casino then reveals to the player which ones of their bets were successful and pays the corresponding reward. To make the game more challenging, the casino may sometimes change the underlying probability of each number without notice.

While placing a few but confident bets may seem to be an economically efficient option, the absolute gain at the end of the day will not be significant. On the other hand, placing many bets may not be a good strategy, as many numbers typically have a low chance to be drawn. To maximise their gain, the player must place as many bets as possible, as long as their expected gain is higher than the amount bet. Whenever the probabilities change, the player needs to adapt their behaviour; otherwise, they may lose most of their bets and experience much regret w.r.t. an optimal (but unknown) strategy.

This game matches not only dependency monitoring (cf. Example 2.1) but also many other real-world applications, such as the placement of online advertisements or the investment in financial portfolios. The S-MAB problem introduces a new trade-off: one wants to maximise the reward, but at the same time minimise the cost of each round/observation. The problem consists of the following challenges:

- ***1: Top-arms Identification.** To maximise the reward from L plays, one needs to find the L arms with the highest expected reward; This is the traditional exploration-exploitation dilemma known from the MP-MAB problem.
- ***2: Scale Decision.** One should not play more arms than necessary: playing many arms leads to high costs, but playing only a few arms leads to low rewards. One should set L to control the efficiency, i.e., the ratio of the rewards to the costs.
- ***3: Change Adaptation.** The environment can either be static or non-static. In the second case, one needs to 'forget' prior knowledge whenever a change occurs. Forgetting that is too aggressive or too conservative leads to suboptimal results.

In the next section, we formalise the S-MAB problem considering those challenges[1]. We use the most common notation from the bandit literature, e.g., as in [BC12].

[1] See also `https://youtu.be/wVogcI3fr7Q` for a 3-minute introduction to this problem.

2.2.3. Problem Definition

Let there be K arms. We associate each arm $i \in [K] = \{1, \ldots, K\}$ with an unknown probability distribution v_i with mean μ_i. At each round $t = 1, \ldots, T$, the forecaster selects arms $I(t) \subset [K]$, then receives a reward vector $X(t)$. $L_t \leq K$ is the number of these arms. The rewards $X_i(t) \in X(t)$ of each arm i are i.i.d. samples from v_i. We make the classical assumption from bandit analysis that the rewards $X_i(t)$ are 0 or 1, i.e., the distribution of rewards from arm $i \in [K]$ follows a Bernoulli distribution with mean μ_i. Selecting an arm $i \in [K]$ leads to a unit cost 1, where cost and reward do not need to have the same unit. Note that it is not very difficult to generalise our results to other reward distributions, as long as they are bounded.

Let $N_i(t)$ and $S_i(t)$ be the number of draws of arm i and the sum of the rewards obtained from it respectively before round t. Let $\hat{\mu}_i(t) = S_i(t)/N_i(t)$ be the empirical estimation of μ_i at time t. The forecaster is interested in maximising the sum of the rewards over arms drawn, under the constraint that the sum of the rewards must be greater than the sum of the costs by an efficiency factor η^*. The parameter $\eta^* \in [0, 1]$ controls the trade-off between the cost of playing and the reward obtained, which is application-dependent.

Let us think of our 'blind roulette' metaphor and assume that, whenever a bet is successful, the casino awards the double of the bet. Then, for positive expectations, the player must set $\eta^* > 0.5$ and control η_t, the admitted cost per arm, to be higher than η^*. Thus, at each step t, the forecaster is facing the following constrained optimisation problem:

$$\max_{I(t)\subset[K]} \sum_{i\in I(t)} S_i(t) \quad s.t. \quad \eta_t = \frac{\sum_{i\in I(t)} \mu_i}{L_t} > \eta^* \tag{2.1}$$

The difficulty here is that the forecaster does not know μ_i, but only has access to an estimate $\hat{\mu}_i$ from previous observations.

$\sum_{i\in I(t)} S_i(t)$ is maximised when the forecaster chooses the arms with the highest expectation μ_i. For simplicity, we assume that all arms have distinct expectations (i.e., $\mu_i \neq \mu_j, \forall i \neq j$) and we assume without loss of generality that $\mu_1 > \mu_2 > \cdots > \mu_K$, and thus $[L_t]$ is the top-L_t arms. Under the assumption that the forecaster always chooses $[L_t]$, the value of η_t is only determined by L_t, i.e., Eq. (2.1) is equivalent to finding the optimal number of plays L^*:

$$L^* = \max_{1\leq L\leq K} L \quad s.t. \quad \frac{\sum_{i=1}^{L} \mu_i}{L} > \eta^* \tag{2.2}$$

Thus, the correct identification of the top-L_t arms (*1) is sine qua non to find the optimal number of plays L^* (*2). Next, in non-static environments, the expected rewards may change, i.e., $\mu_i : t \mapsto [0, 1]$ becomes a function of t, as does L^*. So the forecaster must adapt its estimation $\hat{\mu}_i$ (*2) to correctly select the arms with the highest reward, i.e., it needs to discard outdated observations.

In Part III, we present algorithms that solve the S-MAB problem and those challenges. Using our contribution, one can monitor dependency in data streams very efficiently. In the next section, we start to reflect on the implications of those contributions w.r.t. Knowledge Discovery in high-dimensional data streams.

2.3. Knowledge Discovery

Part IV shows the impact of our contributions towards a higher-level goal: Knowledge Discovery in high-dimensional data streams. Chapter 6 shows that combining our contributions from Part II and III, namely MCDE and S-MAB, helps to transfer the task of subspace search to the streaming setting. We show that the efficient monitoring of subspaces of interest helps with outlier detection in high-dimensional data streams. Then, we deal in Chapter 7 with a real-world use case: the discovery of text outliers from large text corpora and propose a new method to detect such outliers.

2.3.1. Subspace Search in Data Streams

Analysing high-dimensional data is notoriously difficult (cf. Section 1.1.2). As we discussed in Section 1.1.4, a fundamental task of Data Mining is to quantify the dependence between attributes. With this in mind, researchers have proposed *subspace search* methods, mainly for static data, to find interesting low-dimensional projections. Such projections tend to much structure, i.e., high dependence among their dimensions. Subspace search is state-of-the-art to deal with data of high dimensionality and has numerous applications, including Exploratory Data Mining [Ass+07; Tat+12], outlier detection [ZGW08; KMB12; TB19], or clustering [Pro+02; Kai+03; Bau+04; PL07; ZLW07].

One can see subspace search as an ensemble *feature selection* method [GE03], as the goal is to find several projections (subspaces), not just one. The underlying assumption of subspace search is that patterns (e.g., outliers, clusters) may hide in various subspaces [Agg13], and that, when restricting the search to a single subspace, one may miss some patterns. In a nutshell, existing subspace search methods consist of two building blocks:

(1) A *quality measure* to quantify the 'interestingness' of a subspace, i.e., the potential to reveal patterns. Intuitively, subspaces with 'structure' are more likely to contain outliers or clusters [Kel15]. That measure often is a multivariate measure of dependence.

(2) A *search scheme* to explore the set of subspaces. Since this set grows exponentially with dimensionality, inspecting every subspace is not possible, and the search typically is a heuristic, i.e., a trade-off between completeness of the search and result quality.

These two items tend to be specific for a given Data Mining algorithm, e.g., a certain clustering method. The search then is helpful for this particular algorithm, but does not generalise beyond [Tat+12]. Next, existing approaches for subspace search tend to assume that the data is static. A straightforward generalisation to streams – i.e., repeating the search periodically – is computationally expensive and limited by the speed of new observations arriving.

In Chapter 6, we exploit synergies between our contributions [FB19; FKB19; Fou+20a] to facilitate subspace search in streams by fulfilling the constraints of this setting (see Section 1.1.3). The core idea of our method, Streaming Greedy Maximum Random Deviation (SGMRD), is to maintain a set of high-quality subspaces over time, by updating subspace-search results continuously. We show that it is advantageous to detect patterns (e.g., outliers) in high-dimensional data streams, and that existing approaches are much less efficient. To describe our approach, we use the same notation as in Section 2.1.3.

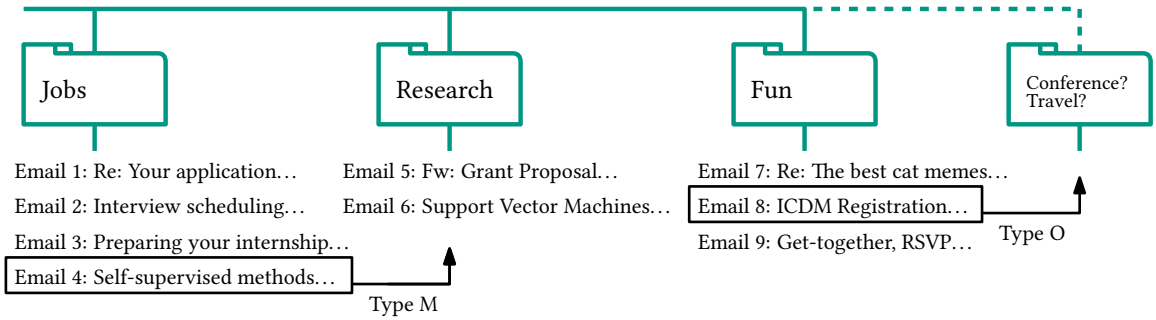


Figure 2.3.: Email Archiving: An Illustrative Example.

2.3.2. Mining Text Outliers

2.3.2.1. Motivations

Recent technological developments have led to an ever-growing number of applications producing, sharing and managing text data. Document repositories, such as email accounts, digital libraries or medical archives, have grown so large that it is now a necessity to classify elements into categories, e.g., folders.

This is far from trivial, as text corpora tend to be highly multi-modal, i.e., there are many classes/folders. Documents also may have ambiguous semantics or more than one semantic focus, so that they do not perfectly fit into just one class.

Humans can order content to some extent and routinely do so: Users organise their emails into folders, authors classify their contributions into existing taxonomies, and physicians issue diagnoses as part of their duties. However, human classification is inherently sloppy and unreliable. Humans may classify an email into an inadequate folder, assign scientific papers to the wrong field, or – even more critical – issue a wrong diagnosis.

Sometimes one may not even notice these errors because the correct class is unknown. For instance, an email does not fit into any existing folder, a paper belongs to an emerging field, or a physician observes a new disease.

Detecting documents classified erroneously is difficult [FV14; GLH15]. This is because folder structures typically are domain-specific and are user-defined taxonomies. At the same time, documents can be misplaced in numerous, unforeseeable ways, which may or may not be folder-specific. So seeing this problem as a supervised one – where a ground truth is available – would be inadequate. Next, orthogonally to this 'semantic' level, two types of errors/outliers can occur: **(O) Out-of-distribution**: A document does not belong to any existing folder — the user should create a new class. **(M) Misclassification**: A document belongs to another folder in the directory — the user misclassified it.

We illustrate this in Figure 2.3, with (fictitious) emails ordered into folders by a user: Email 4 is a Type M outlier, as it belongs to the folder 'Research'. Email 8 in turn is a Type O outlier, because it does not fit into any existing folder — we should create a new one. Intuitively, a document is a Type O outlier when it does not appear to be similar to documents of any single class. In contrast, a document is a Type M outlier when it appears to be most similar to documents from another class.

In this work, we focus on mining text outliers in document directories. It is difficult because documents are outlying w.r.t. their semantics, which is not easy to capture (even for humans). An additional issue, which makes this contribution unique, is that both outlier types must be detected jointly. Namely, the existence of one type harms the detection of the other one. The noise introduced by Type M outliers hinders the detection of Type O outliers. Conversely, Type O outliers may be detected as Type M by mistake, leading to a poor treatment of such outliers. Existing methods only deal with one of these outlier types, cf. Section 3.3.2. Interestingly, we show that the joint detection of both outlier types leads to better performance for each type than approaches only dealing with one of them.

To solve this problem, we propose a new approach to detect text outliers, which we name kj-Nearest Neighbours (kj-NN)[2]. Our approach leverages similarities of documents and phrases, based on state-of-the-art embedding methods [Men+19], to detect both Type O/M outliers. By extracting semantically relevant labels and the documents similar to each outlier, it also supports interpretability. Our approach is efficient and robust to large proportions of erroneous labels thanks to a 'self-supervision' mechanism, which estimates the relevance of the original labels. Our experiments show that our approach improves the current state of the art by a large margin while delivering interpretable results. In the next section, we introduce our notation.

2.3.2.2. Notation

Let $Doc = \{d_1, d_2, \ldots, d_{|Doc|}\}$ and $Phr = \{p_1, p_2, \ldots, p_{|Phr|}\}$ be a set of documents and phrases. We define $O = Doc \cup Phr$ as the set of all text objects.

A common practice in the community is to project each text object in an n-dimensional embedding space as a preprocessing step. In this paper, we use a recent technique [Men+19] capable of projecting both words and documents into the same embedding space, i.e., each object o has an embedding vector representation $V : O \mapsto \mathbb{R}^n$ with n dimensions, with typically $n \geq 100$. The vectors are all normalised, thus $\|V(o)\| = 1,\ \forall o \in O$.

We quantify the similarity between a pair of phrases, documents or phrases/documents with a function $Sim : O^2 \mapsto [0, 1]$ where 1 is the highest possible similarity (identity) and 0 is the lowest one. In our setting, Sim is the normalised cosine similarity:

$$Sim(V(o_i), V(o_j)) = \frac{V(o_i) \cdot V(o_j) + 1}{2} \quad \forall (o_i, o_j) \in O^2 \tag{2.3}$$

For simplicity, we equivalently refer to the vectorial representation of each object, i.e., $o \equiv V(o)$, in what follows. We also assume that there exists an initial classification of documents into a set of classes $C = \{c_1, \ldots, c_{|C|}\}$, expressed as a function $y : Doc \mapsto C$.

Our self-supervision mechanism relies on estimating the *representativeness* of each phrase $p \in Phr$ w.r.t. each class $c \in C$. We denote it as a function $r : Phr \times C \mapsto \mathbb{R}^+$.

In Chapter 7, we extensively describe our approach and experiments. Note that for this study, we restricted our experiments to static text repositories. Nonetheless, our other contributions facilitate the extension of our approach to streams of text, e.g., news or twitter feeds. In that respect, one may see Chapter 7 as preliminary work. Applying our methods for Knowledge Discovery in streams of text is future work (cf. Chapter 9).

[2] See also https://youtu.be/6dl3ZBxB3f0 for a 18-minute talk about our contribution.

3. Related Work

This chapter reviews the related work for each of our contributions.

3.1. Estimating Dependency

Estimating the dependency, or 'correlation', between two or more variables, is a fundamental topic in data analysis and has motivated research for more than a century. Many bivariate measures exist, e.g., [Spe04; Spe04; Ken38; Res+11]. Some of them also target at quantifying the association between two vectors which are possibly multivariate [Gre+07; SR09; LHS13; Bel+18]. However, they can only quantify the dependency between two entities – not between several ones (**R1**). They also may have other drawbacks. The Pearson correlation coefficient, for instance, is parametric (**R4**), targets at linear dependencies (**R2**) and is only applicable to numerical data (**C5**).

There are attempts to extend bivariate dependency measures to the multivariate case. For example, there exists an extension of Spearman's ρ to multidimensional data (Multivariate Spearman [SS07] (MS)), but it is limited to monotonous relationships (**R2**). Several authors also propose multivariate extensions of Mutual Information (MI) [Tim+14]. For example, Interaction Information [McG54] (II) [McG54] quantifies the 'synergy' or 'redundancy' in a set of variables. Similarly, Total Correlation [Wat60] (TC) [Wat60] quantifies the total amount of information. However, information-theoretic measures are difficult to estimate, as they require knowledge about the underlying probability distributions. Density estimation methods, either based on kernels, histograms or local densities, all require setting unintuitive parameters (**R3**) and may be computationally expensive (**C1**). Next, with many attributes, density estimation becomes meaningless due to the *curse of dimensionality* [Bel57]. Information-theoretic measures also are difficult to interpret (**R5**), since they usually correspond to a number of bits or nats, which is theoretically unbounded.

More recently, Cumulative Mutual Information [Ngu+13] (CMI), Multivariate Maximal Correlation [Ngu+14b] (MAC), Universal Dependency Score [NMV16] (UDS), Unbiased Multivariate Correlation [Wan+17] (UMC) and Intrinsic Dimensionality Dependency [Rom+16] (IID) were proposed as multivariate dependency measures. They are remotely related to concepts from information theory, as they rely on the so-called Cumulative Entropy [CL09] (CE). However, these measures are computationally expensive (**C1**) and unintuitive (**R3**). They also are difficult to interpret, because their theoretical maximum and minimum vary with the number of attributes (**R5**).

Another approach, High Contrast Subspaces [KMB12] (HiCS), is somewhat similar to ours, Monte Carlo Dependency Estimation (MCDE). It uses *subspace slicing* as a heuristic to quantify the potential of subspaces to contain outliers. Yet HiCS only addresses static numerical data, and its suitability as a dependency estimator is not known.

Also, most dependency estimators are designed to deal with numerical data only, and one assumes that all relevant observations are available during estimation. In the real world, however, data often consists of an open-ended, ever-evolving stream of measurements or indicators of various types, e.g., numerical, ordinal, or categorical observations.

The current state of the art to handle heterogeneity (C5) is to rely on discretisation, using methods such as the one proposed by [FI93]. Then one can compute an information-theoretic measure, as in [Ngu+14a]. However, any discretisation essentially results in an information loss and may not work as dimensionality increases.

A recent line of work focuses on estimating MI on numerical data streams. Mutual Information Stream Estimation [KMB15] (MISE) is a data summarisation technique to estimate MI over arbitrary time windows. [VRB18] provide dynamic data structures to maintain MI over a sliding window. [VB19] extend this method to propose an anytime estimator for MI, with confidence bounds. However, the resulting estimates inherit the qualities and caveats from MI.

3.2. Monitoring

We refer to 'monitoring' as the continuous surveillance of statistics in a multidimensional, potentially high-dimensional data stream. The existing approaches for monitoring statistics in streams fall into two classes: incremental and approximation schemes.

Incremental Schemes: One can monitor statistics on streams incrementally via a forgetting mechanism, to discard past observations. The approaches are usually based on a sliding window [BG07; VRB18] or a decaying factor [Koy00; Kli04; SG18]. However, not every statistic can be computed incrementally, and the schemes only handle the computation of a single statistic, not of multiple ones.

Approximation Schemes: Another line of research is approximating the statistics via sampling strategies [BDM02; BH17; ABR19] or data transformations, such as Fourier [ZS02; Sel+14] or wavelet [Cha+01; GH05] transform. Other approaches explicitly target at high dependency [Kar+06; ZX08]. However, most methods work only for specific statistics, e.g., Pearson's correlation, which limits their applicability to linear dependencies.

Work on change detection also is related, as detecting whenever a change occurs in the stream helps with monitoring. [NV16], [Rei+16] or [Frí+15] all aim to detect sudden changes. However, these approaches do not apply to stream data or require labels, which make them unsuitable for the high-dimensional streaming setting in general and further applications, e.g., predictive maintenance.

Our approach, Scaling Multi-Armed Bandit (S-MAB), does not fall into these categories, since we target at monitoring virtually any statistics; the estimates are exact, but our algorithm decides when to update them. To do that, we show that our method can also leverage change detectors, such as Adaptive Windowing [BG07] (ADWIN). Thus, our contribution is orthogonal to the existing work. For a broader overview of stream monitoring methods, see [Gam10].

Since our approach bases on bandit theory, the existing bandit models also are related. Work on bandits traces back to [Tho33], with the design of clinical trials. The theoretical guarantees of bandits remained unknown until recently [Aue+00; ACF02; GM08; KKM12].

Our work builds on several facets of bandits, which have been studied separately, such as anytime bandits [Kle06; DP16], Multiple-Play Multi-Armed Bandit (MP-MAB) [AVW87; UNK10; KHN15] and bandits in non-static environments [Aue+02; SU08; GM11].

One can see the S-MAB as an extension of the MP-MAB, with the novelty that the player must control the number of plays over time. In particular, we build on the work from [KHN15]. It shows that Multiple-Play Thompson Sampling [KHN15] (MP-TS) has optimal regret while being efficient. We compare our results with other multiple-play models, such as variants of the celebrated Upper Confidence Bound [Aue+02] (UCB) and the Exponential-weight algorithm for Exploration and Exploitation [Aue+00] (Exp3), namely Combinatorial UCB [Che+16] (CUCB), Multiple-Play Kullback-Leibler UCB [GC11; KHN15] (MP-KL-UCB) and Exp3 with Multiple plays [UNK10] (Exp3.M).

Our problem is different from the profitable bandits [ACG18] since they aim at maximising a static notion of profit – as opposed to efficiency – which boils down to finding the individual arms for which the rewards exceed the costs in expectation. Moreover, our problem is more challenging than the MP-MAB and its extension called Combinatorial MAB (CMAB) in that we are interested in a set of arms where the model parameters μ_i satisfy an efficiency constraint (see Eq. (2.1)), and the algorithm needs to estimate them.

The S-MAB also is related to the budgeted MAB model [Tra+10; Xia+16], because it aims at maximising a notion of efficiency, i.e., the ratio of the reward to the cost of playing arms. In our case, the total number of plays – the 'budget' – is not an external constraint. Instead, the S-MAB decides how many arms to play based on its observations of the environment.

Bandits have readily been applied to many real-world applications, such as packet routing [AK04], online advertising [Cha+08], recommendation systems [Li+10], robotic planning [SRL14] and resource allocation [Li+18]. Nonetheless, the application of bandits to monitoring (see Example 2.1) has received much less attention. For an overview of bandit algorithms, we refer the reader to recent surveys [BC12; BLL15; LS20].

3.3. Knowledge Discovery

Because of the unsupervised nature of tasks in high-dimensional streams, knowledge discovery mainly is limited to two classes of applications: outlier/anomaly detection and clustering. As mentioned in recent studies [Kri+11; Nto+12], little effort was devoted so far to the discovery of patterns in high-dimensional data streams. Most contributions focus instead on only one of the two aspects: high-dimensionality or data streams.

While recent studies, such as [Agg+04; ZGW08; Kri+11; Nto+12], attempt to address high-dimensional data streams, the dimensionality in benchmarks is limited – usually, to less than 50 dimensions. Thus, whether those approaches can scale is not known. Recent surveys [Gup+14; ATS14] provide a good overview of the state-of-the-art methods for Knowledge Discovery in data streams w.r.t. the task of outlier detection and clustering.

While addressing every Knowledge Discovery problem is out of the scope of this dissertation, the fundamental nature of our contributions helps towards this goal. We focus in particular on two Knowledge Discovery tasks: 'Subspace Search in Data Streams' and 'Mining Text Outliers', for which we detail the related work hereafter.

3.3.1. Subspace Search in Data Streams

Many methods for subspace search exist, but almost all of them are either coupled to a specific Data Mining algorithm or are limited to the static setting. For example, various approaches for streams [KPM06; ZLW07; ZGW08; Agg09] only tend to work with a given static algorithm. Other methods in turn [KMB12; Ngu+13; NMB13; NMV16; Wan+17; TB19] decouple the search from the actual task, but none of them can handle streams. The existing work on subspace search mostly focuses on individual applications [PHL04; KKZ09; ZSK12], e.g., clustering or outlier detection, while 'general-purpose' subspace search has received less attention.

To our knowledge, there exist two proposals to extend subspace search to streams in a general way: HCP-StreamMiner [Van+12] and StreamHiCS [Bec16]. But these approaches boil down to a periodic repetition of the procedure in [KMB12] on synopses of the stream. We will see that our method outperforms these approaches. Greedy Maximum Deviation [TB19] (GMD) is the approach most similar to ours. It uses a so-called contrast measure [KMB12] to quantify the interestingness of a given subspace and builds a set of subspaces via a greedy heuristic. However, GMD assumes static data.

Subspace search has been used in the past to improve the results of Data Mining tasks such as outlier detection [ZGW08; KMB12; TB19]. The authors compare their results with full-space static outlier detectors. We perform an analogous evaluation in the streaming setting and compare our results against several baselines and state-of-the-art stream outlier detectors, such as xStream [MLA18] and Randomised Subspace Hashing in Streams [SA18] (RS-Stream). See [Gup+14] for a survey of outlier detection in streams.

3.3.2. Mining Text Outliers

To our knowledge, none of the existing methods handles both outlier types. So we categorise related work into two classes: (1) Type O and (2) Type M outlier detectors.

Type O outlier detectors. Type O is the standard definition of outliers, and detecting such outliers has been studied for decades. Conventional outlier detection approaches typically fall into the following classes: distance-based [KN98; RRS00], neighbour-based [Bre+00; KSZ08], probabilistic-based [KS12; TB99] and subspace-based methods [SA18; LK05; Mül+12; KMB12; Kri+12]. Examples of conventional methods are the well-known Local Outlier Factor [Bre+00] (LOF) or, more recently, Randomised Subspace Hashing [SA16] (RS-Hash). However, textual data is typically extremely sparse, and thus few of the above proposals can detect outlier documents, as they do not model semantics.

There exist a few methods addressing text outliers: [Zhu+17] proposes a generative approach, which models the embedding space as a mixture of von von Mises-Fisher (vMF) distributions. They identify 'outlier regions' that deviate from the majority of the embedded data. TONMF [Kan+17], a Non-negative Matrix Factorisation (NMF) approach, bases on block coordinate descent optimisation. Recently, Ruff et al. proposed Context Vector Data Description [Ruf+19] (CVDD), a one-class classification model leveraging pre-trained language models and a multi-head attention mechanism. However, all of these methods treat outlier detection as a one-class classification problem, i.e., they try to describe an 'abnormal' class and a 'normal' class; none of them addresses Type M outliers.

Type M outlier detectors. Type M outliers represent misclassified text. While this type of outlier is ubiquitous, it has received little attention so far. Few publications try to address the problem directly. Traditional supervised text classification methods, assuming the ground truth to be error-free, have no choice but to tolerate Type M outliers during training. While one can extend the existing supervised document classification models (e.g., [Kim14; Con+17; Zho+16; Lai+15]) to mitigate the effect of Type M outliers, they are in turn not helpful to detect Type O outliers.

As we explained earlier, the existence of both outlier types calls for methods that can detect them simultaneously. In that respect, our method, kj-NN, is the first of its kind. In our experiments, we compare our approach against the methods above and show that we outperform all of them. We refer the reader to [Agg13] for an extensive overview of existing outlier detection methods.

Part II.

Estimating Dependency

4. Monte Carlo Dependency Estimation

The content of this chapter bases on the following publications:

- Edouard Fouché and Klemens Böhm. 'Monte Carlo Dependency Estimation'. In: *SSDBM*. Best Paper Award. ACM, 2019, pp. 13–24. DOI: 10.1145/3335783.3335795
- Edouard Fouché, Alan Mazankiewicz, Florian Kalinke and Klemens Böhm. 'A Framework for Dependency Estimation in Heterogeneous Data Streams'. In: *Distributed and Parallel Databases* (2020). DOI: 10.1007/s10619-020-07295-x

Keywords: Multivariate Statistics; Exploratory Data Analysis; Dependency Estimation

4.1. Chapter Overview

Estimating dependencies from data is a fundamental task of Knowledge Discovery. Identifying the relevant variables leads to a better understanding of data and improves both the runtime and the outcomes of downstream Data Mining tasks. Dependency estimation from static numerical data has received much attention. However, real-world data often occurs as heterogeneous data streams. In this chapter, we make the following contributions:

We present Monte Carlo Dependency Estimation (MCDE), a framework which satisfies both the constraints of heterogeneous data streams and the requirements of dependency estimation. Over a given time window, MCDE estimates the dependency of an attribute set as the average discrepancy between marginal and conditional distributions, via Monte Carlo (MC) simulations. We determine a lower bound for the quality of our estimates, which only depends on the number of MC simulations. Such bound allows users to trade estimation accuracy for a computational advantage.

We explore three instantiations of MCDE, i.e., three new dependency measures, dubbed Mann-Withney-P (MWP), Kolmogorov-Smirnov-P (KSP) and Chi-Squared-P (CSP), which base on the corresponding statistical test. We show that using them in combination allows dealing with heterogeneous data. We describe their implementation and compare them to the existing multivariate methods in our experiments.

We introduce index structures for MWP, KSP, and CSP, to speed up contrast estimation. Our indexes support insertion/deletion operations for efficient estimation in streaming settings, e.g., over a sliding window.

We feature a use case against real-world data from Bioliq (cf. Section 1.1.5) and show how one can leverage MCDE to discover interesting and useful patterns. We release our source code and experiments on GitHub[1,2], with documentation to ensure reproducibility.

[1] https://github.com/edouardfouche/MCDE-experiments
[2] https://github.com/edouardfouche/MCDE-extended

4.2. Theory of MCDE

Dependency estimation determines to which extent a relationship differs from randomness. In this spirit, MCDE quantifies dependence, i.e., a degree of independence violation, based on marginal and conditional distributions. Section 2.1.3 introduced our notation.

4.2.1. Quantifying Dependency via Contrast

A set of variables is *independent* or *uncorrelated* if and only if all variables are pairwise **mutually independent**. By treating the attributes of a subspace as random variables, we can define the independence assumption of a subspace:

Definition 4.1 (Independence Assumption)**.** *The independence assumption $\mathcal{A}$ of a subspace S holds if and only if the random variables $\{X_{s_i} : s_i \in S\}$ are mutually independent, i.e.:*

$$\mathcal{A}(S) \Leftrightarrow p(X) = \prod_{s_i \in S} p_{s_i}(X) \tag{4.1}$$

Under the independence assumption, the joint distribution of subspace S is **expected** to be equal to the product of its marginal distributions. We can define a degree of dependency based on the degree to which $\mathcal{A}$ does not hold:

Definition 4.2 (Degree of Dependency)**.** *The degree of dependency $\mathcal{D}$ of a subspace S is the discrepancy, abbreviated as 'disc', between the **observed** joint distribution $p^o(X)$ and the **expected** joint distribution $p^e(X)$:*

$$\mathcal{D}(S) \equiv disc\left(p^o(X), p^e(X)\right) \tag{4.2}$$

The discrepancy is a random variable. While one can estimate it between two probability distributions, using for instance the Kullback-Leibler divergence, this is not trivial here because $p^o(X)$ and $p^e(X)$ are a priori unknown. We work around this as follows:

Lemma 4.1 (Independence Assumption and Joint Distribution)**.** *The independence assumption $\mathcal{A}$ of subspace S states that the joint distribution for all $S' \subset S$ is equal to its conditional distribution on $S \setminus S'$:*

$$\mathcal{A}(S) \Leftrightarrow p(X_{S'}|\overline{X_{S'}}) = p(X_{S'}) \quad \forall S' \in \mathcal{P}(S) \tag{4.3}$$

Proof of Lemma 4.1. Since all variables in S are mutually independent, for any subspace $S' \in \mathcal{P}(S)$ we also have $p(X_{S'}) = \prod_{s_i \in S'} p_{s_i}(X)$:

$$\begin{aligned}
&\mathcal{A}(S) \Leftrightarrow\ p(X) = \prod_{s_i \in S} p_{s_i}(X) \\
&\mathcal{A}(S) \Leftrightarrow\ p(X) = p(X_{S'}) * \prod_{s_i \in S \setminus S'} p_{s_i}(X) && \forall S' \in \mathcal{P}(S) \\
&\mathcal{A}(S) \Leftrightarrow\ \frac{p(X)}{p(\overline{X_{S'}})} = p(X_{S'}) && \forall S' \in \mathcal{P}(S)
\end{aligned}$$

By the definition of the conditional *pdf*:

$$\mathcal{A}(S) \Leftrightarrow\ p(X_{S'}|\overline{X_{S'}}) = p(X_{S'}) \qquad \forall S' \in \mathcal{P}(S) \qquad \square$$

Lemma 4.1 provides an alternative definition of $\mathcal{A}$. However, it still has issues. First, multivariate density estimation is required to estimate $p(X_{S'})$ and $p(X_{S'}|\overline{X_{S'}})$ with $|S'| \geq 1$. Second, even if one could estimate $p(X_{S'})$ and $p(X_{S'}|\overline{X_{S'}})$, estimating the densities for all $S' \in \mathcal{P}(S)$ is intractable. We instead relax the problem by considering only subspaces with $|S'| = 1$, i.e., we only look at the marginal distribution of single variables.

Definition 4.3 (Relaxed Independence Assumption)**.** *The relaxed independence assumption $\mathcal{A}^*$ of a subspace S states that the marginal distribution $p_{s_i}(X)$ of each variable $s_i \in S$ equals $p_{s_i}(X|\overline{X_{s_i}})$, i.e., the conditional distribution of s_i:*

$$\mathcal{A}^*(S) \Leftrightarrow p_{s_i}(X|\overline{X_{s_i}}) = p_{s_i}(X) \quad \forall s_i \in S$$

Lemma 4.2 (Independence Assumption Relaxation)**.** *$\mathcal{A}(S) \Rightarrow \mathcal{A}^*(S)$, i.e., we can relax $\mathcal{A}$ into $\mathcal{A}^*$ for any subspace S.*

Proof of Lemma 4.2. Using Lemma 4.1:

$$\begin{aligned}
&\mathcal{A}(S) \Leftrightarrow p(X_{S'}|\overline{X_{S'}}) = p(X_{S'}) && \forall S' \in \mathcal{P}(S) \\
&\mathcal{A}(S) \Rightarrow p(X_{S^1}|\overline{X_{S^1}}) = p(X_{S^1}) && \forall S^1 \in \mathcal{P}(S) : |S^1| = 1 \\
&\mathcal{A}(S) \Rightarrow p_{s_i}(X|\overline{X_{s_i}}) = p_{s_i}(X) && \forall s_i \in S
\end{aligned}$$

□

Loosely speaking, the relaxed independence assumption holds if and only if the values of all variables but s_i do not reveal any information on s_i. Next, $\mathcal{A}(S) \Rightarrow \mathcal{A}^*(S)$, then $\neg\mathcal{A}^*(S) \Rightarrow \neg\mathcal{A}(S)$, i.e., showing that $\mathcal{A}^*$ does not hold is sufficient but not necessary to show that $\mathcal{A}$ does not hold. Thus, we can define a relaxed degree of dependency $\mathcal{D}^*$ of a subspace S, namely the discrepancy *disc* of the observed marginal distribution $p^o_{s_i}(X)$ and the expected one $p^e_{s_i}(X)$. Under the relaxed independence assumption $\mathcal{A}^*$, we have $p^e_{s_i}(X) = p^o_{s_i}(X|\overline{X_{s_i}})$. We define $\mathcal{D}^*$ as the expected value $\mathbb{E}[.]$ of this discrepancy:

Definition 4.4 (Relaxed Degree of Dependency)**.**

$$\mathcal{D}^*(S) \equiv \mathop{\mathbb{E}}_{s_i \in S}\left[disc\left(p^o_{s_i}(X), p^o_{s_i}(X|\overline{X_{s_i}})\right)\right] \tag{4.4}$$

This definition includes a whole class of dependency estimators, e.g., [KMB12], which aim at quantifying the so-called *contrast* of a subspace. $\mathcal{D}^*$ – or *contrast* – is a variant of $\mathcal{D}$ which is much easier to estimate. First, it relies on the comparison of marginal against conditional distributions, i.e., multivariate density estimation is not required. Second, the number of degrees of freedom of $\mathcal{A}^*(S)$ increases linearly with $|S|$, but exponentially for $\mathcal{A}(S)$. Thus, estimating $\mathcal{D}^*$ instead of $\mathcal{D}$ allows coping with the strict efficiency requirements for data streams.

By definition, $\mathcal{D}^*$ does not take the dependency between multivariate subsets into account, but only of each variable versus all others. However, we argue that this relaxation is not problematic, and it even supports interpretability. In fact, the detection of dependency is only interesting as long as we can observe effects w.r.t. the marginal and conditional distributions: in real-world scenarios, one is typically looking for interpretable influences of particular variables – so-called targets – on the system and vice versa [HTF09].

4.2.2. Estimating Conditional Distributions

The difficulty when estimating $\mathcal{D}^*$ is estimating the conditional distributions, because the underlying data distributions are unknown. As proposed in [KMB12], one can simulate conditional distributions by applying a set of conditions to S, in a process called *subspace slicing*. The concept of *subspace slice* was defined so far only for numerical data. Here, we extend the definition of *subspace slice* to handle heterogeneity by differentiating between numerical, ordinal and categorical attributes:

Definition 4.5 (Subspace Slice)**.** *A slice c_i in a subspace S w.r.t. attribute s_i is a list of $|S|-1$ conditions C_j which restricts the values of each $s_j \in S \setminus s_i$:*

$$c_i = \left(C_1, \ldots, C_{i-1}, C_{i+1}, \ldots, C_{|S|}\right), \quad \textit{where}$$

$$C_j = \begin{cases} \left[l_j, u_j\right] \ s.t. \ \left|\left\{\vec{x}_k : x_{kj} \in \left[l_j, u_j\right]\right\}\right| = w' & \textit{if } s_j \in \textit{Num} \\ \left[l_j \ldots u_j\right] \ s.t. \ \left|\left\{\vec{x}_k : x_{kj} \in \left[l_j \ldots u_j\right]\right\}\right| = w' & \textit{if } s_j \in \textit{Ord} \\ \left\{v_j : v_j \in s_j\right\} \ s.t. \ \left|\left\{\vec{x}_k : x_{kj} \in \left\{v_j : v_j \in s_j\right\}\right\}\right| = w' & \textit{if } s_j \in \textit{Cat} \end{cases}$$

$$\forall j \in \{1, \ldots, |S|\} \setminus i$$

where $\left[l_j, u_j\right]$ is a continuous interval, $\left[l_j \ldots u_j\right]$ is a discrete interval, and $\left\{v_j : v_j \in s_j\right\}$ is a set of values of s_j. $w' < w$ is the number of observations per condition.

*We call s_i the **reference** attribute, the only attribute without a condition. We write that $\vec{x}_k \in c_i$ if $\vec{x}_k$ fulfils all the conditions in c_i. We define $\bar{c}_i$ as the complementary slice, i.e., it contains all observations which are not in c_i.*

$p_{s_i|c_i}(X)$ and $p_{s_i|\bar{c}_i}(X)$ denote the conditional distribution of the observations in the slice c_i and its complement $\bar{c}_i$, respectively. $\mathcal{P}^c(S)$ is the set of all possible slices in subspace S.

We choose each condition in a slice independently at random, but so that they contain w' observations. Note that ordinal and categorical attributes (e.g., gender) may have many tying values. In such a case, a random condition might not precisely have w' elements. Our solution is to take a random condition containing at least w' observations and randomly remove elements from the condition until only w' observations remain.

We set $w' = \left\lceil w \sqrt[|S|-1]{\alpha} \right\rceil$ with $\alpha \in (0, 1)$, so that, under the independence assumption, the expected share of observations in the slice equals α. As a result, subspace slicing happens in a **dimensionality-aware** fashion. When α is a constant, the expected number of observations per slice does not change with dimensionality. Thus, subspace slicing is a dynamic grid-based method based on the dimensionality of the subspace which alleviates to some extent the effects of the *curse of dimensionality*.

Under the $\mathcal{A}^*$-assumption, the conditional distribution $p_{s_i|c_i}$ is equal to the marginal distribution p_{s_i}, for any attribute s_i and slice c_i. For brevity, we omit '(X)' in $p_{s_i}(X)$ and $p_{s_i|c_j}(X)$ in the following.

Lemma 4.3 ($\mathcal{A}^*$ and Conditional Distributions)**.**

$$\mathcal{A}^*(S) \Leftrightarrow p_{s_i|c_i} = p_{s_i} \quad \forall s_i \in S, \ \forall c_i \in \mathcal{P}^c(S) \tag{4.5}$$

Proof of Lemma 4.3. By contradiction, using Lemma 4.2.

'⇐': From Lemma 4.2, assume $\mathcal{A}^*(S)$ and that

$$\begin{aligned}&\exists s_j \in S : p_{s_j}(X|\overline{X_j}) \neq p_{s_j}\\ &\Rightarrow \exists c_j \in \mathcal{P}^c(S) : p_{s_j|c_j} \neq p_{s_j}\\ &\Rightarrow \text{Contradiction of Lemma 4.3}\end{aligned}$$

'⇒': From Lemma 4.3, assume $\mathcal{A}^*(S)$ and that

$$\begin{aligned}&\exists s_j \in S, \exists c_j \in \mathcal{P}^c(S) : p_{s_j|c_j} \neq p_{s_j}\\ &\Rightarrow p_{s_j}(X|\overline{X_{s_j}}) \neq p_{s_j}\\ &\Rightarrow \text{Contradiction of Lemma 4.2}\end{aligned}$$ □

4.2.3. Discrepancy Estimation

In reality, one only has a limited number of observations, i.e., one only has access to empirical distributions. A solution is to use a statistical test $\mathcal{T}$:

$$disc\left(\hat{p}_{s_i}, \hat{p}_{s_i|c_i}\right) \equiv \mathcal{T}\left(\hat{p}_{s_i}, \hat{p}_{s_i|c_i}\right) \tag{4.6}$$

However, the number of observations is finite, and the observations that we use to estimate $\hat{p}_{s_i|c_i}$ are part of the ones used to estimate $\hat{p}_{s_i}$ so far. This is problematic, as statistical tests assume the samples to be distinct. Plus, when $\alpha \approx 1$, $\hat{p}_{s_i|c_i}$ converges to $\hat{p}_{s_i}$, i.e., the two populations are nearly the same. Conversely, $\alpha \approx 0$ yields spurious effects, since the sample from $\hat{p}_{s_i|c_i}$ then is small. We solve the problem by observing that $p_{s_i|c_i}$ and $p_{s_i|\overline{c}_i}$, the conditional distribution from the complementary slice $\overline{c}_i$, are equal under $\mathcal{A}^*$.

Theorem 4.1 ($\mathcal{A}^*$ and Complementary Conditions)**.**

$$\mathcal{A}^*(S) \Leftrightarrow p_{s_i|\overline{c}_i} = p_{s_i|c_i} \quad \forall s_i \in S,\ \forall c_i \in \mathcal{P}^c(S) \tag{4.7}$$

Proof of Theorem 4.1. By contradiction, using Lemma 4.3.

'⇐': From Lemma 4.3, assume $\mathcal{A}^*(S)$ and that $\exists s_j \in S, \exists c_j \in \mathcal{P}^c(S) : p_{s_j|c_j} \neq p_{s_j}$.

Since $p_{s_j} = p_{s_j|c_j \cup \bar{c}_j}$, then $\exists s_j \in S, \exists c_j \in \mathcal{P}^c(S) : p_{s_j|c_j} \neq p_{s_j|c_j \cup \bar{c}_j}$

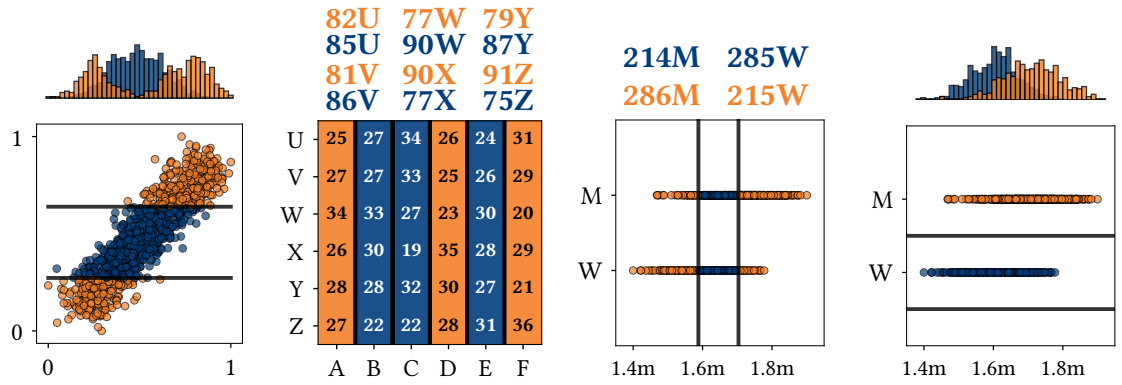
$$\begin{aligned}&\Rightarrow \exists s_j \in S, \exists c_j \in \mathcal{P}^c(S) : p_{s_j|c_j} \neq p_{s_j|\bar{c}_j}\\ &\Rightarrow \text{Contradiction of Theorem 4.1.}\end{aligned}$$

'⇒': From Theorem 4.1, assume $\mathcal{A}^*(S)$ and that $\exists s_j \in S, \exists c_j \in \mathcal{P}^c(S) : p_{s_j|c_j} \neq p_{s_j|\bar{c}_j}$

$$\Rightarrow \exists s_j \in S, \exists c_j \in \mathcal{P}^c(S) : p_{s_j|c_j \cup c_j} \neq p_{s_j|\bar{c}_j \cup c_j}.$$

Since $c_j \cup c_j = c_j$ and $p_{s_j} = p_{s_j|\bar{c}_j \cup c_j}$, then $\exists s_j \in S, \exists c_j \in \mathcal{P}^c(S) : p_{s_j|c_j} \neq p_{s_j}$

$$\Rightarrow \text{Contradiction of Lemma 4.3.}$$ □



(a) Numerical (b) Categorical (c) Heterogeneous i. (d) Heterogeneous ii.

Figure 4.1.: Slicing in numerical, categorical, and heterogeneous subspaces ($|S| = 2$).

Hence, one can evaluate $\mathcal{A}^*$ by looking at the discrepancies between the conditional distribution and its complementary conditional distribution. When doing so, the samples obtained from both distributions are distinct.

We define our dynamic slicing scheme based on a parameter α, the expected share of observations in the slice c_i. Thus, the expected share of observations $\overline{\alpha}$ in $\overline{c}_i$ equals $1 - \alpha$. As a result, we set $\alpha = 0.5$ so that $\overline{\alpha} = \alpha$. This choice is judicious for statistical testing, as samples of equal size lead to higher statistical stability, and we get rid of parameter α.

We illustrate slicing in heterogeneous subspaces in Figure 4.1, with an exemplary numerical and categorical subspace in the left half and a heterogeneous subspace in the right half. The black lines show a random slice. The points in dark blue are in c_i, and the points in light orange are in $\overline{c}_i$. Figure 4.1(a) represents a numerical linear dependency. We can see from the histograms that, after slicing, the distribution of the points in each sample are very different. Figure 4.1(b) depicts the absolute frequencies of observing various symptoms $\{U \dots Z\}$ in different groups of patients $\{A \dots F\}$. Since there is no ordering within symptoms and patient groups, slicing in this case consists in selecting categories at random, here $\{B, C, E\}$. By comparing the absolute frequencies after slicing, we can determine whether there is a statistical association between groups and symptoms. Naturally, the statistical test that we use to estimate the discrepancy between $p_{s_i|\overline{c}_i}$ and $p_{s_i|c_i}$ might differ depending on the type of the reference attribute, as we will discuss later.

Different attribute types can also be part of the same subspace, as we show in Figure 4.1(c) and 4.1(d). We graph the height from a sample of individuals of two sexes. When we slice on the x-axis, the slice is a numerical interval. On the y-axis in turn, the slice is a category drawn at random.

Intuitively, ordinal attributes share features from both numerical and categorical attributes: there exists an ordering between values, but typically also many tying values. In this case, we recommend using a similar slicing methodology as for numerical attributes, by selecting a discrete interval (see Definition 4.5), and a statistical test that is robust to tying values to some extent.

4.2.4. Properties of Contrast

A statistical test $\mathcal{T}(B_1, B_2)$ for two samples B_1 and B_2 typically yields a p-value. Traditionally, one uses p-values to assess the *statistical significance*. Conversely, $\overline{p} = 1 - p$ is known as *confidence level*. The rationale behind estimating the degree of dependency $\mathcal{D}^*$ is to yield values quantifying the independence violation. We define *contrast*, abbreviated as C, as the expected value of the *confidence level* of a statistical test $\mathcal{T}$ between the samples from the conditional distributions for all the possible attributes s_i and slices c_i:

Definition 4.6 (Contrast C)**.**

$$C(S) \equiv \mathop{\mathbb{E}}_{c_i \in \mathcal{P}^c(S)} \left[\mathcal{T}\left(B(c_i), B(\overline{c}_i)\right) \right] \tag{4.8}$$

where $\mathcal{T}$ yields $\overline{p}$-values, and $B(c_i), B(\overline{c}_i)$ are the samples resulting from slicing. We draw the conditions in c_i randomly and independently, w.r.t. any reference attribute s_i in subspace S.

By definition, $\mathcal{T} \sim \mathcal{U}[0, 1]$ when the two samples are independent, and $\mathcal{T} \approx 1$ as the evidence against independence increases. The properties of C follow:

1. C converges to 1 as the dependency in S increases.
2. C converges to 0.5 when S is independent, since $\mathcal{T} \sim \mathcal{U}[0, 1]$.
3. C is bounded between 0 and 1.

4.2.5. Monte Carlo Approximation

It is impossible to compute C exactly; one would need to know the distribution of $B(c_i)$ and $B(\overline{c}_i)$ for every slice. Instead, we approximate C via Monte Carlo (MC) simulations, with M iterations. In each iteration, we choose the reference attribute s_i and slice c_i at random. We define the *approximated contrast* $\widehat{C}$:

Definition 4.7 (Approximated Contrast $\widehat{C}$)**.**

$$\widehat{C}(S) = \frac{1}{M} \sum_{m=1}^{M} \left[\mathcal{T}\left(B\left(c_i\right), B\left(\overline{c}_i\right)\right) \;:\; c_i \overset{m}{\sim} \mathcal{P}^c(S) \right] \tag{4.9}$$

where $c_i \overset{m}{\sim} \mathcal{P}^c(S)$ means that we draw c_i randomly from $\mathcal{P}^c(S)$ in iteration m.

Interestingly, we can bound the quality of the approximation. From Hoeffding's inequality [Hoe63], we derive a bound on the probability of $\widehat{C}$ to deviate not less than ε from C. The bound decreases exponentially with increasing M:

Theorem 4.2 (Hoeffding's Bound of $\widehat{C}$)**.**

$$\Pr\left[|\widehat{C} - C| \geq \varepsilon \right] \leq 2e^{-2M\varepsilon^2} \tag{4.10}$$

where M is the number of MC iterations, and $0 < \varepsilon < 1 - C$.

Proof of Theorem 4.2. Let us first restate Theorem 1 from [Hoe63] (see also Lemma 5.5): Let $X_1, X_2, \ldots, X_n$ be independent random variables $0 \leq X_i \leq 1$ for $i = 1, \ldots, n$ and let $\bar{X} = \frac{1}{n}(X_1 + X_2 + \cdots + X_n)$ be their mean with expected value $E[\bar{X}]$. Then, for $0 < t < 1 - E[\bar{X}]$, it holds that $\Pr[\bar{X} - E[\bar{X}] \geq t] \leq \mathrm{e}^{-2nt^2}$ and $\Pr[\bar{X} - E[\bar{X}] \leq t] \leq \mathrm{e}^{-2nt^2}$.

We can treat each MC iteration $m_1, \ldots, m_M$ as i.i.d. random variables $X_{m_1}, \ldots, X_{m_M}$ with $0 \leq X_{m_i} \leq 1$. $\hat{C}$ is the mean of the iterations with $E[\hat{C}] = C$ (Definition 4.6). Thus, for $0 < \varepsilon < 1 - C$ it holds that

$$\Pr[\hat{C} - C \geq \varepsilon] \leq \mathrm{e}^{-2M\varepsilon^2} \qquad \Pr[\hat{C} - C \leq \varepsilon] \leq \mathrm{e}^{-2M\varepsilon^2} \tag{4.11}$$

since $\Pr[|\hat{C} - C| \geq \varepsilon] = \Pr[\hat{C} + C \geq \varepsilon] + \Pr[\hat{C} - C \geq \varepsilon]$ it is easy to verify Theorem 4.2. □

This is very useful. For instance, when $M = 200$, the probability of $\widehat{C}$ to deviate more than 0.1 from its expected value is less than $2e^{-4} \approx 0.04$, and this bound decreases exponentially with M. Thus, one can adjust the computational requirements of $\widehat{C}$ given the available resources, the desired quality level, or the rate of arrival of new observations in a stream. In other words, users can set M intuitively, as it leads to an expected quality, and vice versa. Observe that M is our only parameter.

4.2.6. Instantiating MCDE

One must instantiate $\mathcal{T}$ as a suitable statistical test. Ideally, $\mathcal{T}$ is non-parametric (**R4**) and suitable for the type of the reference attribute (numerical, ordinal, categorical). To facilitate meaningful experiments, we investigate instantiations of MCDE based on three well-known non-parametric tests: the Kolmogorov-Smirnov, the Mann-Whitney U and the Chi-Squared test. We call the respective instantiations KSP, MWP and CSP.

The Kolmogorov-Smirnov test assumes the data to be continuous, i.e., it should be adequate for numerical attributes. The Mann-Whitney U test specifically handles tying values, so it might work well with ordinal attributes. Lastly, the Chi-Squared test bases on frequencies from a finite number of categories, i.e., we hypothesise it to be suitable for categorical attributes.

Algorithm 4.1 MCDE($S = \{s_1, \ldots, s_{|S|}\}$)

```
1: I ← ConstructIndex(S) ; result ← 0
2: for m ← 1 to M do
3:     r ← random integer in [1, |S|]
4:     slice ← Slice(I, r)
5:     result ← result + Test(I, slice, r)
6: return (result/M) ∈ (0, 1)
```

Algorithm 4.1 summarises the general idea behind MCDE for any arbitrary subspace $S = \{s_1, \ldots, s_{|S|}\}$ of dimensionality $|S|$. In practice, we can significantly improve the efficiency of slicing operations, which require the values of each attribute to be ordered, with an index structure (Line 1). Afterwards, for M iterations, we slice the data (Line 4) and carry out the statistical test (Line 5). The final estimate is the average of the test outcomes over M iterations. In what follows, we present the specifics of the procedures ConstructIndex, Slice and Test for each instantiation of MCDE.

4.3. Instantiation as Mann-Withney-P (MWP)

We first consider the instantiation of $\mathcal{T}$ as a two-sided Mann-Whitney U test [MW47]. An advantage of this test is that it does not assume the data to be continuous, as it operates on ranks. So it is robust and applicable to numeric and ordinal measurements.

4.3.1. Estimating The Mann-Whitney U Statistics

In a nutshell, the Mann-Whitney U test compares the difference between the median of two samples. We review the definition of this test [SC88] between two samples B_1 and B_2 with size n_1 and n_2, and $n_1 + n_2 = w$. It tests the null hypothesis that it is equally likely that a randomly selected value from one sample will be less than or greater than a randomly selected value from the other sample. R_1 and R_2 are the sums of ranks of the objects in B_1 and B_2, obtained by ranking the values of B_1 and B_2 together, starting with 0. In case of ties, the ranks of the tying objects are *adjusted*, i.e., become the average of their ranks.

$$p = \Phi(Z) \text{ or } 1 - \Phi(Z) \qquad Z = \frac{U_i - \mu}{\sigma} \qquad U_i = R_i - \frac{n_i(n_i - 1)}{2} \tag{4.12}$$

Here, one can choose $U_i = U_1$ or $U_i = U_2$ equivalently. Φ is the cumulative distribution function of the normal distribution, and μ, σ are defined as:

$$\mu = \frac{n_1 n_2}{2} \qquad \sigma = \sqrt{\frac{n_1 n_2}{12}\left((w+1) - \sum_{i=1}^{k} \frac{t_i^3 - t_i}{w(w-1)}\right)} \tag{4.13}$$

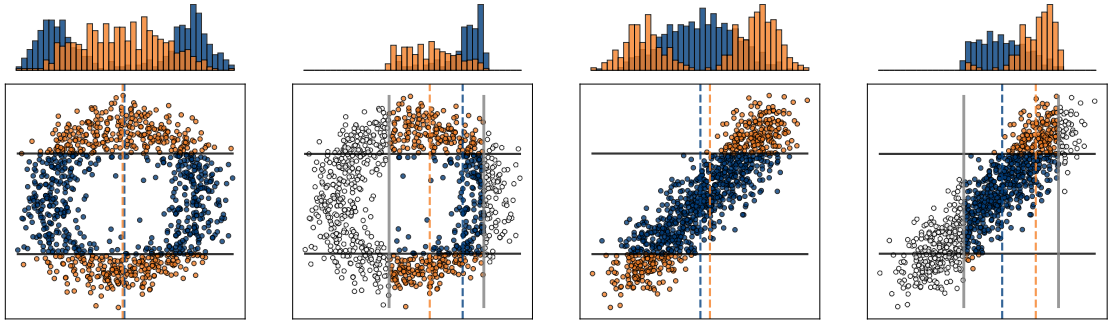
The summation term of σ is a correction for ties, where t_i is the number of observations sharing rank i, and k is the number of distinct ranks. Typically, for $w > 20$, the values of U are normally distributed [MW47] with mean μ and standard deviation σ. Z is the standardised score, since $Z \sim \mathcal{N}(0, 1)$. If $U = U_1$, then $Z \ll 0$ and $p \approx 0$ when the ranks of A_1 are stochastically smaller than those of A_2. Conversely, when the ranks of A_1 are stochastically larger, then $Z \gg 0$ and $p \approx 1$. Both cases indicate an independence violation. As both directions are relevant, our test should capture them equally:

$$\text{MWP} = \Phi^{1/2}(Z') \qquad Z' = |Z| \qquad U = U_1 \tag{4.14}$$

Since $Z \sim \mathcal{N}(0, 1)$, Z' follows the so-called half-normal distribution with *cdf* $\Phi^{1/2}$. Since $|U_1 - \mu| = |U_2 - \mu|$, we can simply set $U = U_1$ or $U = U_2$ arbitrarily.

However, the ability of this test to detect dependency – the 'power' – declines in the case of unequal variance of the two samples [Zim03; FS09]. Thus, we include an additional step into the slicing process. It restricts the domain of the reference attribute s_i to a share α of observations. Formally, we define the marginal restriction as follows:

Definition 4.8 (Marginal Restriction)**.** *A marginal restriction is a condition on the reference attribute s_i, i.e., an interval $r_i : [l_i, u_i]$ or $r_i : [l_i \dots u_i]$, so that $|\{\vec{x}_j : x_{ji} \in r_i\}| = \lceil \alpha \cdot w \rceil = \lceil w' \rceil$ and the subspace slice becomes $c_i \cup r_i$.*



(a) without MR (b) with MR (c) without MR (d) with MR

Figure 4.2.: Marginal Restriction (MR) w.r.t. a circular and linear dependency ($|S| = 2$).

We illustrate in Figure 4.2 the impact of the marginal restriction. We show in the left half a circular dependency and in the other half a linear dependency. Two grey lines show a marginal restriction (in 4.2(b) and 4.2(d)), and two vertical dashed lines stand for the median of each sample. We can see that in both cases, the marginal restriction leads to the median of the two samples having a larger difference. Thus, the discrepancy between both samples will be better detected by the Mann Whitney U test. Intuitively, the marginal restriction 'breaks the symmetry' between both distributions. Because of that, the MWP estimator with marginal restriction generally has higher statistical power.

4.3.2. Implementation Details

Algorithm 4.2 is the pseudo-code for the index construction. The index $\mathcal{I}$ is a 1-dimensional structure containing the adjusted ranks and tying value corrections for each attribute. It consists of $|S|$ elements $\{I_1, \ldots, I_{|S|}\}$, where I_i is an array of 4-tuples sorted by s_i in ascending order. In this index, l_i is the row numbers, $\tilde{x}_i$ are the sorted values of s_i, a_i are the *adjusted* ranks and b_i the accumulated correction terms of the standard deviation σ. $I_i[j]$ stands for the j-th tuple of I_i, and l_{ji}, $\tilde{x}_{ji}$, a_{ji}, b_{ji} are its components.

Algorithm 4.3 shows the slicing process. We can slice the input data efficiently because the tuples are already sorted in the index structure. We successively mask the row numbers based on a random condition for all but one reference attribute s_r. Additionally, we ensure that the condition boundaries do not split any tying values and that each condition has exactly w' observations. The algorithm returns a *slice*, i.e., a list of w Boolean values, so we write $slice \in \mathbb{Z}_2^w$. Since we visit each value at most once, the complexity is in $O(|S| \cdot w)$.

Algorithm 4.4 implements the statistical test based on our index. We determine a restriction $[start, end]$ on s_r and sum the adjusted ranks of the observations in the slice. Since the ranks in this subset may not start from 0, we adjust the sum of the ranks R_1 (Line 10). Then we compute a correction (Line 13) using the cumulative correction b_r to adjust σ for ties (Line 14). $\Phi^{1/2}$ is the cumulative distribution function of the half-Gaussian distribution. We compute the statistical test via a single pass, considering only elements between *start* and *end*. Each operation requires constant time; The complexity is in $O(w)$.

Algorithm 4.2 MWP-ConstructIndex($S = \{s_1, \ldots, s_{|S|}\}$)

for $i = 1$ to $|S|$ **do**
  $r_i \leftarrow [0, \ldots, w-1]$
  $(l_i, \tilde{x}_i) \leftarrow$ sort (r_i, x_i) by s_i in ascending order, break ties randomly
  $I_i \leftarrow [(l_{1i}, \tilde{x}_{1i}, r_{1i}), \ldots, (l_{wi}, \tilde{x}_{wi}, r_{wi})]$
  $j \leftarrow 1$; $correction \leftarrow 0$
  while $j \leq w$ **do**
    $k \leftarrow j$; $t \leftarrow 1$; $adjust \leftarrow 0$
    while $(k < w-1) \wedge (s_i[l_{ki}] = s_i[l_{k+1,i}])$ **do**
      $adjust \leftarrow adjust + r_{ki}$; increment k and t
    if $k > j$ **then**
      $adjusted \leftarrow (adjust + r_{ki})/t$; $correction \leftarrow correction + t^3 - t$
      for $m \leftarrow j$ to k **do** $I_i[m] \leftarrow (l_{mi}, \tilde{x}_{mi}, adjusted, correction)$
    else $I_i[j] \leftarrow (l_{ji}, \tilde{x}_{ji}, r_{ji}, correction)$
    $j \leftarrow j + t$
return $\mathcal{I} : \{I_1, \ldots, I_{|S|}\}$ with $I_i : (l_i, \tilde{x}_i, a_i, b_i)$

Algorithm 4.3 MWP-Slice($\mathcal{I} : \{I_1, \ldots, I_{|S|}\}, r \in \{1, \ldots, |S|\}$)

$w' \leftarrow \lceil w \cdot \sqrt[|S|-1]{\alpha} \rceil$
for $I_i \in \mathcal{I} \setminus I_r$ **do**
  $slice_i \leftarrow$ Array of w Boolean values initialised to *false*
  $start \leftarrow$ random integer in $[1, w - w']$
  $end \leftarrow start + w'$
  while $r_{start,i} = r_{start-1,i}$ **do** $start = start - 1$
  while $r_{end,i} = r_{end+1,i}$ **do** $end = end + 1$
  for $j \leftarrow start$ to end **do** $slice[l_{ji}] \leftarrow true$
  if $end - start > w'$ **then**
    $nb \leftarrow end - start - w'$
    $exclude \leftarrow$ draw nb sample from $[start, end]$ without replacement
    for $el \in exclude$ **do** $slice[el] \leftarrow false$
$slice_r \leftarrow$ Array of w Boolean values initialised to *true*
$slice \leftarrow slice_1 \wedge \cdots \wedge slice_{|S|}$
return $slice \in \mathbb{Z}_2^w$

Algorithm 4.4 MWP-Test($\mathcal{I} : \{I_1, \ldots, I_{|S|}\}, slice \in \mathbb{Z}_2^w, r \in \{1, \ldots, |S|\}$)

$start \leftarrow$ random integer in $[1, w \cdot (1-\alpha)]$
$end \leftarrow start + \lceil w \cdot \alpha \rceil$
$R_1 \leftarrow 0$; $n_1 \leftarrow 0$
for $j \leftarrow start$ to end **do**
  if $slice[l_{jr}] = true$ **then**
    $R_1 \leftarrow R_1 + a_{jr}$
    $n_1 \leftarrow n_1 + 1$
$w' \leftarrow end - start$
if $n_1 = 0$ or $n_1 = w'$ **return** 1
$U_1 \leftarrow R_1 - start \cdot n_1$
$n_2 \leftarrow w' - n_1$
$\mu \leftarrow (n_1 \cdot n_2)/2$
$correction \leftarrow (b_{end-1,r} - b_{start-1,r})/(w' \cdot (w'-1))$
$\sigma \leftarrow \sqrt{((n_1 \cdot n_2)/12) \cdot (w' + 1 - correction)}$
return $\Phi^{1/2}(|U_1 - \mu|/\sigma) \in (0, 1)$

4.4. Instantiation as Kolmogorov-Smirnov-P (KSP)

4.4.1. Estimating Kolmogorov-Smirnov Statistic

We now describe another instantiation of MCDE, which uses the two-sample Kolmogorov-Smirnov (KS) test. The KS test is non-parametric, and it is widely used to test the equality of two continuous one-dimensional probability distributions. It is adequate in case the reference attribute is numerical. However, it is known that the KS test has less power in the presence of ties [LR05]. So it may not work well with ordinal attributes.

In a nutshell, the two-sample Kolmogorov-Smirnov statistic D is the supremum of the absolute differences of the empirical cumulative distribution functions $F_1(x)$ and $F_2(x)$ of Samples 1 and 2 with n_1 and n_2 elements:

$$D = \sup_x |F_1(x) - F_2(x)| \tag{4.15}$$

High Contrast Subspaces [KMB12] (HiCS) employed this test statistic to quantify contrast. However, to comply with the MCDE framework, one must first derive the corresponding $\overline{p}$-value. The $\overline{p}$-values are not trivial to obtain, because the distribution of D does not have any known closed form, and the time required for an exact computation increases with n_1 and n_2 in particular. To obtain the $\overline{p}$-values, we approximate them using the asymptotic Kolmogorov-Smirnov distribution proposed in [Dur73]:

$$\Pr\left[D\sqrt{\frac{n_1 \cdot n_2}{n_1 + n_2}} \leq x\right] = 1 - 2\sum_{i=1}^{\infty}(-1)^{i-1}e^{-2i^2x^2} \tag{4.16}$$

Empirically, we found that summing up the first 1000 terms of the expansion provides enough accuracy, without much impact on the execution time. Using this approximation is common practice within most modern statistical software, such as R [R C19].

4.4.2. Implementation Details

Algorithm 4.5 is the pseudo-code for the index construction. The difference to MWP is that we do not need to adjust ranks or precompute tie correction, because the Kolmogorov-Smirnov test assumes that there are no tying values. The resulting data structure contains the indices l_i and the values x_i of each attribute s_i, ordered by s_i in ascending order.

Algorithm 4.5 KSP-ConstructIndex($S = \{s_1, \ldots, s_{|S|}\}$)

1: **for** $i = 1$ to $|S|$ **do**
2: $\quad r_i \leftarrow [0, \ldots, w-1]$
3: $\quad (l_i, \tilde{x}_i) \leftarrow$ sort (r_i, x_i) by s_i in ascending order, break ties randomly
4: $\quad I_i \leftarrow [(l_{1i}, \tilde{x}_{1i}), \ldots, (l_{wi}, \tilde{x}_{wi})]$
5: **return** $\mathcal{I} : \{I_1, \ldots, I_{|S|}\}$ with $I_i : (l_i, \tilde{x}_i)$

Similarly, Algorithm 4.6 is responsible for slicing, but does not require any further step to handle ties. Algorithms 4.2 and 4.5 as well as Algorithms 4.3 and 4.6 respectively behave in the same way whenever the data does not have ties.

Algorithm 4.6 KSP-Slice($\mathcal{I} : \{I_1, \ldots, I_{|S|}\}, r \in \{1, \ldots, |S|\}$)

1: $w' \leftarrow \lceil w \cdot \sqrt[|S|-1]{\alpha} \rceil$
2: **for** $I_i \in \mathcal{I} \setminus I_r$ **do**
3: $\quad slice_i \leftarrow$ Array of w Boolean values initialised to *false*
4: $\quad start \leftarrow$ random integer in $[1, w - w']$
5: $\quad end \leftarrow start + w'$
6: $\quad$ **for** $j \leftarrow start$ to end **do** $slice[l_{ji}] \leftarrow true$
7: $slice_r \leftarrow$ Array of w Boolean values initialised to *true*
8: $slice \leftarrow slice_1 \wedge \cdots \wedge slice_{|S|}$
9: **return** $slice \in \mathbb{Z}_2^w$

Algorithm 4.7 implements the KS test. We compute the statistic D, i.e., the largest difference of the two empirical cumulative distribution functions in Lines 7–10. Then we approximate the $\overline{p}$-value with Equation 4.16 in Line 12.

Algorithm 4.7 KSP-Test($\mathcal{I} : \{I_1, \ldots, I_{|S|}\}, slice \in \mathbb{Z}_2^w, r \in \{1, \ldots, |S|\}$)

1: $n_1 \leftarrow |\{i : i \in [1 \ldots w] \wedge slice[l_{ir}] = true\}|$
2: $n_2 \leftarrow w - n_1$
3: **if** $n_1 = 0$ **or** $n_2 = 0$ **return** 1
4: $u \leftarrow 1/n_1$; $v \leftarrow 1/n_2$
5: $\zeta_1 \leftarrow 0$; $\zeta_2 \leftarrow 0$
6: $D \leftarrow 0; \phi \leftarrow 0$
7: **for** $i \leftarrow 1$ to w **do**
8: $\quad$ **if** $slice[l_{ir}] = true$ **then** $\zeta_1 \leftarrow \zeta_1 + u$
9: $\quad$ **else** $\zeta_2 \leftarrow \zeta_2 + v$
10: $\quad D \leftarrow max\{D, |\zeta_2 - \zeta_1|\}$
11: $z \leftarrow D\sqrt{n_1 n_2/(n_1 + n_2)}$
12: **for** $i \leftarrow 1$ to 1000 **do** $\phi \leftarrow \phi + (-1)^{i-1} e^{-2i^2 z^2}$ **return** $(1 - 2\phi) \in (0, 1)$

4.5. Instantiation as Chi-Squared-P (CSP)

4.5.1. Estimating Chi-Squared Statistic

The Chi-Squared test, also known as Pearson's Chi-Squared test, perhaps is the most famous non-parametric test. In short, it determines whether there is a significant difference between the expected frequencies and the observed frequencies of a set of categories.

For a reference variable $s_i \in Cat$ with categories $A = \{a_1, \ldots, a_{|A|}\}$, we sketch the contingency table w.r.t. the two samples $B(c_i)$ and $B(\overline{c}_i)$ in Table 4.1, where o_j^i is the absolute frequency of Category a_j in Sample i, and we have:

$$\sum_{i=1}^{|A|} o_i^j = o^j \qquad j \in \{1, 2\} \tag{4.17}$$

$$\sum_{j=1}^{2} o_i^j = o_i \qquad i \in \{1, \ldots, |A|\} \tag{4.18}$$

Table 4.1.: Exemplary Contingency Table.

	a_1	a_2	...	$a_{\|A\|}$	Total
Sample 1: $B(c_i)$	o_1^1	o_2^1	...	$o_{\|A\|}^1$	o^1
Sample 2: $B(\overline{c}_i)$	o_1^2	o_2^2	...	$o_{\|A\|}^2$	o^2
Total	o_1	o_2	...	$o_{\|A\|}$	w

Then we can compute the test statistic Q as follows:

$$Q = \sum_{i=1}^{|A|} \sum_{j=1}^{2} \frac{(o_i^j - e_i^j)^2}{e_i^j} \tag{4.19}$$

where $e_i^j = o_i \cdot o^j / w$ is the expected absolute frequency.

Under independence, Q follows the χ^2 distribution with cumulative distribution function $\chi_k^2 : \mathbb{R}^+ \mapsto [0, 1]$, where $k = |A| - 1$ is the number of degrees of freedom. Thus, $\chi_k^2(Q)$ leads to the $\overline{p}$-value that we use for CSP.

4.5.2. Implementation Details

As with the other instantiations, we improve the execution time with an index. It contains the position of each occurrence of a categorical value binned into its respective category. Algorithm 4.8 is our pseudo-code for its construction. We can construct the index in linear time with a single pass over each attribute, as it does not require any sorting.

Algorithm 4.9 is our slicing procedure for CSP. The main difference to MWP and KSP is that the index is not ordered. Thus, slicing consists in selecting a random set of categories per attribute. The algorithm ensures that exactly w' observations are part of each condition.

Finally, Algorithm 4.10 shows how to compute the Chi-Squared statistic, based on the information from the index and a subspace slice.

4.5.3. Complexity

The overall complexity of MWP and KSP is in $O(|S| \cdot (w \cdot log(w) + w) + M \cdot (|S| \cdot w + w))$. Since $|S| \ll w$, this simplifies to $O(w \cdot log(w) + M \cdot w)$. The index construction is asymptotically the most expensive step, as it is in $O(w \cdot log(w))$. Since the index construction for CSP does not require sorting, this step simplifies to $O(w)$. However, one only needs to construct the index once for a given data set or window. When the index is available, one can compute the estimator in linear time for the exponential number of subspaces.

Interestingly, MCDE is trivial to parallelise: one can compute the elements of the index structure $I_1, \ldots, I_{|S|}$ in parallel, as they are independent of each other. Similarly, one can parallelise each Monte Carlo iteration. This is useful, as multi-core architectures are ubiquitous in modern database systems. Thus, MCDE scales well with the size of the data set. We will verify this claim via experiments in Section 4.7.2.7.

Algorithm 4.8 CSP-ConstructIndex($S = \{s_1, \ldots, s_{|S|}\}$)

for $i = 1$ to $|S|$ **do**
  Define I_i as a mapping of categories to $\{positions \subset \{0, \ldots, w-1\}, counts \in \mathbb{N}^+\}$
  for $x_{ji} \in s_i$ **do**
    if $I_i[x_{ji}] \neq \emptyset$ **then**
      $I_i[x_{ji}] \leftarrow \{I_i[x_{ji}].positions \oplus x, I_i[x_{ji}].counts + 1\}$
    else
      $I_i[x_{ji}] \leftarrow \{j, 1\}$
return $\mathcal{I} : \{I_1, \ldots, I_{|S|}\}$ where $I_i : el \in s_i \mapsto \mathbb{N}^+$

Algorithm 4.9 CSP-Slice($\mathcal{I} : \{I_1, \ldots, I_{|S|}\}, r \in \{1, \ldots, |S|\}$)

$w' \leftarrow \lceil w \cdot \sqrt[d-1]{\alpha} \rceil$
for $I_i \in \mathcal{I} \setminus I_r$ **do**
  $slice_i \leftarrow$ Array of w Boolean values initialised to *false*
  $slicesize \leftarrow 0$
  $positions \leftarrow \emptyset$
  $categories \leftarrow \mathcal{I}_i.keys$
  while $slicesize < w'$ **do**
    $category \leftarrow$ draw a random category from $categories$
    $categories \leftarrow categories \setminus category$
    $slicesize \leftarrow slicesize + \mathcal{I}_i[category].counts$
    $positions \leftarrow positions \oplus \mathcal{I}_i[category].positions$
  if $slicesize > w'$ **then**
    Delete $slicesize - w'$ random elements from $positions$
  for $pos \in positions$ **do**
    $slice_i[pos] \leftarrow true$
$slice_r \leftarrow$ Array of w Boolean values initialised to *true*
$slice \leftarrow slice_1 \wedge \cdots \wedge slice_{|S|}$
return $slice \in \mathbb{Z}_2^w$

Algorithm 4.10 CSP-Test($\mathcal{I} : \{I_1, \ldots, I_{|S|}\}, slice \in \mathbb{Z}_2^w, r \in \{1, \ldots, |S|\}$)

$Q = 0$; $k = 0$
$o^1 = |\{pos \in [0 \ldots w-1] : slice[pos] = true\}|$
$o^2 = w - o_1$
for $\{positions, counts\} \in I_r$ **do**
  $o_x^1 = |\{pos \in positions : slice[pos] = true\}|$
  $o_x^2 = counts - o_x^1$
  $o_x = o_x^1 + o_x^2$
  $e_x^1 = o_x \cdot o^1 / w$
  $e_x^2 = o_x \cdot o^2 / w$
  $k = k + 1$
  $Q = Q + (o_x^1 - e_x^1)^2 / e_x^1 + (o_x^2 - e_x^2)^2 / e_x^2$
return $\chi^2_{k-1}(Q) \in (0, 1)$

4.6. MCDE in Heterogeneous Data Streams

4.6.1. Heterogeneity

For simplicity, we have described KSP, MWP and CSP, assuming a homogeneous data set, being numerical, ordinal and categorical respectively.

In fact, each attribute is treated independently in each of our algorithms. So we can easily extend Algorithm 4.1 to the heterogeneous setting, as we show in Algorithm 4.11. We construct the index of each attribute depending on its type (Lines 1–3) and use the corresponding slicing methodology (Lines 7-9). The resulting slice is the element-wise conjunction for each type (Line 10). The type of the reference attribute determines which test we should use (Lines 11-13). Independently of the underlying statistical test, the $\overline{p}$-values have the properties described in Section 4.2.4. So the final MCDE score is the average of the $\overline{p}$-values over each iteration.

Algorithm 4.11 HETEOROGENEOUS-MCDE ($S = \{s_1, \ldots, s_{|S|}\}$)

1: $\mathcal{I}_N \leftarrow$ KSP-CONSTRUCTINDEX($\{s_i \in S : s_i \in Num\}$)
2: $\mathcal{I}_O \leftarrow$ MWP-CONSTRUCTINDEX($\{s_i \in S : s_i \in Ord\}$)
3: $\mathcal{I}_C \leftarrow$ CSP-CONSTRUCTINDEX($\{s_i \in S : s_i \in Cat\}$)
4: $result \leftarrow 0$
5: **for** $m \leftarrow 1$ to M **do**
6: $\quad r \leftarrow$ random integer in $[1, |S|]$
7: $\quad slice_N \leftarrow$ KSP-SLICE($\mathcal{I}_N, r$)
8: $\quad slice_O \leftarrow$ MWP-SLICE($\mathcal{I}_O, r$)
9: $\quad slice_C \leftarrow$ CSP-SLICE($\mathcal{I}_C, r$)
10: $\quad slice \leftarrow slice_N \oplus slice_O \oplus slice_C$
11: $\quad$ **if** $s_r \in Num$ **do** $result \leftarrow result +$ KSP-TEST($\mathcal{I}_N, slice, r$)
12: $\quad$ **if** $s_r \in Ord$ **do** $result \leftarrow result +$ MWP-TEST($\mathcal{I}_O, slice, r$)
13: $\quad$ **if** $s_r \in Cat$ **do** $result \leftarrow result +$ CSP-TEST($\mathcal{I}_C, slice, r$)
14: **return** $(result/M) \in (0, 1)$

4.6.2. Adaptation to the Streaming Setting

To deal with streams, we adopt the well-known sliding window model, i.e., we only consider the w most recent observation. A naive way to support this model is to recompute the index at the arrival of each new observation. Instead, we propose efficient insertion and deletion operations for our indexes.

Furthermore, to maintain a dependency estimate over time, we propose to perform a number M of MC iterations periodically and report the exponential moving average:

$$MCDE_t = \gamma \cdot MCDE(W_{t-\Delta}) + (1 - \gamma) \cdot MCDE(W_t) \tag{4.20}$$

where γ is the so-called decaying factor, and Δ is the step size.

We update the MWP index in Algorithm 4.12 in two steps: STEP 1: INSERT/DELETE and STEP 2: REFRESH. Our algorithm maintains two data structures: a queue, which determines for each new point the value of the point to delete in the current window, in a first-in-first-out fashion, and a variant of our static index which supports binary search. In the first

Table 4.2.: Algorithmic Complexity of MCDE instantiations.

	MWP	KSP	CSP
Index Construction	$O(\|S\| \cdot w \cdot log(w))$	$O(\|S\| \cdot w \cdot log(w))$	$O(\|S\| \cdot w)$
Slicing	$O(\|S\| \cdot w)$	$O(\|S\| \cdot w)$	$O(\|S\| \cdot w)$
Test	$O(w)$	$O(w)$	$O(w)$
Update (Step 1)	$O(\|S\| \cdot log(w))$	$O(\|S\| \cdot log(w))$	$O(1)$
Update (Step 2)	$O(\|S\| \cdot w)$	$O(\|S\| \cdot w)$	$O(\|S\| \cdot w)$

step, we store the values for each attribute in a queue, in chronological order. Then we find the positions where to insert the new point and where to delete the oldest point in the index via binary search. Then, we shift all the values to insert the point. In the second step, we recompute the adjusted ranks and cumulative correction as in Algorithm 4.2.

Using an adequate data structure like a binary tree, Step 1 only has logarithmic complexity, while Step 2 has linear complexity. Besides this, one must perform Step 1 for each new point, but Step 2 only once before slicing. So when re-estimating contrast only every Δ-th step, we perform Step 1 for each point, but Step 2 lazily. As a result, we can update the index in $O(|S| \cdot log(w))$ in Step 1 for each observation and postpone Step 2, which is in $O(|S| \cdot w)$, to contrast estimation. Updating the KSP index is somewhat simpler because KSP does not handle tying values. The CSP index is unsorted, and thus Step 1 only requires constant time. We refer to KSP-Update and CSP-Update as variants of Algorithm 4.12 for those indexes. We summarise the complexity of each step in Table 4.2.

For efficiency, Algorithm 4.12 simultaneously inserts and deletes observations. Note that one could also perform each operation via two independent methods. This way, one may handle time-based windows, in which observations may arrive at arbitrary time steps.

Algorithm 4.12 MWP-Update($\mathcal{I} : \{I_1, \ldots, I_{|S|}\}, \vec{x}_{new} = \langle x_{new,i} \rangle_{i \in \{1,\ldots,|S|\}}$)

for $i = 1$ to $|S|$ **do**
  $queue_i.insert(x_{new,i})$; $x_{old,i} = queue_i.pop()$ ▷ Step 1: Insert/Delete
  $offset_i = offset_i + 1$
  $insert = binarysearch(I_i, x_{new,i})$; $delete = binarysearch(I_i, x_{old,i})$
  if $insert < delete$ **for** $x \leftarrow insert$ to $delete$ **do** $I_i[x+1] = I_i[x]$
  else for $x \leftarrow delete$ to $insert$ **do** $I_i[x] = I_i[x+1]$
  $I_i[insert] = (w + offset_i, x_{new,i}, -1, -1)$
  for $pos \leftarrow 1$ to w **do** $I_i[pos] = (l_{pos,i} - offset_i, \tilde{x}_{pos,i}, pos, 0)$ ▷ Step 2: Refresh
  $offset_i \leftarrow 0$; $j \leftarrow 1$; $correction \leftarrow 0$
  while $j \leq w$ **do**
    $k \leftarrow j$; $t \leftarrow 1$; $adjust \leftarrow 0$
    while $(k < w - 1) \wedge (s_i[l_{ki}] = s_i[l_{k+1,i}])$ **do**
      $adjust \leftarrow adjust + a_{ki}$; increment k and t
    if $k > j$ **then**
      $adjusted \leftarrow (adjust + a_{ki})/t$; $correction \leftarrow correction + t^3 - t$
      for $m \leftarrow j$ to k **do** $I_i[m] \leftarrow (l_{mi}, \tilde{x}_{mi}, adjusted, correction)$
    else $I_i[j] \leftarrow (l_{ji}, \tilde{x}_{ji}, a_{ji}, correction)$
    $j \leftarrow j + t$
return $\mathcal{I} : \{I_1, \ldots, I_{|S|}\}$ with $I_i : (l_i, \tilde{x}_i, a_i, b_i)$

4.7. Experiments

4.7.1. Methodology

To evaluate our dependency estimators, i.e., MWP, KSP, CSP and our competitors, we look at how they behave w.r.t. an assortment of dependencies. See Figure 4.3; The dependencies are scaled to $[0, 1]$. [FB19] displays in appendix our algorithms for generating each dependency. For benchmarking, we repeatedly draw w objects with $|S|$ dimensions, to which we add Gaussian noise with standard deviation σ, which we call *noise level*. Intuitively, noise-free dependencies should lead to higher scores than noisier ones.

We also show that MCDE handles heterogeneity by simulating numerical, ordinal, and categorical attributes. To simulate ordinal attributes, we discretise numerical attributes into a number Ω of values from 1 to 20. To simulate categorical attributes, we randomly permute the discretised values, to mimic the absence of an order. As in other studies [NMV16; Res+11; KA14], we compute the statistical power, defined as follows:

Definition 4.9 (Power)**.** *The power of an estimator $\mathcal{E}$ w.r.t. dependency O with σ, w and $|S|$ is the probability of the score of $\mathcal{E}$ to be larger than a γ-th percentile of the scores w.r.t. the independent subspace* I*:*

$$\Pr\left[\mathcal{E}\left(Inst^{\mathrm{O},\sigma}_{w\times|S|}\right) > \left\{\mathcal{E}\left(Inst^{\mathrm{I},0}_{w\times|S|}\right)\right\}^{P_\gamma}\right]$$

$Inst^{\mathrm{O},\sigma}_{w\times|S|}$ is a random instantiation of a subspace as dependency O with noise level σ, which has w objects and $|S|$ dimensions. $\{x\}^{P_\gamma}$ stands for the γ-th percentile of the set $\{x\}$, i.e., a value v so that $\gamma\%$ of the values in $\{x\}$ are smaller than v.

The attributes of the independent subspace I are i.i.d. in $\mathcal{U}[0, 1]$. Note that, since the attributes of I are independent, adding noise does not have any effect on dependence, so we set noise to 0 when instantiating I. To estimate the power, we draw two sets of 500 estimates from O, σ and I, respectively:

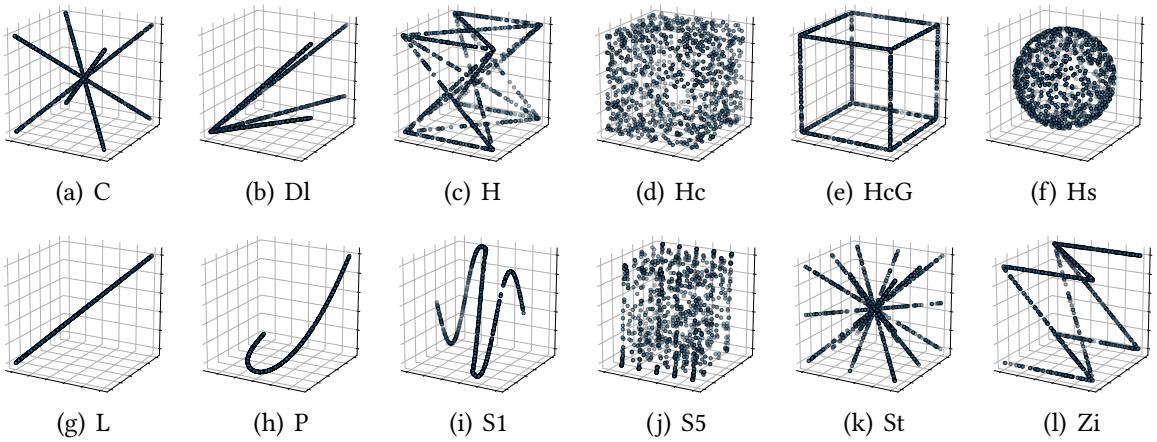
$$\Sigma^{\mathcal{E}}_{\mathrm{O},\sigma} : \left\{\mathcal{E}\left(Inst^{\mathrm{O},\sigma}_{w\times|S|}\right)\right\}_{i=1}^{500} \qquad \Sigma^{\mathcal{E}}_{\mathrm{I}} : \left\{\mathcal{E}\left(Inst^{\mathrm{I},0}_{w\times|S|}\right)\right\}_{i=1}^{500}$$

Then we count the elements in $\Sigma^{\mathcal{E}}_{\mathrm{O},\sigma}$ greater than $\left\{\Sigma^{\mathcal{E}}_{\mathrm{I}}\right\}^{P_\gamma}$:

$$\mathrm{power}^{\mathrm{O},\sigma,\gamma}_{w\times|S|}(\mathcal{E}) = \frac{\left|\left\{x : x \in \Sigma^{\mathcal{E}}_{\mathrm{O},\sigma} \ \wedge \ x > \left\{\Sigma^{\mathcal{E}}_{\mathrm{I}}\right\}^{P_\gamma}\right\}\right|}{500}$$

One can interpret 'power' as the probability to correctly reject the independence hypothesis with $\gamma\%$ confidence. I.e., the power quantifies how well a dependency measure can differentiate between the independence I and a given dependency O with noise level σ. For our experiments, we set $\gamma = 95$, $w = 1000$. We let the noise σ vary linearly from 0 to 1, with 30 distinct levels. We consider dependencies with dimensionality $|S|$ from 2 to 20.

In our figures, O_Ω denotes each dependency, where O stands for the dependency type (e.g., L stands for 'Linear'), and Ω is the discretisation level, i.e., the number of distinct values; O means that the attributes are numerical. '$|S|$D' indicates the dimensionality, and O^*_Ω indicates that the attributes are categorical, with a number Ω of nominal values.



(a) C (b) Dl (c) H (d) Hc (e) HcG (f) Hs

(g) L (h) P (i) S1 (j) S5 (k) St (l) Zi

Figure 4.3.: An assortment of 12 dependencies (displayed here with three dimensions, $\sigma = 0$). C: Cross, Dl: Double linear, H: Hourglass, Hc: Hypercube, HcG: Hypercube Graph, Hs: Hypersphere, L: Linear, P: Parabolic, S1: Sine (P=1), S5: Sine (P=5), St: Star, Zi: Z inversed.

4.7.2. Evaluation

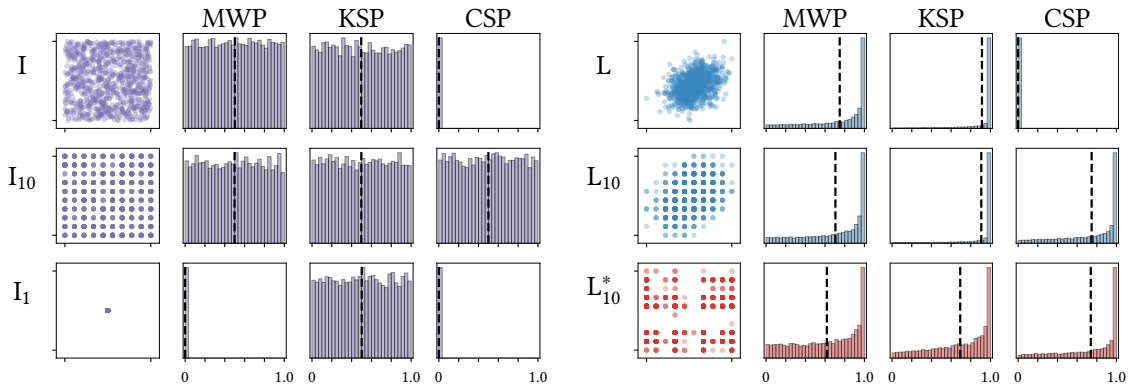
In our evaluation, we first compare our three estimators (MWP, KSP and CSP) together. Then, we compare MWP to our competitors and present our case study.

4.7.2.1. Specifics of Attributes Types

We first observe how MCDE handles numerical, ordinal, and categorical attributes. Figure 4.4 displays the empirical distribution of MWP, KSP and CSP iterations w.r.t. a 2-dimensional independent (I) and a linearly dependent (L) space. Each distribution is estimated as a histogram from 10 000 independent iterations. The vertical dashed lines show the means of the distributions, and we display a scatter plot illustrating the corresponding scenario.

According to Figure 4.4(a), MWP, KSP and CSP values are uniformly distributed in the case of an independent subspace, with a few exceptions: First, the CSP values are all close to 0 in the case of numerical attributes. Since each point is unique, they are all treated as a single category, and the Chi-Squared test does not have power in this setting. We observe the same effect with L (in 4.4(b)). Second, we see that the values of MWP and CSP are also 0 with I_1. This corresponds to the desired behaviour. In this situation, every observation in the subspace is equal, so contrast is undefined. KSP assumes that values are continuous, and thus does not handle this case.

Also, we can see from 4.4(b) that the KSP values are generally closer to 1 with L and L_{10}. This indicates a larger power than MWP and CSP. However, we can see that the CSP distribution does not change between L_{10} and L^*_{10}, while the mean of the MWP and KSP distribution decreases significantly. Thus, CSP detects dependency from categorical attributes better than MWP/KSP.



(a) Independent

(b) Dependent (Linear, $\sigma = 0.4$)

Figure 4.4.: Distribution of the contrast estimation iterations ($|S| = 2$).

4.7.2.2. Statistical Power of MWP, KSP and CSP

We first look at the statistical power of MWP, KSP and CSP with confidence level $\gamma = 0.95$ against a linear dependency of increasing dimensionality $|S|$, discretisation level Ω, and noise level σ. Please note that the figures are best seen in colour. The expectation is that the scores and statistical power are high for noiseless dependencies, i.e., the left side of the plot is blue, and decrease gradually as we add noise. A noise level $\sigma = 1$ is comparably high since the data is scaled to $[0, 1]$. Thus, the right side of the plot should be red, standing for low scores, or low power. From Figure 4.5 (upper part), we see that MWP without marginal restriction does not work well in high-dimensional and highly discretised spaces.

The marginal restriction alleviates this problem to some extent, in particular for numerical subspaces. In fact, as dimensionality increases, it becomes more and more likely that the points selected in the slice are 'at the centre' of the distribution. As a result, the mean of the point in the slice and the point outside of the slice become nearly equal, leading to low power with the Mann-Whitney U test, and thus a slight performance decrease of MWP. This calls for further research on the MWP slicing scheme, or alternatives to the Mann-Whitney U test. Nonetheless, the results indicate that MWP with marginal restriction works well against numerical attributes. Next, we see that KSP has high power in every case, although slightly decreasing with Ω. CSP does not work with numerical spaces but has more power in discrete spaces. CSP works best with categorical attributes.

We now compare the power of our estimators against the assortment of dependencies from Figure 4.3. CSP does not apply to numerical attributes. Thus, for comparability, we discretise the values with $\Omega = 10$. We can see from Figure 4.5 (lower part) that KSP consistently has more power than MWP, and even alleviate some of its drawbacks, such as the low power against the noiseless Hypercube (Hc). CSP generally has less power than KSP but can detect categorical dependencies.

Our experiments show that MWP has a slight performance decrease in high-dimensional discrete spaces. KSP seems to perform better, but its statistical power decreases with discretisation. Overall, we recommend to use KSP for numerical and ordinal data but to use CSP for categorical data. MWP still is a valid alternative with numerical attributes.

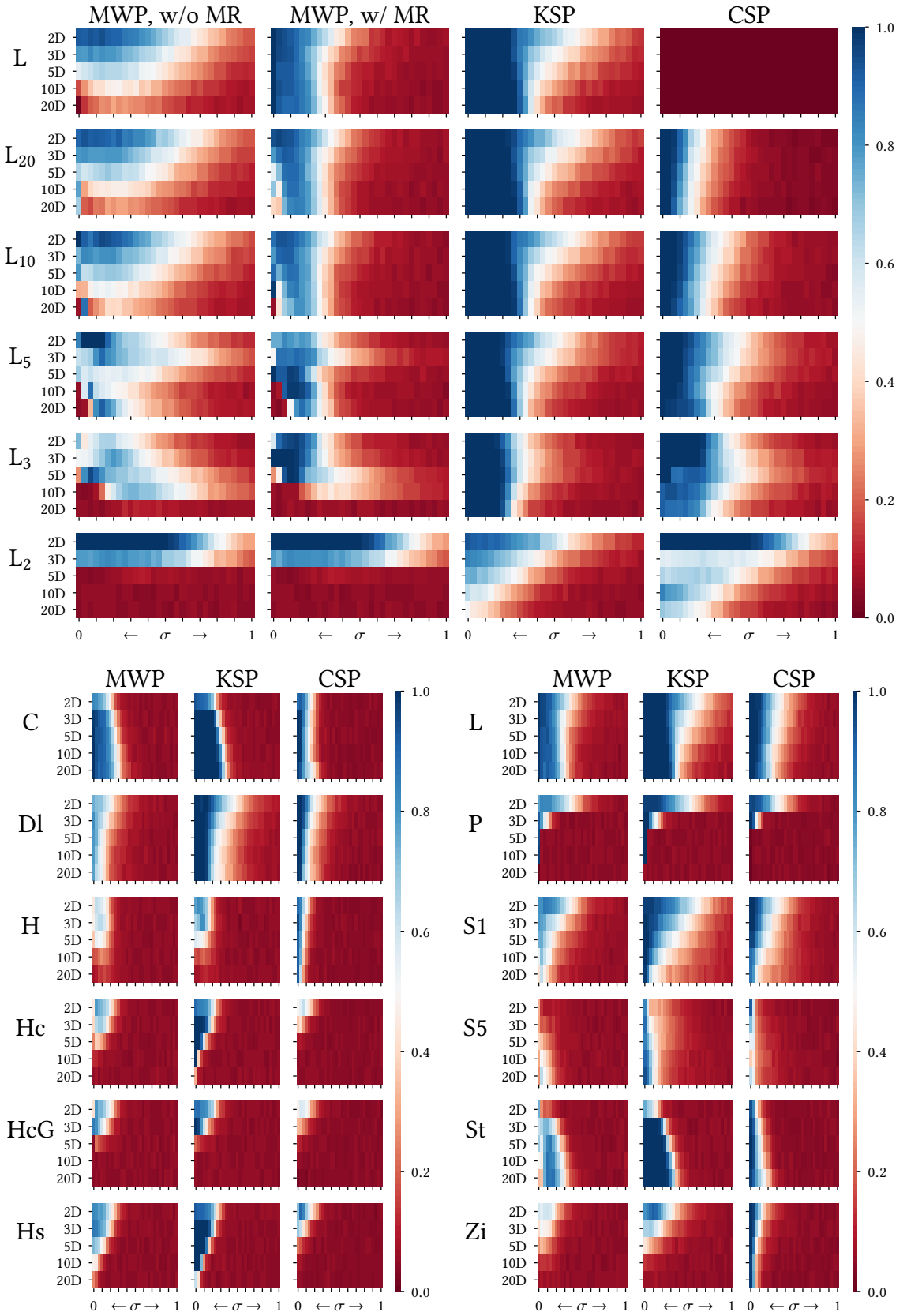


Figure 4.5.: Power of MWP/KSP/CSP against continuous/discrete linear distributions (upper part) and our assortment of dependencies (lower part).

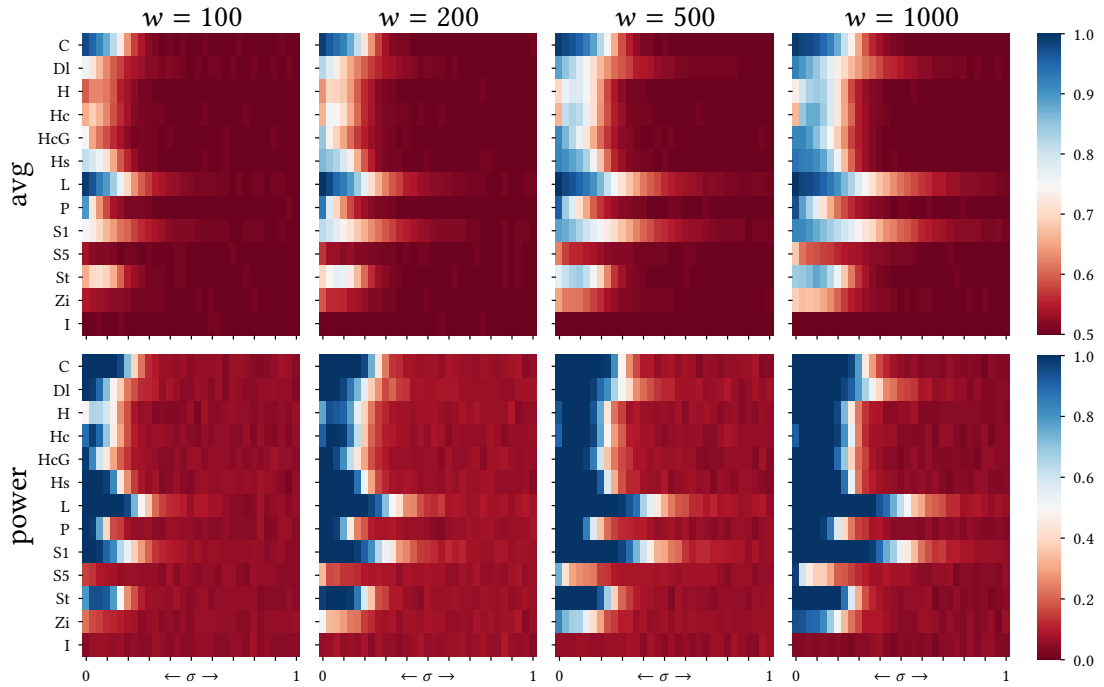


Figure 4.6.: Power of MWP w.r.t. w.

4.7.2.3. Influence of parameters, w, M and $|S|$

Figure 4.6 shows that power globally increases with w, but it is still high for most dependencies with low w, provided noise is moderate. As we can see, the average score of MWP tends to increase with w, which explains the gain in power. That is because MWP is sensitive (**R6**), as we discuss in Section 4.7.2.5. Similarly, Figure 4.6 shows that power increases slightly as M increases, but the effect is visible only for S5 and Zi. We can explain this increase of power easily by the fact that the standard deviation of MWP decreases, which is what Theorem 4.2 predicted: with more iterations, the values concentrate around C. In the end, we see that MWP is already useful for small w or small M, even though more iterations or more data samples yield higher power when data is noisy.

Figure 4.8 graphs the evolution of MWP for $|S| = 2, 3, 5$. As we see, the average MWP decreases gradually for each dependency. The same level of noise does not seem to affect each estimate equally, also regarding dimensionality. For instance, the estimates of L, P and S1 are larger at $|S| = 2$. While the estimates of Hc, HcG, P and Zi decrease with increasing $|S|$, they increase for C and St. While some dependencies (e.g., Hc, HcG) appear to have lower contrast without noise, we see that this effect is not as visible when we decrease the threshold for measuring power. The standard deviation of MWP increases with noise and decreases with $|S|$. In particular, L, C and Hs have a low standard deviation. This means that fewer iterations are required to estimate stronger dependencies. Power does not seem to vary much with dimensionality for most dependencies. It decreases with $|S|$ for Hc, HcG, Hs, P and Zi, while it increases for C, S5 and St.

All in all, each dependency yields a score larger than I up to a certain level of noise, leading to high power. Thus, MWP, and, by extension, MCDE, are general-purpose (**R2**).

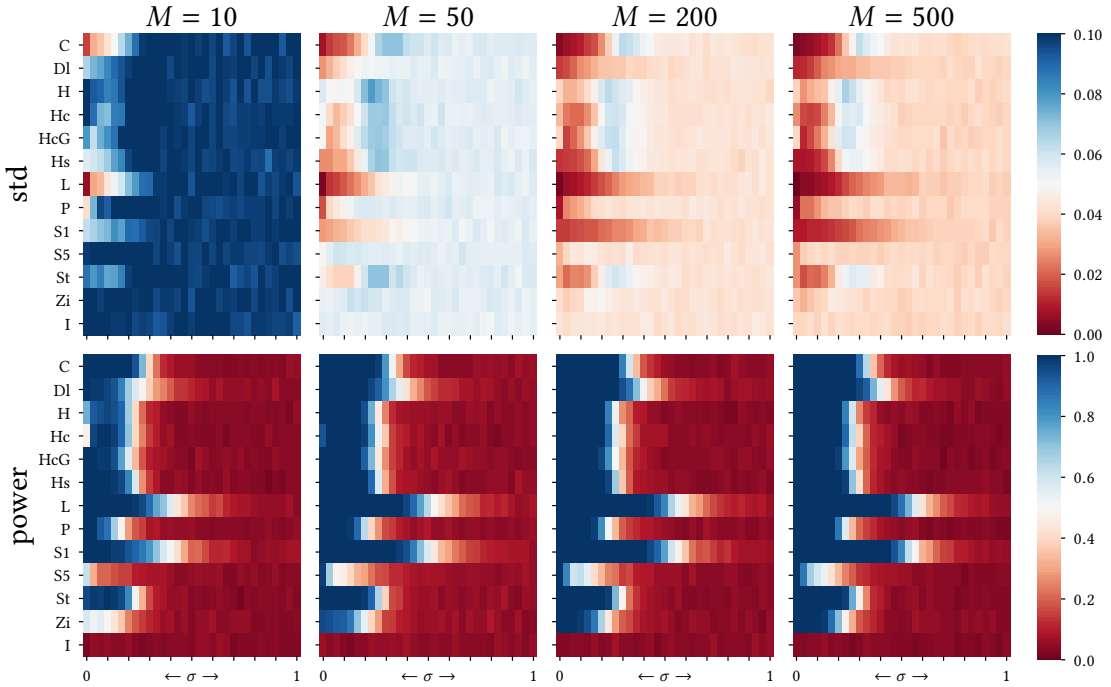


Figure 4.7.: Power of MWP w.r.t. M.

4.7.2.4. Comparison to Competing Approaches

We compare the score distribution for each competing approach with MWP in Figure 4.9. First, we see that the average score of MWP is most similar to Total Correlation [Wat60] (TC). However, TC is unbounded, and its scores follow a logarithmic scale. This means that the estimates of TC change very abruptly. We see that MWP scores are slightly smaller for noiseless dependencies with many ties w.r.t. marginal distributions, such as H and Hc, which we attribute to the correction for ties in the Mann-Whitney U test.

Interaction Information [McG54] (II) can yield positive or negative values. We visualise the absolute value of II with a logarithmic scale. We mark the dependencies which obtain a positive score in their noiseless form with a plus sign. II assigns high scores to every noiseless dependency. However, the score decreases rapidly with noise, except for L.

HiCS shows a similar behaviour as MWP, except that the scores decrease faster, and that many dependencies start with a relatively low score, even in the noiseless form, such as C, Dl, H, Hs, S5 and St. Next, Multivariate Spearman [SS07] (MS) and Universal Dependency Score [NMV16] (UDS) are restricted to monotonous and bijective functional relationships, respectively. They can detect only 3 out of 12 dependencies. Multivariate Maximal Correlation [Ngu+14b] (MAC) and Cumulative Mutual Information [Ngu+13] (CMI) behave curiously. Their scores change noticeably only for C, Dl, L, P and S1. The values of MAC also change abruptly and even non-monotonously with noise. For example, L and S1 obtain lower scores with a noise level of 0.3 than with higher noise levels. CMI evolves smoothly. However, for many dependencies, including I, the score increases again with more noise: the shades on the right are lighter, which shows a bias towards noise, independently of the underlying relationship.

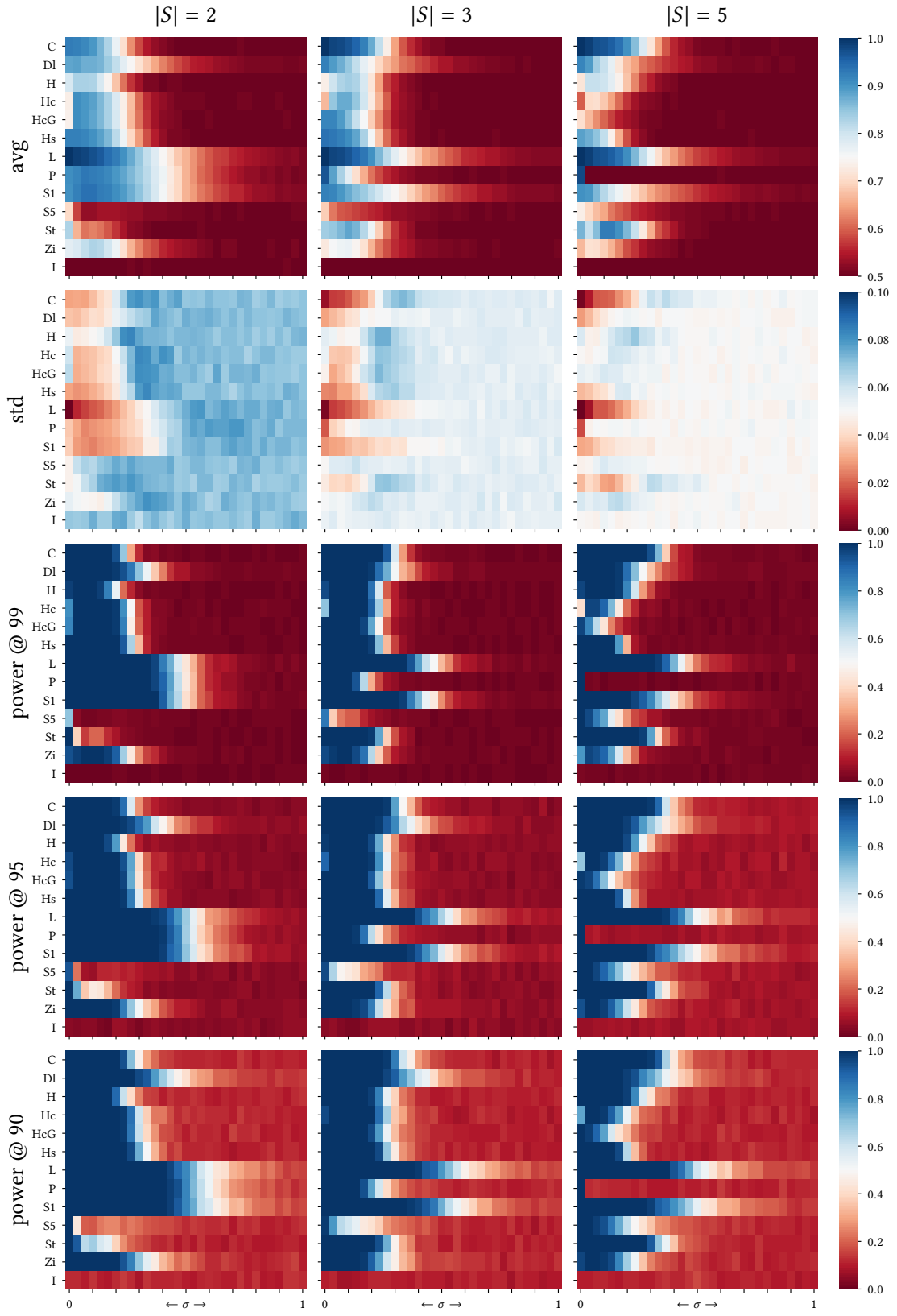


Figure 4.8.: MWP w.r.t. dimensionality $|S|$.

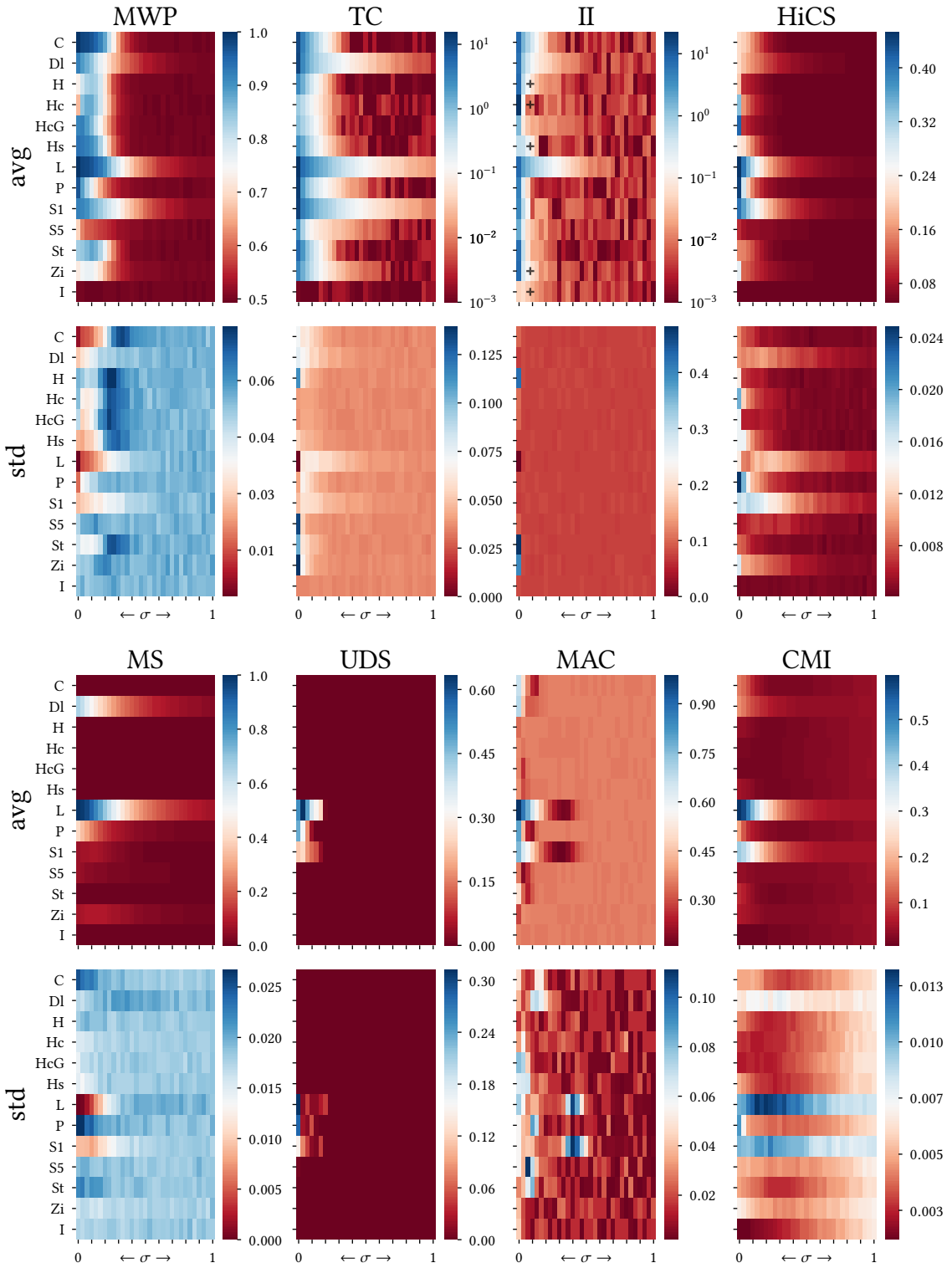


Figure 4.9.: Distribution of dependency estimation scores, $|S| = 3$.

By looking at MWP and MS, we see that the standard deviation behaves similarly: it decreases as the score increases. We observe the opposite for HiCS. The standard deviation of CMI reaches its highest level at a certain noise level, around 0.2 for L, and tends to increase slightly again with more noise.

While HiCS, UDS, MAC and CMI are expected to be in $[0, 1]$, the theoretical maximum or minimum is never reached, even if our benchmark features both strong and weak dependencies. On the other hand, MWP and MS exploit all the values of their range, being $[0.5, 1]$ and $[0, 1]$ respectively. Thus, they are interpretable (**R6**).

Figure 4.10 reveals that MWP, TC and HiCS achieve high power in any situation up to a certain extent of noise. MWP shows slightly more power with C, H, Hc, HcG, Hs and St. II can detect almost every dependency, but the power decreases rapidly with noise and dimensionality. MS detects Dl, L, P, S1, S5 and Zi, but misses all other dependencies. MAC looks unstable since its power evolves in a non-monotonous way and decreases with increasing dimensionality by much. In fact, it is not able to detect most dependencies for $|S| = 5$. UDS can only detect L, P and S1, a clear limitation. CMI has maximal power for each dependency and noise level for $|S| = 3$, which is unrealistic: CMI reaches its lowest score against the noiseless I, our baseline for power. This means that CMI does not clearly distinguish noise from dependence.

4.7.2.5. Sensitivity

R6 states that estimators should reflect the strength of the observed effect w.r.t. the number of observations. Figure 4.11 graphs the average score from 500 instances of each dependency with a minimal noise of 1/30. The average of MWP obtained for each dependency converges to 1 consistently with more samples, except for I, which stabilises around 0.5. This means that MWP is sensitive (**R6**).

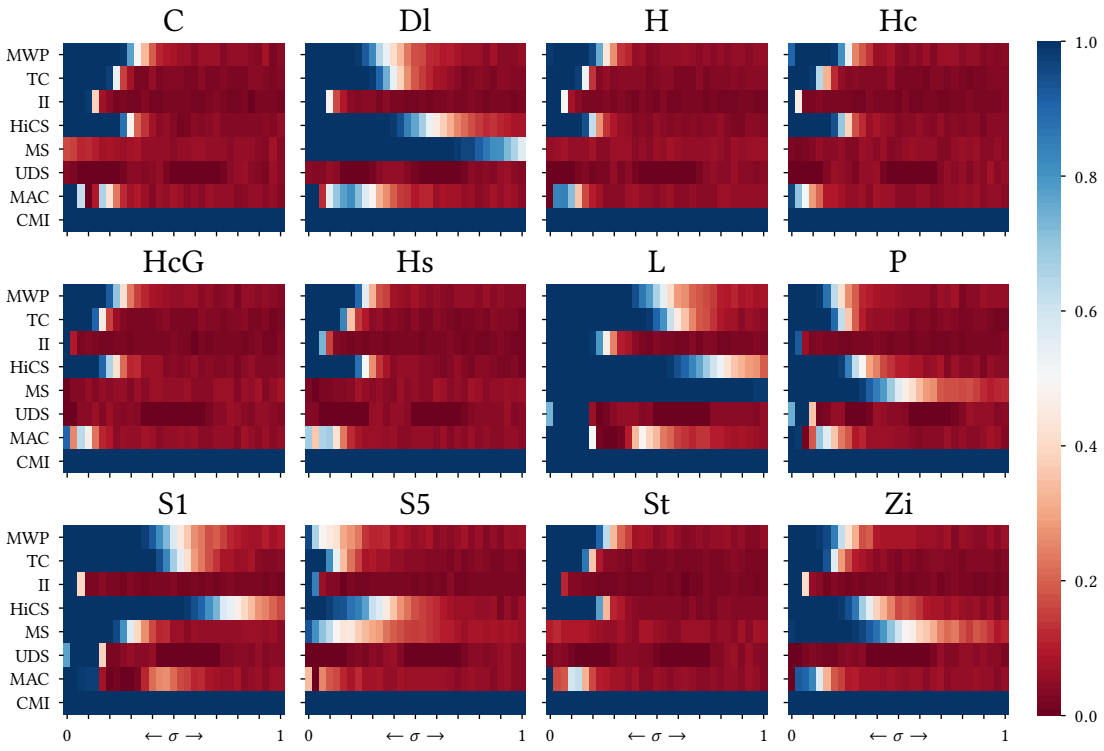
TC behaves similarly to MWP. However, it is not bounded. While the scores of II seem to increase with sample size, their absolute value decreases. MS is insensitive to changes in sample size. HiCS, UDS, MAC and CMI behave differently: their scores tend to go down as the sample size increases, even with I. This implies that their minimum or maximum score varies with the sample size, also highlighting interpretability (**R5**) problems.

4.7.2.6. Robustness

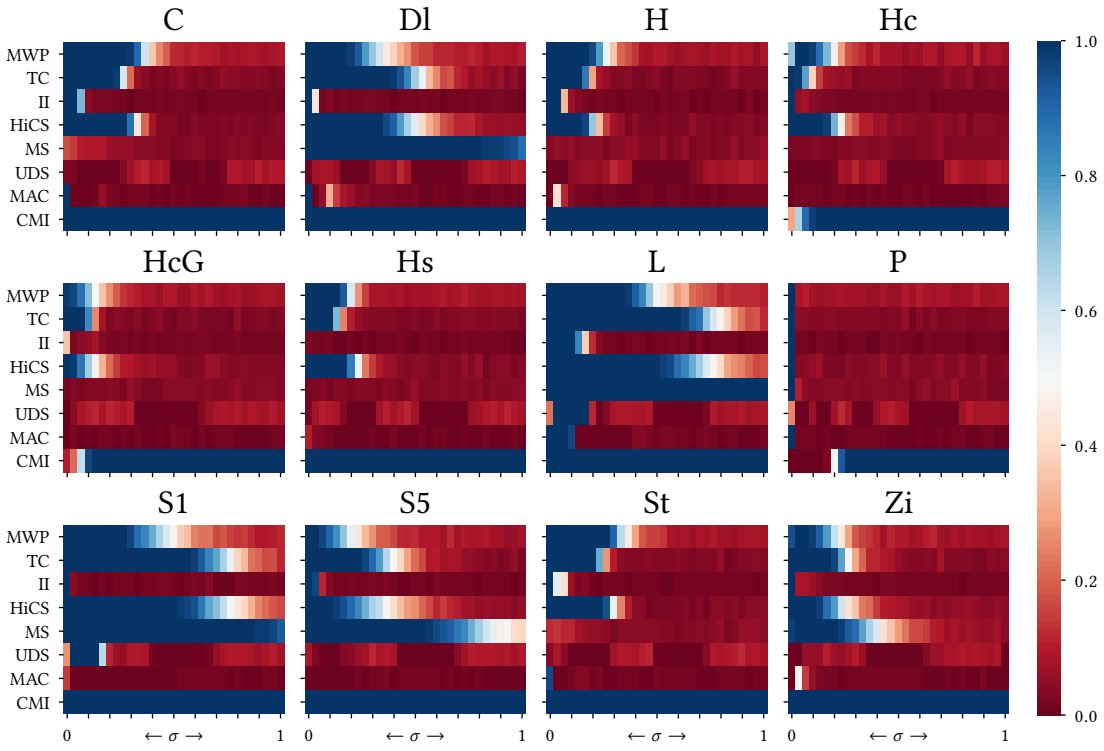
Data is often imperfect, i.e., values are rounded or trimmed. In some cases, this may lead to wrong estimates, e.g., an independent space is declared as strongly dependent.

We simulate data imperfections by discretising a 3-dimensional linear dependency into a number ω of discrete values from 100 to 1. With only one value, the space is entirely redundant, i.e., its *contrast* should be minimal. We compare the power of MWP and the other approaches against L and I for different levels of discretisation. Since TC and II base on local neighbourhoods, they do not work in this setting; We exclude them from the analysis. Figure 4.12(a) displays the results.

HiCS yields high power in the case of discrete values, even for I. Thus, HiCS is not robust. Also, the power of CMI wrongly increases as we add noise to I, provided that $\omega \geq 10$. This is why the power of CMI is high for every dependency in Figure 4.10. CMI



(a) $|S| = 3$



(b) $|S| = 5$

Figure 4.10.: Power against each dependency.

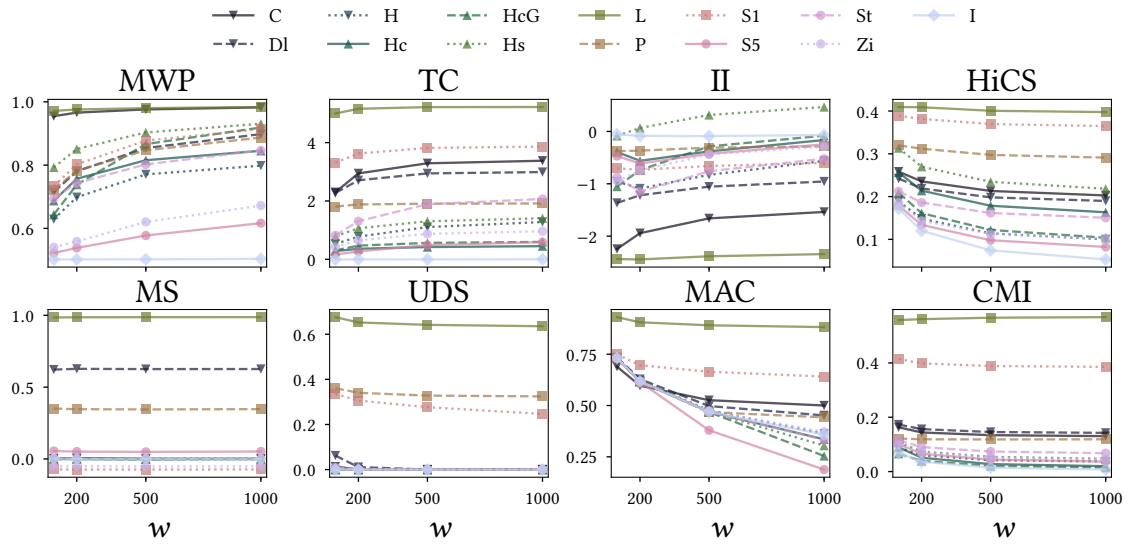
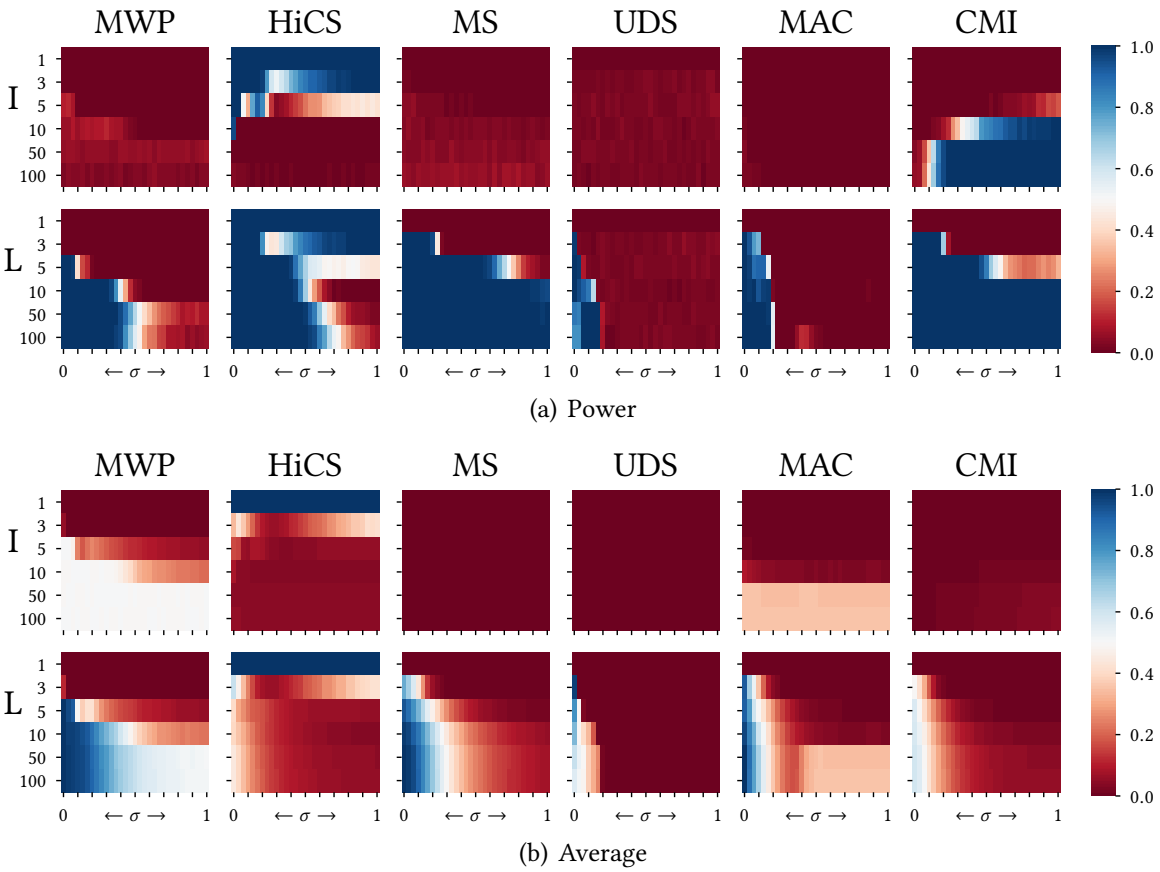


Figure 4.11.: Average score w.r.t. w, $\sigma = 1/30$.



(a) Power

(b) Average

Figure 4.12.: Power and average score w.r.t. ω.

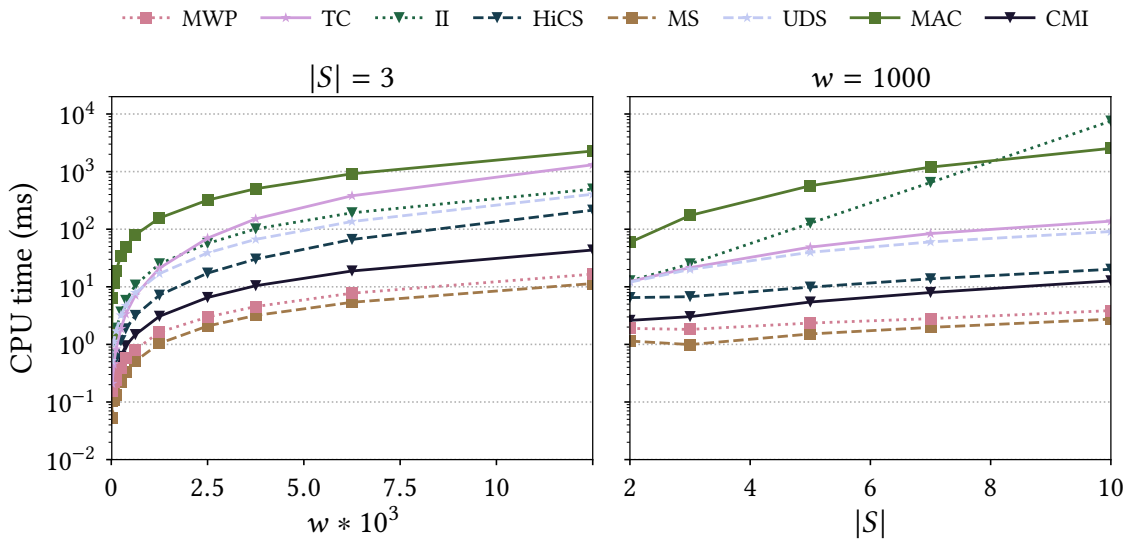


Figure 4.13.: Execution time w.r.t. w and $|S|$.

rejects the independence for independent spaces, i.e., it is not robust. On the other hand, MWP, MS, UDS and MAC seem robust (**R7**).

In Figure 4.12(b), we see that the score of CMI tends to increase slightly for I as we add noise, whenever $\omega > 5$. Also, the score of HiCS increases for both I and L when $\omega \leq 5$. MAC converges to 0.4 as noise increases for $\omega > 10$. On the other hand, MWP converges to 0 as the space becomes discrete. This is an interesting feature of our estimator: discrete spaces are of lower interest since the notion of *contrast* is not clearly defined there. It allows analysts to draw a line between discrete and real-valued attributes, characterizing their relevance for further analysis.

4.7.2.7. Scalability

We now look at the runtime requirements of our approach. We measure the average time for each estimator against 500 independent data sets with growing w and $|S|$. Note that which data set we use only has a marginal effect on the measured time. For consistency, we use instantiations of I for every estimator. Figure 4.13 graphs the results. As we can see, MWP is the second fastest after MS. HiCS and CMI scale relatively well with w and $|S|$. There is a second group formed by TC, II and UDS one order of magnitude slower. However, II does not scale well with $|S|$. MAC is way behind all others. One should note that the runtime of MWP can be further improved via parallelisation and prior indexing.

We evaluate the scalability of index construction for each approach, by increasing the size of the sliding window w from 10^2 to 10^5 in an independent space I with three dimensions. The red line ('Construction') is the average time for creating the index with window size w (Algorithms 4.2/4.5/4.8). The other lines show the average time to insert a new point into the window.

In Figure 4.14, we can see that the construction time of each index increases almost linearly with increasing window size w. The KSP index is less expensive to update than

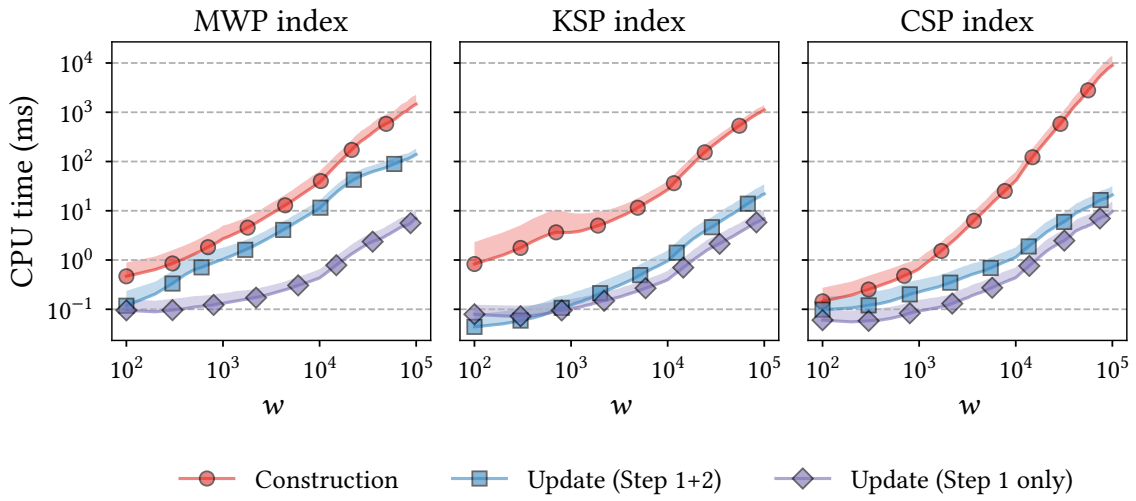


Figure 4.14.: Time required for index construction and update w.r.t. window size w.

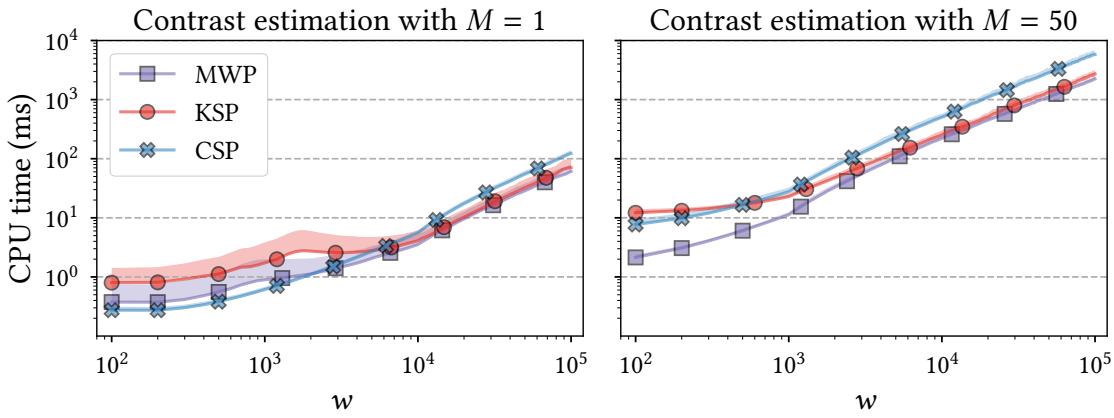


Figure 4.15.: Time required for contrast estimation w.r.t. window size w.

MWP, regarding Step 2 in particular. The first step of the CSP index update is very efficient, as it requires more or less constant time (see Table 4.2). We can see that only performing Step 1 in our update operations, while delaying Step 2 to the contrast estimation step, reduces the execution time by up to 3 orders of magnitude, compared to standard index construction. So we can significantly speed up the monitoring of MCDE contrast using the index update operations.

In Figure 4.15, we compare the execution time estimating MWP, KSP, and CSP, with increasing window size w. We can see that the three approaches have a comparable execution time. KSP is slightly slower for small window sizes because the $\overline{p}$-values are more difficult to obtain than with the other approaches. However, as the window size increases, KSP and MWP have the same execution time. CSP contrast estimation appears to be slightly slower as the window size increases but scales similarly.

4.7.2.8. Deployment to the Streaming Setting

We monitor contrast when the dependency gradually evolves. We simulate this setting by concatenating 100 three-dimensional linear dependencies with 1 000 observations each and a level of noise σ linearly increasing from 0 to 1. We use the approach described in Section 4.6.2, Equation 4.20 and instantiate MCDE with MWP. We estimate the dependency over a sliding window of size $w = 1\,000$ and with a decaying factor $\gamma = 0.99$.

We let Δ vary from 1 to 1 000 and M from 1 to 500. We compare each configuration to a baseline, which is the most expensive configuration ($\Delta = 1$, $M = 500$), without the benefit of our update operations. When $\Delta > 1$, we simply set the current contrast estimate to the value from the latest estimation. We define the following measures:

- The **Absolute Error** is the average absolute difference between the values obtained with the tested configuration and the values from the baseline.
- The **Relative Time** is the ratio of the time required by the tested configuration over the time required by the baseline.
- The **Index Speedup** is the ratio between the time required by the tested configuration without/with our index update operations.

We can see from Figure 4.16 that the absolute error decreases with Δ, while the relative execution time increases. The speedup obtained by our index operations is responsible for this gain in efficiency to a large part. As we increase the number of iterations M, the errors tend to decrease, but at the same time, contrast estimation dominates the overall execution time. In such cases, we see less benefit from our efficient insert/delete operations.

We identify the configuration $M = 50$, $\Delta = 50$ as a sweet spot: for an absolute error as small as ≈ 0.01, the computational burden is reduced by up to 100 times, with a consistent index speedup of 3. We mark this configuration with a star $*$ in the figure.

4.7.3. Case Study: Discovering Dependency Patterns

We now apply MCDE to a real-world multivariate time series. The data was collected during a 4-day production campaign at Bioliq, a pyrolysis plant in the surroundings of Karlsruhe [Pfi+16]. It contains one measurement per second, i.e., 345 600 observations, from a selection of 20 physical sensors in various components of the plant. We monitored the evolution of dependency between each sensor pair with $w = 1000$, $\Delta = 50$, $M = 50$, as just explained. We obtained the evolution of dependency between the 20 sensors (i.e., 190 pairs) using a single CPU core in about 2 hours. Note that it would be easy to shorten the computation time significantly by parallel processing.

We have presented the results of our monitoring technique to the plant operators. They have identified several patterns which they deemed ‘interesting’, i.e., patterns yielding insights that could help with plant operation.

Figure 4.17 displays one of these patterns. It is the result of monitoring two sensors, namely the pressure at the reactor input, and the oxygen concentration at the output. As we see, the dependency between these two measures changes significantly over time. Some changes, marked in the figure from 1 to 4, appear to represent different stages in the production process. A better understanding of the dynamics of the physical measures involved in the reaction will help the plant owners to operate smoothly and efficiently.

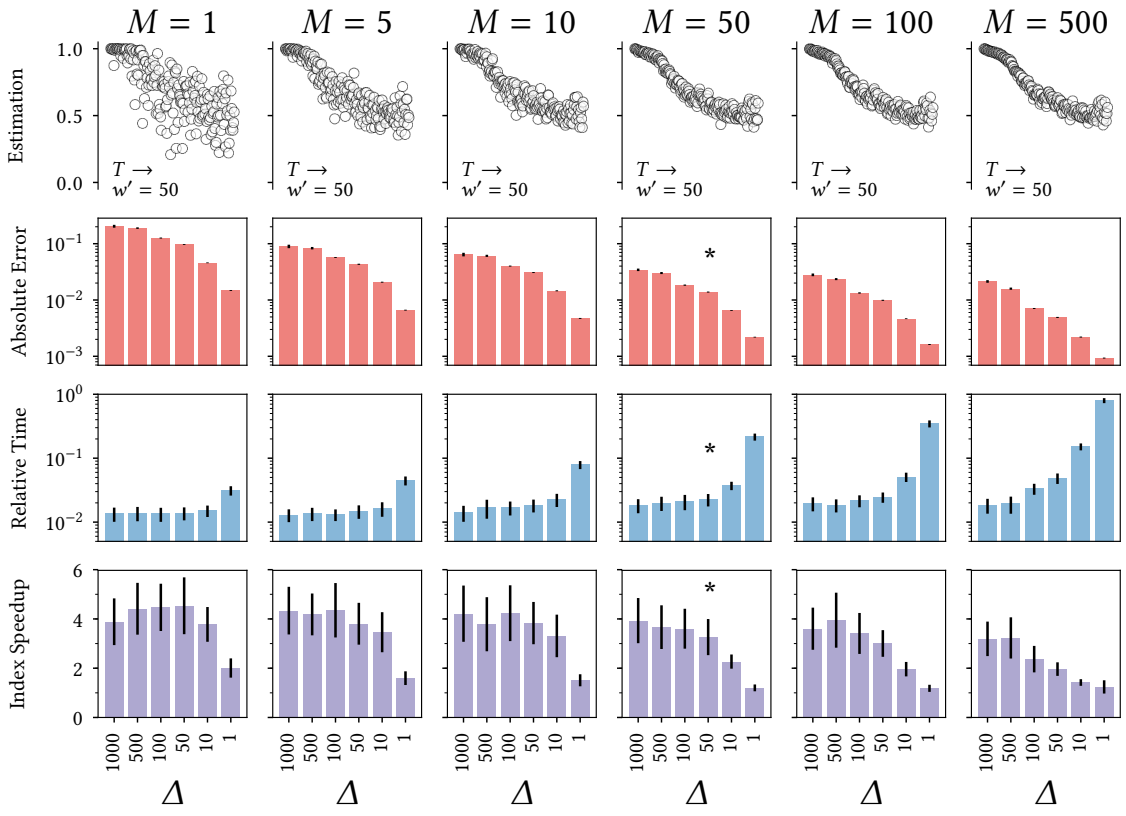


Figure 4.16.: Performance of contrast estimation with *concept drift* (* ≡ sweet spot).

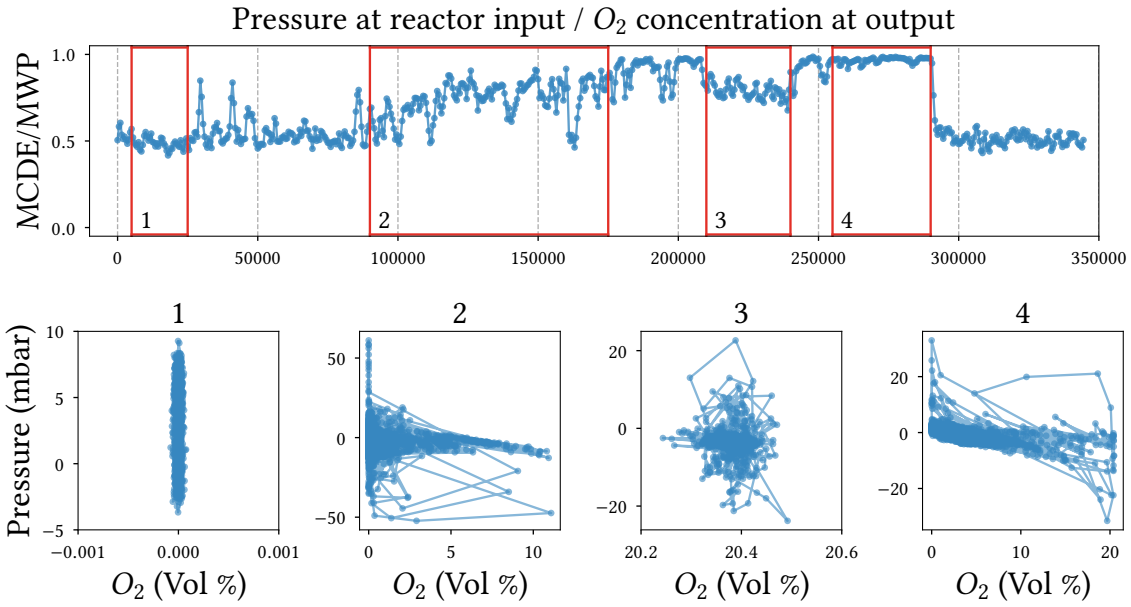


Figure 4.17.: Example of an interesting dependency pattern in the pyrolysis data.

4.8. Discussion

Our experiments show that MCDE fulfils all requirements linked to heterogeneous data streams and has the desirable features of a framework for dependency estimation.

First, we can see that MCDE is efficient (**C1**), and, thanks to the index update operations, one can use it in combination with the sliding window model to mine data streams in a single scan (**C2**) and adapt (**C3**) the contrast estimation over time, taking *concept drift* into account. Second, one can also reduce or increase the number of MC iterations M to trade between accuracy and computation time, in an anytime (**C4**) fashion. MCDE can handle heterogeneity (**C5**) via the instantiation of various statistical tests.

Each approach, except MCDE and MS, has at least one unintuitive parameter (**R3**): TC and II require $k \in \mathbb{N}$, CMI requires $Q \in \mathbb{N}$, MAC requires $\epsilon \in (0, 1)$, UDS requires $\beta \in \mathbb{N}$, HiCS requires $\alpha \in (0, 1)$. Next, only MCDE and HiCS allow trading accuracy for a computational advantage (**C4**). Note that, by adapting recent anytime estimators for Mutual Information (MI) such as [VB19], TC and II maybe potentially also fulfil **C4**.

Last, MCDE is non-parametric (**R4**) and interpretable (**R5**) by design. Our experiments against an assortment of dependencies show that it is multivariate (**R1**), general-purpose (**R2**) and robust (**R7**). MCDE is sensitive (**R6**) because estimates are the average of $\overline{p}$-values. Table 4.3 summarises our findings.

Table 4.3.: Fulfilment of Constraints and Requirements.

Estimator	C1	C2	C3	C4	C5	R1	R2	R3	R4	R5	R6	R7
MS	++	✗	✗	✗	✗	✓	✗	✓	✓	✓	✗	✓
TC	-	✓	✓	✓	✗	✓	✓	✗	✓	✗	✓	✗
II	--	✓	✓	✓	✗	✓	✗	✗	✓	✗	✗	✗
CMI	+	✗	✗	✗	✗	✓	✗	✗	✓	✗	✗	✗
MAC	--	✗	✗	✗	✗	✓	✗	✗	✓	✗	✗	✓
UDS	-	✗	✗	✗	✗	✓	✗	✗	✓	✗	✗	✓
HiCS	+	✗	✗	✓	✗	✓	✓	✗	✓	✗	✗	✗
MCDE	++	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

All in all, MCDE yields to state-of-the-art estimators: it is versatile, allowing quality-runtime trade-offs and parallelisation, which is useful when time is critical, e.g., in large data streams. At the same time, it shows excellent detection quality with no restriction on the dependency type, while being easy to use and interpret. MCDE features a blend of properties that so far no competitor offers.

In this chapter, we have described a framework to estimate multivariate dependency in heterogeneous data streams. It fulfils all requirements one would expect from a state-of-the-art dependency estimator. Compared to other approaches, it provides high statistical power on a large panel of dependencies, while being very efficient. Furthermore, we introduced index operations for the streaming setting and illustrated the benefits of our framework against a real-world use case. In the next part, we generalise the task of dependency estimation to monitoring large sets of statistics.

Part III.

Monitoring

5. Scaling Multi-Armed Bandit Algorithms

The content of this chapter bases on the following publication:

- Edouard Fouché, Junpei Komiyama and Klemens Böhm. 'Scaling Multi-Armed Bandit Algorithms'. In: *KDD*. ACM, 2019, pp. 1449–1459. DOI: 10.1145/3292500.3330862

Keywords: Bandit Algorithms; Thompson Sampling; Adaptive Windowing

5.1. Chapter Overview

The Multi-Armed Bandit (MAB) is a fundamental model capturing the dilemma between exploration and exploitation in sequential decision-making. At every time step, the decision-maker selects a set of arms and observes a reward from each of the chosen arms. In this chapter, we present a variant of the problem, which we call the Scaling Multi-Armed Bandit (S-MAB): the goal of the decision-maker is not only to maximise the cumulative rewards, i.e., choosing the arms with the highest expected reward, but also to decide how many arms to select so that, in expectation, the cost of selecting arms does not exceed the rewards. This problem is relevant to many real-world applications, e.g., online advertising, financial investments or data stream monitoring. In Section 2.2.3, we introduced the required notation to describe and analyse our algorithms. This chapter makes the following contributions:

We address the S-MAB problem, a novel generalisation of the MAB problem. The novelty is that the decision-maker not only decides which arms to play, but also how many, to maximise her cumulative reward under an efficiency constraint. To our knowledge, we are the first to consider this setting.

We first propose Scaling Thompson Sampling (S-TS), an algorithm to solve this problem in the static setting, i.e., when the distribution of rewards does not change over time. We leverage existing bandit algorithms, e.g., Thompson Sampling [Tho33] (TS), and show that the regret of our method (i.e., the difference from the outcome of a perfect oracle) only grows logarithmically with the number of time steps.

Then, we generalise our method for the non-static setting. To do so, we combine our algorithm with Adaptive Windowing [BG07] (ADWIN), a state-of-the-art change detector, which is at the same time efficient and offers theoretical guarantees.

Finally, we validate our findings via experiments. We illustrate the benefits of our contribution via a real-world use case on predictive maintenance. The comparison with existing approaches shows that our method achieves state-of-the-art results. – We release our source code and data on GitHub[1], to ensure reproducibility.

[1] https://github.com/edouardfouche/S-MAB

5.2. Scaling Thompson Sampling

Let us first assume a static environment. Our algorithm consists of:

1. A Multiple-Play Multi-Armed Bandit (MP-MAB) to identify the top-L_t arms (*1).
2. A so-called 'scaling policy', to determine the value of L_{t+1} based on L_t and the observations at time t (*2).

For (1), we use an existing algorithm, Multiple-Play Thompson Sampling [KHN15] (MP-TS). It is a Bayesian-inspired bandit algorithm, which maintains a Beta posterior with parameters α_i, β_i over each arm i. In each round, MP-TS samples an observation θ_i from each posterior and selects the top-L_t arms according to these observations. Then, the parameters of this posterior are adjusted based on the reward vector $X(t)$.

For (2), we propose to use a scaling policy, i.e., a strategy to control the number of plays, such that the empirical efficiency $\hat{\eta}_t$ remains larger than η^*. Whenever $\hat{\eta}_t \leq \eta^*$, we 'scale down', i.e., we set $L_{t+1} = L_t - 1$. Otherwise, we 'scale up'. When we are confident that adding one arm will lead to $\hat{\eta}_t \leq \eta^*$, we stop scaling. To do so, we estimate B_t, an upper confidence bound for $\hat{\eta}_{t+1}$, assuming that $L_{t+1} = L_t + 1$. $\hat{B}_t$ is our estimator for B_t, based on the observations from the environment so far. The confidence is derived from the Kullback-Leibler divergence, as the so-called Kullback-Leibler UCB (KL-UCB) index [GC11; Mai17]. We name our policy Kullback-Leibler Scaling (KL-S):

$$L_{t+1} = \begin{cases} L_t - 1 & \text{if } \hat{\eta}_t \leq \eta^* \\ L_t + 1 & \text{if } \hat{\eta}_t > \eta^* \text{ and } \hat{B}_t > \eta^* \\ L_t & \text{otherwise} \end{cases} \tag{5.1}$$

where $1 \leq L_{t+1} \leq K$ and

$$\hat{\eta}_t = \frac{1}{L_t} \sum_{i}^{I(t)} \hat{\mu}_i \qquad \hat{B}_t = \frac{L_t}{L_t + 1} \hat{\eta}_t + \frac{1}{L_t + 1} b_{\widehat{L_t+1}}(t) \tag{5.2}$$

$\hat{B}_t$ is the empirical estimator of

$$B_t = \frac{1}{L_t + 1} \sum_{i=1}^{L_t} \mu_i(t) + \frac{1}{L_t + 1} b_{L_t+1}(t).$$

where $b_i(t)$ is the KL-UCB index of arm i and $\widehat{L_t + 1}$ is the arm of the $(L_t + 1)$-th largest index. The KL-UCB index is as follows:

$$b_i(t) = \max_q \{N_i(t) \cdot d_{\mathrm{KL}}(\hat{\mu}_i(t), q) \leq \log(t/N_i(t))\} \tag{5.3}$$

where d_{KL} is the Kullback-Leibler divergence. Intuitively, our policy maximises L_t such that the empirical efficiency $\hat{\eta}_t$ remains larger than η^* at any time t. We stop scaling up whenever $\hat{B}_t$, which is an upper confidence bound of $\hat{\eta}_t$, is greater than η^*.

In S-TS, the algorithms of (1) MP-TS and (2) KL-S are intertwined. See Algorithm 5.1. S-TS successively calls the two procedures MP-TS and KL-S, while maintaining the statistics

$N_i(t), S_i(t)$ for each arm. We initialise the scaling policy with $L_1 = K$ (Lines 1-2). The rationale is that nothing is known about the reward distribution of the arms initially, so pulling a maximum number of arms is informative. Previous studies showed that optimistic initialisation improves the empirical performance of bandit algorithms [KP14; SB18].

Computational complexity of Algorithm 5.1: At each round, MP-TS draws a sample from a Beta distribution (Line 12), and KL-S computes the KL-UCB index for each arm (Line 25), which can be done efficiently via Newton's method. Given that these operations are done in constant time and that finding the top-L_t elements among K elements takes $O(K \log K)$ in the worst case ($L_t = K$), each round of the proposed algorithm takes $O(K \log K)$ time. Therefore, the total computational complexity of the algorithm is $O(TK \log(K))$. The space complexity of the algorithm is $O(K)$, as it only keeps four statistics (α_i, β_i, N_i, S_i) per arm $i \in [K]$.

Algorithm 5.1 S-TS($[K], \eta^*$)

Require: Set of arms $[K] = \{1, 2, \dots, K\}$, target efficiency η^*

1: $\alpha_i(1) = 0, \beta_i(1) = 0 \quad \forall i \in [K]$
2: $N_i(1) = 0, S_i(1) = 0 \quad \forall i \in [K]$
3: $L_1 \leftarrow K$
4: **for** $t = 1, 2, \dots, T$ **do**
5: $\quad I(t), X(t) \leftarrow$ MP-TS(L_t) ▷ Play L_t arms (as in MP-TS)
6: $\quad$ **for** $i \in I(t)$ **do**
7: $\quad\quad N_i(t+1) = N_i(t) + 1$
8: $\quad\quad S_i(t+1) = S_i(t) + X_i(t)$
9: $\quad L_{t+1} \leftarrow$ KL-S(L_t) ▷ Scale L_t for the next round

10: **procedure** MP-TS(L_t)
11: $\quad$ **for** $i = 1, \dots, K$ **do**
12: $\quad\quad \theta_i(t) \sim \text{Beta}(\alpha_i(t) + 1, \beta_i(t) + 1)$
13: $\quad$ Play arms $I(t) := \arg\max_{K' \subset [K], |K'| = L_t} \sum_i^{K'} \theta_i(t)$
14: $\quad$ Observe reward vector $X(t)$
15: $\quad$ **for** $i \in I(t)$ **do** ▷ Update parameters
16: $\quad\quad \alpha_i(t+1) = \alpha_i(t) + X_i(t)$
17: $\quad\quad \beta_i(t+1) = \beta_i(t) + (1 - X_i(t))$
18: $\quad$ **return** $I(t), X(t)$

19: **procedure** KL-S(L_t)
20: $\quad S_i = S_i(t+1), N_i = N_i(t+1) \quad \forall i \in [K]$
21: $\quad \hat{\mu}_i = S_i / N_i \quad \forall i \in [K]$ ▷ $\hat{\mu}_i = 1$, if $N_i = 0$
22: $\quad \hat{\eta}_t = \sum_{i \in I(t)} \hat{\mu}_i$
23: $\quad$ **if** $\hat{\eta}_t \leq \eta^*$ **then return** $\max(L_t - 1, 1)$ ▷ Scale down
24: $\quad$ **else**
25: $\quad\quad KL = \left\{ \max_q \{ N_i \cdot d_{\text{KL}}(\hat{\mu}_i, q) \leq \log \frac{t+1}{N_i} \} : \forall i \in [K] \right\}$
26: $\quad\quad b_{\widehat{L_t+1}} = (L_t + 1)$-th largest element from KL
27: $\quad\quad \hat{B}_t = \frac{L_t}{L_t+1} \hat{\eta}_t + \frac{1}{L_t+1} b_{\widehat{L_t+1}}(t)$
28: $\quad\quad$ **if** $\hat{B}_t > \eta^*$ **then return** $\min(L_t + 1, K)$ ▷ Scale up
29: $\quad\quad$ **else return** L_t ▷ Do not scale

5.3. Scaling Thompson Sampling with ADWIN

To handle the non-static setting (*3), we combine S-TS with ADWIN [BG07]. We call the resulting algorithm Scaling Thompson Sampling with ADWIN (S-TS-ADWIN).

ADWIN (Algorithm 5.2) monitors the expected value from a single (virtually infinite) stream of values $\{x_1, x_2, \dots\}$, where $x_i \in [0, 1]$. ADWIN maintains a window W of varying size $|W| = w$ so that the nature of the stream is consistent. ADWIN reduces the size of the window whenever two neighbouring subwindows have different mean, based on a statistical test with confidence δ. For every two subwindows of size $|W_1| + |W_2| = |W|$ with corresponding means $\hat{\mu}_{W_1}, \hat{\mu}_{W_2}$, ADWIN shrinks the windows to W_2 if

$$|\hat{\mu}_{W_1} - \hat{\mu}_{W_2}| \geq \epsilon^{\delta}_{\text{cut}} \quad \text{where} \quad \epsilon^{\delta}_{\text{cut}} = \sqrt{\frac{1}{2m} \log\left(\frac{4|W|}{\delta}\right)} \tag{5.4}$$

and $m = 1/((1/|W_1|) + (1/|W_2|))$. [BG07] showed that ADWIN efficiently adapts to both gradual and abrupt changes with theoretical guarantees (see Theorem 3.1 therein).

Algorithm 5.2 ADWIN(B, δ)

Require: Stream of values $B = (x_1, x_2, \dots)$, confidence level δ

1: $W \leftarrow \{\}$
2: **for** $t = 1, 2, \dots$ **do**
3: $\quad W \leftarrow W \cup \{x_t\}$
4: $\quad$ Drop elements from the tail of W until $|\hat{\mu}_{W_1} - \hat{\mu}_{W_2}| < \epsilon^{\delta}_{\text{cut}}$ holds for every split $W = W_1 \cup W_2$.

The idea behind S-TS-ADWIN, showed in Algorithm 5.3, is to create an ADWIN instance A_i per arm i. At each step t, A_i obtains as input the reward from the corresponding arm $X_i(t)$ if $i \in I(t)$. Thus, each instance A_i maintains a time window W_i of variable size, which shrinks whenever ADWIN detects a change in μ_i.

However, for any bandit algorithm with logarithmic regret, the number of plays of suboptimal arms grows with $\log(T)$. That is, after some time, the A_j of any suboptimal arm j does not obtain any input, and thus no change can be detected for arm j.

Thus, we use $w_t = \min\{|W_i| : \forall i \in [K]\}$, i.e., the smallest window from each A_i, to estimate the statistics of any arm $i \in [K]$ at each step t. Here, we implicitly assume that the change points are 'global', i.e., that they are shared across the $\mu_i, \forall i \in [K]$. In principle, changes may also be 'local', e.g., a single μ_i changes. But we will show that despite this assumption, it works well in practice.

By default, we set $\delta = 0.1$ for each instance, since [BG07] showed that it leads to a very low empirical false positive rate and good performance. We show in our experiments that this parameter does not have a significant impact on our results and that S-TS-ADWIN performs very well against synthetic and real-world scenarios.

Computational complexity of Algorithm 5.3: We use the improved version of ADWIN, dubbed ADWIN2 [BG07]. For a window of size w, ADWIN2 takes $O(\log w)$ time per object. Since we have K instances of them, the time complexity of the ADWIN2 part is in $O(K \log w) = O(K \log T)$ per round. The space complexity of ADWIN2 is in $O(w)$, but the window typically shrinks rapidly in the case of a non-static environment. We show in our experiments that the scalability of S-TS-ADWIN is almost the same as S-TS.

Algorithm 5.3 S-TS-ADWIN($[K], \eta^*, \delta$)

Require: Set of arms $[K]$, target efficiency η^*, delta δ

$\alpha_i(1) = 0, \beta_i(1) = 0 \quad \forall i \in [K]$
$\alpha_i(1) = 0, \beta_i(1) = 0 \quad \forall i \in [K]$
$N_i(1) = 0, S_i(1) = 0 \quad \forall i \in [K]$
$A_i \leftarrow$ instantiate ADWIN with parameter $\delta, \forall i \in [K]$
$L_1 \leftarrow K$
for $t = 1, 2, \ldots, T$ **do**
  $I(t), X(t) \leftarrow$ MP-TS(L_t) ▷ Play L_t arms (as in MP-TS)
  for $i \in I(t)$ **do**
    $N_i(t+1) = N_i(t) + 1$
    $S_i(t+1) = S_i(t) + X_i(t)$
    Add $X_i(t)$ into A_i
  $L_{t+1} \leftarrow$ KL-S(L_t) ▷ Scale L_t for the next round
  $w_t \leftarrow \min\{|W_i| : \forall i \in [K]\}$ ▷ Keep the smallest window
  $N_i(t+1) = \sum_{j=t-w_t}^{t} \mathbf{1}(i \in I(j))$
  $S_i(t+1) = \sum_{j=t-w_t}^{t} X_i(j) * \mathbf{1}(i \in I(j))$
  $\alpha_i(t+1) = S_i(t+1), \beta_i(t+1) = N_i(t+1) - S_i(t+1)$

5.4. Theoretical Analysis

We analyse the properties of scaling bandits. In particular, we measure the capability of an algorithm to control the size of L_t by introducing a quantity called 'pull regret'. Our analysis is general: we show that not only S-TS (Algorithm 5.1) but that KL-S, combined with any MP-MAB algorithm of logarithmic regret, has logarithmic pull regret. We introduce our notation in Section 5.4.1 and proceed to our main theorem in Section 5.4.2.1.

5.4.1. The General Scaling Bandit

We assume there is a unique L^*, such that $\sum_{i=1}^{L^*} \mu_i / L^* > \eta^*$ and $\sum_{i=1}^{L^*+1} \mu_i / (L^*+1) < \eta^*$. Let $\Delta = \min(\Delta_a, \Delta_b)$ be the 'gap', i.e., the absolute difference between η^* and the closest possible η_t, with $\Delta_a = (\sum_{i=1}^{L^*} \mu_i - \eta^*) > 0$ and $\Delta_b = (\eta^* - \sum_{i=1}^{L^*+1} \mu_i) > 0$.

Let us first generalise S-TS in Algorithm 5.4. A 'base bandit' (MP-Base-Bandit, Line 4) is an abstract bandit algorithm that, given the reward information up to the last round and the current number of plays L_t, decides on $I(t)$, i.e., which arms to draw, and returns the reward vector $X(t)$ at each round t.

Algorithm 5.4 General Scaling Bandit ($[K], \eta^*$)

Require: Set of arms $[K]$, target efficiency η^*

$N_i(1) = 0, S_i(1) = 0 \quad \forall i \in [K]$
$L_1 \leftarrow K$
for $t = 1, 2, \ldots, T$ **do**
  $I(t), X(t) \leftarrow$ MP-Base-Bandit(L_t) ▷ Play L_t arms
  for $i \in I(t)$ **do**
    $N_i(t+1) = N_i(t) + 1$
    $S_i(t+1) = S_i(t) + X_i(t)$
  $L_{t+1} \leftarrow$ KL-S(L_t) ▷ Scale L_t for the next round

If we set MP-Base-Bandit ≡ MP-TS, then Algorithm 5.4 becomes Algorithm 5.1. As an alternative to MP-TS, one could consider for example Multiple-Play Kullback-Leibler UCB [GC11; KHN15] (MP-KL-UCB) (resp. Combinatorial UCB [Che+16] (CUCB)) that draws the top-L_t arms in terms of the KL-UCB indices (resp. Upper Confidence Bound [Aue+02] (UCB) indices) or Exp3 with Multiple plays [UNK10] (Exp3.M) that uses exponential weighting.

To evaluate whether our scaling strategy converges to the optimal number of pulls L^*, we define a new notion of 'pull regret' as the absolute difference between the number of pulls L_t and L^*:

$$\mathrm{PReg}(T) = \sum_{t=1}^{T} |L^* - L_t| \tag{5.5}$$

The 'standard' multiple-play regret, with varying L_t, measures how many suboptimal arms the algorithm draws. It is defined as:

$$\mathrm{Reg}(T) = \sum_{t=1}^{T} \left[\max_{\mathcal{I}\subseteq[K],|\mathcal{I}|=L_t} \sum_{i\in\mathcal{I}} \mu_i - \sum_{i\in I(t)} \mu_i \right] \tag{5.6}$$

Notice that, when an algorithm uses $L_t = L^*$ in each round, the pull regret is 0, and the regret boils down to the existing MP-MAB. Achieving sublinear pull regret means that the algorithm satisfies the efficiency constraint, while sublinear regret means that it maximises the total reward, so we need to minimise both regrets.

5.4.2. Regret Bound

5.4.2.1. Logarithmic Regret

For any event $\mathcal{X}$, let $\mathcal{X}^c$ be its complementary. For $\mathcal{X}$, $\mathbf{1}(\mathcal{X}) = 1$ if $\mathcal{X}$ holds or 0 otherwise.

Definition 5.1 (Top-L_t Set)**.** *$I(t) : |I(t)| = L_t$ is a top-L_t set if it contains the L_t arms with highest expectation μ_i. Let $\mathcal{A}_t$ be the event that $I(t)$ is the top-L_t set.*

Definition 5.2 (Logarithmic Regret Algorithm)**.** *A base bandit algorithm has logarithmic regret if there exists a distribution-dependent constant $C_{\mathrm{alg}} = C_{\mathrm{alg}}(\{\mu_i\})$ such that*

$$\sum_{t=1}^{T} \Pr\left[\mathcal{A}_t^c\right] \leq C^{\mathrm{alg}} \log T$$

Remark 5.1 (Logarithmic Regret of Multiple-Play Bandits)**.** *Note that Definition 5.2 is equivalent to state that the regret of Eq. (5.6) is logarithmic. Based on the existing analyses [Che+16], one can prove that MP-KL-UCB [GC11] and MP-TS [KHN15] have logarithmic regret for varying L_t. For completeness, we show that MP-TS has logarithmic regret in Section 5.4.2.2, using techniques from [AG13].*

The following theorem states that our policy has logarithmic pull regret, i.e., that the number of pulls converges to L^* when we combine it with a base bandit algorithm of logarithmic regret.

Theorem 5.1 (Logarithmic Pull Regret). *Let the general scaling bandit of Algorithm 5.4 with a base bandit algorithm of logarithmic regret be given. Then, there exist two distribution-dependent constants* $C_*^{\text{preg}}, C_*^{\text{reg}} = C_*^{\text{preg}}(\{\mu_i\}), C_*^{\text{reg}}(\{\mu_i\})$ *such that*

$$\mathbb{E}[PReg(T)] \leq C_*^{\text{preg}} \log T, \tag{5.7}$$

Moreover, the standard regret of the proposed algorithm is bounded as

$$\mathbb{E}[Reg(T)] \leq C_*^{\text{reg}} \log T. \tag{5.8}$$

Let us first define the events needed for the proof:

$$\mathcal{B}_t = \{L_t \leq L^* \cap \hat{\eta}_t > \eta^*\} \cup \{L_t > L^* \cap \hat{\eta}_t \leq \eta^*\} \tag{5.9}$$

$$C_t = \{L_t \geq L^* \cup \hat{B}_t > \eta^*\} \tag{5.10}$$

$$\mathcal{D}_t = \{L_t < L^* \cup \hat{B}_t \leq \eta^*\} \tag{5.11}$$

The following lemmas are key to bound the pull regret:

Lemma 5.1 (Scaling). $L_t \in \{L^*, L^* + 1\}$ *holds if*

$$\bigcap_{t'=t-K,\ldots,t-1} \mathcal{B}_{t'} \cap C_{t'}.$$

Proof of Lemma 5.1. $\mathcal{B}_{t'} \cap C_{t'}$ implies

- $L_{t'+1} = L_{t'} + 1$ if $L_{t'} < L^*$,
- $L_{t'+1} \in \{L^*, L^* + 1\}$ if $L_{t'} = L^*$
- $L_{t'+1} = L_{t'} - 1$ if $L_{t'} \geq L^* + 1$

As $L^* - L_{t-K} < K$, there exists $t'_c \in \{t - K, t - K + 1, \ldots, t - 1\}$ such that $L_{t'_c} = L^*$, and after the round t'_c, $L_{t'} \in \{L^*, L^* + 1\}$ holds. □

Lemma 5.2 (Sufficient Condition of No-regret). $L_t = L^*$ *holds if*

$$\bigcap_{t'=t-K,\ldots,t-1} \{\mathcal{B}_{t'} \cap C_{t'}\} \cap \mathcal{D}_{t-1}.$$

Proof of Lemma 5.2. Lemma 5.1 implies $L_{t-1} \in \{L^*, L^* + 1\}$, which, combined with $\mathcal{B}_{t-1} \cap C_{t-1} \cap \mathcal{D}_{t-1}$ implies that, at round $t - 1$, it scales to $L_t = L^*$. □

We can now proceed to the proof of Theorem 5.1:

Proof of Theorem 5.1. Lemma 5.2 implies that, if the direction of scaling is correct and the confidence bound is sufficiently small, then L_t goes to L^*. We decompose the pull regret using Lemma 5.2:

$$\begin{aligned}
&\mathrm{PReg}(T) \leq K \sum_{t=1}^{T} \mathbf{1}[L_t \neq L^*] \\
&\text{(since } \mathrm{PReg}(t) \text{ increases at most by } K \text{ at each round)} \\
&\leq K \sum_{t=1}^{T} \mathbf{1}\left[\left(\bigcup_{t'=t-K}^{t-1} \left(\mathcal{A}_{t'}^c \cup \mathcal{B}_{t'}^c \cup C_{t'}^c\right) \cup \mathcal{D}_{t-1}^c\right) \cap L_t \neq L^*\right] \\
&\text{(by the contraposition of Lemma 5.2)} \\
&\leq K + K \sum_{t=K+1}^{T} \left(\mathbf{1}\left[\bigcup_{t'=t-K}^{t-1} \left(\mathcal{A}_{t'}^c \cup \mathcal{B}_{t'}^c \cup C_{t'}^c\right)\right]\right. \\
&\qquad \left. + \mathbf{1}\left[\bigcap_{t'=t-K}^{t-1} \left(\mathcal{A}_{t'}^c \cap \mathcal{B}_{t'}^c \cap C_{t'}^c\right) \cap \mathcal{D}_{t-1}^c \cap L_t \neq L^*\right]\right) \\
&\leq K + K^2 \sum_{t'=K+1}^{T} \left(\mathbf{1}[\mathcal{A}_{t'}^c] + \mathbf{1}[\mathcal{A}_{t'} \cap \mathcal{B}_{t'}^c] + \mathbf{1}[\mathcal{A}_{t'} \cap C_{t'}^c]\right) \\
&\quad + K \sum_{t=K+1}^{T} \mathbf{1}\left[\bigcap_{t'=t-K}^{t-1} \left(\mathcal{A}_{t'} \cap \mathcal{B}_{t'} \cap C_{t'}\right) \cap \mathcal{D}_{t-1}^c \cap L_t \neq L^*\right] \qquad (5.12)
\end{aligned}$$

The following lemma bounds each term in Eq. (5.12) in expectation.

Lemma 5.3 (Bounds on each Term)**.** *The following bounds hold:*

$$\underbrace{\sum_{t=1}^{T} \Pr[\mathcal{A}_{t'}^c]}_{\text{(A)}} = O(\log T) \quad ; \quad \underbrace{\sum_{t=1}^{T} \Pr[\mathcal{A}_t \cap \mathcal{B}_t^c]}_{\text{(B)}} = O(1/\Delta^2)$$

$$\underbrace{\sum_{t=1}^{T} \Pr[\mathcal{A}_t \cap C_t^c]}_{\text{(C)}} = O(1/\Delta^2) + O(\log\log T)$$

$$\underbrace{\sum_{t=K+1}^{T} \Pr\left[\bigcap_{t'=t-K}^{t-1} (\mathcal{A}_{t'} \cap \mathcal{B}_{t'} \cap C_{t'}) \cap \mathcal{D}_{t-1}^c, L_t \neq L^*\right]}_{\text{(D)}} = O(1/\Delta^2).$$

Eq. (5.7) now follows from Lemma 5.3. Eq. (5.8) follows from the fact that the base bandit algorithm has logarithmic regret. We will prove Lemma 5.3 shortly after. □

Discussion: Following the bandit literature, we assume that parameters $\{\mu_i\}$ (and thus the gap Δ) are constants. Although Lemma 5.3 uses the Landau notation, all the terms[2] in the analysis are explicitly written, i.e., they are finite-time. One can also see that the leading term of the pull regret and the regret are $O(\log T/\Delta^2)$ when the regret of the base bandit algorithm (Definition 5.2) is $O(\log T/\Delta^2)$: our scaling bandit algorithm targets at the largest L such that $(1/L)\sum_{i\le L}\mu_i > \eta^*$, and the gap Δ characterises the hardness of finding such an L. For ease of analysis, we omit the dependence on K.

In what follows, we show that Remark 5.1 holds for MP-TS. Then, we prove Lemma 5.3.

5.4.2.2. Performance of MP-TS as a 'base bandit' algorithm

Let the posterior sample of TS at round t be $\theta_i(t) \sim \text{Beta}(\alpha_i(t)+1, \beta_i(t)+1)$. Note that $\mathcal{A}_t^c, T = l$ implies that there exists $i \le l, j > l$ such that $i \notin I(t), j \in I(t)$. Let $d = \mu_i - \mu_j > 0$ and $x, y = \mu_j + d/3, \mu_j + 2d/3$. Let the events $\mathcal{E}_{i,\mu}(t) = \{\hat{\mu}_i(t) \le x\}$ and $\mathcal{E}_{i,\theta}(t) = \{\theta_i(t) \le y\}$. Let $\theta_{(l)}(t)$ be the l-th largest from $\{\theta_i(t)\}$ (ties are broken arbitrarily). We have

$$\begin{aligned}\sum_{t=1}^{T}\Pr\left[\mathcal{A}_t^c\right] &= \sum_{t=1}^{T}\sum_{l\in[K]}\Pr\left[\mathcal{A}_t^c \cap T = l\right] \\ &\le \sum_{t=1}^{T}\sum_{l\in[K]}\sum_{i\le l, j>l}\Pr\left[T = l \cap i \notin I(t) \cap j \in I(t)\right]. \end{aligned} \tag{5.13}$$

Here,

$$\begin{aligned}\Pr\left[T = l \cap i \notin I(t) \cap j \in I(t)\right] &\le \Pr\left[T = l \cap y \le \theta_j(t)\right] \\ &+ \Pr\left[T = l \cap \theta_{(l)}(t) \le y \cap \theta_i(t) \le y\right]. \end{aligned} \tag{5.14}$$

Let $p_{i,n} = \Pr[\theta_i(t) > y \cap N_i(t) = n]$. The following discussion is essentially equivalent to Lemma 9 in [KHN15]. Let $\theta_{(l)\setminus i}(t)$ be the value of the l-th largest among $\{\theta_j\}_{j\in[K]\setminus i}$. Thus,

$$\begin{aligned}&\sum_{t=1}^{T}\mathbf{1}\left[T = l \cap \theta_{(l)}(t) \le y \cap \theta_i(t) \le y\right] \\ &\le \sum_{n=1}^{T}\sum_{t=1}^{T}\mathbf{1}\left[T = l \cap \theta_{(l)}(t) \le y \cap \theta_i(t) \le y \cap N_i(t) = n\right] \\ &\le \sum_{n=1}^{T}\sum_{t=1}^{T}\mathbf{1}\left[T = l \cap \theta_{(l)\setminus i}(t) \le y \cap \theta_i(t) \le y \cap N_i(t) = n\right] \\ &\le \sum_{n=1}^{T}\sum_{m=1}^{T}\mathbf{1}\left[m \le \sum_{t=1}^{T}\mathbf{1}\left[T = l \cap \theta_{(l)\setminus i}(t) \le y \cap \theta_i(t) \le y \cap N_i^t = n\right]\right].\end{aligned}$$

The event

$$m \le \sum_{t=1}^{T}\mathbf{1}\left[\theta_{(l)\setminus i}(t) \le y \cap \theta_i(t) \le y \cap N_i(t) = n\right] \tag{5.15}$$

[2] (including the involved underestimation term of Lemma 5.7, cf. [Mai17] for an explicit bound)

implies that $\{\theta_{(l)\setminus i}(t) \leq y \cap \theta_i(t) \leq y \cap N_i(t) = n\}$ occurred at least m rounds, and $\{\theta_i(t) \leq y\}$ occurred for the first m rounds such that Eq. (5.15) holds. By the statistical independence of $\theta_{(l)\setminus i}(t)$ and $\theta_i(t)$, we have

$$\Pr\left[m \leq \mathbf{1}\left[T = l \cap \theta_{(l)\setminus i}(t) \leq y \cap \theta_i(t) \leq y \cap N_i^t = n\right]\right] \leq (1 - p_{i,n})^m.$$

and following the same steps as Lemma 9 in [KHN15], we have

$$\begin{aligned}\sum_{n=1}^{T}\sum_{m=1}^{T}(1-p_{i,n})^m &\leq \sum_{n=1}^{T}\frac{1-p_{i,n}}{p_{i,n}} \\ &\leq \frac{1}{(\mu_i - y)^2} \text{ (by Lemma 2 in [AG13]).}\end{aligned} \tag{5.16}$$

Moreover, by Lemma 4 and Lemma 3 in [AG13]:

$$\begin{aligned}\Pr\left[T = l \cap y \leq \theta_j(t)\right] &\leq \Pr\left[T = l \cap j \in I(t) \cap y \leq \theta_j(t) \cap x > \hat{\mu}_j\right] \\ &\quad + \Pr\left[T = l \cap j \in I(t) \cap x \leq \hat{\mu}_j\right] \\ &\leq \left(\frac{\log T}{d_{\mathrm{KL}}(x,y)} + 1\right) + \left(\frac{1}{d_{\mathrm{KL}}(x,\mu_j)} + 1\right),\end{aligned} \tag{5.17}$$

where $d_{\mathrm{KL}}(p,q) = p\log(p/q) + (1-p)\log((1-p)/(1-q))$ be the KL divergence between two Bernoulli distributions. From Eqs. (5.13), (5.14), (5.16), and (5.17), $\sum_{t=1}^{T}\Pr\left[\mathcal{A}_t^c\right] = O(\log T)$.

5.4.2.3. Proof of Lemma 5.3

In this section, we bound each of the terms (A)–(D) in Lemma 5.3. Let us first describe the following required lemma:

Lemma 5.4 (Uniform Bound)**.** *Let*

$$\mathcal{G}_l(t) = \bigcap_{i \leq l}\left\{|\hat{\mu}_i(t) - \mu_i| \leq \Delta\right\}.$$

For $l \in [K]$, the following inequality holds:

$$\sum_{t=1}^{T}\Pr[\mathcal{A}_t \cap T = l \cap \mathcal{G}_l^c(t)] = O(1/\Delta^2). \tag{5.18}$$

Proof of Lemma 5.4. Event $\mathcal{A}$ implies each arm $i \leq l$ is drawn, and thus

$$\begin{aligned}\sum_{t=1}^{T}\Pr[\mathcal{A}_t \cap T = l \cap \mathcal{G}_l^c(t)] &\leq 1 + \sum_{n=1}^{T}\Pr[|\hat{\mu}_{i,n} - \mu_i| \leq \Delta] \\ &\leq 1 + \frac{1}{2\Delta^2}e^{-2n_c\Delta^2} \text{ (by Lemma 5.6)} = O\left(\frac{1}{\Delta^2}\right).\end{aligned}$$

□

Bounding Term (A): Term (A) is directly bounded by the fact that the base bandit algorithm has logarithmic regret.

Bounding Term (B): Note that $\mathcal{B}_t = \{T \le L^* \cap \hat{\eta}_t < \eta^*\} \cup \{T > L^* \cap \hat{\eta}_t \ge \eta^*\}$, and $\{\mathcal{A}_t \cap \mathcal{G}_l(t)\}$ implies $\mathcal{B}_t$. By Lemma 5.4,

$$\sum_{t=1}^{T} \Pr[\mathcal{A}_t \cap \mathcal{B}_t^c] \le \sum_{l\in[K]} \sum_{t=1}^{T} \Pr[\mathcal{A}_t \cap \mathcal{G}_l^c(t)] = O(1/\Delta^2). \tag{5.19}$$

Note that $\mathcal{A}_t \cap L_t = l$ implies arms $1, \dots, l$ are drawn.

Bounding Term (C): The event $\mathcal{A}_t \cap B_i^l(t) < \eta^*$ implies $\mathcal{G}_l^c(t) \cup b_{l+1}(t) \le \mu_{l+1} - \Delta$. By using this, we have

$$\begin{aligned}
&\sum_{t=1}^{T} \Pr[\mathcal{A}_t \cap C_t^c] \le \sum_{l=1}^{L^*-1} \sum_{t=1}^{T} \Pr[\mathcal{A}_t \cap T = l \cap B_i^l(t) \le \eta^*] \\
&\le \sum_{l=1}^{L^*-1} \sum_{t=1}^{T} \Pr\left[\mathcal{A}_t \cap T = l \cap \left(\mathcal{G}_l^c(t) \cup b_{l+1}(t) \le \mu_{l+1} - \Delta\right)\right] \\
&\le \sum_{l=1}^{L^*-1} \sum_{t=1}^{T} \left(\Pr[\mathcal{A}_t \cap T = l \cap \mathcal{G}_l^c(t))] + \Pr[b_{l+1}(t) \le \mu_{l+1} - \Delta]\right) \\
&\le \sum_{l=1}^{L^*-1} \sum_{t=1}^{T} \Pr[\mathcal{A}_t \cap T = l \cap \mathcal{G}_i^c(t))] + O(\log\log T) \\
&\quad \text{(By the union bound of Lemma 5.7 over } t \in [T]) \\
&= O(1/\Delta^2) + O(\log\log T) \quad \text{(by Lemma 5.4).}
\end{aligned} \tag{5.20}$$

Bounding Term (D): Note that $\bigcap_{t'=t-K}^{t-1} (\mathcal{A}_{t'} \cap \mathcal{B}_{t'} \cap C_{t'})$ implies $L_{t-1} \in \{L^*, L^*+1\}$. Thus, $T \ne L^*$ implies $B_i^{L_{t-1}}(t-1) > \eta^*$, and $B_i^{L_{t-1}}(t-1) > \eta^*$ implies $\mathcal{G}_{L_{t-1}}^c(t) \cup b_{L_{t-1}+1}(t) > \mu_{L_{t-1}+1} + \Delta$. Moreover, $\bigcap_{t'=t-K}^{t-1} (\mathcal{A}_{t'} \cap \mathcal{B}_{t'} \cap C_{t'}) \cap \mathcal{D}_{t-1}^c$ implies that arm L^*+1 is drawn in either round $t-1$ or round t, and thus the event

$$\bigcap_{t'=t-K}^{t-1} (\mathcal{A}_{t'} \cap \mathcal{B}_{t'} \cap C_{t'}) \cap \mathcal{D}_{t-1}^c \cap N_{L^*+1}(t) = n$$

occurs at most twice for each n.

By using these, we obtain

$$\begin{aligned}
&\sum_{t=K+1}^{T} \Pr\left[\bigcap_{t'=t-K}^{t-1} (\mathcal{A}_{t'} \cap \mathcal{B}_{t'} \cap \mathcal{C}_{t'}) \cap \mathcal{D}_{t-1}^{c} \cap T \neq L^*\right] \\
&\leq K \sum_{l \in \{L^*, L^*+1\}} \sum_{t=1}^{T} \Pr[\mathcal{A}_t \cap \mathcal{G}_l^c(t)] + K + \frac{4 \log T}{\Delta^2} \\
&\quad + \sum_{t=K+1}^{T} \Pr\left[b_{L^*+1}(t-1) \geq \mu_{L^*+1} + \Delta \cap N_i(t) \geq \frac{2 \log T}{\Delta^2}\right] \\
&\leq O(1/\Delta^2) + K + \frac{4 \log T}{\Delta^2} \\
&\quad + \sum_{t=K+1}^{T} \Pr\left[b_{L^*+1}(t-1) \geq \mu_{L^*+1} + \Delta \cap N_{L^*+1}(t) \geq \frac{2 \log T}{\Delta^2}\right] \\
&\quad \text{(by Lemma 5.4)} \\
&\leq O(1/\Delta^2) + K + \frac{4 \log T}{\Delta^2} \\
&\quad + 2 \sum_{n=\frac{2 \log T}{\Delta^2}}^{T} \Pr\left[\bigcup_{t} \left(b_{L^*+1}^t \geq \mu_{L^*+1} + \Delta \cap N_{L^*+1}^t = n\right)\right], \qquad (5.21)
\end{aligned}$$

and the last term is bounded as

$$\begin{aligned}
&\sum_{n=\frac{2 \log T}{\Delta^2}}^{T} \Pr\left[\bigcup_{t} (b_{L^*+1}(t) \geq \mu_{L^*+1} + \Delta \cap N_{L^*+1}(t) = n)\right] \\
&\leq \sum_{n=\frac{2 \log T}{\Delta^2}}^{T} \Pr\left[n d_{\mathrm{KL}}(\hat{\mu}_{L^*+1,n}, \mu_{L^*+1} + \Delta) \leq \log T\right] \\
&\leq \sum_{n=\frac{2 \log T}{\Delta^2}}^{T} \Pr\left[2n(\hat{\mu}_{L^*+1,n} - \mu_{L^*+1} - \Delta)^2 \leq \log T\right] \\
&\quad \text{(by Pinsker's inequality)} \\
&= \sum_{n=\frac{2 \log T}{\Delta^2}}^{T} \Pr\left[\hat{\mu}_{L^*+1,n} \leq \mu + \Delta - \sqrt{\frac{\log T}{2n}}\right] \\
&\leq \sum_{n=\frac{2 \log T}{\Delta^2}}^{T} \Pr\left[\hat{\mu}_{L^*+1,n} \leq \mu + \Delta/2\right] \leq \sum_{n=\frac{2 \log T}{\Delta^2}}^{T} \mathrm{e}^{-2n(\Delta/2)^2} = O(1/T) \\
&\quad \text{(by Hoeffding inequality).} \qquad (5.22)
\end{aligned}$$

5.4.2.4. Concentration Inequalities

The following inequalities are used to derive Lemma 5.3.

Lemma 5.5 (Hoeffding's Inequality). *Let $X_1, \ldots, X_n$ be independent random variables taking values in $[0,1]$ with mean $\mu = (1/n)\sum_i^n X_i$. Let $\hat{\mu} = (1/n)\sum_{i=1}^n X_i$. Then we have:*

$$\Pr[\hat{\mu} \geq \mu + \epsilon] \leq \mathrm{e}^{-2n\epsilon^2} \text{ and } \Pr[\hat{\mu} \leq \mu - \epsilon] \leq \mathrm{e}^{-2n\epsilon^2}. \tag{5.23}$$

Lemma 5.6 (High-Probability Bound). *Let $\hat{\mu}_{i,n}$ be the empirical estimate of μ_i at $N_i(t) = n$. For any $\epsilon > 0$ and $n_c > 0$, the following bound holds:*

$$\Pr\left[N_i(t) = n \cap \bigcup_{n=n_c}^{\infty} |\hat{\mu}_{i,n} - \mu| \geq \epsilon\right] \leq \frac{1}{2\epsilon^2}\mathrm{e}^{-2n_c\epsilon^2}.$$

Lemma 5.6 is derived by using the Hoeffding inequality and the union bound over n.

Proof of Lemma 5.6.

$$\begin{aligned}
&\Pr\left[N_i(t) = n \cap \bigcup_{n=n_c}^{\infty} |\hat{\mu}_i(t) - \mu| \geq \epsilon\right] \\
&\leq \sum_{n=n_c}^{\infty} \Pr\left[N_i(t) = n \cap |\hat{\mu}_i(t) - \mu| \geq \epsilon\right] \\
&\leq \mathrm{e}^{-2n_c\epsilon^2} \sum_{n=0}^{\infty} \mathrm{e}^{-2n\epsilon^2} \text{ (by Hoeffding inequality)} \\
&= \mathrm{e}^{-2n_c\epsilon^2} \frac{\mathrm{e}^{2\epsilon^2}}{\mathrm{e}^{2\epsilon^2} - 1} \leq \mathrm{e}^{-2n_c\epsilon^2} \frac{1}{2\epsilon^2}.
\end{aligned}$$

□

Lemma 5.7 (KL-UCB Index Underestimation, Corollary 23 in [Mai17]). *The following inequality holds: Let $\epsilon > 0$ be arbitrary. There exists constants $T_c, C_{KL} = C_{KL}(\{\mu_i\}, \epsilon)$ such that, for $t > T_c$*

$$\Pr[b_i(t) \leq \mu_i - \epsilon] \leq \frac{C_{KL}}{t \log t}. \tag{5.24}$$

5.5. Experiments

This section evaluates the performance of S-TS and S-TS-ADWIN. We compare against alternative 'base bandits' and to the state-of-the-art non-static bandit algorithms. We also highlight the benefits of scaling by comparing against non-scaling bandits. We simulate scenarios with 10^5 steps to evaluate our approach in static (Section 5.5.1) and non-static (Section 5.5.2) environments. Then, we present a study where we have monitored real-world data streams (Section 5.5.3). We will also verify the scalability of our approach.

We have implemented every approach in Scala and averaged our experimental results across 100 runs. Each algorithm was run single-threaded in a server with 64 cores at 3 GHz and 128 GB RAM.

5.5.1. Static Environment

In this section, our goal is to verify the capability of S-TS to find L^* and maximise the reward in a static environment. We compare S-TS in terms of pull regret and standard regret against alternative scaling bandits, i.e., by replacing the base bandit with MP-KL-UCB [GM11; KHN15], CUCB [Che+16], and Exp3.M [UNK10]. We adapt each algorithm so that they 'scale'. The prefix 'S-' stands for the use of KL-S. For instance, Scaling Kullback-Leibler UCB (S-KL-UCB) is Algorithm 5.4 where the MP-Base-Bandit chooses the top-L arms based on the KL-UCB index [GC11]. We simulate a static scenario with K Bernoulli arms with known means $\mu_1, \dots \mu_K$ and $T = 10^5$, such that

$$\{\mu_i\}_{i=1}^{K} = \left\{\frac{i}{K} - \frac{1}{3K}\right\}_{i=1}^{K} \tag{5.25}$$

Given this, the means of the K arms are distributed linearly between 0 and 1, such that, when $\eta^* = 0.9$, then $L^* = K/5$, and when $\eta^* = 0.8$, then $L^* = 2K/5$, and so on. We set $K = 100$. We measure the regret and the pull regret of each approach against a Static Oracle (SO), which always pulls the top-L^* arms in expectation. S-SO is a 'Scaling' Static Oracle, i.e., it uses KL-S to determine L_t.

In Figure 5.1, the first row shows the convergence to L^*. The second and last rows show the pull regret and standard regret, respectively. We see that S-TS and S-KL-UCB perform best since they obtain the lowest regret for both measures. When η^* is smaller, the number of pulls T converges faster to L^*, for two reasons: (i) the optimal number of pulls L^* is closer to the starting condition L_1, and (ii) a lower η^* allows more exploration and more plays per rounds. So the top-L^* arms are found in fewer rounds with higher confidence.

We also see that S-Exp3.M does not perform very well. S-Exp3.M targets at the adversarial bandit problem [UNK10]. I.e., its assumptions regarding the rewards distribution are weaker. The policy, based on exponential weighting, forces Exp3.M to explore much so that our scaling policy KL-S lets T quickly drop to 1. Nonetheless, we see that after many steps, S-Exp3.M increases T again.

5.5.2. Non-Static Environment

In this section, we want to verify whether S-TS-ADWIN adapts to changes in the reward distribution. We compare our results against the following state-of-the-art non-static bandit algorithms:

- Discounted Thompson Sampling [RK17] (dTS) applies a discounting factor γ to the parameter α_i, β_i of the Beta posterior for each arm $i \in K$ at each time step t.
- Epsilon-Greedy [SB18] (EG) successively selects with probability ϵ the arm with the highest reward seen so far. Otherwise, it selects an arm randomly.
- Sliding Window UCB [GM08] (SW-UCB) discards any information older than a sliding window of fixed size w.

We set $\eta^* = 0.6$ and use the previous static setup to generate our non-static scenarios. In line with the literature [Gam+14], we simulate 'gradual' and 'abrupt' changes:

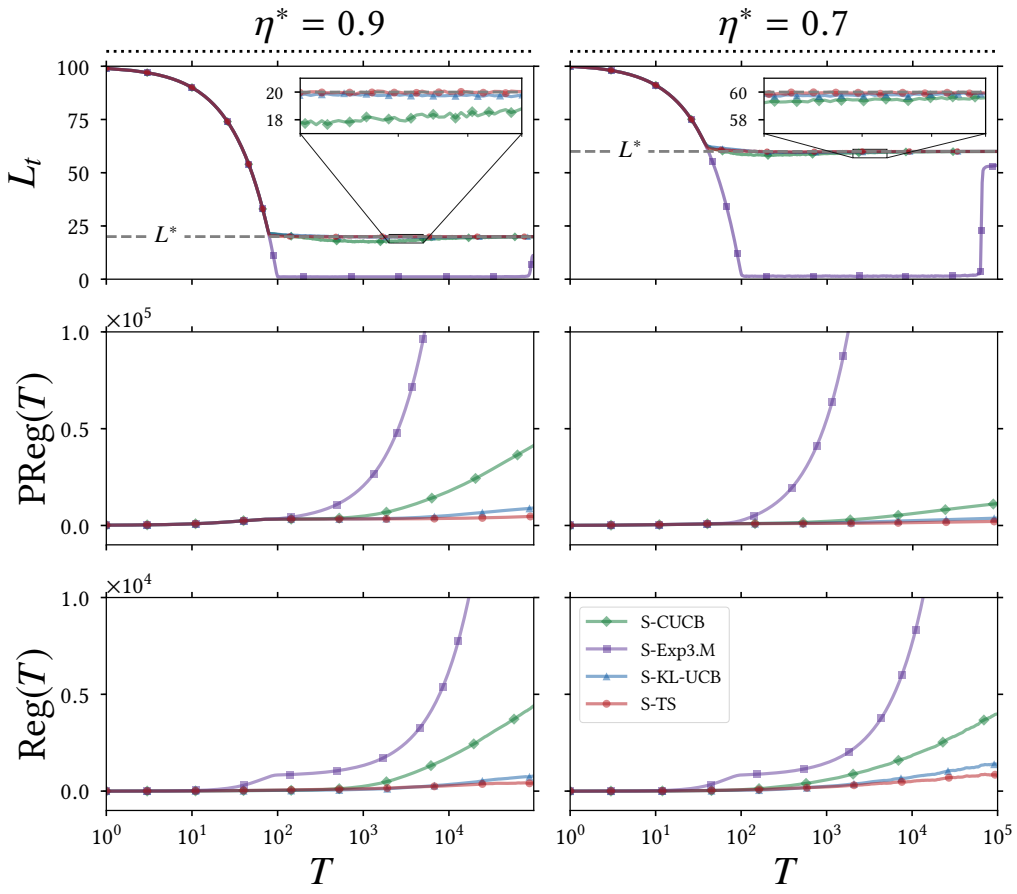


Figure 5.1.: Static experiment: S-TS minimises both regrets.

- **Gradual**: We place 60 equidistant change points over the time axis. For the first 30 change points, we set $\mu_h = 0$ for the arm $h \in K$ with the current highest expected reward. Then, we revert those changes in a 'last in – first out' way. Thus, L^* evolves gradually from 80 to 20, and back.
- **Abrupt**: We place two change points, equidistant from the start and end. At the first one, we set $\mu_h = 0$ for the top-30 arms. We revert this change at the second change point. Thus, L^* abruptly changes from 80 to 20 and back.

Since the environment is non-static, μ_i and L^* now vary as a function of t. Thus, we measure regret against a piecewise static oracle, which 'knows' $\mu_i(t)$ and $L^*(t)$.

A key result from this experiment is that S-TS, which assumes that arms do not change over time, fails to adapt to a changing environment. In contrast, our improvement, S-TS-ADWIN, does (Figure 5.2) and even outperforms all alternatives (Figure 5.3).

Figure 5.2 shows that S-CUCB(-ADWIN) and S-KL-UCB(-ADWIN) behave similarly to S-TS(-ADWIN), but have slightly higher regret and pull regret. S-Exp3.M has very high pull regret. Overall, we see that our adaptation based on ADWIN made it possible to handle both gradual and abrupt changes.

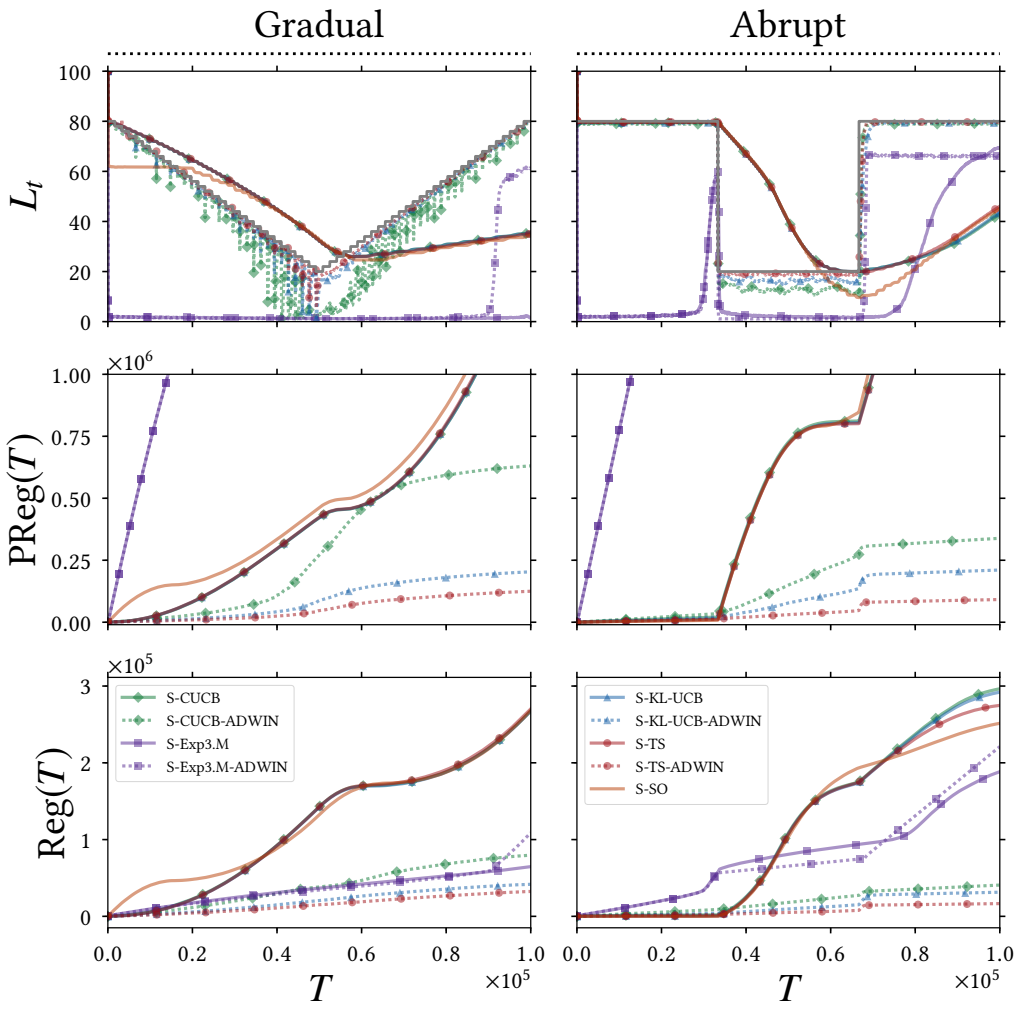


Figure 5.2.: Non-static experiment: S-TS vs. S-TS-ADWIN.

Figure 5.3 compares our approach to the existing non-static bandit alternatives. S-dTS tends to underestimate L^* in the case of a strong discounting factor, e.g., for $\gamma = 0.7$. On the contrary, S-SW-UCB overestimates L^*, in particular when the window size w is small. S-EG behaves similarly as the static approaches: it does not adapt to change quickly.

We also see that S-TS-ADWIN is robust for a large range of δ, except for very small values, e.g., 0.01 and 0.001. The best results are obtained with $\delta = 0.1$, which is consistent with the results in [BG07]. Other approaches, in turn, are quite sensitive to their parameters. For example, we can see that a weak discounting factor of $\gamma = 0.99$ is beneficial for dTS in the case of a gradual change, but that more aggressive discounting is better with abrupt changes. The figure shows that our approach adapts to different kinds of change, as opposed to the other approaches, without tuning its parameter.

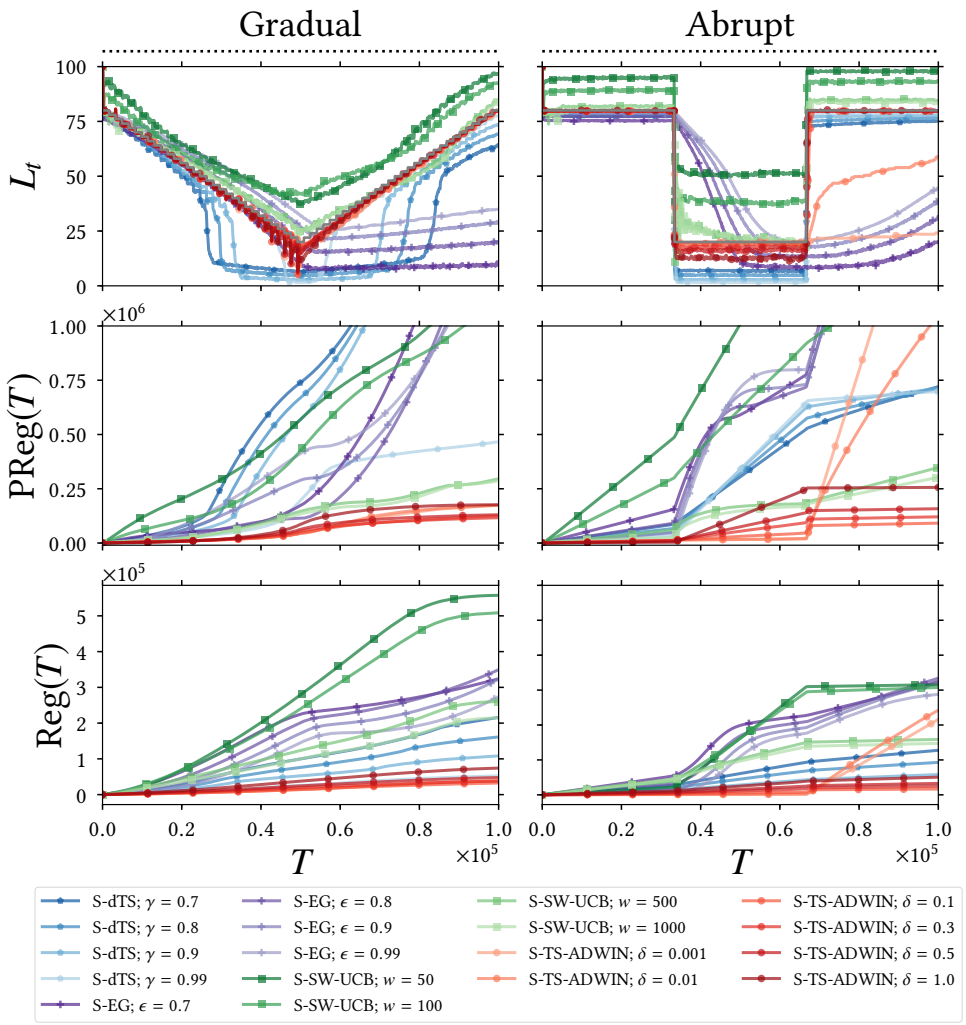


Figure 5.3.: Non-static experiment: Non-static bandits.

5.5.3. Case Study: Monitoring Dependency

In this section, we look at our real-world example: the Bioliq power plant. We create a data set corresponding to a week of measurements. It contains one measurement per second from a selection of 20 sensors, such as temperature, pressure, in various components.

We consider Mutual Information (MI) [KSG04] as a measure of correlation, which we have computed pair-wise between all attributes over a sliding window of size 1000 (~ 15 minutes) with step size 100 (~ 1.5 minute). Our goal in this use case is to employ bandit algorithms as a 'monitoring system' to keep an overview of large correlation values in the stream. Whenever the monitoring system detects a MI value higher than a threshold Γ, it obtains a reward of 1, otherwise 0. The challenge is to decide which coefficients to re-compute and how many of them at each step. This results in a S-MAB problem with 6048 steps and $(20 * 19)/2 = 190$ arms.

Figure 5.4 is the reward matrix for $\Gamma = 2$. We see that there are fewer rewards at the beginning and end of time. This is because the week is bordered by periods of lower activity in the plant. In the weekends, we observe fewer correlations than during weekdays.

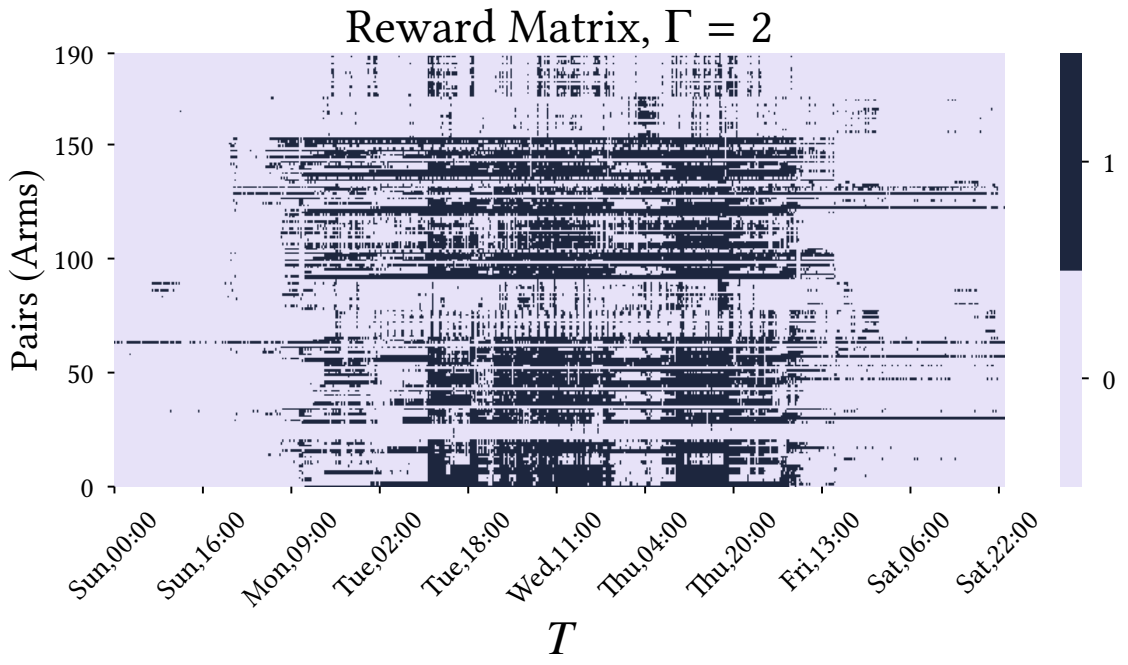


Figure 5.4.: Real-world experiment: Distribution of rewards.

Since there is no ground truth $\{\mu_i(t)\}$, it is not possible to assess the pull regret nor the standard regret. Instead, we compare the rewards and costs across algorithms: whenever an Algorithm A obtains more rewards than an Algorithm B for the same cost (i.e., number of plays), we conclude that A is superior to B. We compare against oracles with different levels of knowledge: Random Oracle (RO), Static Oracle (SO) and Dynamic Oracle (DO) are shown as a black, green and gold dotted lines respectively.

In Figure 5.5, we set $\eta^* = 0.8$ and visualise the evolution of the number of plays over time for various approaches. S-TS-ADWIN is the closest match to S-DO-ADWIN, our strongest baseline. This indicates that S-TS-ADWIN adapts to changes of rewards to find a proper value for T, unlike static algorithms such as S-TS.

Figure 5.6 shows the relationship between the average reward and the average cost (in terms of number of plays) of each algorithm. S-TS-ADWIN consistently yields higher rewards than DO for the same costs. TS-ADWIN (without scaling) also is superior to SO. Here we see the full benefit of our scaling policy with S-DO-ADWIN: the scaling dynamic oracle consistently achieves nearly maximal reward, while pulling fewer arms than a non-scaling algorithm. In other words, it outperforms its non-scaling counterpart.

Surprisingly, the UCB-based approaches do not perform well without scaling; they are close to the Random Oracle (RO). We hypothesise that ADWIN keeps the size of the dynamic window small in the real-world setting; w_t remains small, affecting the sharpness of the confidence bound. However, when our scaling policy is used, both approaches perform slightly better than DO.

We also verify that S-TS-ADWIN can adapt to different environments by changing Γ, which influences the availability of rewards. We see here that the improvement against our baselines is consistent, i.e., our algorithm also adapts the number of plays per round.

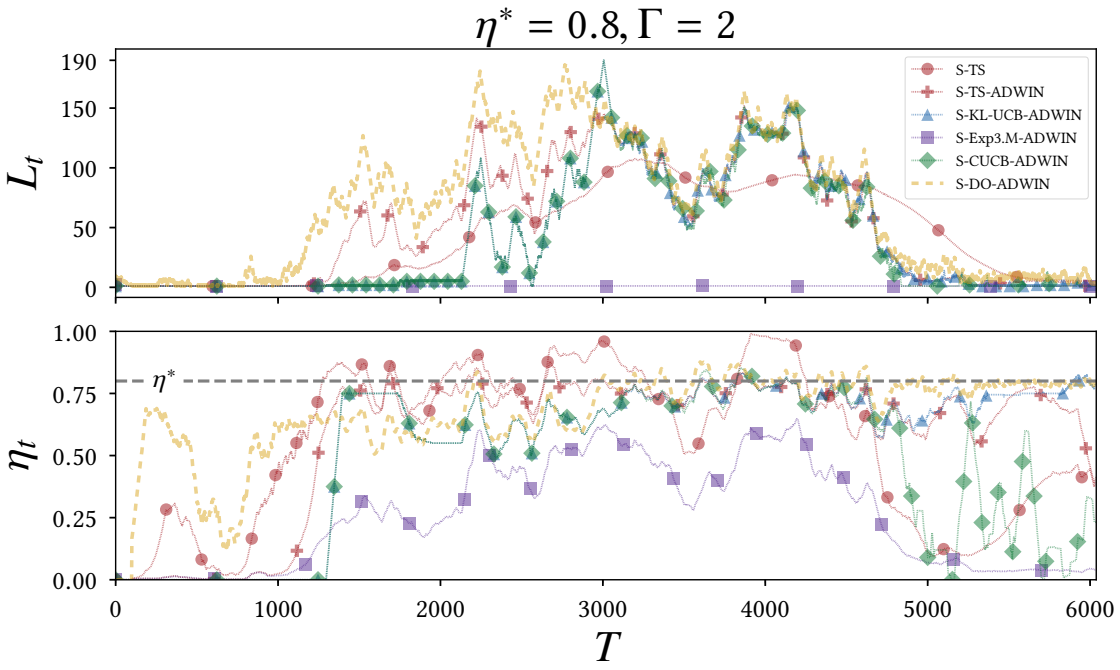


Figure 5.5.: Real-world experiment: Variation of T and η_t.

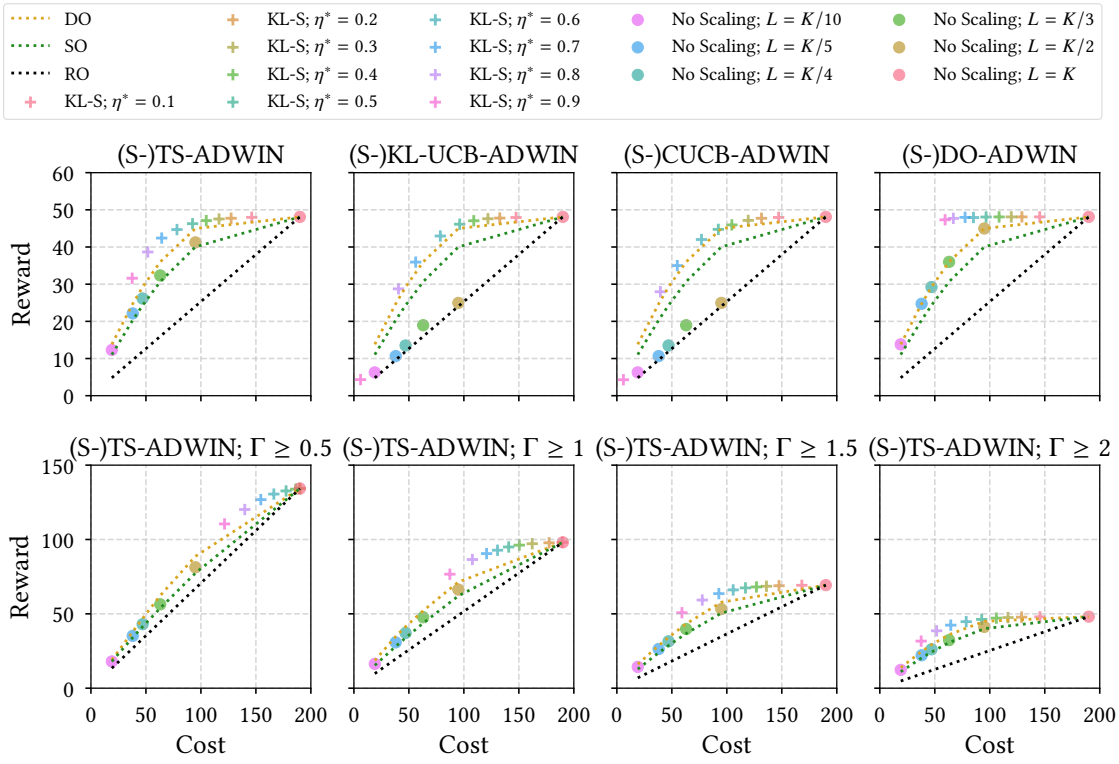


Figure 5.6.: Real-world experiment: Scaling versus No Scaling.

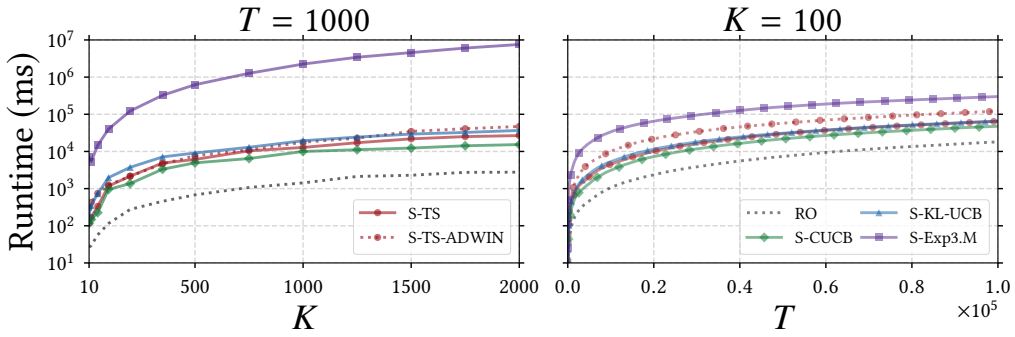


Figure 5.7.: Scalability of bandit algorithms w.r.t. K and T.

Finally, we evaluate the scalability of our approach w.r.t. K and T. To do so, we stick to our real-world example and create versions of the problem of different size with 10 to 2000 arms and up to 10^5 steps, by resampling the arms and observations. Then, we run our real-world experiment with $\Gamma = 2$ and average the runtime with scaling parameter η^* from 0.1 to 0.9. Figure 5.7 shows the result.

We see that each bandit approach scales linearly with the number of arms and the number of steps. S-CUCB is the fastest one, closely followed by S-TS and S-KL-UCB. S-Exp3.M is two orders of magnitude slower w.r.t. K. By comparing S-TS and S-TS-ADWIN, we see that the added computational burden from ADWIN is small and scales alike with an increasing number of arms and time steps. Each bandit approach, except Exp3.M, is at most one order of magnitude slower than choosing arms at random (RO). We see that our approach requires on average one millisecond to decide which arms to play when $K = 100$. This is typically less than the time required at each step to estimate MI on a single pair, using state-of-the-art estimators [KSG04].

Altogether, our experiments verified that our algorithms, S-TS and S-TS-ADWIN, are both effective and efficient. S-TS outperforms state-of-the-art bandits in the static setting, while S-TS-ADWIN adapts better to different kinds of change than its competitors. In our real-world example, S-TS-ADWIN obtains almost all the rewards in the environment for a cost reduced by up to 50%, outperforming very competitive baselines, such as a non-scaling dynamic oracle.

5.6. Discussion

We have proposed a new algorithm, S-TS, which combines Multiple-Play Thompson Sampling [KHN15] (MP-TS) with a strategy to decide on the number of arms played per round, a so-called 'scaling policy'. Our analysis and experiments showed that it enjoys strong theoretical guarantees and very good empirical behaviour. We also proposed an extension of our algorithm for the non-static setting. We applied the proposed model to data stream monitoring and showed its utility. However, we expect the impact of our contribution to extend beyond this one application. In the next part, we discuss applications to Knowledge Discovery in high-dimensional data streams.

Part IV.

Knowledge Discovery

6. Subspace Search in Data Streams

This chapter builds on our contributions from Chapter 4 and 5. Our main results have not been officially published yet, but we present them in the following manuscript:

- Edouard Fouché, Florian Kalinke and Klemens Böhm. *Efficient Subspace Search in Data Streams.* 2020. arXiv: 2011.06959 [cs.LG]

Keywords: Subspace Search; Data Stream Monitoring; Outlier Detection

6.1. Chapter Overview

In the real world, data streams are ubiquitous – think of network traffic or sensor data. Mining patterns, e.g., outliers or clusters, from such data must take place in real time. This is challenging because (1) streams often have high dimensionality, and (2) the data characteristics may change over time. Existing approaches tend to focus on only one aspect, either high dimensionality or the specifics of streams. For static data, a common approach to deal with high dimensionality – known as subspace search – extracts low-dimensional, 'interesting' projections (subspaces), in which patterns are easier to find.

However, searching for subspaces is difficult with static data already, because the number of subspaces increases exponentially with dimensionality. In [Agg13], Aggarwal compared the task of finding a pattern (e.g., an outlier) in high-dimensional spaces to that of searching for a needle in a haystack, while the haystack is one from an exponential number of haystacks. At the same time, [TB19] showed that to ensure high-quality results, the set of subspaces found must also be diverse, i.e., have low redundancy.

The streaming setting comes as an additional but orthogonal challenge. Stream mining algorithms are complex and must satisfy several constraints [DH03] (see Section 1.1.3). While a periodic recomputation of existing, static methods may cope with **C2** (Single Scan) and **C3** (Adaptability), this is not efficient (**C1**) and results may not be available in an anytime fashion (**C4**). Last but not least, data streams may also be heterogeneous (**C5**). Considering these challenges, the analogy above becomes more complex: The haystacks (subspaces) of interest are not only hidden but also change over time.

In this chapter, we address those challenges by extending the idea of subspace search to data streams. We propose a new approach, Streaming Greedy Maximum Random Deviation (SGMRD), to monitor subspaces of interest in high-dimensional data streams. In a nutshell, we show that (1) our approach leads to efficient monitoring of relevant subspaces, and (2) such monitoring also improves subsequent Knowledge Discovery tasks, such as outlier detection. We compare our approach to competitive baselines and state-of-the-art methods. We release our source code and benchmark on GitHub[1], to ensure reproducibility.

[1] https://github.com/edouardfouche/SGMRD

	Dimension-based	Other
1:	$\{s_1, s_6\}$	$\{s_8, s_9\}$
2:	$\{s_2, s_5\}$	$\{s_8, s_9, s_{10}\}$
3:	$\{s_3, s_7, s_8\}$	$\{s_7, s_8, s_9\}$
4:	$\{s_4, s_3, s_1\}$	$\{s_6, s_9, s_{10}\}$
5:	$\{s_5, s_2\}$	$\{s_7, s_8\}$
6:	$\{s_6, s_4, s_3\}$	$\{s_7, s_9, s_{10}\}$
7:	$\{s_7, s_8, s_9\}$	$\{s_8, s_{10}\}$
8:	$\{s_8, s_1\}$	$\{s_2, s_9\}$
9:	$\{s_9, s_{10}, s_2\}$	$\{s_3, s_8, s_9\}$
10:	$\{s_{10}, s_9, s_5\}$	$\{s_7, s_{10}\}$

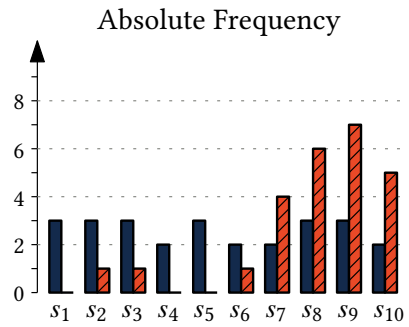


Figure 6.1.: Dimension-based subspace search versus other methods ($d = 10$).

6.2. Problem Formulation

6.2.1. Dimension-Based Subspace Search

Subspace search in the static setting has already been formalised in the literature. The goal is to find a set of subspaces that fulfils a specific notion of optimality. Such subspaces must at the same time (1) be likely to reveal patterns (the 'haystacks' from the analogy above) and (2) be diverse, i.e., have low redundancy with each other.

To this end, the idea is to deem a set of subspaces optimal if adding or removing a subspace to/from this set makes the search results worse. To ensure diversity, the notion of optimality of each subspace must be tied to a specific dimension. This way, the resulting set may consist of the best subspaces w.r.t. each dimension, and each dimension is represented in this manner. This is the essence of what we call 'dimension-based' search.

To illustrate this, we show in Figure 6.1 an exemplary result from a dimension-based search and from another scheme. Dimension-based results are more diverse compared to other results, which tend to over-represent some dimensions. Previous work [TB19] showed that the diversity from dimension-based approaches is key to improve the performance of subspace search algorithms.

Formally, we define a so-called Dimension-Subspace Quality Function (D-SQF), to capture how much a dimension s_i helps to reveal patterns in a subspace S:

Definition 6.1 (D-SQF). *For any $S \in \mathcal{P}(D)$ and any $s_i \in D$, a Dimension-Subspace Quality Function (D-SQF) is a function of type $q : \mathcal{P}(D) \times D \mapsto [0, 1]$ with $q(S, s_i) = 0, \forall s_i \notin S$.*

$q(S, s_i) = 1$ means that S has the maximum potential to reveal patterns w.r.t. s_i. Put differently, patterns may become more visible as one includes s_i in S. In turn, $q(S, s_i) = 0$ means that S cannot reveal any patterns w.r.t. s_i. By definition, $q(S, s_i) = 0$ if s_i is not part of S because the subspace cannot reveal any pattern in s_i.

Recent studies [Wan+17; TB19] instantiate such D-SQF as a measure of correlation, which one can estimate without any ex-post evaluation, i.e., it is not specific to any Data Mining algorithm. With this, subspace search remains independent of any downstream task. We can define a notion of subspace optimality:

Definition 6.2 (Optimal Subspace)**.** *A subspace $S \in \mathcal{P}(D)$ is optimal w.r.t. $s_i \in D$ and a D-SQF q if and only if*

$$q(S, s_i) \geq q(S', s_i) \quad \forall S' \in \mathcal{P}(D).$$

Then the optimal subspace set $\mathbb{S}^*$ is the set of optimal subspaces in D for all $s_i \in D$:

Definition 6.3 (Optimal Subspace Set)**.** *A set $\mathbb{S}^*$ is optimal w.r.t. D and a Dimension-Subspace Quality Function (D-SQF) q if and only if*

$$\forall s_i \in D,\ \exists S \in \mathbb{S}^*\ s.t.\ \forall S' \in \mathcal{P}(D),\ q(S, s_i) \geq q(S', s_i).$$

Thus, the optimal set $\mathbb{S}^*$ contains one subspace S for each dimension $s_i \in D$, each one maximising the quality q w.r.t. s_i. In other words, we can see $\mathbb{S}^*$ as a mapping of each dimension $s_i \in D$ to an optimal subspace w.r.t. that dimension:

$$\mathbb{S}^* : s_i \in D \mapsto S \in \mathcal{P}(D) \quad s.t. \quad \forall S' \in \mathcal{P}(D) \quad q(S, s_i) \geq q(S', s_i).$$

Finding $\mathbb{S}^*$ is NP-hard [Ngu15]. This is because one needs to assess the D-SQF of a set whose size grows exponentially with the number of dimensions. In fact, even finding a single optimal subspace is NP-hard. Thus, existing techniques do not guarantee the optimality of the results, but instead target at a good approximation of $\mathbb{S}^*$, while keeping the number of subspaces considered small.

This idea is suitable in the static setting [KMB12; Wan+17]. [TB19] showed that it leads to diverse sets of subspaces with better downstream mining results. Here, we propose a generalisation for streams.

6.2.2. Dimension-Based Subspace Search in Data Streams

The quality of subspaces, estimated via statistical correlation measures, may change over time, manifesting a phenomenon known as 'concept drift' [BGE15] (see Section 1.1.3). Thus, in streams, the quality function q is time-dependent, and so we write $\mathbb{S}^*$ and q_t. Then, the problem becomes more complex — observe the following example:

Example 6.1 (Variation of Mutual Information)**.** *We obtained measurement data from the Bioliq power plant (cf. Section 1.1.5) and computed the evolution of Mutual Information between a set of 10 sensor pairs for a single day. Figure 6.2 graphs the results. The Mutual Information for pairs* 1 *and* 2 *remains stable for the whole duration, while it is more volatile for pairs* 3 *to* 6*. The pairs* 7 *to* 10 *in turn show some change, but with less variance.*

As the example shows, some subspaces remain optimal for a longer period of time, while others frequently become sub-optimal. So the difficulty is that, even if one finds the set of subspaces $\mathbb{S}^*_t$, there is no guarantee that this set is optimal at time $t + 1$. Next, it is impossible to even test the optimality of $\mathbb{S}^*$, because one would need to evaluate an exponential number of subspaces.

Let us assume that the cost of evaluating the quality of a subspace is constant across different subspaces and time. This is the case when considering existing correlation estimators. We define a function $\mathcal{E}_t : \mathcal{P}(D) \times D \mapsto \{0, 1\}$ such that $\mathcal{E}_t(S, s_i) = 1$ if one computes $q_t(S, s_i)$, otherwise 0. We can now formulate subspace search in data streams as a multi-objective optimisation problem at time t with two conflicting objectives:

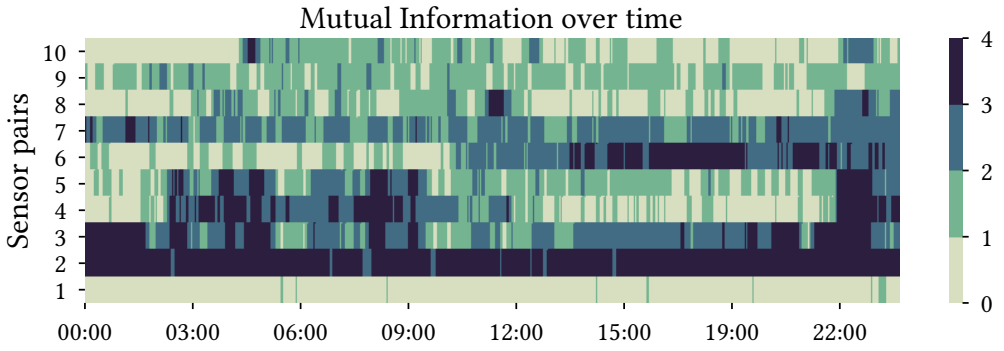


Figure 6.2.: Evolution of Mutual Information between 10 sensor pairs at Bioliq.

1. Find a set $\mathbb{S}_t$ which approximates $\mathbb{S}_t^*$ well. I.e., minimise the sum of the differences between the quality of the optimal set and the quality of the approximate set for each dimension. This is the first objective $O_1 = \sum_{i=1}^{|D|} \left[q_t(\mathbb{S}_t^*(s_i), s_i) - q_t(\mathbb{S}_t(s_i), s_i) \right]$.
2. Reduce the computation of the search, i.e., minimise the number of subspaces for which one computes the quality at time t. This is objective $O_2 = \sum_{s_i}^{D} \sum_{S}^{\mathcal{P}(D)} \mathcal{E}_t(S, s_i)$.

If $O_1 = 0$, then $\mathbb{S}_t \equiv \mathbb{S}_t^*$. Conversely, if $O_2 = 0$, then choosing $\mathbb{S}_t$ boils down to random guessing. Thus, objectives O_1 and O_2 conflict. They capture the trade-off between the quality of the set of subspaces and the computation effort.

Definition 6.3 implies that the search is independent for each dimension. Thus, to optimise O_1, we must find a dimension-based search algorithm, which for any dimension $s_i \in D$, returns a near-optimal subspace S w.r.t. the dimension. More formally:

Definition 6.4 (Dimension-based Search Algorithm)**.** *We define a dimension-based search algorithm as a function* $\textsc{Search}_t : D \mapsto \mathcal{P}(D)$*, which for any dimension* $s_i \in D$*, returns a subspace* S *minimising* $q_t(\mathbb{S}_t^*(s_i), s_i) - q_t(S, s_i)$ *at time* t*.*

Such an algorithm is associated with a cost, which depends on the number of subspaces evaluated. I.e., each run of $\textsc{Search}_t$ negatively impacts objective O_2. Thus, an additional challenge is to find an update policy $\pi : t \mapsto \mathcal{P}(D)$ to decide at any time, for which dimension(s) one should repeat the search. We define it as follows:

Definition 6.5 (Update Policy)**.** *An update policy is a function* $\pi : t \mapsto \mathcal{P}(D)$ *which returns* $\forall t$ *a set* $I(t) \in \mathcal{P}(D)$*, so that one repeats the* $\textsc{Search}_t$ *algorithm for* $s_i \in I(t)$*.*

Overall, to achieve subspace search in data streams, we must come up with an adequate instantiation of the following elements:

- A quality measure $q_t : \mathcal{P}(D) \times D \mapsto [0, 1]$.
- A dimension-based search algorithm $\textsc{Search}_t : D \mapsto \mathcal{P}(D)$.
- An update policy $\pi : t \mapsto \mathcal{P}(D)$.

Our approach, Streaming Greedy Maximum Random Deviation (SGMRD), addresses the challenges described previously by instantiating and combining each of these elements in a general framework.

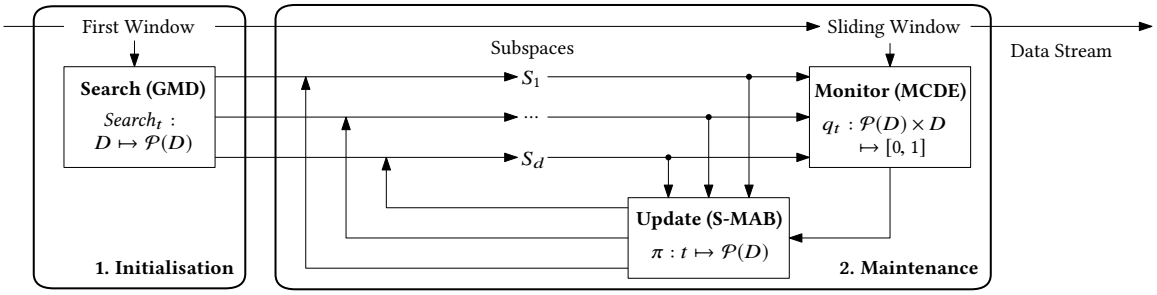


Figure 6.3.: SGMRD: A High-Level Overview.

6.3. Subspace Search in Data Streams

6.3.1. Our Approach: SGMRD

Figure 6.3 shows an overview of our framework, which consists of two steps:

1. **Initialisation**. We find an initial set of subspaces $\mathbb{S}_0$, using the first w observations in the stream. To do so, we run the algorithm $\textsc{Search}_t$ for each dimension. The outcome of this step is a set of subspaces $S_1, \ldots, S_d$, one for each dimension.
2. **Maintenance**. For any new observation in the stream, we monitor the quality of subspaces $q_t(S, s_i)$ for $S \in \mathbb{S}_t$, and decide, by learning a policy π, for which dimension(s) we repeat the search, i.e., algorithm $\textsc{Search}_t$.

Our approach is general to some extent, as one could consider various instantiations of each building block (Search, Monitor and Update). With SGMRD, we instantiate the search function $\textsc{Search}_t$ as a greedy hill-climbing heuristic, q_t as an efficient dependency estimator, MCDE (Chapter 4), and the policy π as a Multi-Armed Bandit (MAB) algorithm with variable number of plays, S-MAB (Chapter 5). We describe the specifics of each block and explain our design decisions in the following sections.

6.3.1.1. Search

SGMRD's initialisation searches for an initial set of subspaces using the first observation window. Since finding the optimal set (cf. Definition 6.3) is not feasible, we instantiate $\textsc{Search}_t$ as a greedy hill-climbing heuristic. Our heuristic constructs subspaces in a bottom-up manner. Algorithm 6.1 is our pseudo-code, and Figure 6.4 illustrates our idea.

In a first step (Line 1), we select the 2-dimensional subspace maximising the quality w.r.t. s_i. In our example, $s_i = s_1$, and the subspace with the highest quality is $\{s_1, s_2\}$. Then, we iteratively test whether adding the dimension associated with the next best 2-dimensional subspace containing s_i (Line 4) increases the quality of the current subspace (Line 5). If this is the case, we add it into the current subspace (Line 6), otherwise, we discard it. In our example, the heuristic first considers adding s_3, then s_4.

The advantage of only considering 2-dimensional subspaces is that it keeps the runtime of the search linear w.r.t. the number of dimensions. More precisely, we can see that the heuristic computes the quality of exactly $(|D| - 1) + (|D| - 2) = 2|D| - 3$ subspaces. Thus,

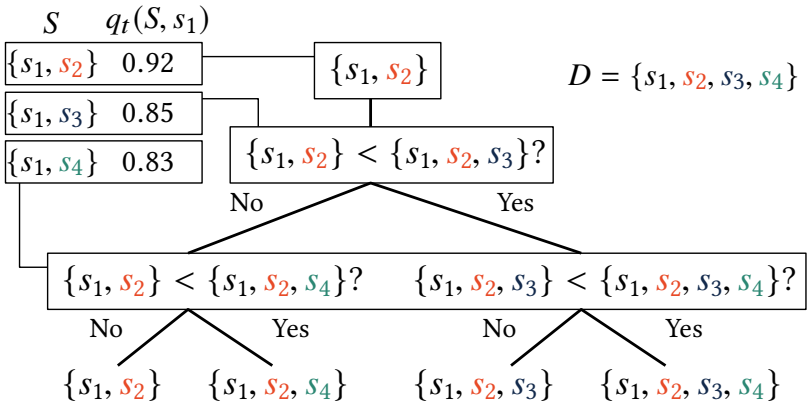


Figure 6.4.: Example of search w.r.t. s_1 with four dimensions.

the runtime of Algorithm 6.1 is in $O(|D|)$. The search is independent for each dimension. At initialisation, we run it for each dimension, so the initialisation is in $O(|D|^2)$.

Algorithm 6.1 SEARCH$_t(s_i)$

Require: A dimension/attribute $s_i \in D$

1: $\mathcal{S}^{max} \leftarrow s_i \cup \left(\arg\max_{s_j \in S} q_t(s_i \cup s_j, s_i)\right)$
2: $S \leftarrow S \setminus \mathcal{S}^{max}$
3: **while** S is not empty **do**
4: $\quad \mathcal{S}^{cand} \leftarrow \arg\max_{s_j \in S} q_t(s_i \cup s_j, s_i)$
5: $\quad$ **if** $q_t(\mathcal{S}^{max} \cup \mathcal{S}^{cand}, s_i) > q_t(\mathcal{S}^{max}, s_i)$ **then**
6: $\quad\quad \mathcal{S}^{max} \leftarrow \mathcal{S}^{max} \cup \mathcal{S}^{cand}$
7: $\quad S \leftarrow S \setminus \mathcal{S}^{cand}$
8: **return** subspace $\mathcal{S}^{max} \subseteq D$

6.3.1.2. Monitor

So far, we did not discuss any concrete instantiation of the quality q_t. In practice, one has only a sample of observations, and thus, the quality can only be *estimated* from a limited number of points. In what follows, we describe a new method to estimate the quality. Considering the constraints from the streaming setting and the nature of the required quality function, our method must fulfil the following technical requirements:

Efficiency. Since the search includes estimating the quality of numerous subspaces, the quality-estimation procedure must be efficient, to cope with the streaming constraints.

Multivariate. Since subspaces can have an arbitrary number of dimensions, the quality measure must be multivariate. Traditional dependency estimators are bivariate [Joe89].

Asymmetric. Since the quality values are specific to each dimension, the measure is not symmetric. The quality of subspace $\{s_1, s_2\}$ does not need to be the same w.r.t. s_1 or s_2.

We define the quality as a measure of non-independence in subspace S w.r.t. s_i. Recall that Definition 4.1 implies that the marginal distributions of each variable $X_i \in X$ must be equal to their conditional distribution w.r.t. $X \setminus X_i$. We can estimate the quality as a degree of non-independence w.r.t. a dimension X_i, and quantify it as the discrepancy between the

empirical marginal distribution $\hat{p}_{X_i}$ and the conditional distribution $\hat{p}_{X_i|(X\setminus X_i)}$. For the ease of discussion, let us consider $q_t(\{s_1, s_2\}, s_1)$. Then,

$$q_t(\{s_1, s_2\}, s_1) \propto disc\left(\hat{p}_{s_1}, \hat{p}_{s_1|s_2}\right), \tag{6.1}$$

where *disc* is the discrepancy between both distributions.

To estimate this discrepancy, we propose a bootstrap method. Iteratively, we take a random condition w.r.t. the dimensions $S \setminus s_1$ (i.e., restricting the other dimensions to a random interval) and perform a statistical test between the sets of observations within and outside of this random condition w.r.t. s_1. This is the idea behind our main contribution from Chapter 4, Monte Carlo Dependency Estimation (MCDE).

In the end, the quality is as follows:

$$q(S, s_i) = \sum_{m=1}^{M} \left[\mathcal{T}\left(B\left(c_i\right), B\left(\overline{c}_i\right)\right)\right] \tag{6.2}$$

where c_i is chosen randomly over M independent iterations, but the condition c_i is chosen such that s_i is the reference dimension (see also Equation 4.7). The underlying statistical test $\mathcal{T}$ depends on the type of attribute s_i. For example, if the attribute is of numerical type, we use the two-sample Kolmogorov-Smirnov test (see Section 4.2.6).

Thus, our quality function inherits the benefits of MCDE. The incremental index structure gives way to an efficient (**C1**) estimation of subspace quality on streams (see Section 4.6.2). Our index operates on a sliding window, so it only requires a single scan (**C2**) and supports efficient insert/deletion operations. Furthermore, the anytime flexibility of MCDE helps to control the trade-off between the accuracy of the search and execution time (**C4**). Finally, MCDE handles heterogeneous streams (**C5**).

To monitor the quality estimates of each subspace in $\mathbb{S}_t$ over time, we smooth out the statistical fluctuations of the estimation process via exponential smoothing with parameter γ (as in Equation 4.20). Previous experiments from Chapter 4 showed that $M = 100$ with $\gamma = 0.9$ leads to good estimation quality and performance w.r.t. downstream tasks. The outcome is a smoothed quality function $Q_t : D \mapsto [0, 1]$:

$$Q_{t+1}(s_i) = \gamma \cdot q_t(\mathbb{S}_t(s_i), s_i) + (1 - \gamma) \cdot q_{t+1}(\mathbb{S}_t(s_i), s_i). \tag{6.3}$$

6.3.1.3. Update

The initialisation is expensive, because one needs to run the search (Algorithm 6.1) for each dimension. While running the search for every dimensions optimises the quality of the subspace set (Objective O_1), it curbs efficiency (Objective O_2). However, as in Example 6.1, the dependence between dimensions can unexpectedly change over time. Thus, without any assumption, it is not possible to exploit our knowledge from step $t - 1$. We mitigate the computational cost over time by only repeating the search for a few dimensions.

The challenge is to find a policy $\pi : t \mapsto \mathcal{P}(D)$ to decide at any time step t for which L dimensions to repeat the search, where $L < |D|$ is a budget per time step. The budget L can either be set by the user, or it is based on the available computational time available between two subsequent observations. Basically, it is a trade-off between O_1 and O_2.

To solve this challenge, we cast the decisions of the policy π as a MAB problem with multiple plays [UNK10]. Multi-Armed Bandit (MAB) models are useful tools to capture the trade-offs of sequential decision making problems. We model each dimension $s_i \in D$ as an 'arm'. In each round $t = \{1, \ldots, T\}$, the policy π selects $L < |D|$ arms $I(t) \subset D$ and runs the search for each $s_i \in I(t)$. Then, there are two possible outcomes for each s_i:

1. **Success**: $\textsc{Search}_t(s_i)$ yields a better subspace than $\mathbb{S}_{t-1}(s_i)$, so we update $\mathbb{S}_t(s_i)$. The policy receives a reward of 1.
2. **Failure**: $\textsc{Search}_t(s_i)$ does not yield a better subspace, so we set $\mathbb{S}_t(s_i) \leftarrow \mathbb{S}_{t-1}(s_i)$. The reward is 0.

The S-MAB model [FKB19] (Chapter 5) provides an adequate framework to optimise the gains from the policy π in such setting. Let each dimension be an arm, i.e., $K = |D|$. We associate arm $i \in [K]$ with a Bernoulli distribution $\mathcal{B}_i$ with mean μ_i. At each round $t = 1, \ldots, T$, a policy selects a set of L_t arms $I(t) \subset [K]$ and repeat the search for these arms/dimensions. The policy receives a reward vector $X(t)$. The reward $X_i(t) = 1$ if the search is successful, 0 otherwise. Selecting an arm $i \in [K]$ leads to a cost in $O(|D|)$, normalised to 1. The policy aims to maximise the sum of the rewards, but such that it is greater than the sum of the costs by an efficiency factor $\eta^* \in [0, 1]$. The parameter η^* models the trade-off between objective O_1 and O_2. If $\eta^* \approx 0$, then minimising O_1 (improving quality) is much more important than O_2 (reducing computation). Conversely, whenever $\eta^* \approx 1$, one must minimise computation as much as possible (see Section 5.2).

Thus, we use Scaling Thompson Sampling (S-TS) as an update strategy for SGMRD. Algorithm 6.2 is the pseudo-code for our strategy. We initialise the statistics for each arm/dimension, α_i, β_i, N_i and S_i to 1 and the initial number of plays $L = |D|$. The scaling policy, Kullback-Leibler Scaling (KL-S), requires an additional parameter η^*, as discussed earlier (cf. Algorithm 5.1).

Algorithm 6.2 $\textsc{Update}(\mathbb{S}_t)$

Require: A set of subspaces $\mathbb{S}_t$, A target efficiency η^* and number of plays L (For KL-S, the scaling procedure)

for $i = 1, \ldots, K$ **do**
  $\theta_i(t) \sim \text{Beta}(\alpha_i(t), \beta_i(t))$
$I(t) := \arg\max_{K' \subset [K], |K'|=L} \sum_i^{K'} \theta_i(t)$
for $i \in I(t)$ **do**
  $S \leftarrow \textsc{Search}_t(s_i)$ ▷ Serarching for a new subspace w.r.t s_i (Algorithm 6.1)
  if $q_{t+1}(S, s_i) > Q_{t+1}(\mathbb{S}_t(s_i), s_i)$ and $S \neq \mathbb{S}_t(s_i)$ **then**
    $\mathbb{S}_{t+1}(s_i) \leftarrow S$
    $X_i(t) \leftarrow 1$
  else
    $\mathbb{S}_{t+1}(s_i) \leftarrow \mathbb{S}_t(s_i)$
    $X_i(t) \leftarrow 0$
  $\alpha_i(t+1) = \alpha_i(t) + X_i(t)$ ▷ Update parameters
  $\beta_i(t+1) = \beta_i(t) + (1 - X_i(t))$
  $L \leftarrow \text{KL-S}(L)$ ▷ Scaling policy (Algorithm 5.1)
return $\mathbb{S}_{t+1}$

6.3.1.4. SGMRD

Algorithm 6.3 summarises our approach as pseudo-code. SGMRD finds an initial set of subspaces using the first window (Line 2). Then SGMRD monitors and updates the set of subspaces (Line 5 to 7) for each new observation $\vec{x}_{t+1}$. The parameter v controls the frequency of the update steps.

Algorithm 6.3 SGMRD((D, B), w, v, γ)

Require: A data stream (D, B), a window size $w > 0$, the frequency of updates $v > 0$, decay $\gamma \in (0, 1)$

1: $t \leftarrow w$ ▷ **1. Initialisation**
2: $\mathbb{S}_t \leftarrow \{s_i : \text{SEARCH}_t(s_i), \forall s_i \in D\}$ ▷ Initial search. Algorithm 6.1
3: **while** B has a new observation $\vec{x}_{t+1}$ **do** ▷ **2. Maintenance**
4: $t \leftarrow t + 1$
5: **for** $s_i \in D$ **do** ▷ Monitoring
6: $Q_t(s_i) = \gamma \cdot q_{t-1}(\mathbb{S}_{t-1}(s_i), s_i) + (1 - \gamma) \cdot q_t(\mathbb{S}_t(s_i), s_i)$
7: **if** $t - w \equiv 0 \pmod{v}$ **then** $\mathbb{S}_t \leftarrow \text{UPDATE}(\mathbb{S}_{t-1})$ **else** $\mathbb{S}_t \leftarrow \mathbb{S}_{t-1}$ ▷ Updating. Algorithm 6.2
8: **return at anytime** the set of subspaces $\mathbb{S}_t$

Overall, our method is efficient (**C1**), as our quality estimates can be computed in linear time. It also requires a single scan of the data (**C2**), as we monitor subspaces over a sliding window. By design, SGMRD adapts (**C3**) to the environment by updating the subspace search results with parsimonious resource consumption, and our experiments will confirm this. Finally, results are available at any point in time (**C4**).

6.3.2. Downstream Knowledge Discovery

High-quality subspaces can be useful for virtually any downstream Data Mining task. For example, previous work [KMB12; NMV16; Wan+17; TB19] leverages subspaces to build ensemble-like outlier detectors. Other Data Mining tasks are possible as well, such as clustering [PL07; ZLW07; Agr+05; Agg09; KPM06]. Our approach, SGMRD, yields a set of subspaces $\mathbb{S}_t$ at any time. While our approach is not tied to any specific Data Mining task, we use outlier detection as an exemplary task in our evaluation.

The articles just cited apply an outlier detector to each subspace in $\mathbb{S}_t$, and the final score is a combination of the individual scores. The outcome is a ranking of objects by decreasing 'outlierness'. In our experiments, we use the Local Outlier Factor [Bre+00] (LOF) detector because it is a common baseline in the outlier detection literature [KMB12].

The best combination of the individual scores depends on the concrete application. Literature has discussed this extensively [AS15; Ngu+16]. Several studies [LK05; PLL07] argue that the average of the scores with LOF yields the best results overall in the static case, so we stick to this choice. The final outlier score of $\vec{x}_t$ is the average of the scores from each subspace across every window containing $\vec{x}_t$:

$$score\,(\vec{x}_t) = \frac{1}{w \cdot |D|} \sum_{i=0}^{w} \sum_{s_i}^{\mathbb{S}_{t-i}} score_{W_{t-i}}^{\mathbb{S}_{t-i}(s_i)}(\vec{x}_t) \tag{6.4}$$

Again, one may also reduce the computation effort of outlier detection by evaluating the scores only once every v time steps, as with SGMRD.

6.4. Experiment Setup

We evaluate the performance of our approach w.r.t. two aspects: (1) the quality of subspace monitoring, i.e., how efficiently and effectively can SGMRD maintain a set of high-quality subspaces over time, and (2) the benefits w.r.t. outlier detection, as an exemplary downstream Data Mining task. To facilitate reproducibility, we release our source code with documentation on GitHub[2].

6.4.1. Evaluation Measures

6.4.1.1. Subspace Monitoring

Evaluating the quality of subspace monitoring is difficult because finding $\mathbb{S}_t^*$ is computationally infeasible for non-trivial data. Thus, based on the definition of objective O_1 (cf. Section 6.2.2), we measure the regret and the average quality:

$$R_T = \frac{1}{|D|} \sum_{t=0}^{T} \sum_{i=1}^{D} \left[Q_t^*(s_i) - Q_t(s_i) \right], \qquad \overline{Q}_t = \frac{1}{|D|} \sum_{i=1}^{|D|} Q_t(s_i), \tag{6.5}$$

where $Q_t(s_i)$ is the quality w.r.t. s_i as defined in Equation 6.3, and $Q_t^*(s_i)$ is the quality obtained when one always updates the corresponding subspace, i.e., it the same as repeating the initialisation. R_T is the regret, defined as the sum of the differences between $Q_t^*(s_i)$ and $Q_t(s_i)$ up to T. $\overline{Q}_t$ is the average quality at time t, and $\overline{Q}_T$ is the average quality up to T.

To characterise the behaviour of our update strategy, we also look at the relative update frequency $F_T(s_i)$ for each dimension $s_i \in D$ and the rate of successful updates U_T:

$$F_T(s_i) = \sum_{t=0}^{T} \frac{\mathbf{1}\left[s_i \in I(t)\right]}{T}, \qquad U_T = \sum_{t=0}^{T} \sum_{i=1}^{|D|} \frac{\mathbf{1}\left[A_t(s_i) \wedge B_t(s_i)\right]}{|D| \cdot T}, \tag{6.6}$$

where $s_i \in I(t)$ means that SGMRD has selected dimension s_i at time t, and $A_t(s_i)$ and $B_t(s_i)$ are the conditions capturing whether the search was successful (i.e., the new subspace is different and of higher quality than the previous one):

$$A_t(s_i) = \mathbb{S}_{t-1}(s_i) \neq \mathbb{S}_t(s_i), \tag{6.7}$$

$$B_t(s_i) = Q_t(\mathbb{S}_{t-1}(s_i), s_i) < q_t(\mathbb{S}_t(s_i), s_i). \tag{6.8}$$

6.4.1.2. Outlier Detection

By definition, outliers are rare, so the detection of outliers is an imbalanced classification problem. We report the area under the Receiver Operating Characteristic (ROC) curve (AUC) and the Average Precision (AP), which are popular measures for evaluating outlier detection algorithms. Outlier detectors typically ranks the observation by decreasing 'outlierness' score. In most applications, end users only check the top X% items. So we report the Recall (R) and Precision (P) within the top X% instances, with X $\in \{1, 2, 5\}$.

[2] https://github.com/edouardfouche/SGMRD

6.4.2. Data Sets

We use an assortment of data sets for our evaluation. We use a data set from our real-world use case, Bioliq (cf. Section 1.1.5), and two real-world data sets frequently used in the literature, KDDCup99 and Activity. To cope with the lack of publicly available data sets for outlier detection in the streaming setting, we also generate three synthetic benchmark data sets: Synth10, Synth20 and Synth50. We describe each data set in detail hereafter; Table 6.1 summarises their main characteristics. Our data sets only contain continuous attributes, so we use KSP for our quality measure (cf. Section 4.4).

Table 6.1.: Characteristics of the Benchmark Data Sets (SGMRD).

Benchmark	Type	# Instances	# Dimensions	% Outliers
Bioliq	Real	10,000	100	NA
KDDCup99	Real	25,000	38	7.12
Activity	Real	22253	51	10
Synth10	Synthetic	10,000	10	0.86
Synth20	Synthetic	10,000	20	0.88
Synth50	Synthetic	10,000	50	0.81

6.4.2.1. Real-World Data Sets

- Bioliq: This data set contains 10,000 measurements (one per second) from a selection of 100 sensors, such as temperature or pressure, in various components of the Bioliq plant. We use this data set to evaluate how well our method can search for subspaces in data streams. However, there is no ground truth, so we cannot use this data set for our downstream Knowledge Discovery application.
- Activity: This data set, initially proposed in [RS12], describes different subjects performing various activities (e.g., walking, running), monitored via body-mounted sensors. Analogously to [SA18], we took the *walking data* of a single subject and replaced 10% of the data with *nordic walking* data, which we marked as outliers. The rest of the elements are inliers. We obtained the original data set from [DG17].
- KDDCup99: This data set was part of the KDD Cup Challenge 1999. It is a network intrusion data set. Analogously to [SA18], we excluded DDoS (Denial-of-Service) attacks and marked all other attacks as outliers. We take a contiguous subset of 25,000 data points. We obtained this data set from [DG17].

Unfortunately, there exist only a few publicly-available data sets with outlier ground truth, in particular in the streaming setting. Thus, we create three additional data sets, inspired by the static benchmarks in [KMB12; TB19]. Our data sets simulate *concept drift* [BGE15] via random variations of the data distribution over time. In the next section, we describe how we generate our synthetic benchmark.

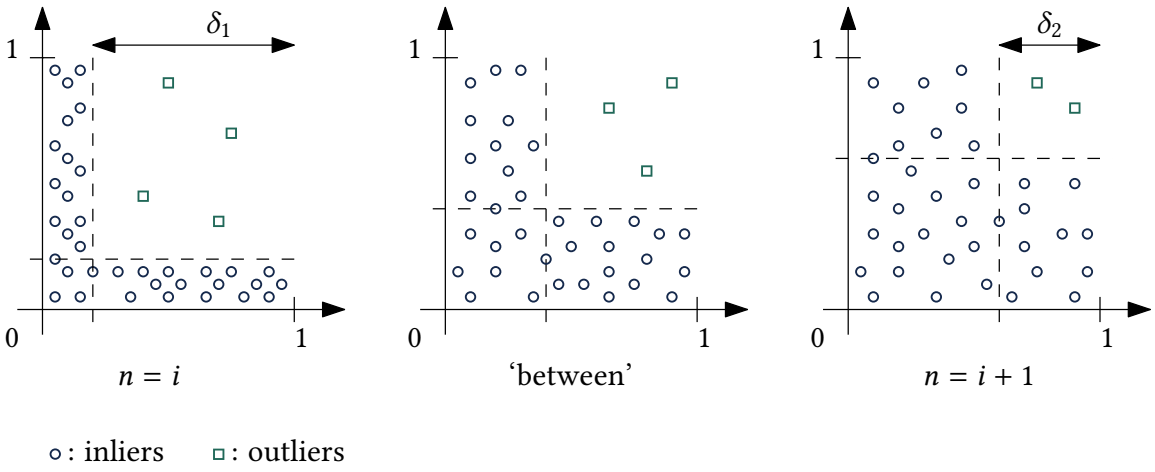


Figure 6.5.: SGMRD: Synthetic Benchmark Generation (Example).

6.4.2.2. Synthetic Benchmark Generation

In addition, we create three data sets simulating *concept drift* [BGE15] via random variations of the data distribution over time. We generate $n + 1$ distributions $\Gamma_0, \Gamma_1, \ldots, \Gamma_n$, and sample from each distribution Γ_i a number e of observations, while letting the distribution Γ_i gradually drift towards the distribution Γ_{i+1} as we sample from it.

We initialise $\Gamma_0 \equiv \mathcal{U}[0, 1]$ over the full space D. Then we select a set of distinct subspaces (i.e., the subspaces do not have any dimension in common) from $\mathcal{P}(D)$ for each other distribution, so that 50% of the subspaces change from one distribution to the next one. For each subspace and, with a small probability $p \in (0, 1)$, we sample the next point from $\mathcal{U}[\delta, 1]$ for each dimension with $\delta \in (0, 1)$ chosen randomly. We call this point an 'outlier'. With probability $1 - p$, we sample the next point uniformly from the rest of the unit hypercube – this is an 'inlier'. Figure 6.5 illustrates this principle with two dimensions. For the dimensions not part of any subspace, every observations are i.i.d. in $\mathcal{U}[0, 1]$.

Outliers placed this way are said to be 'non-trivial' [KMB12], since they do not appear as outliers in any other subspace — they are 'hidden' in the data. Our goal is to evaluate to what extent different approaches can detect such outliers.

We generate three benchmark data sets with $n = 10$ distributions and $e = 1000$ observations. We set $p < 0.01$, since outliers are rare by definition. Synth10, Synth20, Synth50 have 10, 20 and 50 dimensions respectively, with subspaces up to 5 dimensions.

Note that we release the code for our data generator and our real-world benchmark data sets via our GitHub repository as well.

6.4.3. Baselines and Competitors

6.4.3.1. Subspace Monitoring

Our goal is to assess how effectively SGMRD can handle the trade-off between computational cost and monitoring quality. We compare SGMRD to several alternative update strategies and against several baselines.

- SGMRD-S-TS uses Scaling Thompson Sampling (S-TS) as an update strategy. We let the efficiency parameter η^* vary from 0.1 to 0.9.
- SGMRD-TS uses Thompson Sampling [Tho33] (TS) as update strategy. This approach is similar to SGMRD-S-TS, but the number of updates does not change over time.
- SGMRD-RD uses a random (RD) update strategy. We update a single subspace per time step, chosen randomly from the current set of subspaces.
- SGMRD-GD uses a greedy (GD) update strategy. We update a single subspace per time step and choose the subspace with the lowest quality in the current set.
- Batch repeats the initialisation of SGMRD periodically for every batch of data with size $w = 1000$.
- Init runs the initialisation and then keeps the same set of subspaces for the rest of the experiment (no update).
- Gold repeats the initialisation of SGMRD at every step. This baseline represents the highest level of quality that one can reach with this instantiation of SGMRD, but it also is the most expensive configuration. In fact, we can only afford to run it on the Bioliq data set.

6.4.3.2. Outlier Detection

We compare the results obtained with SGMRD-TS against the following outlier detectors:

- RS-Stream is an adaptation of the RS-Hash [SA16] outlier detector to the streaming setting, presented in [SA18]. It estimates the outlierness of each observation via randomised hashing over random projections. We reproduce the approach and use the default parameters recommended by the authors.
- LOF [Bre+00] is a well-known outlier detector. We run it periodically and average the scores over a sliding window in the full space. We use the implementation from ELKI [Sch+15], which profits from efficient index structures.
- xStream [MLA18] is an ensemble outlier detector. xStream estimates densities via randomised ensemble binning from a set of random projections. xStream declares the points lying in low density areas as outliers. We use the implementation from the authors with the recommended parameters.
- StreamHiCS [Bec16] is an adaptation of [KMB12]. Based on a change detector, StreamHiCS repeats the computation from [KMB12] on a data synopsis (so-called micro-clusters). We use the reference implementation with default parameters.

We average the scores obtained from each detector over a sliding window of size $w = 1000$ for every $v = 100$ time steps (cf. Section 6.3.2). For approaches based on LOF, such as ours, we repeat the computation with parameter $k \in \{1, 2, 5, 10, 20, 50, 100\}$ and report the best result in terms of AUC. The performance may vary widely w.r.t. this parameter; this is a well-known caveat of LOF [Cam+16]. We average every result from 10 independent runs. Each approach runs single-threaded on a server with 20 cores at 2.2GHz and 64GB RAM. We implement our algorithms in Scala.

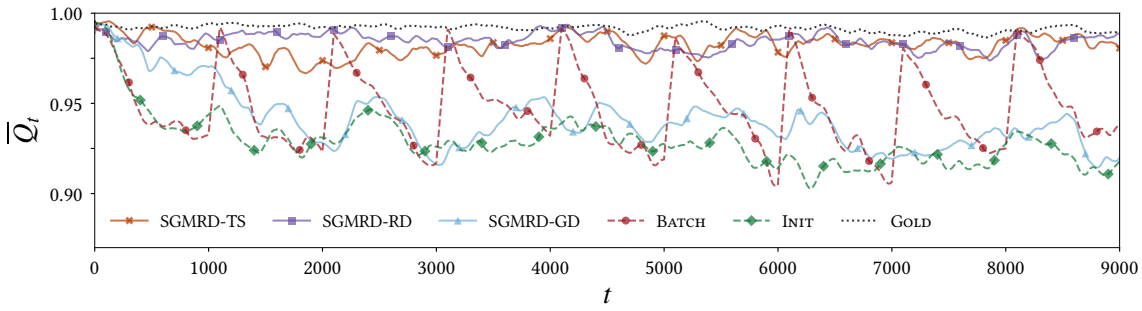


Figure 6.6.: Average Quality at time t (Bioliq, $L = 1$, $v = 2$).

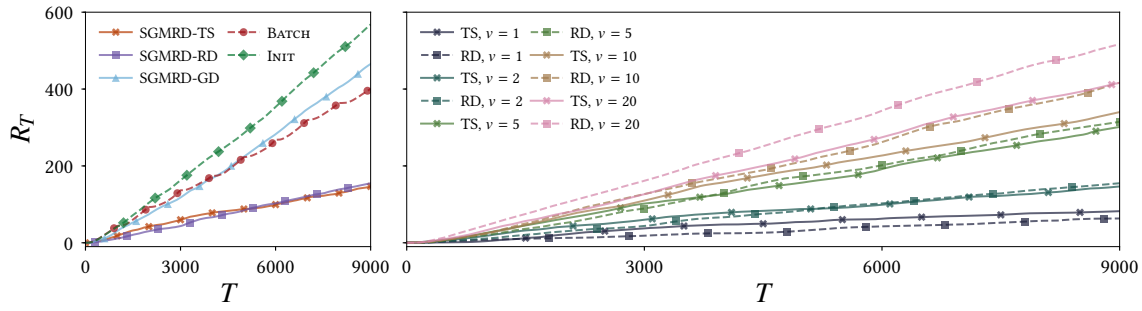


Figure 6.7.: Regret up to T (Bioliq, $L = 1$, left: $v = 2$).

6.5. Results

6.5.1. Subspace Monitoring

We first evaluate the quality of monitoring from SGMRD. We set $w = 1000$, $v = 2$ and $L = 1$, i.e., for each update strategy, SGMRD only keeps the latest 1000 observations, and, for any new two observations, SGMRD attempts to replace one of the current subspaces.

As we can see in Figure 6.6, both SGMRD-TS and SGMRD-RD can keep the quality $\overline{Q}_t$ close to that of Gold, our strongest and most expensive baseline. In the beginning, SGMRD-TS seems to perform slightly worse than SGMRD-RD, but after some time (once SGMRD-TS has learned its update strategy), it tends to dominate SGMRD-RD. We can see that Batch occasionally leads to the same quality as Gold, but the quality drops quickly between the update steps. SGMRD-GD is not much better than Init (no monitoring).

Figure 6.7 confirms our observations: While the regret of SGMRD-TS is slightly worse than the one of SGMRD-RD at the beginning, it becomes better afterwards. The other approaches lead to much higher regret. For larger step size v, we see that SGMRD-TS is superior to SGMRD-RD. For smaller v, the environment does not change much between observations, and it is more difficult to learn which subspaces to update more frequently.

In Figure 6.8, the update frequencies show how the strategies differ. As expected, Random (RD) updates each subspace uniformly. Greedy (GD) tends to focus only on a few subspaces; most subspaces are never replaced, although they may become suboptimal as well. TS focuses more on some subspaces, the ones requiring more frequent updates.

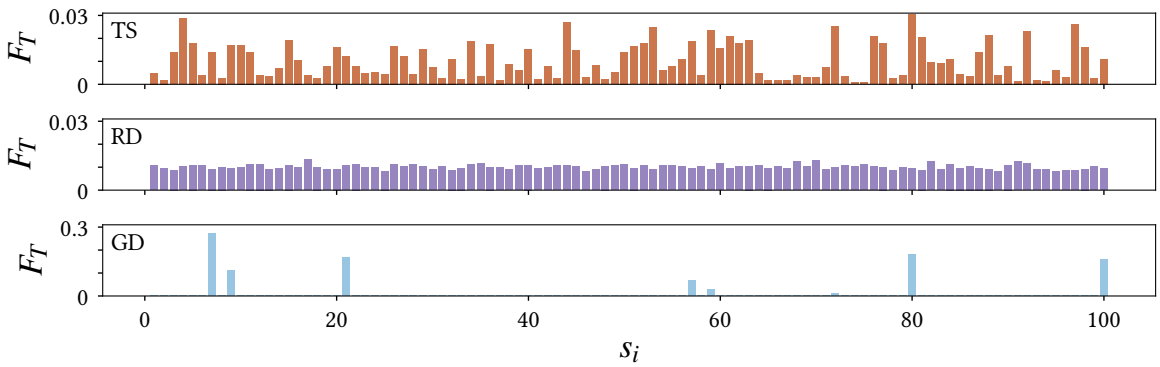


Figure 6.8.: Frequency of update (BIOLIQ, $v = 2$, $L = 1$).

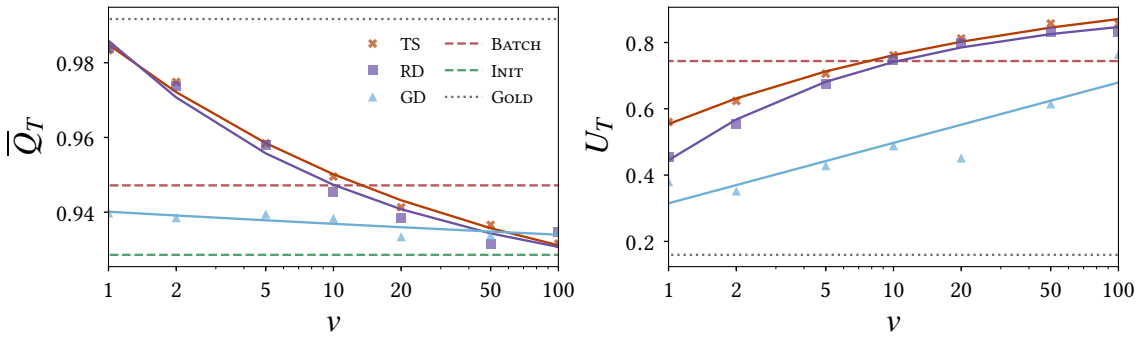


Figure 6.9.: Quality / Success Rate w.r.t. step size v (BIOLIQ, $L = 1$).

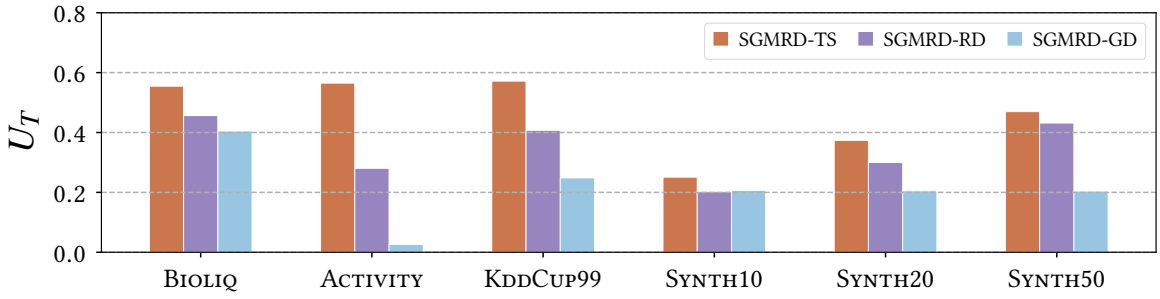


Figure 6.10.: Success Rate ($v = 1$, $L = 1$).

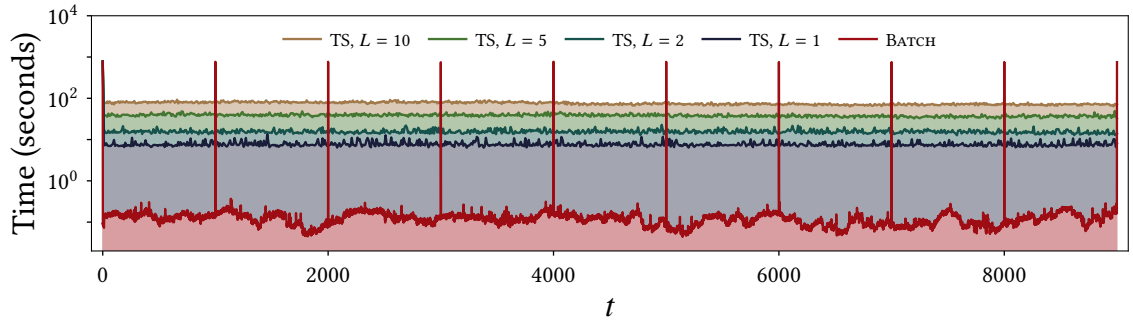


Figure 6.11.: Stream Processing Time (Bioliq, $v = 1$).

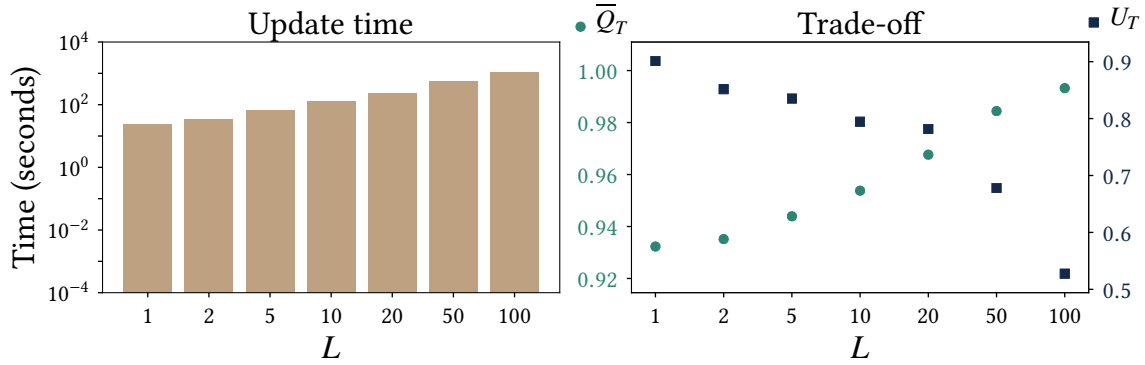


Figure 6.12.: Quality/efficiency (Bioliq, SGMRD-TS, $v = 100$).

Figure 6.9 shows that the average quality $\overline{Q}_T$ tends to decrease as we increase the update step v. However, the rate of successful updates U_T (Equation 6.6) increases. As v increases, it is more likely for any subspace to become suboptimal. For Batch, U_T is high, but the quality $\overline{Q}_T$ is low. We observe the opposite for Gold. SGMRD is a trade-off between these two extremes, and the strategy based on Thompson Sampling [Tho33] (TS) appears superior to others, both w.r.t. $\overline{Q}_T$ and U_T. Figure 6.10 shows that our observations are not only valid for the Bioliq data set, but also for other benchmarks. SGMRD-TS consistently achieves a higher rate of successful updates than other approaches.

Figure 6.11 highlights an important drawback of previous methods: The computation for batch-wise techniques is concentrated in a few discrete time steps. For stream mining, it is better to distribute computation uniformly over time. Computation-intensive episodes can lead to long response times of the system, and this contradicts the efficiency sought (**C1**) and anytime behaviour (**C4**). Besides this, the system becomes unable to adapt to the environment (**C3**) between the different episodes.

Next, we set $v = 100$ and let L vary to observe the trade-off between quality of the subspaces and the efficiency of the search in SGMRD-TS (see Figure 6.12). As L increases, the cost of updating subspaces increases linearly. Similarly, the quality Q_T increases while the rate of successful updates U_T decreases.

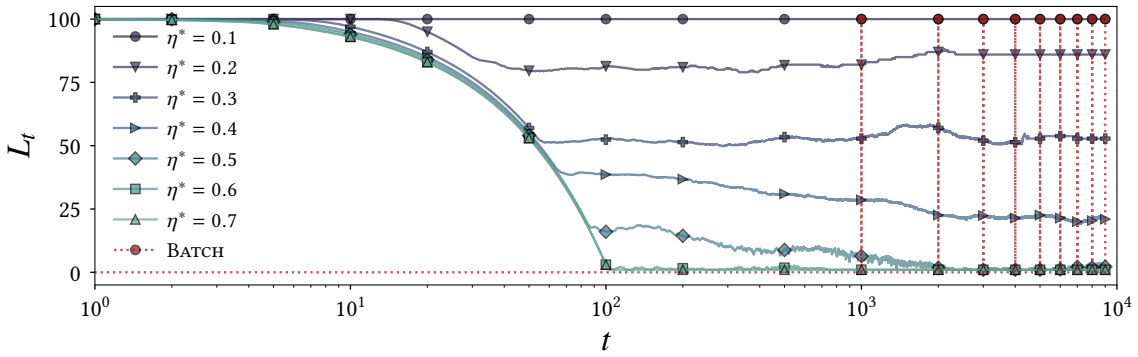


Figure 6.13.: SGMRD-S-TS (Biolig, $v = 1$).

Table 6.2.: Comparison with Scaling Bandits (Biolig, $v = 1$).

Baseline	L_T	$\overline{Q}_T$	U_T	R_T/T	**SGMRD-S-TS**	L_T	$\overline{Q}_T$	U_T	R_T/T
TS	1	95.43	0.61	3.74	$\eta^* = 0.1$	100.00	99.45	0.16	0.09
RD	1	95.20	0.54	3.97	$\eta^* = 0.2$	74.13	99.19	0.20	0.08
GD	1	93.72	0.43	5.45	$\eta^* = 0.3$	54.48	99.22	0.29	0.06
Batch	–	95.16	0.81	4.01	$\eta^* = 0.4$	36.78	99.33	0.37	0.78
Init	–	92.50	–	6.67	$\eta^* = 0.5$	5.73	98.79	0.46	2.46
Gold	100	99.16	0.16		$\eta^* = 0.6$	1.68	97.62	0.53	2.81
					$\eta^* = 0.7$	1.58	97.68	0.53	2.26
					$\eta^* = 0.8$	1.57	97.75	0.55	2.32
					$\eta^* = 0.9$	1.56	97.31	0.55	2.59

Scaling Multi-Armed Bandit (S-MAB) algorithms are convenient tools for controlling this trade-off. Figure 6.13 shows the number of plays/updates L_t over time for SGMRD-S-TS. As we can see, L_t converges to an optimal amount of updates according to the efficiency criterion η^* for SGMRD-S-TS. Our scaling policy (KL-S, Algorithm 5.1) scales up the number of updates/plays until we are confident that doing so violates the efficiency constraint. Note that, in non-static environments, we can further adapt the number of updates using Adaptive Windowing [BG07] (ADWIN), e.g., as in Section 5.3.

As we can see in Table 6.2, SGMRD-TS has the highest quality after Gold, the best success rate U_T after Batch and the smallest average regret among the baselines. It is interesting to see that the Scaling Bandits consistently lead to higher quality and smaller regret. Also, SGMRD-S-TS adjusts the number of plays so that U_T approximately matches η^*. Naturally, this only works up to a certain point, depending on the distribution of rewards in the environment.

In conclusion, the experiments show that SGMRD is a useful tool to monitor high-quality subspaces over time, and it is highly versatile. Based on the available hardware, users can set the number of plays per round, as with SGMRD-TS, to obtain the highest quality from this budget. If the computation is no bottleneck, then users can set an efficiency threshold, which leads to an adequate use of resources for subspaces monitoring.

In the next section, we show that SGMRD-TS helps to detect outliers and compare the results with state-of-the-art outlier detectors for data streams.

Table 6.3.: Outlier Detection Performance.

Benchmark	Approach	AUC	AP	P1%	P2%	P5%	R1%	R2%	R5%
Activity	**SGMRD**	**97.32**	**85.39**	**94.59**	**94.83**	**94.24**	**9.44**	**18.97**	**47.10**
	LOF	93.93	61.80	74.32	64.72	64.03	7.42	12.94	32.00
	StreamHiCS	88.52	47.38	70.72	54.61	51.89	7.06	10.92	25.93
	RS-Stream	95.95	68.23	71.62	72.58	75.00	7.15	14.52	37.48
	xStream	77.71	20.41	3.60	10.14	16.31	0.36	2.02	8.13
KddCup99	**SGMRD**	**69.98**	**10.29**	**0.00**	**0.20**	**0.56**	**0.00**	**0.06**	**0.39**
	LOF	65.07	9.57	0.00	0.00	0.08	0.00	0.00	0.06
	StreamHiCS	57.11	7.89	0.00	0.00	0.08	0.00	0.00	0.06
	RS-Stream	43.21	5.73	0.00	0.00	0.08	0.00	0.00	0.06
	xStream	52.70	8.23	0.00	0.20	0.08	0.00	0.06	0.06
Synth10	**SGMRD**	**92.70**	**59.93**	**50.00**	**26.00**	**12.00**	**58.14**	**60.47**	**69.77**
	LOF	88.77	31.44	33.00	18.50	10.40	38.37	43.02	60.47
	StreamHiCS	88.81	31.16	33.00	19.00	10.40	38.37	44.19	60.47
	RS-Stream	71.23	1.87	0.00	0.00	2.80	0.00	0.00	16.28
	xStream	68.51	2.58	5.00	3.00	4.00	5.81	6.98	23.26
Synth20	**SGMRD**	**85.05**	**41.19**	**36.00**	**19.50**	**9.20**	**40.91**	**44.32**	**52.27**
	LOF	72.55	5.57	8.00	6.00	4.40	9.09	13.64	25.00
	StreamHiCS	71.71	5.37	8.00	6.00	4.00	9.09	13.64	22.73
	RS-Stream	48.39	0.80	0.00	0.00	0.00	0.00	0.00	0.00
	xStream	63.64	1.58	1.00	1.50	2.20	1.14	3.41	12.50
Synth50	**SGMRD**	**75.87**	**31.27**	**27.00**	**16.00**	**7.60**	**33.33**	**39.51**	**46.91**
	LOF	61.38	1.08	0.00	0.50	0.60	0.00	1.23	3.70
	StreamHiCS	63.90	12.00	11.00	6.00	3.40	13.58	14.81	20.99
	RS-Stream	46.52	0.73	0.00	0.00	0.00	0.00	0.00	0.00
	xStream	48.43	0.90	1.00	0.50	1.40	1.23	1.23	8.64

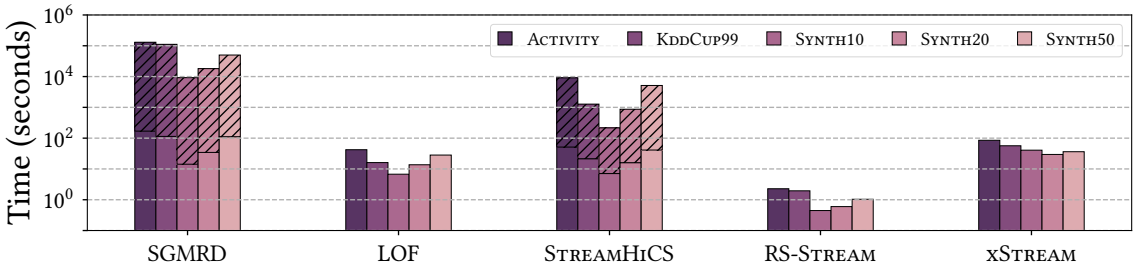


Figure 6.14.: Outlier Detection Time (hatched area: search time).

6.5.2. Outlier Detection

We leverage the subspaces obtained from SGMRD-TS to detect outliers, as in Section 6.3.2. We set $w = 1000$, $v = 1$ and $L = 1$. Table 6.3 shows the results. SGMRD clearly leads to the best results w.r.t. each benchmark. It is interesting to see that LOF turns out to be our most competitive baseline and often outperforms our competitors.

In Figure 6.14, we can see that our competitors, in particular RS-Stream and xStream, are much faster than SGMRD, but they often are not much better than random guessing. Most of the computation required by SGMRD and StreamHiCS is due to the search. Nonetheless, one may reduce the required computation with SGMRD, e.g., increase v, without decreasing detection quality by much.

6.6. Discussion

Finding interesting subspaces is fundamental to any step of the Knowledge Discovery from Data (KDD) process. We have proposed a new method, SGMRD, to bring subspace search to streams. Our approach leverages our previous contributions: on the one hand, we use MCDE (Chapter 4) to estimate the quality of subspaces. We can monitor the resulting estimates efficiently, thanks to our incremental index structure. On the other hand, SGMRD learns an update strategy to determine the subspaces to update at each time step. We use bandit theory and S-MAB algorithms (Chapter 5) to address the trade-off between the quality of monitoring and the required computation.

Our experiments not only show that SGMRD leads to efficient monitoring of subspaces, but also to state-of-the-art results w.r.t. downstream Knowledge Discovery tasks, such as outlier detection. One may expect similar benefits for other mining tasks on streams.

An administrator can control monitoring via two parameters: the number of plays per round L and the step size v. While SGMRD can leverage S-MAB to decide the appropriate number of updates per steps L automatically, finding the most adequate step size v for a specific problem is not trivial. In future work, it would be interesting to extend the S-MAB framework, such that the underlying bandit algorithm can not only decide which arms and how many arms to play, but also whether to play at all.

Next, we focus on a more specific Knowledge Discovery task: mining text outliers. This task is more challenging than typical outlier detection tasks, so we first address it in the static setting. Nonetheless, it is easy to see how SGMRD could help to transfer the problem to the streaming setting, as we discuss in our future work (Chapter 9).

7. Mining Text Outliers

The results from this chapter have been published as follows:

- Edouard Fouché, Yu Meng, Fang Guo, Honglei Zhuang, Klemens Böhm and Jiawei Han. 'Mining Text Outliers in Document Directories'. In: *ICDM*. In press. IEEE Computer Society, 2020

Keywords: Text Mining; Anomaly Detection; Data Cleaning; Nearest-Neighbour Search

7.1. Chapter Overview

Nowadays, it is common to classify collections of documents into (human-generated, domain-specific) directory structures, such as email or document folders. But documents may be classified wrongly, for a multitude of reasons. Then they are outlying w.r.t. the folder they end up in. Orthogonally to this, and more specifically, two kinds of errors can occur: (O) Out-of-distribution: the document does not belong to any existing folder in the directory; and (M) Misclassification: the document belongs to another folder. It is this specific combination of issues that we address in this article, i.e., we mine text outliers from massive document directories, considering both error types (see also the motivations in Section 2.3.2). We make the following contributions:

We explore the problem of text outlier detection in document directories. The task is challenging because text can be outlying in numerous ways, and directories are domain-specific. At the same time, text outliers fall into two categories: Types O/M. To our knowledge, we are first to propose an integrated outlier detection framework for text documents that builds on this conceptual distinction.

We propose a new proximity-based approach to detect text outliers, which we name kj-Nearest Neighbours (kj-NN). Our approach leverages similarities of documents and phrases, based on state-of-the-art embedding methods [Men+19], to detect both Type O/M outliers. By extracting semantically relevant labels and the documents similar to each outlier, it also supports interpretability.

We introduce a 'self-supervision' mechanism to make our approach more robust. Our idea is to weigh each decision by the relevance of neighbouring documents. A document is said to be relevant w.r.t. a given class when its semantics, characterised by its closest phrases in the embedding space, is representative of its class.

We conduct extensive experiments to compare our method to competitors and provide example outputs. The experiments show that our approach improves the current state of the art by a large margin while delivering interpretable results.

We release our source code on GitHub[1], together with our benchmark data, to ensure the reproducibility of our experiments.

[1] https://github.com/edouardfouche/MiningTextOutliers

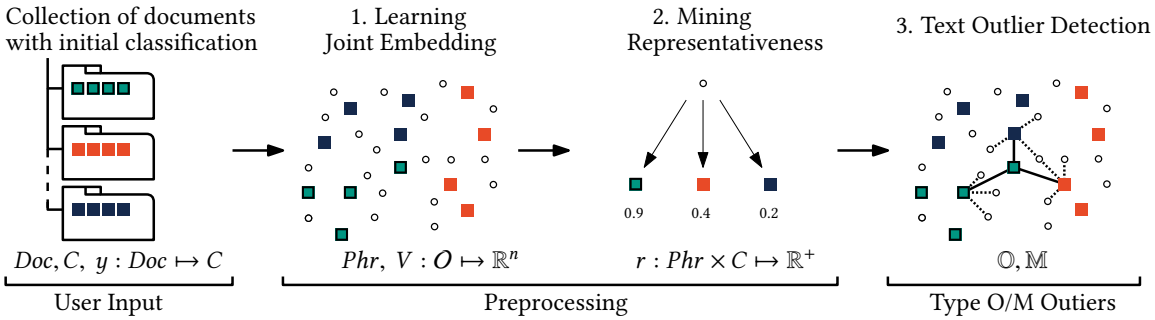


Figure 7.1.: Our Framework: A High-Level Overview. In the illustrations, squares are documents and dots are phrases.

7.2. The kj-NN Algorithm

7.2.1. Our Framework

Figure 7.1 provides a high-level overview of our framework. We assume as input a collection of documents *Doc*, with an initial but imperfect classification $y : Doc \mapsto C$ provided by the user. Squares stand for documents, and the colours represent their initial class. Documents are composed of phrases, represented in turn as dots. We design our framework in three steps:

1. Learning Joint Embedding: Text outlier detection relies on textual similarity/dissimilarity measures, which can be effectively captured by embeddings. We segment the phrases *Phr* from every document using AutoPhrase [Sha+18] and obtain the phrase and document embeddings $V : O \mapsto \mathbb{R}^n$ in a joint spherical space via Joint Spherical Embedding [Men+19] (JoSE).

The benefits of using JoSE are twofold: (1) JoSE trains document and phrase embeddings jointly in the same space, where text similarity between phrases and documents can be directly derived. (2) JoSE captures directional similarity by training in the spherical space, which characterises textual similarity more effectively than Euclidean embeddings.

2. Mining Representativeness: We estimate the representativeness of each phrase for each class, based on the initial classification. The representative phrases for a class are indicative of the semantics of documents. As in [Tao+16], we define the representativeness as a function of three criteria:

- **Integrity:** A phrase with high integrity in a given corpus is meaningful, understandable and of high quality.
- **Popularity:** A phrase is popular in a given class if it has many occurrences.
- **Distinctiveness:** A phrase is distinctive if it distinguishes a class from others.

For each phrase $p \in Phr$ and class $c \in C$, we estimate the integrity $int(p, c) \in [0, 1]$, popularity $pop(p, c) \in \mathbb{R}^+$ and the distinctiveness $disti(p, c) \in [0, 1]$ as described in [ZH19]. The representativeness is the product of those three criteria:

$$r(p, c) = int(p, c) \cdot pop(p, c) \cdot disti(p, c). \tag{7.1}$$

The idea is that, even if each class may contain ambiguous documents or erroneous labels, these are 'rare', so the impact on the estimation of representativeness is low. Our experiments show that our method is robust against erroneous labels.

3. Text Outlier Detection: We introduce kj-NN, an outlier detector inspired by the well-known k-NN classifier [FH89]. The main novelty is that inferring the class of each element (document) does not only base on the class of its k nearest documents but also on their relevance. We estimate the relevance of a document as the average representativeness of its j nearest sub-elements (phrases) for its class. If a document d is closer to documents which are (i) relevant and (ii) from another single class, then d likely is a Type M outlier. On the other hand, if d is similarly close to relevant documents of various classes, then d likely is a Type O outlier. This final step yields two ranked lists $\mathbb{O}$ and $\mathbb{M}$ of outliers for Type O and Type M, respectively.

In our approach, self-supervision consists of estimating the relevance of the original label, recognising that less relevant labels must have a lower influence on our predictions. The rationale is to perceive irrelevant documents as such because they either were misclassified or have ambiguous semantics.

In the next section, we present the technical details of our outlier detector. We first provide a Bayesian formalisation, then describe the practical implementation of kj-NN.

7.2.2. Formalisation

We define the k nearest documents and the j nearest phrases of $d \in Doc$ as follows:

$$\mathcal{K}(d) = \{x \in Doc \setminus d : |\{x' \in Doc \setminus d : Sim(x', d) > Sim(x, d)\}| < k\},$$
$$\mathcal{J}(d) = \{x \in Phr : |\{x' \in Phr : Sim(x', d) > Sim(x, d)\}| < j\}.$$

where $Sim : \mathcal{O}^2 \mapsto [0, 1]$ is a similarity function between text objects (documents, phrases) defined as in Equation 2.3. We call $\mathcal{K}(d)$ and $\mathcal{J}(d)$ the k- and j-neighbourhood of d. Then one can build a classifier based on local densities. For each document $d \in Doc$, the k-neighbourhood provides an estimate of the density within each class, and the j-neighbourhood provides a pseudo-posterior probability for each neighbour.

Bayesian inference formulates the posterior probability of class membership of document $d \in Doc$ as follows:

$$\Pr[c|d] = \frac{\Pr[d|c] \Pr[c]}{\Pr[d]}. \tag{7.2}$$

Now think of a sphere of volume v centred at d that contains k points. Then we can express the likelihood as

$$\Pr[d|c] = \frac{|\mathcal{K}_c(d)|}{|D_c|v}, \tag{7.3}$$

where $\mathcal{K}_c(d)$ is the set of documents in $\mathcal{K}(d)$ of class c, and D_c is the set of all documents of class c. Traditionally, the class prior is

$$\Pr[c] = \frac{|D_c|}{|Doc|}. \tag{7.4}$$

Since v, $|Doc|$ and $\Pr[d]$ are independent of c, we have

$$\Pr[c|d] \propto |\mathcal{K}_c(d)|. \tag{7.5}$$

To minimise the misclassification probability, one must assign d to the class with the highest density in its neighbourhood.

This only works under the assumption that all labels are correct, which is unrealistic in our setting. Our idea is to weight each document d' in $\mathcal{K}_c(d)$ by a pseudo-posterior probability $\widehat{\Pr}(c|d')$, capturing our belief that they indeed are of class c:

$$\Pr[c|d] \propto \sum_{d'}^{\mathcal{K}_c(d)} \widehat{\Pr}[c|d'], \tag{7.6}$$

where we define $\widehat{\Pr}[c|d']$ to be proportional to the representativeness $r(p,c)$ of the phrases $p \in \mathcal{J}(d')$ for class c:

$$\widehat{\Pr}[c|d'] \propto \sum_{p}^{\mathcal{J}(d')} r(p,c). \tag{7.7}$$

With this specification of $\widehat{\Pr}[c|d']$, we exploit additional information from the j-neighbourhood as 'self-supervision signals'. In contrast, the standard k-NN classification rule assumes that $\Pr[c|d'] = 1, \forall d' \in \mathcal{K}_c(d)$, i.e., the labels of neighbours are accurate. Finally, predicting the class of document d boils down to

$$\hat{y}(d) = \arg\max_{c \in C} \Pr[c|d] = \arg\max_{c \in C} \sum_{d'}^{\mathcal{K}_c(d)} \sum_{p}^{\mathcal{J}(d')} r(p,c). \tag{7.8}$$

Whenever $\hat{y}(d) \neq y(d)$, we may declare that a user misclassified document$d$, i.e., it is a Type M outlier. However, the reliability of such predictions may vary widely. For example, if each class has a very similar posterior probability, deciding for one or the other might not be meaningful. When a document does not prominently belong to any existing class, we must declare a Type O outlier. We quantify the reliability of a prediction via the entropy of the posterior probabilities, which measures the uncertainty of the prediction:

$$I(d) = -\sum_{c \in C} \Pr[c|d] \cdot \log \Pr[c|d]. \tag{7.9}$$

We obtain the posterior probabilities via normalisation:

$$\Pr[c|d] = \frac{\sum_{d'}^{\mathcal{K}_c(d)} \sum_{p}^{\mathcal{J}(d')} r(p,c)}{\sum_{c}^{C} \sum_{d'}^{\mathcal{K}_c(d)} \sum_{p}^{\mathcal{J}(d')} r(p,c)}. \tag{7.10}$$

We decide whether a prediction is uncertain using a threshold $\Gamma > 0$ that we set to a percentile p^* of the empirical distribution of the entropy for every document in the corpus:

$$\Gamma > 0 \quad s.t. \quad \frac{|\{d \in Doc \mid I(d) < \Gamma\}|}{|Doc|} = p^*. \tag{7.11}$$

In other words, our idea is to declare that $p^*\%$ of the documents with the most uncertain predictions are Type O outliers. For the remaining $1 - p^*\%$ documents, we declare that they are Type M outliers if $\hat{y}(d) \neq y(d)$.

7.2.3. Implementation

A well-known caveat of neighbour-based classifiers is that they tend to be sensitive to the choice of parameter k. [Dud76] first proposed to weigh each neighbour by the distance to the queried point. There exist many weighting schemes [BL16; Gou+19]. While finding the best scheme is out of our scope, the consensus is that weighting improves empirical performance and leads to more flexible parameter choice (see [HS04]). So we propose the following score:

$$score_{d,c} = \sum_{d'}^{\mathcal{K}_c(d)} Sim(d, d') \sum_{p}^{\mathcal{J}(d')} Sim(d', p) \cdot r(p, c), \tag{7.12}$$

which uses both the inter-document and the document-phrase similarities for weighting. By definition, if $\mathcal{K}_c(d) = \emptyset$, then $score_{d,c} = 0$. We compute the entropy as follows:

$$I(d) = -\sum_{c}^{C} \overline{score}_{d,c} \cdot \log \overline{score}_{d,c}, \tag{7.13}$$

where $\overline{score}_{d,c}$ is the normalised score over the classes $c \in C$:

$$\overline{score}_{d,c} = \frac{score_{d,c}}{\sum_{c}^{C} score_{d,c}}. \tag{7.14}$$

By convention, $\overline{score}_{d,c} \cdot \log \overline{score}_{d,c} = 0$ if $\overline{score}_{d,c} = 0$. Finally, the outcome is a list of outliers for each type:

$$\begin{aligned}
&\mathbb{O} = \left\langle d_i,\ d_j,\ \ldots\ , d_{|\mathbb{O}|} \right\rangle \quad s.t. \quad I(d_i) > \Gamma \ \wedge\ I(d_i) \geq I(d_j), \\
&\mathbb{M} = \left\langle d_i,\ d_j,\ \ldots\ , d_{|\mathbb{M}|} \right\rangle \quad s.t. \quad I(d_i) \leq \Gamma \ \wedge\ \hat{y}(d_i) \neq y(d_i)\ \ldots \\
&\ldots\ \wedge\ I(d_i) \leq I(d_j), \quad \forall\ i < j,\ (i, j) \in Doc^2.
\end{aligned}$$

Here, Γ is set by parameter p^* (see Equation 7.11). The prediction is:

$$\hat{y}(d) = \arg\max_{c \in C} score_{d,c}. \tag{7.15}$$

Note that we sort $\mathbb{O}$ by decreasing uncertainty, while we sort $\mathbb{M}$ by increasing uncertainty. The rationale is that the more uncertain the decision for document d, the more likely d is a Type O outlier. On the other hand, the less uncertain a misclassification, the more likely d is a Type M outlier. So we output a ranking of outliers for both. In particular, if users only have a limited amount of time, they may only examine the most 'flagrant' outliers.

Algorithm 7.1 is our approach as pseudo-code. Since vectors are normalised, the cosine similarity Sim (cf. Equation 2.3) is proportional to the euclidean distance. Thus, we can use R*-trees to speed up the neighbourhood queries, and we cache the results for each document. From our algorithm, it is easy to see that the complexity of the approach is quasi-linear w.r.t. $|Doc|$, $|Phr|$, $|C|$, k and j, i.e., it can scale to very large corpora.

Algorithm 7.1 kj-NN(k, j, Doc, Phr, y, r, p^*)

Require: k, j, corpus Doc, phrases Phr, initial classification $y : Doc \mapsto C$, representativeness $r : Phr \times C \mapsto \mathbb{R}^+$, threshold $p^* \in [0, 1]$

$\mathbb{O} = \langle\rangle$; $\mathbb{M} = \langle\rangle$ ▷ Initialisation
$K \leftarrow$ index Doc with a R*-tree
$J \leftarrow$ index Phr with a R*-tree
for $d_i \in Doc$ **do** ▷ Cache neighbourhood queries
    $\mathcal{K}(d_i) \leftarrow$ the k nearest neighbours of d_i in K
    $\mathcal{J}(d_i) \leftarrow$ the j nearest neighbours of d_i in J
for $d_i \in Doc$ **do** ▷ Get scores and entropy for each document
    for $c \in C$ **do**
        $\mathcal{K}_c(d_i) \leftarrow \{d' \in \mathcal{K}(d_i) : y(d') = c\}$
        $score_{d_i,c} = \sum_{d'}^{\mathcal{K}_c(d_i)} Sim(d_i, d') \sum_{p}^{\mathcal{J}(d_i)} Sim(d', p) \cdot r(p, c)$
    $I(d_i) = \sum_{c}^{C} \overline{score}_{d_i,c} \cdot \log \overline{score}_{d_i,c}$
Choose $\Gamma \quad s.t. \quad |\{d_i \in Doc \mid I(d_i) < \Gamma\}|/|Doc| = p^*$
Sort Doc by increasing $I(d_i)$, $d_i \in Doc$
for $d_i \in Doc$ **do** ▷ Populate outlier lists
    if $I(d_i) > \Gamma$ **then** $\mathbb{O} \leftarrow \langle d_i \rangle \cup \mathbb{O}$
    else if $\hat{y}(d_i) \neq y(d_i)$ **then** $\mathbb{M} \leftarrow \mathbb{M} \cup \langle d_i \rangle$
return $\mathbb{O}$, $\mathbb{M}$

7.3. Experiment Setup

We evaluate the performance of our approach w.r.t. both types of outliers. We compare with the current state of the art, as well as with competitive baselines and ablations. We create real-world benchmark data sets from publicly available data.

7.3.1. Evaluation Measures

Outlier detection typically is an imbalanced classification problem. For Type O outliers, we report the area under the ROC curve (AUC) and the Average Precision (AP). These are popular measures for the evaluation of outlier detection algorithms. Since Type M outliers are more frequent, we report precision (P), recall (R) and the F1 score. In most applications, users are more concerned with the recall [Zhu+17]. So we measure the *recall at a certain percentage*, i.e., the share of detected outliers when the user checks the top X% items (RX) from the ranked list of outliers. Our measures are in line with Chapter 6.

7.3.2. Data Sets

We evaluate our approach against an assortment of benchmark data sets of various size, outlier ratio and number of inlier/outlier classes. Since emails and medical records typically contain highly confidential information, they are not adequate for the reproducibility of our study. Instead, we create the following sets of benchmarks from publicly available news articles and paper abstracts:

- **NYT:** We crawl 10,000 articles from 5 topics (*Business*, *Sports*, *Arts*, *Politics*, *Science*) with the New York Times API[2] and add 1% articles (i.e., 100 articles) from 4 other topics (*Real estate*, *Health*, *Education*, *Technology*), i.e., they are Type O outliers. We also increase the ratio of outliers to 2% and 5% and downsample the data by 50%, 20% and 10%. Thus, we create six benchmark data sets: **NYT-1**, **NYT-2**, **NYT-5**, **NYT-50**, **NYT-20**, **NYT-10**.
- **ARXIV:** We crawl 21,467 abstracts of the articles published on ArXiv [3] from 10 computer science categories (*cs.AI*, *cs.CC*, *cs.CL*, *cs.CR*, *cs.CY*, *cs.DB*, *cs.DS*, *cs.LG*, *cs.PL*, *cs.SE*). Then we choose 1 to 5 inlier classes at random and inject 1% outliers from 5 other classes. We repeat the procedure, but let the number of outlier classes vary. In the end, we create nine benchmark data sets: **ARXIV-15**, **ARXIV-25**, **ARXIV-35**, **ARXIV-45**, **ARXIV-55**, **ARXIV-54**, **ARXIV-53**, **ARXIV-52**, **ARXIV-51**.

Table 7.1 shows the features of each benchmark data set, in particular: the number of inliers (# Inliers), Type O outliers (# O), inlier classes (# Classes), Type O outlier classes (# O Classes), and the ratio of Type O outliers (O%). As mentioned, we release the data sets together with our source code.

Table 7.1.: Characteristics of the Benchmark Data Sets (kj-NN).

Benchmark	# Inliers	# O	# Classes	# O Classes	O%
NYT-1	10,000	100	5	4	1.00
NYT-2	10,000	200	5	4	2.00
NYT-5	10,000	500	5	4	5.00
NYT-50	5000	50	5	4	1.00
NYT-20	2000	20	5	4	1.00
NYT-10	1000	10	5	4	1.00
ARXIV-15	3267	33	1	5	1.00
ARXIV-25	4619	46	2	5	1.00
ARXIV-35	6299	63	3	5	1.00
ARXIV-45	10,136	101	4	5	1.00
ARXIV-55	11,115	111	5	5	1.00
ARXIV-54	11,115	111	5	4	1.00
ARXIV-53	11,115	111	5	3	1.00
ARXIV-52	11,115	111	5	2	1.00
ARXIV-51	11,115	111	5	1	1.00

7.3.3. Baseline and Competitors

Since none of the existing approaches detects both Type O and Type M outliers, we must compare with two different sets of competitors/baselines. To validate our design choices, we also compare against a set of ablations derived from our method (see Section 7.4.2.3).

[2] http://developer.nytimes.com
[3] https://arxiv.org/archive/cs

7.3.3.1. Type O

We compare kj-NN w.r.t. Type O outlier detection against the following competitors:

- **Local Outlier Factor [Bre+00] (LOF)** is a well-known density-based outlier detector. We report the best result with parameter $k \in [1, 100]$ in terms of AUC, as performance may vary widely w.r.t. k [Cam+16]. We use the implementation from ELKI [Sch+15].
- **Randomised Subspace Hashing [SA16] (RS-Hash)** is a subspace outlier detector. It estimates the outlierness of each data point via randomised hashing. We implement it as described by the authors, and we use the recommended parameters.
- **Average Negative Cosine Similarity (ANCS)** is a baseline measuring the outlierness of a document as the average negative cosine similarity to every other document. We also propose **k-ANCS**, a variant which only uses the k nearest neighbours for each document. We select $k \in [1, 100]$ maximising the AUC.
- **VMF-Q [Zhu+17]** models word embeddings as a vMF mixture and penalises lexically general words to identify semantically deviating documents. We use the implementation provided by the authors with the recommended parameters.
- **TONMF [Kan+17]** uses Non-negative Matrix Factorisation to detect documents with unusual word frequencies. We use the implementation released by the and let the parameters k, α and β vary from 1 to 30. We report the best result in terms of AUC.
- **Context Vector Data Description [Ruf+19] (CVDD)** is a one-class classification model using a pre-trained language model and multi-head attention. We use the authors' implementation with the recommended parameters.

7.3.3.2. Type M

To our knowledge, there is no approach explicitly handling the detection of misclassifications (Type M outliers). However, we can adapt virtually any supervised classifier to this task. We compare against the following state-of-the-art approaches for text classification:

- **Word-level CNN [Kim14] (W-CNN)** applies convolution kernels on stacked word embedding matrix of a document followed by pooling operations.
- **Very Deep CNN [Con+17] (VD-CNN)** uses up to 29 convolution layers that perform feature learning starting from characters, aiming to capture the hierarchical semantic structure encoded in characters, n-grams, words and sentences.
- **Attention-Based Hierarchical RNN [Yan+16] (AT-RNN)** employs attention mechanisms both at the word and sentence level. It learns to focus on the most relevant words and sentences for text classification.
- **Recurrent CNN [Lai+15] (RCNN)** combines both bi-directional recurrent structures and max-pooling layers to capture contextual information and extract relevant features.

For each approach, we train the classifier and predict the class of each instance. If the prediction differs from the actual class, we conclude that it is of Type M. The rationale is that such instances do not fit their class as well as other instances. Note that this is a rather common approach for outlier detection.

Since one typically does not have any ground truth in outlier detection tasks, performing a hyper-parameter search is not realistic and falls out of the scope of our study. We run each approach with default parameters.

7.3.4. Data Preparation

Since every Type O methods are unsupervised, they operate without labels. To evaluate the approaches, we set the labels of inliers to 0 and the labels of Type O outliers to 1. LOF, RS-Hash, ANCS and k-ANCS use as input the embedding representation that we mine in our preprocessing step (cf. Section 7.2). For VMF-Q, TONMF and CVDD, we implement the preprocessing steps recommended by their respective inventors.

For Type M methods, we control the proportion of Type M outliers by randomly assigning a proportion $m \in [0, 0.5]$ of labels to a wrong class, i.e., when $m = 0.5$, we misclassify half of the documents. The algorithms train with those erroneous labels, but we use the original labels for evaluation. We assign Type O outliers to an inlier class at random.

For each method, we first tokenise and segment the raw text data using AutoPhrase [Sha+18]. We learn the joint embeddings via JoSE [Men+19] with $n = 100$ dimensions. We process the input data similarly for every method, taking into account the recommendation of the respective authors. Our goal is to make the comparison as fair as possible.

7.4. Results

We perform our experiments on a machine running Ubuntu 18.04 with 20GB RAM and a quad-core CPU at 2.40GHz. We average each of our results from 10 independent runs.

7.4.1. Parameter Sensitivity

We first study the sensitivity of our approach to its parameters k, j and p^* against the benchmark NYT-1, w.r.t. varying Type M outlier ratio m in particular. We first simulate an extreme scenario by setting $m = 0.5$, i.e., we assign half of the articles to the wrong topic. We can see from Figure 7.2 that the Type O recall and precision tend to increase with k and j, but saturate for $j > 30$ and $k > 30$. We also see that p^* captures a trade-off between precision and recall: higher values of p^* leads to higher Type O precision, but lower recall. On the other hand, higher values of p^* lead to higher Type M recall, but lower precision. We see that the Type M precision and recall are high for $p^* \geq 0.9$ for large enough k and j. Given that the initial classification is very noisy ($m = 0.5$), the quality of the detection of both outlier types is impressive.

Next, we set $k = j = 30$ and let m vary from 0 to 0.5. In Figure 7.3, we see that the Type O recall and precision improve significantly for lower m, but the Type M precision decreases. While imperfect labels negatively impact the quality of Type O outlier detection, the Type M recall appears stable. Based on our observations, we recommend setting $k = j = 30$ and $p^* = 0.9$. For the remaining of our experiments, we set the parameters as such and assume $m = 0.2$.

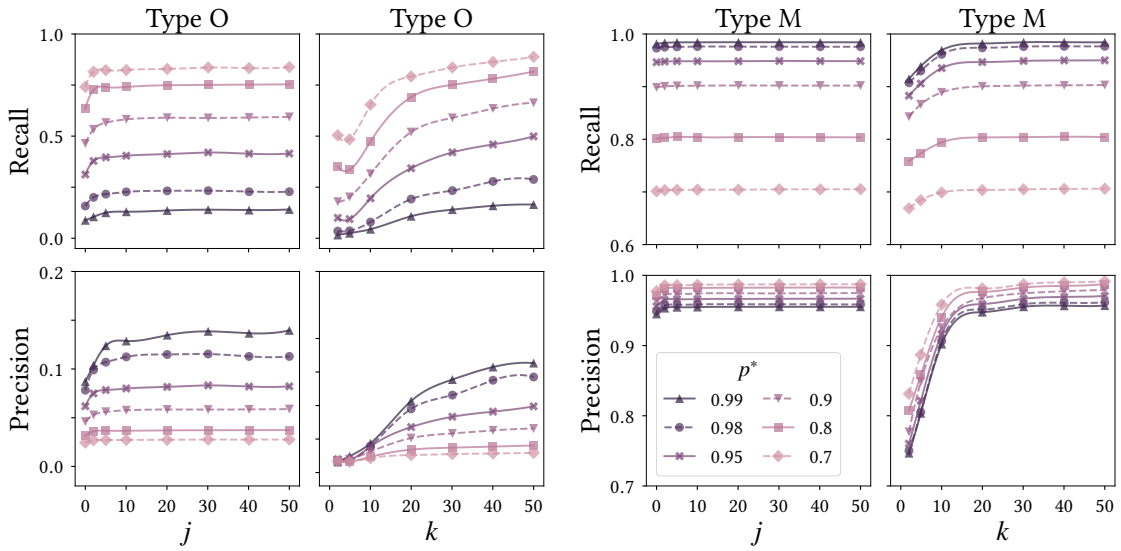


Figure 7.2.: Type O/M – k, j, p^* sensitivity, NYT-1.

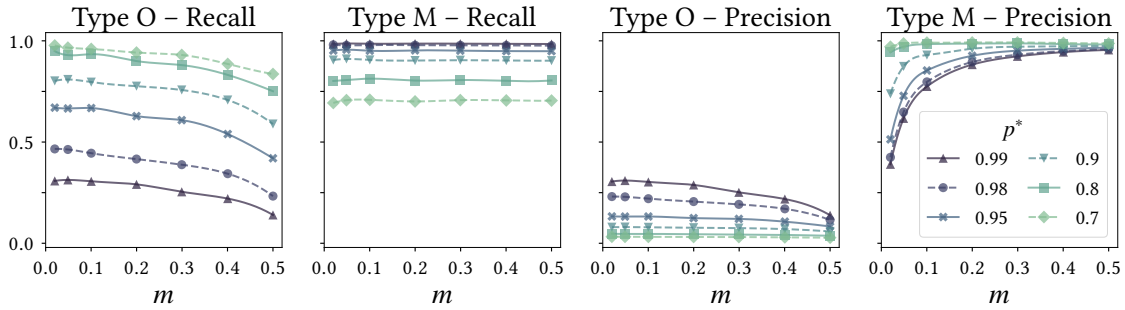


Figure 7.3.: Type O/M – p^*, m sensitivity, NYT-1.

7.4.2. Performance Comparison

7.4.2.1. Type O

Table 7.2 and Table 7.3 list the results of our comparison against the NYT and ARXIV benchmark. First, the results of every approach are generally better against the NYT benchmark. The ARXIV benchmark is much more challenging because the corpus is composed of abstracts from computer science sub-fields, with much semantic overlap.

Then, we see from both tables that kj-NN outperforms every competitor w.r.t. Type O outlier detection. The performance in terms of recall becomes lower for smaller data sets (e.g., NYT-20 and NYT-10). For small data sets, CVDD and our ANCS/k-ANCS baselines appear competitive.

Finally, we see that our approach handles particularly well multi-modal settings, i.e., with multiple inlier/outlier classes. By design, our approach cannot handle only one inlier class (e.g., ARXIV-15) — this does not fit our scenario. When there only is a single outlier class, LOF performs best (see ARXIV-51).

Table 7.2.: Comparison w.r.t. our Competitors (Type O, NYT).

Benchmark	Approach	AUC	AP	R1%	R2%	R5%
NYT-1	LOF	66.45	1.62	3.00	3.00	9.00
	RS-Hash	46.62	0.87	0.00	1.00	1.00
	ANCS	66.63	2.57	6.00	10.00	21.00
	k-ANCS	83.89	3.65	3.00	7.00	19.00
	TONMF	58.66	7.30	0.00	2.00	9.00
	VMF-Q	76.34	2.21	2.00	3.00	11.00
	CVDD	78.47	7.75	11.7	18.09	22.34
	kj-NN	**92.51**	**17.57**	**25.00**	**39.20**	**61.40**
NYT-2	LOF	60.66	2.45	2.00	2.50	6.00
	RS-Hash	48.05	1.87	0.50	1.00	5.50
	ANCS	67.59	5.19	8.00	12.00	18.00
	k-ANCS	82.51	5.94	3.00	5.50	14.50
	TONMF	54.61	1.78	2.50	3.00	9.00
	VMF-Q	83.67	6.87	4.00	11.00	20.00
	CVDD	73.10	10.00	11.58	14.74	22.11
	kj-NN	**94.51**	**42.64**	**30.40**	**44.50**	**64.50**
NYT-5	LOF	52.28	5.44	1.80	3.40	7.00
	RS-Hash	48.76	4.58	0.80	1.40	2.80
	ANCS	67.67	11.13	6.60	9.80	19.20
	k-ANCS	75.56	10.45	3.40	6.40	11.40
	TONMF	52.95	1.50	1.40	2.40	6.40
	VMF-Q	77.11	12.92	4.60	7.40	15.60
	CVDD	72.81	18.69	9.04	13.05	21.49
	kj-NN	**97.04**	**71.69**	**19.12**	**36.96**	**68.28**
NYT-50	LOF	60.91	1.77	2.00	6.00	12.00
	RS-Hash	46.14	0.90	0.00	0.00	2.00
	ANCS	63.94	4.35	14.00	16.00	20.00
	k-ANCS	81.08	4.43	12.00	14.00	20.00
	TONMF	61.42	**31.01**	0.90	0.90	4.90
	VMF-Q	85.76	4.00	6.00	8.00	22.00
	CVDD	76.29	15.89	21.62	27.03	29.73
	kj-NN	**93.33**	27.76	**37.20**	**46.80**	**62.80**
NYT-20	LOF	74.76	4.12	5.00	5.00	15.00
	RS-Hash	42.91	0.89	0.00	0.00	0.00
	ANCS	78.52	5.60	10.00	15.00	**40.00**
	k-ANCS	88.56	6.59	15.00	20.00	30.00
	TONMF	64.94	6.35	0.00	0.00	15.00
	VMF-Q	83.98	4.59	5.00	10.00	20.00
	CVDD	88.83	**24.00**	**25.00**	**25.00**	25.00
	kj-NN	**91.42**	6.57	3.00	11.00	38.00
NYT-10	LOF	77.93	2.77	0.00	0.00	20.00
	RS-Hash	56.94	1.76	0.00	0.00	10.00
	ANCS	83.89	13.65	30.00	40.00	40.00
	k-ANCS	91.37	10.93	30.00	30.00	30.00
	TONMF	71.13	29.92	0.00	0.00	0.00
	VMF-Q	63.39	2.55	0.00	10.00	20.00
	CVDD	85.13	**44.99**	**42.86**	**42.86**	**42.86**
	kj-NN	**91.52**	8.45	10.00	16.00	38.00

Table 7.3.: Comparison w.r.t. our Competitors (Type O, ARXIV).

Benchmark	Approach	AUC	AP	R1%	R2%	R5%
ARXIV-15	LOF	63.44	1.78	0.00	5.13	7.69
	RS-Hash	54.22	1.17	2.56	2.56	2.56
	ANCS	47.46	0.96	0.00	0.00	5.13
	k-ANCS	67.14	1.98	0.00	7.69	10.26
	TONMF	57.65	**7.32**	0.00	7.69	15.38
	VMF-Q	**71.71**	2.69	**5.13**	**10.26**	15.38
	CVDD	69.85	2.27	0.00	5.13	**17.95**
	kj-NN	54.21	1.17	0.00	0.00	0.00
ARXIV-25	LOF	68.41	2.17	4.35	6.52	13.04
	RS-Hash	44.21	0.91	0.00	0.00	4.35
	ANCS	47.88	1.09	2.17	4.35	6.52
	k-ANCS	67.87	2.00	2.17	4.35	10.87
	TONMF	60.03	**6.37**	0.00	4.35	13.04
	VMF-Q	71.71	2.69	5.13	10.26	15.38
	CVDD	**74.55**	2.19	2.17	4.35	8.70
	kj-NN	70.64	5.10	**10.44**	**16.52**	**27.82**
ARXIV-35	LOF	71.00	2.16	0.00	3.23	11.29
	RS-Hash	47.77	0.96	1.61	1.61	4.84
	ANCS	52.94	1.07	0.00	0.00	3.23
	k-ANCS	73.43	2.36	1.61	4.84	16.13
	TONMF	56.90	1.16	1.61	3.22	6.45
	VMF-Q	60.97	1.28	0.00	0.00	1.61
	CVDD	53.89	1.06	0.00	0.00	1.61
	kj-NN	**73.58**	**3.27**	**6.45**	**10.97**	**20.97**
ARXIV-45	LOF	62.99	2.43	2.02	2.02	5.05
	RS-Hash	49.18	0.99	0.00	0.00	6.06
	ANCS	51.50	1.32	1.01	1.01	2.02
	k-ANCS	70.11	2.23	1.01	3.03	17.17
	TONMF	57.49	**17.23**	0.00	0.00	2.02
	VMF-Q	66.59	2.20	3.03	7.07	15.15
	CVDD	**71.32**	2.10	2.02	4.04	11.11
	kj-NN	69.94	3.28	**6.46**	**9.09**	**18.99**
ARXIV-55	LOF	59.47	1.47	3.23	3.23	11.29
	RS-Hash	49.00	0.94	0.81	0.81	3.23
	ANCS	51.69	1.80	1.61	3.23	6.45
	k-ANCS	67.59	2.25	3.23	6.45	14.52
	TONMF	56.86	0.92	1.61	3.23	9.68
	VMF-Q	72.99	2.65	4.84	8.06	16.94
	CVDD	60.92	1.63	1.61	4.84	11.29
	kj-NN	**76.74**	**3.22**	**5.49**	**9.84**	**20.48**
ARXIV-54	LOF	61.92	1.33	0.00	2.42	6.45
	RS-Hash	47.95	0.92	0.00	0.81	2.42
	ANCS	50.46	1.01	0.81	0.81	3.23
	k-ANCS	68.93	1.83	0.00	1.61	8.87
	TONMF	59.46	**5.94**	0.00	0.00	6.45
	VMF-Q	61.81	1.33	0.81	1.61	6.45
	CVDD	60.30	1.62	1.61	5.65	10.48
	kj-NN	**76.65**	3.85	**5.81**	**10.00**	**21.61**
ARXIV-53	LOF	57.37	1.33	2.42	3.23	9.68
	RS-Hash	42.32	0.79	0.00	0.00	0.00
	ANCS	47.55	0.93	0.81	1.61	3.23
	k-ANCS	63.56	1.54	2.42	4.03	9.68
	TONMF	56.12	4.26	0.00	1.61	8.06
	VMF-Q	77.29	3.16	3.23	8.06	25.00
	CVDD	69.28	1.98	2.42	4.03	12.90
	kj-NN	**79.16**	**5.10**	**7.74**	**15.65**	**32.42**
ARXIV-52	LOF	55.58	1.09	0.81	0.81	4.84
	RS-Hash	52.52	1.08	0.00	1.61	5.65
	ANCS	45.02	0.84	0.00	0.00	0.81
	k-ANCS	66.26	1.61	0.81	3.23	8.87
	TONMF	54.04	2.70	1.61	3.23	8.06
	VMF-Q	67.57	2.06	2.42	7.26	13.71
	CVDD	55.64	1.39	1.61	4.84	11.29
	kj-NN	**78.10**	**3.07**	**3.87**	**7.90**	**19.35**
ARXIV-51	LOF	**82.57**	**4.50**	**7.26**	**14.52**	**25.81**
	RS-Hash	51.22	1.10	1.61	4.03	6.45
	ANCS	65.91	2.36	5.65	10.48	15.32
	k-ANCS	54.19	1.68	4.03	7.26	11.29
	TONMF	55.23	1.01	1.61	1.61	7.26
	VMF-Q	62.54	1.54	1.61	4.84	8.87
	CVDD	66.93	2.54	4.84	8.87	23.39
	kj-NN	65.24	2.19	4.68	7.90	14.36

Table 7.4.: Comparison w.r.t. our Competitors (Type M, NYT).

Benchmark	Approach	P	R	F1	R10%	R20%
NYT-1	W-CNN	54.38	86.04	66.64	27.28	53.51
	VD-CNN	90.71	69.22	78.52	15.04	30.18
	AT-RNN	67.12	51.88	58.52	32.92	51.34
	RCNN	**96.02**	9.65	17.54	9.55	9.55
	kj-NN	95.93	**90.02**	**92.88**	**50.19**	**90.02**
NYT-2	W-CNN	54.16	88.02	67.06	27.30	54.56
	VD-CNN	88.99	69.69	78.17	14.71	29.61
	AT-RNN	80.36	49.03	60.90	40.29	48.14
	RCNN	89.60	5.59	10.52	5.49	5.49
	kj-NN	**94.63**	**91.15**	**92.86**	**50.74**	**91.15**
NYT-5	W-CNN	51.12	89.07	64.96	25.14	50.38
	VD-CNN	58.21	82.39	68.22	8.98	18.46
	AT-RNN	61.71	70.49	65.81	31.38	61.62
	RCNN	90.82	44.93	60.12	42.86	42.86
	kj-NN	**92.43**	**93.44**	**92.93**	**52.27**	**93.44**
NYT-50	AT-RNN	47.38	87.61	61.50	22.28	47.62
	RCNN	**98.58**	55.44	70.97	49.31	54.95
	VD-CNN	89.75	69.22	78.16	14.83	29.92
	W-CNN	57.46	87.31	69.31	27.23	56.93
	kj-NN	95.78	**90.36**	**92.99**	**50.22**	**90.36**
NYT-20	W-CNN	39.34	**91.56**	55.03	20.54	40.59
	VD-CNN	93.50	81.80	87.26	15.20	30.81
	AT-RNN	50.44	85.11	63.34	25.25	52.23
	RCNN	39.63	**91.56**	55.32	20.54	40.59
	kj-NN	**93.80**	90.64	**92.19**	**50.52**	**90.64**
NYT-10	W-CNN	36.12	88.94	51.38	16.83	35.15
	VD-CNN	86.20	69.40	76.89	14.10	27.26
	AT-RNN	36.12	88.94	51.38	16.83	35.15
	RCNN	36.12	88.94	51.38	16.83	35.15
	kj-NN	**88.57**	**90.77**	**89.65**	**50.50**	**89.97**

7.4.2.2. Type M

Our approach also outperforms its competitors w.r.t. Type M outliers. See Tables 7.4 and 7.5. RCNN and VD-CNN have high precision, but much lower recall. Neural networks tend to fit the data very well, including Type M outliers. The performance decreases dramatically with smaller data sets. VD-CNN occasionally ranks high in terms of F1-score. However, those approaches do not rank outliers, so the recall at a certain percentage (e.g., R10, R20) is equivalent to that from a list of detected outliers in random order.

7.4.2.3. Ablation Analysis

We verify each of our design choices by comparing against the following ablations:

- **A1: No self-supervision**; we set $j = 0$, i.e., our approach boils down to a k-NN classifier as in Equation 7.5.
- **A2: No entropy**; we do not estimate the entropy of the prediction, i.e., objects can be in both outlier lists $\mathbb{O}$ and $\mathbb{M}$.
- **A3: No neighbourhood**; the predictions only base on the relevance of document d, and not on the relevance of its neighbours, i.e., we set $\mathcal{K}(d) = \{d\}$.
- **A4: Unweighted kj-NN**; we do not weigh the score by the inter-document and document-phrase similarities, as explained in Section 7.2.3. Each neighbouring document and phrase have the same impact on the decision.

Table 7.5.: Comparison w.r.t. our Competitors (Type M, ARXIV).

Benchmark	Approach	P	R	F1	R10%	R20%
ARXIV-25	W-CNN	95.33	82.51	88.46	47.77	82.15
	VD-CNN	72.97	59.71	65.68	15.52	31.60
	AT-RNN	65.90	72.24	68.92	32.97	65.94
	RCNN	96.75	26.01	41.00	25.90	25.90
	kj-NN	**96.09**	**88.85**	**92.33**	**49.22**	**88.85**
ARXIV-35	W-CNN	54.97	84.40	66.58	27.45	54.49
	VD-CNN	66.06	80.08	72.40	11.73	23.01
	AT-RNN	**84.37**	38.59	52.96	38.12	38.12
	RCNN	80.21	25.02	38.14	24.72	24.72
	kj-NN	80.04	**86.98**	**83.36**	**48.25**	**83.40**
ARXIV-45	W-CNN	51.47	81.26	63.02	24.80	49.40
	VD-CNN	73.68	72.24	72.95	10.48	21.00
	AT-RNN	**74.20**	49.97	59.72	37.70	49.60
	RCNN	65.49	26.10	37.32	25.90	25.90
	kj-NN	70.29	**86.87**	**77.70**	**45.78**	**76.89**
ARXIV-55	W-CNN	53.62	87.71	66.55	27.79	53.66
	VD-CNN	**92.65**	**91.97**	**92.31**	12.01	24.08
	AT-RNN	86.50	54.56	66.91	43.07	54.06
	RCNN	83.80	33.87	48.24	33.56	33.56
	kj-NN	70.70	88.23	78.50	**46.77**	**78.67**
ARXIV-54	W-CNN	58.07	87.94	69.95	28.03	57.40
	VD-CNN	**87.22**	85.12	**86.16**	11.38	22.71
	AT-RNN	69.81	56.33	62.35	34.79	55.77
	RCNN	58.10	41.09	48.14	28.98	40.68
	kj-NN	70.38	**88.05**	78.23	**46.83**	**77.78**
ARXIV-53	W-CNN	47.12	85.91	60.86	24.56	47.97
	VD-CNN	**82.78**	82.64	**82.71**	10.95	21.62
	AT-RNN	58.65	60.31	59.47	29.22	58.48
	RCNN	63.25	22.58	33.28	22.33	22.33
	kj-NN	70.77	**88.46**	78.63	**46.69**	**78.24**
ARXIV-52	W-CNN	49.10	84.35	62.07	24.84	49.12
	VD-CNN	**86.92**	**91.59**	**89.19**	11.49	22.77
	AT-RNN	78.94	61.65	69.23	39.41	60.87
	RCNN	45.24	27.02	33.83	22.53	26.67
	kj-NN	71.08	88.43	78.81	**47.28**	**78.79**
ARXIV-51	W-CNN	56.00	**86.47**	67.98	28.14	56.49
	VD-CNN	78.89	82.56	**80.68**	10.24	20.42
	AT-RNN	76.27	59.04	66.56	38.57	58.36
	RCNN	**79.67**	25.09	38.16	24.80	24.80
	kj-NN	70.91	88.63	78.79	**46.25**	**78.18**

From Table 7.6 and 7.7, we can see that kj-NN consistently outperforms every ablation on average. A2 leads to higher Type M recall, and often high Type O AUC, but much lower Type M precision. A1 and A3 yield worse results for both outlier types. A4 values are consistently below, but only by a few hundredth. The overall average ranks are as follows:

- **kj-NN:** 1.67, **A2:** 2.50, **A4:** 2.69, **A1:** 3.30, **A3:** 4.29

Thus, we can see that taking the neighbourhood into account improves the performance of our approach the most, followed by self-supervision. Measuring decision uncertainty and weighting by similarity, as in Equation 7.12, improves the performance further.

Table 7.6.: Ablation Analysis (NYT).

		Type A		Type B			
Benchmark	Ablation	AUC	AP	P	R	F1	Rank
NYT-1	A1	89.57	16.65	94.56	90.04	92.25	3.6
	A2	**92.51**	**17.57**	86.43	**99.07**	92.32	2.2
	A3	90.56	8.45	91.85	89.28	90.54	4.6
	A4	92.48	17.07	95.88	90.00	92.85	2.8
	kj-NN	**92.51**	**17.57**	**95.93**	90.02	**92.88**	**1.4**
NYT-2	A1	93.29	**43.23**	93.61	91.23	92.40	2.6
	A2	**94.51**	42.64	83.15	**99.32**	90.52	2.6
	A3	91.23	18.62	90.22	89.89	90.05	4.8
	A4	94.50	42.12	94.55	91.07	92.78	2.8
	kj-NN	**94.51**	42.64	**94.63**	91.15	**92.86**	**1.6**
NYT-5	A1	96.61	**72.33**	92.12	93.55	92.83	2.6
	A2	**97.04**	71.69	75.92	**99.18**	86.01	2.8
	A3	92.33	41.43	86.12	91.97	88.95	4.6
	A4	**97.04**	71.63	92.23	93.45	92.84	2.4
	kj-NN	**97.04**	71.69	**92.43**	93.44	**92.93**	**1.8**
NYT-50	A1	91.20	27.29	94.20	90.36	92.24	3.2
	A2	**93.33**	**27.76**	85.40	**99.34**	91.84	2.4
	A3	90.78	13.34	92.68	89.11	90.86	4.8
	A4	93.30	27.44	95.64	90.34	92.91	2.8
	kj-NN	**93.33**	**27.76**	**95.78**	90.36	**92.99**	**1.2**
NYT-20	A1	89.01	5.94	93.10	90.29	91.67	4.0
	A2	**91.42**	6.57	83.56	**98.90**	90.58	2.6
	A3	89.93	**9.45**	91.19	87.44	89.27	3.8
	A4	**91.42**	6.51	93.60	90.49	92.02	2.2
	kj-NN	**91.42**	6.57	**93.80**	90.64	**92.19**	**1.4**
NYT-10	A1	91.25	8.05	88.18	90.47	89.31	3.6
	A2	**91.52**	8.45	80.98	**98.50**	88.88	2.6
	A3	89.65	14.06	87.30	85.17	86.20	4.6
	A4	91.23	**8.49**	88.19	90.57	89.36	2.4
	kj-NN	**91.52**	8.45	**88.57**	90.77	**89.65**	**1.4**

7.4.2.4. Execution Time

Figure 7.4 graphs the execution time for each approach. We neglect the common preprocessing time for each of them and report the sum of training and testing time. RS-Hash is extremely fast, but as we saw the performance is not better than guessing. LOF and k-ANCS, which leverage index support, are relatively fast. Our method, kj-NN, is only slightly slower than LOF but seems to scale better (observe the difference in execution time between NYT-1 and NYT-10). The other methods appear slower.

Table 7.7.: Ablation Analysis (ARXIV).

		Type A		Type B			
Benchmark	Ablation	AUC	AP	P	R	F1	Rank
ARXIV-25	A1	67.75	3.70	95.01	88.93	91.87	4.4
	A2	70.64	5.10	91.22	**98.17**	**94.56**	2.2
	A3	**81.23**	**5.93**	**97.30**	88.16	92.50	**2.0**
	A4	70.48	5.09	96.10	88.98	92.40	3.0
	kj-NN	70.64	5.10	96.09	88.85	92.33	3.0
ARXIV-35	A1	71.98	3.02	80.03	87.14	**83.43**	2.8
	A2	73.58	**3.27**	72.75	**95.60**	82.62	2.4
	A3	**74.14**	2.72	71.60	84.32	77.44	4.2
	A4	73.54	**3.27**	79.92	86.99	83.30	2.8
	kj-NN	73.58	**3.27**	**80.04**	86.98	83.36	**2.0**
ARXIV-45	A1	69.42	2.96	70.02	86.64	77.45	4.0
	A2	69.94	**3.28**	63.93	**95.41**	76.56	2.6
	A3	**71.56**	3.17	65.43	84.83	73.87	3.8
	A4	69.88	3.27	70.09	86.88	77.59	2.6
	kj-NN	69.94	**3.28**	**70.29**	86.87	**77.70**	**1.6**
ARXIV-55	A1	75.84	3.07	70.33	88.24	78.27	3.2
	A2	**76.74**	**3.22**	63.78	**96.82**	76.90	2.4
	A3	72.31	2.24	66.06	87.02	75.10	4.8
	A4	76.60	3.18	70.49	88.19	78.35	2.8
	kj-NN	**76.74**	**3.22**	**70.70**	88.23	**78.50**	**1.4**
ARXIV-54	A1	75.54	3.63	70.19	87.95	78.07	3.6
	A2	**76.65**	**3.85**	63.38	**96.94**	76.65	2.4
	A3	76.33	3.40	66.09	86.68	75.00	4.6
	A4	76.54	3.82	70.19	88.01	78.10	2.6
	kj-NN	**76.65**	**3.85**	**70.38**	88.05	**78.23**	**1.2**
ARXIV-53	A1	78.55	4.91	70.58	88.35	78.47	3.6
	A2	**79.16**	**5.10**	63.54	**96.75**	76.71	2.4
	A3	76.48	3.22	65.89	86.98	74.98	4.8
	A4	79.04	5.08	70.65	88.51	**78.58**	2.4
	kj-NN	**79.16**	**5.10**	**70.77**	88.46	78.63	**1.4**
ARXIV-52	A1	77.44	**3.39**	70.86	88.53	78.71	2.4
	A2	**78.10**	3.07	63.76	**96.73**	76.85	2.6
	A3	69.53	1.56	65.85	86.51	74.78	4.4
	A4	77.93	3.03	70.90	88.45	78.71	3.0
	kj-NN	**78.10**	3.07	**71.08**	88.43	**78.81**	**2.0**
ARXIV-51	A1	**68.41**	**3.03**	70.76	88.33	78.58	2.6
	A2	65.24	2.19	63.81	**96.89**	76.94	2.8
	A3	57.35	1.12	65.83	86.86	74.90	4.2
	A4	64.76	2.11	70.80	88.68	78.74	3.0
	kj-NN	65.24	2.19	**70.91**	88.63	**78.79**	**2.0**

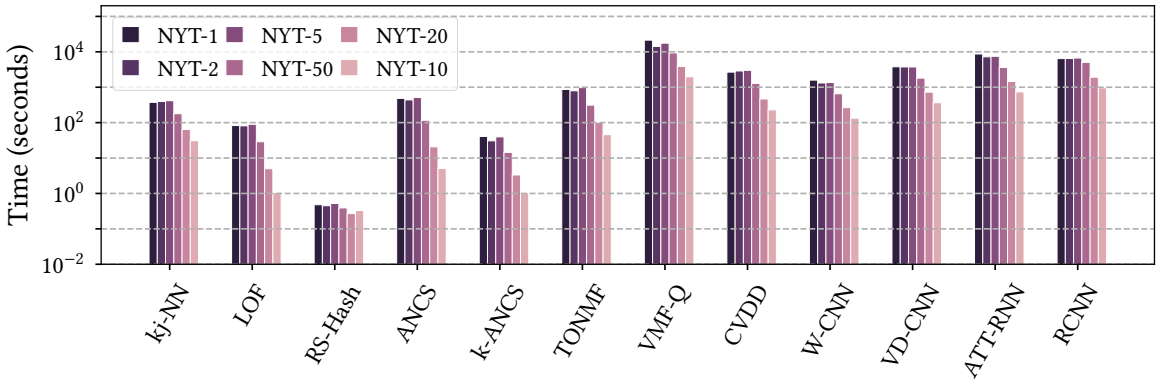


Figure 7.4.: Execution time of each approach.

7.4.3. Interpretation

The design of our method gives way to interpretable results. To find similar documents, we can simply perform a nearest-neighbour search in the embedding space. We find in turn the most representative phrases for a given outlier d_{out} via a similar approach as in preprocessing (see Section 7.2.1), with this time only two classes: an outlier class c_{out} containing the k-neighbourhood of document d, i.e., $\mathcal{K}(d) \cup d$ and a class c_{in} containing the rest of the documents in the corpus.

Figure 7.5 shows the two top-ranked Type O and Type M outliers we found in NYT-1, along with the most representative phrases and excerpts from the most similar documents.

It is interesting to see that the three nearest neighbours of the Type O outlier either relate to building construction projects or political decisions in education (or both). Actually, the ground truth label '*Education*' even appears at position five within the most representative phrases — a useful information in practice.

With the Type M outlier, the nearest documents all relate to science and business in some way, and the top phrases (energy, fuel, ...) strongly relate to their specific topic.

We then run our approach on the ARXIV benchmark with all the ten classes. Figure 7.6 shows the top-ranked outliers. The Type O outliers are indeed abnormal: they are not paper abstracts, so one should remove them from the corpus. The Type M outlier is from [Bor+11]. The authors chose to publish this article in the category *cs.AI* (Artificial Intelligence), but our approach suggests that it should be under *cs.CL* (Computation and Language). Intuitively, this makes sense, given the general topic of the abstract and the most representative phrases.

7.5. Discussion

We have studied the detection of text outliers in document directories. This task is challenging because text outliers are manifold, domain-specific, and the task is unsupervised.

We observe that such outliers fall into two types: out-of-distribution (Type O) and misclassified (Type M) documents. We are first to propose an approach that (i) detects text outliers from multiple folders and (ii) effectively distinguishes between both outlier types. Our algorithm, kj-NN, leverages self-supervision signals mined from an initial but imperfect classification. Interestingly, our experiments show that detecting both outlier types simultaneously leads to better performance than with existing approaches, which only deal with one type. Our method also yields interpretable results, by finding adequate alternative names for folders and by describing the particular semantics of text outliers.

This chapter only addressed text outlier detection in the static setting, while text can also be seen as a stream of information. An interesting future work would be to combine it with our other methods, such as MCDE (Chapter 4) and S-MAB (Chapter 5), similarly as in Chapter 6. Doing so would facilitate Data Mining in massive streams of textual information, such as news feeds or twitter data. Text outlier detection in such a setting is an interesting application of Knowledge Discovery in high-dimensional data streams.

Figure 7.5.: Interpretation of Type O/M outliers (NYT-1 benchmark).

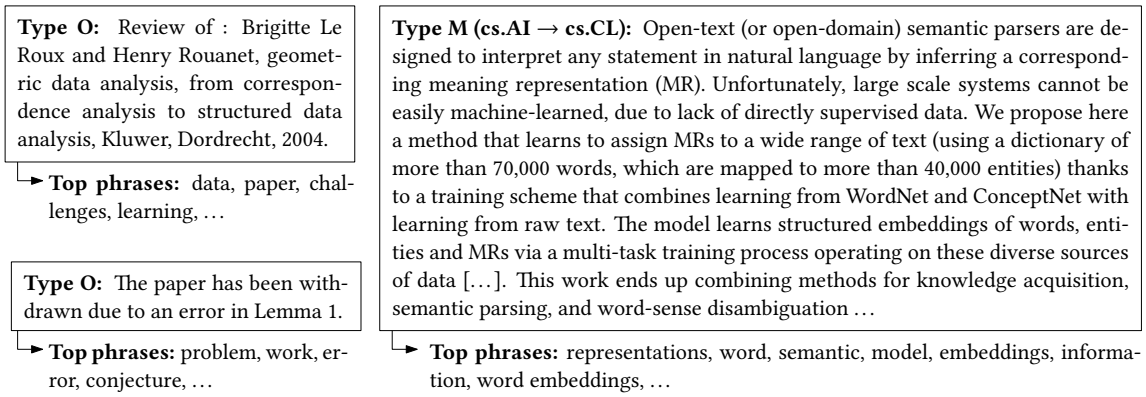


Figure 7.6.: Interpretation of Type O/M outliers (ARXIV benchmark).

Part V.

Conclusions

8. Outcome

This dissertation deals with fundamental topics of Data Mining under the constraints of high-dimensional data streams. We addressed them from the most specific to the most general: Estimating Dependency (**Q1**), Monitoring (**Q2**) and Knowledge Discovery (**Q3**).

Those topics are intertwined: virtually any Knowledge Discovery task benefits from estimating dependency (c.f. Section 1.1.4). However, in the high-dimensional streaming setting, one also needs solutions for the efficient monitoring of such estimates. Data Mining in such context is difficult because one must simultaneously address two orthogonal challenges: high-dimensionality and data streams (c.f. Sections 1.1.2 and 1.1.3).

In Part II, we addressed the challenges of estimating dependency (**Q1**) in high-dimensional data streams. After identifying the constraints and requirements, we introduced Monte Carlo Dependency Estimation (MCDE), a framework to estimate multivariate dependency in heterogeneous data streams. In a nutshell, MCDE quantifies dependency as the statistical discrepancy between marginal and conditional distributions over multiple Monte Carlo simulations. Based on different statistical test, we derived three new estimators: Kolmogorov-Smirnov-P (KSP), Mann-Withney-P (MWP) and Chi-Squared-P (CSP), and showed that they fulfil all the requirements described earlier. Compared to other approaches, MCDE provides high statistical power on a large panel of dependencies, while being very efficient. Furthermore, we introduced index operations for the streaming setting and illustrated the benefits of our framework against a real-world use case: the Bioliq power plant. Due to its anytime nature and efficient index structure, MCDE gives way to efficient monitoring of dependency in data streams.

Then, we generalised the task of monitoring (**Q2**) statistics in Part III. We proposed a novel bandit model, named the Scaling Multi-Armed Bandit (S-MAB), which captures the efficiency trade-off that is central to many real-world applications. We presented a new algorithm, Scaling Thompson Sampling (S-TS), which combines Multiple-Play Thompson Sampling [KHN15] (MP-TS) with a new procedure to decide on the number of arms played per round, a so-called 'scaling policy'. Our analysis and experiments showed that it enjoys strong theoretical guarantees and excellent empirical behaviour. We also proposed an extension of our algorithm for the non-static setting, by combining it with Adaptive Windowing [BG07] (ADWIN), a state-of-the-art change detector. We illustrated with the example of Bioliq that one can use our algorithms to monitor multiple statistics online.

In Part IV, we addressed Knowledge Discovery (**Q3**) in high-dimensional data streams. While the knowledge of dependencies in streams already is valuable as such, we illustrated the impact of efficient dependency monitoring systems on downstream Data Mining tasks. First, we achieved Subspace Search in Data Streams (Chapter 6) by proposing a new algorithm, Streaming Greedy Maximum Random Deviation (SGMRD), that leverages MCDE and S-MAB algorithms. We showed that SGMRD leads to state-of-the-art performance against a typical Data Mining task: the detection of outliers. Then, we looked at a specific

application: detecting text outlier from large corpora (Chapter 7). This task is particularly challenging because text outliers are manifold, domain-specific, and the task is unsupervised in nature. We are the first to make the distinctions between two types of outliers: out-of-distribution (Type O) and misclassified (Type M) documents. We proposed an approach which simultaneously detects both types of outliers. Our algorithm, kj-Nearest Neighbours (kj-NN), leverages self-supervision signals from an initial but imperfect classification. Our experiments show that our approach outperforms each competitor and baseline w.r.t. both outlier types while delivering interpretable results. Using our results may increase the quality of massive text archives by much and assist the annotators.

All in all, this dissertation presents fundamental contributions to the field of Data Mining, focusing on the particularly challenging setting of high-dimensional data streams. Our contributions have been well-received in peer-review conferences: MCDE [FB19] was presented at the 31st International Conference on Scientific and Statistical Database Management (SSDBM'19) and received the 'Best Paper Award'. We published an extended study of the MCDE framework [Fou+20a] in the 'Distributed and Parallel Databases' (DAPD) journal (Springer). We presented the S-MAB [FKB19] algorithms at the 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'19), and kj-NN [Fou+20b] at the 20th IEEE International Conference on Data Mining (ICDM'20).

Our contributions raised several interesting questions, in particular about their impact on downstream analysis tasks. While we did make a few strides towards answering those questions with SGMRD [FKB20] and kj-NN [Fou+20b], we elaborate on the open problems and future challenges of this dissertation hereafter.

9. Future Work

Our contributions raise several interesting questions. While we partially addressed some of them, the required research efforts extend beyond the scope of this dissertation. We identify the following open research directions:

Multi-Scale Dependency Monitoring: While we showed that Monte Carlo Dependency Estimation (MCDE) (Chapter 4) leads to efficient monitoring of dependency in streams, e.g., via exponential weighting, MCDE could benefit from a more flexible update mechanism, using for instance a sliding window of adaptive size [BG07]. Breaking free from the fixed-size sliding window model may help to generalise dependency estimation to time contexts of various scales. While doing so, one may need to investigate the integration of newer statistical test into the MCDE framework, such as [FP81] or [BM00].

Regret Analysis of the Non-static Scaling Multi-Armed Bandit: In Part III, we solved the Scaling Multi-Armed Bandit (S-MAB) problem in the static setting with Scaling Thompson Sampling (S-TS). For the non-static environment, we proposed an improvement based on Adaptive Windowing [BG07] (ADWIN). Our algorithm performs better than the existing approaches empirically. However, the static regret analysis already is quite involved, and extending it to non-static environments is a challenge. We hypothesise that this success is due to a class of non-stationarity that our solution exploits – but which we have not formalised yet. A first step is to leverage the theoretical guarantees of ADWIN to derive general guarantees for Scaling Bandits with ADWIN. Still, the technical difficulty arises in guaranteeing that change points are accurately detected. For the analysis, non-trivial modifications of the ADWIN algorithm might be necessary.

Anytime Trade-off Strategies: We proved in Chapter 4 that the estimates from MCDE converge as one increases the number of simulations. This is a crucial characteristic of MCDE that we call *anytime flexibility*. Based on a computational budget, users can bound the estimate quality. Conversely, one can interrupt the computation whenever the estimate reaches a desired quality level. The same year, [VB19] published an estimator for Mutual Information (MI) with the same characteristic. With the popularisation of edge computing, reducing the cost of basic tasks – e.g., dependency estimation – is very appealing, because computation typically is distributed among systems with limited resources. However, in Knowledge Discovery, one is often interested in finding sets of attributes with high dependency. In other words, the value of each estimate, which is unknown a priori, may not all be equally interesting to the end-user. The question is then how to distribute in an anytime fashion a global computational budget among concurrent estimates? While our contribution [FB19] and [VB19] addressed this question for individual dependency estimates, generalising the optimisation of anytime algorithms w.r.t. multiple problems remains open. Now, it would be interesting to refine bounds, like the one presented in Theorem 4.2, via further assumptions, and to leverage user-specific criterion to guide the allocation of resources between concurrent problems.

High-Dimensional Stream Mining with User Constraints: In the literature, integrating user constraints into Data Mining algorithms often is seen as a complication [HLN99]. With the S-MAB model (Chapter 5), we have seen that user information (such as a threshold on the interestingness of estimates) may also help to reduce computational costs drastically. Integrating such information into algorithms might be an essential aspect to tackle Data Mining in high-dimensional data streams. This idea is in line with our future work on Anytime Trade-off Strategies (see above). The question is how to formally integrate user constraints into mining algorithms, in particular, considering the type of information one may receive from users with different levels of expertise?

Mining Causality: With MCDE (Chapter 4) and S-MAB (Chapter 5), one can estimate the dependence between attributes in high-dimensional data sets and discover interesting, previously unknown associations. The next step is to ask, for each of those associations, whether they emerge from a causal relationship, or turn out to be spurious. Recent advances in a related field, Causal Inference [Pea09; Nan16; Lop16; Pea19], provide elements to answer this question solely based on observational data. While passive causal inference methods are somewhat limited, because they require relatively strong assumptions, it would be interesting to extend the MCDE framework in this direction. Our intuition is that slight modifications of our subspace slicing scheme may allow testing the existence of confounding variables for a given association. One could then build a so-called 'dependency network' [Hec+00; Sch+16; DM17], i.e., a graphical representation of dependency in the stream, providing new insights to end-users as well. By extending our methods, we could mine such networks in real-time in high-dimensional data streams.

Analysis of Multivariate Times Series Embeddings: In Chapter 7, we showed that exploiting local neighbourhoods in word/document embeddings helps to detect outlying text. Integrating the methods from Chapter 6 may bring an incremental improvement in the static setting already, and solve similar tasks in massive streams of text. Perhaps even more interesting is to transfer our approach to other domains, e.g., multivariate times series. For example, we could project sequences of measurements from the Bioliq power plant into an embedding space. Then, using kj-Nearest Neighbours (kj-NN), we may automatically detect normal and abnormal states in the plant, and, with Streaming Greedy Maximum Random Deviation (SGMRD), adapt to changes in the streaming setting. In predictive maintenance, there typically are only very few labels, such setting is known as 'weakly-supervised' [Zho17; Men+18]. It would be interesting to investigate extensions of our approach to handle weak supervision or further complications.

To conclude, this dissertation advances the state of the art of the Data Mining field. We provided fundamental contributions with significant impacts on the general task of Knowledge Discovery in high-dimensional data streams. Last but not least, our work paves the way for multiple exciting future research topics.

Appendix

Acronyms

ADWIN Adaptive Windowing [BG07]. 9, 22, 65, 68, 69, 79, 82, 84, 103, 127, 129, 143

ANCS Average Negative Cosine Similarity. 114–118

AP Average Precision. 96, 112, 117, 118, 121, 122

AT-RNN Attention-Based Hierarchical RNN [Yan+16]. 114, 119, 120

AUC Area Under the Curve (e.g., of the ROC curve). 96, 99, 112, 114, 117, 118, 121, 122

Bioliq Biomass-to-Liquid. vii, 4, 8, 9, 11, 14, 29, 59, 81, 90, 97, 127, 130, 139, 140, 145

CE Cumulative Entropy [CL09]. 21

CMAB Combinatorial MAB. 23

CMI Cumulative Mutual Information [Ngu+13]. 21, 51, 54, 57, 61

CSP Chi-Squared-P. viii, 29, 36, 41–49, 58, 127, 139, 143

CUCB Combinatorial UCB [Che+16]. 23, 70, 78, 79, 84

CVDD Context Vector Data Description [Ruf+19]. 24, 114–118

DO Dynamic Oracle. 82

dTS Discounted Thompson Sampling [RK17]. 78, 80

D-SQF Dimension-Subspace Quality Function. 88, 89, 138, 145

EDM Exploratory Data Mining. 7

EG Epsilon-Greedy [SB18]. 78, 80

Exp3 Exponential-weight algorithm for Exploration and Exploitation [Aue+00]. 23

Exp3.M Exp3 with Multiple plays [UNK10]. 23, 70, 78, 79, 84

GMD Greedy Maximum Deviation [TB19]. 24

JoSE Joint Spherical Embedding [Men+19]. 108, 115

HiCS High Contrast Subspaces [KMB12]. 21, 24, 40, 51, 54, 57, 61, 99, 104

II Interaction Information [McG54]. 21, 51, 54, 57, 61

IID Intrinsic Dimensionality Dependency [Rom+16]. 21

k-ANCS ANCS from the k nearest neighbours. 114–118, 121

KDD Knowledge Discovery from Data. 3, 4, 105, 139

kj-NN kj-Nearest Neighbours. iii, vi, ix, 9, 10, 19, 25, 107–109, 111–114, 116–123, 128, 130, 141, 143

KL-UCB Kullback-Leibler UCB. 66, 67, 70, 77, 78, 145

KL-S Kullback-Leibler Scaling. 66, 67, 69, 78, 94, 103

KSP Kolmogorov-Smirnov-P. viii, 29, 36, 40–42, 44–49, 57, 58, 97, 127, 139, 143

LOF Local Outlier Factor [Bre+00]. 24, 95, 99, 104, 114–118, 121

MAB Multi-Armed Bandit. iii, v, 9, 10, 14, 15, 23, 65, 137, 139

MAC Multivariate Maximal Correlation [Ngu+14b]. 21, 51, 54, 57, 61

MCDE Monte Carlo Dependency Estimation. iii, v, viii, 9, 10, 12, 13, 17, 21, 29–36, 38, 40, 42, 44–48, 50, 52, 54, 56, 58–61, 93, 105, 123, 127–130, 141, 143

MI Mutual Information. 8, 11, 21, 22, 61, 81, 84, 129

MISE Mutual Information Stream Estimation [KMB15]. 22

MP-KL-UCB Multiple-Play Kullback-Leibler UCB [GC11; KHN15]. 23, 70, 78

MP-MAB Multiple-Play Multi-Armed Bandit. 15, 23, 66, 69, 70, 137

MP-TS Multiple-Play Thompson Sampling [KHN15]. 23, 66, 67, 69, 70, 73, 84, 127

MS Multivariate Spearman [SS07]. 21, 51, 54, 57, 61

MWP Mann-Withney-P. viii, 29, 36, 37, 39, 40, 42, 44–52, 54, 57–59, 127, 139, 143

RCNN Recurrent CNN [Lai+15]. 114, 119, 120

RO Random Oracle. 82, 84

ROC Receiver Operating Characteristic. 96, 112

RS-Hash Randomised Subspace Hashing [SA16]. 24, 99, 114, 115, 117, 118, 121

RS-Stream Randomised Subspace Hashing in Streams [SA18]. 24, 99, 104

S-KL-UCB Scaling Kullback-Leibler UCB. 78, 79, 84

S-MAB Scaling Multi-Armed Bandit. iii, v, viii, 9, 10, 14–17, 22, 23, 65, 66, 68, 70, 72, 74, 76, 78, 80–82, 84, 94, 103, 105, 123, 127–130, 138

S-TS Scaling Thompson Sampling. viii, 65–69, 77–80, 82, 84, 94, 99, 103, 127, 129, 139, 140, 143

S-TS-ADWIN Scaling Thompson Sampling with ADWIN. viii, 68, 69, 77–80, 82, 84, 139, 143

SGMRD Streaming Greedy Maximum Random Deviation. iii, vi, ix, 9, 10, 17, 87, 90, 91, 94–100, 102–105, 127, 128, 130, 140, 141, 143

SO Static Oracle. 78, 82

SW-UCB Sliding Window UCB [GM08]. 78, 80

TC Total Correlation [Wat60]. 21, 51, 54, 57, 61

TONMF Text Outliers using Non-negative Matrix Factorisation [Kan+17]. 24, 114, 115, 117, 118

TS Thompson Sampling [Tho33]. 9, 65, 73, 82, 99, 100, 102–104, 140

UCB Upper Confidence Bound [Aue+02]. 23, 70, 82

UDS Universal Dependency Score [NMV16]. 21, 51, 54, 57, 61

UMC Unbiased Multivariate Correlation [Wan+17]. 21

VD-CNN Very Deep CNN [Con+17]. 114, 119, 120

vMF von Mises-Fisher. 24, 114

VMF-Q Von Mises-Fisher Quantiles [Zhu+17]. 114, 115, 117, 118

W-CNN Word-level CNN [Kim14]. 114, 119, 120

Notation

D A set of attributes, $D = \{s_1, \ldots, s_{|D|}\}$. 13, 88–96, 98, 137, 143

B An open list of observations, $B = (\vec{x}_1, \vec{x}_2, \ldots)$. 13, 95, 137, 143

$\vec{x}_i$ An observation, i.e., a vector of values with $|D|$ attributes, $\vec{x}_i = \langle x_{ij} \rangle_{j \in \{1,\ldots,|D|\}}$. 13, 32, 37, 45, 95, 137, 143

s_i An attribute in a data stream D, a.k.a. dimension, variable. 13, 30–33, 35, 37, 38, 40, 41, 43, 44, 88–96, 137, 143

t The current time step. 13, 16, 44, 66–79, 82, 89, 90, 93–96, 100, 103, 137, 138, 145

w The number of observations in a sliding window. 13, 32, 37–46, 50, 56–59, 68, 69, 78, 80, 82, 91, 95, 100, 104, 137, 139, 143

W_t A window containing the latest w observations, $W_t = (\vec{x}_{t-w+1}, \ldots, \vec{x}_t)$. 13, 44, 137

S A subspace, i.e., a projection on $|S|$ attributes, $S \subseteq D$, $|S| \leq |D|$. 13, 30–36, 38–46, 48, 50, 52–55, 57, 88–94, 137, 139, 143

X_{s_i} A random variable associated with an attribute $s_i \in D$. 13, 30, 137

Num A set of attributes of numerical type. 13, 32, 44

Ord A set of attributes of ordinal type. 13, 32, 44

Cat A set of attributes of categorical type. 13, 32, 41, 44

$p(X)$ The joint probability distribution function (*pdf*) of a random vector X. 13, 30

X A random vector, $X = \left\langle X_{s_i} \right\rangle_{s_i \in S}$. 13, 30–33, 92, 137

$p_{s_i}(X)$ The marginal probability distribution function (*pdf*) of variable s_i. 13, 30–32

$\mathcal{P}$ The power set (e.g., of a subspace S or set of attributes D). 13, 30, 31, 88–90, 93, 98

K The number of arms in a MAB problem. 14–16, 66–76, 78, 79, 84, 94, 139, 140, 143

L The number of plays per round in a MP-MAB problem. 15, 16, 54, 66, 67, 69–73, 75, 78, 93, 94, 100–104, 140, 145

μ_i The expected reward of arm i. 16, 66, 68–71, 73, 74, 77–79, 82, 94

T The total number of time steps (potentially infinite in streams). 16, 67–76, 78, 82–84, 94, 96, 102, 103, 139, 140

$I(t)$ The set of arms selected at time t. 16, 66–70, 73, 74, 90, 94, 96

$X(t)$ The reward vector received at time t. 16, 66, 67, 69, 94

$X_i(t)$ The reward vector received from arm i at time t. 16, 67–69, 94

$N_i(t)$ The number of draws of arm i before time t. 16, 66, 67, 69, 73, 74, 76, 77

$S_i(t)$ The sum of the rewards obtained from arm i before time t. 16, 67, 69

η^* The efficiency factor of a S-MAB, controling the trade-off between the cost of playing the reward obtained. 16, 66, 67, 69, 71, 73, 75, 78, 82, 84, 94, 99, 103, 143

L^* The optimal number of plays of a S-MAB, see Equation 2.2. 16, 69–72, 75, 76, 78–80

q A Dimension-Subspace Quality Function (D-SQF) (cf. Definition 6.1). 88, 89, 93, 94, 138

q_t The D-SQF q at time t. 89–93, 96

$\mathbb{S}^*$ An optimal set of subspaces (cf. Definition 6.3). 89, 138

$\mathbb{S}_t^*$ The optimal set of subspaces $\mathbb{S}^*$ at time t. 89, 90, 96, 138

$\mathbb{S}_t$ An approximation of $\mathbb{S}_t^*$. 90, 91, 93–95, 143

$\mathcal{S}$ An arbitrary set of subspaces. 92

Doc A set of documents, $Doc = \{d_1, d_2, \ldots, d_{|Doc|}\}$. 19, 108–112, 138, 143

d A document, $d \in Doc$. 19, 109–112, 119, 123, 138

Phr A set of phrases, $Phr = \{p_1, p_2, \ldots, p_{|Phr|}\}$. 19, 108, 109, 111, 112, 138, 143

p A phrase, $p \in Phr$. 19, 108, 110–112, 138

O A set of text objects (documents and phrases), $O = Doc \cup Phr$. 19, 108, 109, 138

o An object (document or phrase), $o \in O$. 19, 138

V An embedding vector representation $V : O \mapsto \mathbb{R}^n$ with n dimensions. 19, 108, 138

Sim A similarity function between text objects, $Sim : O^2 \mapsto [0, 1]$. 19, 109, 111, 112, 138

C A set of classes, $C = \{c_1, \ldots, c_{|C|}\}$. 19, 108, 110–112, 138

c A class, $c \in C$. 19, 108–112, 123, 138

y An initial classification function of documents, $y : Doc \mapsto C$. 19, 108, 110, 112, 138, 143

r A representativeness function of phrases, $r : Phr \times C \mapsto C$. 19, 108, 110–112, 138, 143

List of Figures

List of Tables

List of Algorithms

List of Theorems

Bibliography

[ABR19] Vadim Arzamasov, Klemens Böhm and Ignaz Rutter. 'Minimizing Bias in Estimation of Mutual Information from Data Streams'. In: *SSDBM*. ACM, 2019, pp. 1–12. DOI: 10.1145/3335783.3335796 (page 22).

[ACF02] Peter Auer, Nicolò Cesa-Bianchi and Paul Fischer. 'Finite-time Analysis of the Multiarmed Bandit Problem'. In: *Mach. Learn.* 47.2-3 (2002), pp. 235–256. DOI: 10.1023/A:1013689704352 (page 22).

[ACG18] Mastane Achab, Stéphan Clémençon and Aurélien Garivier. 'Profitable Bandits'. In: *ACML*. Vol. 95. Proceedings of Machine Learning Research. PMLR, 2018, pp. 694–709. URL: http://proceedings.mlr.press/v95/achab18a.html (page 23).

[AG13] Shipra Agrawal and Navin Goyal. 'Further Optimal Regret Bounds for Thompson Sampling'. In: *AISTATS*. Vol. 31. JMLR Workshop and Conference Proceedings. JMLR.org, 2013, pp. 99–107. URL: http://proceedings.mlr.press/v31/agrawal13a.html (pages 70, 74).

[Agg+04] Charu C. Aggarwal, Jiawei Han, Jianyong Wang and Philip S. Yu. 'A Framework for Projected Clustering of High Dimensional Data Streams'. In: *VLDB*. Morgan Kaufmann, 2004, pp. 852–863. DOI: 10.1016/B978-012088469-8.50075-9 (page 23).

[Agg09] Charu C. Aggarwal. 'On High Dimensional Projected Clustering of Uncertain Data Streams'. In: *ICDE*. IEEE Computer Society, 2009, pp. 1152–1154. DOI: 10.1109/ICDE.2009.188 (pages 24, 95).

[Agg13] Charu C. Aggarwal. *Outlier Analysis*. Springer, 2013. DOI: 10.1007/978-1-4614-6396-2 (pages 4, 5, 17, 25, 87).

[Agg15] Charu C. Aggarwal. *Data Mining - The Textbook*. Springer, 2015. DOI: 10.1007/978-3-319-14142-8 (page 3).

[Agr+05] Rakesh Agrawal, Johannes Gehrke, Dimitrios Gunopulos and Prabhakar Raghavan. 'Automatic Subspace Clustering of High Dimensional Data'. In: *Data Min. Knowl. Discov.* 11.1 (2005), pp. 5–33. DOI: 10.1007/s10618-005-1396-1 (page 95).

[AK04] Baruch Awerbuch and Robert D. Kleinberg. 'Adaptive Routing with End-to-End feedback: Distributed Learning and Geometric Approaches'. In: *STOC*. ACM, 2004, pp. 45–53. DOI: 10.1145/1007352.1007367 (page 23).

[AS15] Charu C. Aggarwal and Saket Sathe. 'Theoretical Foundations and Algorithms for Outlier Ensembles'. In: *SIGKDD Explorations* 17.1 (2015), pp. 24–47. DOI: 10.1145/2830544.2830549 (page 95).

[Ass+07] Ira Assent, Ralph Krieger, Emmanuel Müller and Thomas Seidl. 'VISA: Visual Subspace Clustering Analysis'. In: *SIGKDD Explorations* 9.2 (2007), pp. 5–12. DOI: 10.1145/1345448.1345451 (pages 7, 17).

[Ass+12] Ira Assent, Philipp Kranen, Corinna Baldauf and Thomas Seidl. 'AnyOut: Anytime Outlier Detection on Streaming Data'. In: *DASFAA (1)*. Vol. 7238. Lecture Notes in Computer Science. Springer, 2012, pp. 228–242. DOI: 10.1007/978-3-642-29038-1_18 (page 4).

[ATS14] Amineh Amini, Ying Wah Teh and Hadi Saboohi. 'On Density-Based Data Streams Clustering Algorithms: A Survey'. In: *J. Comput. Sci. Technol.* 29.1 (2014), pp. 116–141. DOI: 10.1007/s11390-014-1416-y (pages 4, 23).

[Aue+00] Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund and Robert E. Schapire. 'Gambling in a rigged casino: The adversarial multi-armed bandit problem'. In: *Electronic Colloquium on Computational Complexity (ECCC)* 7.68 (2000). URL: http://eccc.hpi-web.de/eccc-reports/2000/TR00-068/index.html (pages 22, 23, 133).

[Aue+02] Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund and Robert E. Schapire. 'The Nonstochastic Multiarmed Bandit Problem'. In: *SIAM J. Comput.* 32.1 (2002), pp. 48–77. DOI: 10.1137/S0097539701398375 (pages 23, 70, 135).

[AVW87] V. Anantharam, P. Varaiya and J. Walrand. 'Asymptotically efficient allocation rules for the multiarmed bandit problem with multiple plays-Part I: I.I.D. rewards'. In: *IEEE Transactions on Automatic Control* 32.11 (Nov. 1987), pp. 968–976. DOI: 10.1109/TAC.1987.1104491 (pages 15, 23).

[Bau+04] Christian Baumgartner, Claudia Plant, Karin Kailing, Hans-Peter Kriegel and Peer Kröger. 'Subspace Selection for Clustering High-Dimensional Data'. In: *ICDM*. IEEE Computer Society, 2004, pp. 11–18. DOI: 10.1109/ICDM.2004.10112 (page 17).

[BC12] Sébastien Bubeck and Nicolò Cesa-Bianchi. 'Regret Analysis of Stochastic and Nonstochastic Multi-armed Bandit Problems'. In: *Foundations and Trends in Machine Learning* 5.1 (2012), pp. 1–122. DOI: 10.1561/2200000024 (pages 15, 23).

[BDM02] Brian Babcock, Mayur Datar and Rajeev Motwani. 'Sampling From a Moving Window Over Streaming Data'. In: *SODA*. ACM/SIAM, 2002, pp. 633–634. ISBN: 089871513X. URL: http://dl.acm.org/citation.cfm?id=545381.545465 (page 22).

[Bec16] Vincent Becker. 'Concept Change Detection in Correlated Subspaces in Data Streams'. MA thesis. Karlsruhe Institute of Technology, 2016 (pages 24, 99).

[Bel+18] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeswar, Sherjil Ozair, Yoshua Bengio, R. Devon Hjelm and Aaron C. Courville. 'Mutual Information Neural Estimation'. In: *ICML*. Vol. 80. Proceedings of Machine Learning Research. PMLR, 2018. URL: http://proceedings.mlr.press/v80/belghazi18a.html (page 21).

[Bel57] Richard E. Bellman. *Dynamic Programming*. 1st ed. Princeton University Press, 1957. ISBN: 978-0-691-07951-6 (pages 5, 21).

[Bey+99] Kevin Beyer, Jonathan Goldstein, Raghu Ramakrishnan and Uri Shaft. 'When Is "Nearest Neighbor" Meaningful?' In: *ICDT*. Vol. 1540. Lecture Notes in Computer Science. Springer, 1999, pp. 217–235. DOI: 10.1007/3-540-49257-7_15 (page 5).

[BG07] Albert Bifet and Ricard Gavaldà. 'Learning from Time-Changing Data with Adaptive Windowing'. In: *SDM*. SIAM, 2007, pp. 443–448. DOI: 10.1137/1.9781611972771.42 (pages 7, 9, 22, 65, 68, 80, 103, 127, 129, 133).

[BGE15] Jean Paul Barddal, Heitor Murilo Gomes and Fabrício Enembreck. 'A Survey on Feature Drift Adaptation'. In: *ICTAI*. IEEE Computer Society, 2015, pp. 1053–1060. DOI: 10.1109/ICTAI.2015.150 (pages 6, 89, 97, 98).

[BH17] Jonathan Boidol and Andreas Hapfelmeier. 'Fast Mutual Information Computation forDependency-Monitoring on Data Streams'. In: *SAC*. ACM, 2017, pp. 830–835. DOI: 10.1145/3019612.3019669 (page 22).

[BHS15] Mohamed Bennasar, Yulia Hicks and Rossitza Setchi. 'Feature selection using Joint Mutual Information Maximisation'. In: *Expert Syst. Appl.* 42.22 (2015), pp. 8520–8532. DOI: 10.1016/j.eswa.2015.07.007 (page 7).

[BL16] Manuele Bicego and Marco Loog. 'Weighted K-Nearest Neighbor revisited'. In: *ICPR*. IEEE Computer Society, 2016, pp. 1642–1647. DOI: 10.1109/ICPR.2016.7899872 (page 111).

[BLL15] Giuseppe Burtini, Jason Loeppky and Ramon Lawrence. *A Survey of Online Experiment Design with the Stochastic Multi-Armed Bandit*. 2015. arXiv: 1510.00757 [stat.ML] (page 23).

[BM00] Edgar Brunner and Ullrich Munzel. 'The Nonparametric Behrens-Fisher Problem: Asymptotic Theory and a Small-Sample Approximation'. In: *Biometrical journal* 42.1 (2000), pp. 17–25. DOI: 10.1002/(SICI)1521-4036(200001)42:1<17::AID-BIMJ17>3.0.CO;2-U (page 129).

[Bor+11] Antoine Bordes, Xavier Glorot, Jason Weston and Yoshua Bengio. *Towards Open-Text Semantic Parsing via Multi-Task Learning of Structured Embeddings*. 2011. arXiv: 1107.3663 [cs.AI] (page 123).

[Bre+00] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng and Jörg Sander. 'LOF: Identifying Density-Based Local Outliers'. In: *SIGMOD Conference*. ACM, 2000, pp. 93–104. DOI: 10.1145/342009.335388 (pages 24, 95, 99, 114, 134).

[Cam+16] Guilherme O. Campos, Arthur Zimek, Jörg Sander, Ricardo J. G. B. Campello, Barbora Micenková, Erich Schubert, Ira Assent and Michael E. Houle. 'On the Evaluation of Unsupervised Outlier Detection: Measures, Datasets, and an Empirical Study'. In: *Data Min. Knowl. Discov.* 30.4 (2016), pp. 891–927. DOI: 10.1007/s10618-015-0444-8 (pages 99, 114).

[Cha+01] Kaushik Chakrabarti, Minos N. Garofalakis, Rajeev Rastogi and Kyuseok Shim. 'Approximate Query Processing Using Wavelets'. In: *VLDB J.* 10.2-3 (2001), pp. 199–223. DOI: 10.1007/s007780100049 (page 22).

[Cha+08] Deepayan Chakrabarti, Ravi Kumar, Filip Radlinski and Eli Upfal. 'Mortal Multi-Armed Bandits'. In: *NIPS*. Curran Associates, Inc., 2008, pp. 273–280. URL: http://papers.nips.cc/paper/3580-mortal-multi-armed-bandits (page 23).

[Che+16] Wei Chen, Wei Hu, Fu Li, Jian Li, Yu Liu and Pinyan Lu. 'Combinatorial Multi-Armed Bandit with General Reward Functions'. In: *NIPS*. 2016, pp. 1651–1659. URL: http://papers.nips.cc/paper/6511-combinatorial-multi-armed-bandit-with-general-reward-functions (pages 23, 70, 78, 133).

[Che+17] Jinghui Chen, Saket Sathe, Charu C. Aggarwal and Deepak S. Turaga. 'Outlier Detection with Autoencoder Ensembles'. In: *SDM*. SIAM, 2017, pp. 90–98. DOI: 10.1137/1.9781611974973.11 (page 4).

[CHY96] Ming-Syan Chen, Jiawei Han and Philip S. Yu. 'Data Mining: An Overview from a Database Perspective'. In: *IEEE Trans. Knowl. Data Eng.* 8.6 (1996), pp. 866–883. DOI: 10.1109/69.553155 (pages 7, 11).

[CL09] Antonio Di Crescenzo and Maria Longobardi. 'On Cumulative Entropies and Lifetime Estimations'. In: *IWINAC (1)*. Vol. 5601. Lecture Notes in Computer Science. Springer, 2009, pp. 132–141. DOI: 10.1007/978-3-642-02264-7_15 (pages 21, 133).

[Con+17] Alexis Conneau, Holger Schwenk, Loïc Barrault and Yann LeCun. 'Very Deep Convolutional Networks for Text Classification'. In: *EACL (1)*. Association for Computational Linguistics, 2017, pp. 1107–1116. DOI: 10.18653/v1/e17-1104 (pages 25, 114, 135).

[DG17] Dheeru Dua and Casey Graff. *UCI Machine Learning Repository*. 2017. URL: http://archive.ics.uci.edu/ml (page 97).

[DH03] Pedro Domingos and Geoff Hulten. 'A General Framework for Mining Massive Data Streams'. In: *Journal of Computational and Graphical Statistics* 12.4 (2003), pp. 945–949. DOI: 10.1198/1061860032544 (pages 6, 87).

[Dit+15] Gregory Ditzler, Manuel Roveri, Cesare Alippi and Robi Polikar. 'Learning in Nonstationary Environments: A Survey'. In: *IEEE Comp. Int. Mag.* 10.4 (2015), pp. 12–25. DOI: 10.1109/MCI.2015.2471196 (pages 6, 7).

[DJ03] Tamraparni Dasu and Theodore Johnson. *Exploratory Data Mining and Data Cleaning*. John Wiley, 2003. ISBN: 0-471-26851-8. DOI: 10.1002/0471448354 (page 7).

[DM17] Mathias Drton and Marloes H. Maathuis. 'Structure Learning in Graphical Modeling'. In: *Annual Review of Statistics and Its Application* 4.1 (2017), pp. 365–393. DOI: 10.1146/annurev-statistics-060116-053803 (page 130).

[DP16] Rémy Degenne and Vianney Perchet. 'Anytime optimal algorithms in stochastic multi-armed bandits'. In: *ICML*. Vol. 48. JMLR Workshop and Conference Proceedings. JMLR.org, 2016, pp. 1587–1595. URL: http://proceedings.mlr.press/v48/degenne16.html (page 23).

[Dud76] Sahibsingh A. Dudani. 'The Distance-Weighted k-Nearest-Neighbor Rule'. In: *IEEE Trans. Systems, Man, and Cybernetics* 6.4 (1976), pp. 325–327. DOI: 10.1109/TSMC.1976.5408784 (page 111).

[Dur73] J. Durbin. *Distribution Theory for Tests Based on the Sample Distribution Function*. 1st ed. SIAM, 1973. DOI: 10.1137/1.9781611970586 (page 40).

[FB19] Edouard Fouché and Klemens Böhm. 'Monte Carlo Dependency Estimation'. In: *SSDBM*. Best Paper Award. ACM, 2019, pp. 13–24. DOI: 10.1145/3335783.3335795 (pages 9, 17, 29, 46, 128, 129).

[FH89] Evelyn Fix and J. L. Hodges. 'Discriminatory Analysis – Nonparametric Discrimination: Consistency Properties'. In: *International Statistical Review / Revue Internationale de Statistique* 57.3 (1989), pp. 238–247. DOI: 10.2307/1403797 (page 109).

[FI93] Usama M. Fayyad and Keki B. Irani. 'Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning'. In: *IJCAI*. Morgan Kaufmann, 1993, pp. 1022–1029. URL: http://ijcai.org/Proceedings/93-2/Papers/022.pdf (page 22).

[FKB19] Edouard Fouché, Junpei Komiyama and Klemens Böhm. 'Scaling Multi-Armed Bandit Algorithms'. In: *KDD*. ACM, 2019, pp. 1449–1459. DOI: 10.1145/3292500.3330862 (pages 9, 17, 65, 94, 128).

[FKB20] Edouard Fouché, Florian Kalinke and Klemens Böhm. *Efficient Subspace Search in Data Streams*. 2020. arXiv: 2011.06959 [cs.LG] (pages 9, 87, 128).

[Fou+20a] Edouard Fouché, Alan Mazankiewicz, Florian Kalinke and Klemens Böhm. 'A Framework for Dependency Estimation in Heterogeneous Data Streams'. In: *Distributed and Parallel Databases* (2020). DOI: 10.1007/s10619-020-07295-x (pages 9, 17, 29, 128).

[Fou+20b] Edouard Fouché, Yu Meng, Fang Guo, Honglei Zhuang, Klemens Böhm and Jiawei Han. 'Mining Text Outliers in Document Directories'. In: *ICDM*. In press. IEEE Computer Society, 2020 (pages 9, 107, 128).

[FP81] Michael A. Fligner and George E. Policello. 'Robust Rank Procedures for the Behrens-Fisher Problem'. In: *J. Amer. Statist. Assoc.* 76.373 (1981), pp. 162–168. DOI: 10.2307/2287062 (page 129).

[Frí+15] I. Frías-Blanco, J. d. Campo-Ávila, G. Ramos-Jiménez, R. Morales-Bueno, A. Ortiz-Díaz and Y. Caballero-Mota. 'Online and Non-Parametric Drift Detection Methods Based on Hoeffding's Bounds'. In: *IEEE Trans. Knowl. Data Eng.* 27.3 (2015), pp. 810–823. DOI: 10.1109/TKDE.2014.2345382 (page 22).

[Fri97] Jerome H. Friedman. 'Data Mining and Statistics: What's the Connection?' In: *Proceedings of the 29th Symposium on the Interface Between Computer Science and Statistics*. 1997. URL: docs.salford-systems.com/dm-stat.pdf (page 4).

[FS09] Morten W. Fagerland and Leiv Sandvik. 'The Wilcoxon–Mann–Whitney test under scrutiny'. In: *Statistics in Medicine* 28.10 (2009), pp. 1487–1497. DOI: 10.1002/sim.3561 (page 37).

[FV14] Benoît Frénay and Michel Verleysen. 'Classification in the Presence of Label Noise: A Survey'. In: *IEEE Trans. Neural Netw. Learning Syst.* 25.5 (2014), pp. 845–869 (page 18).

[Gam+14] João Gama, Indre Zliobaite, Albert Bifet, Mykola Pechenizkiy and Abdelhamid Bouchachia. 'A Survey on Concept Drift Adaptation'. In: *ACM Comput. Surv.* 46.4 (2014), 44:1–44:37. DOI: 10.1145/2523813 (page 78).

[Gam10] João Gama. *Knowledge Discovery from Data Streams*. 1st ed. Chapman & Hall/CRC, 2010. ISBN: 1439826110 (pages 4, 22).

[Gam12] João Gama. 'A survey on learning from data streams: current and future trends'. In: *Progress in AI* 1.1 (2012), pp. 45–55. DOI: 10.1007/s13748-011-0002-6 (pages 4, 6, 13).

[GC11] Aurélien Garivier and Olivier Cappé. 'The KL-UCB Algorithm for Bounded Stochastic Bandits and Beyond'. In: *COLT*. Vol. 19. JMLR Proceedings. JMLR.org, 2011, pp. 359–376. URL: http://proceedings.mlr.press/v19/garivier11a/garivier11a.pdf (pages 23, 66, 70, 78, 134).

[GE03] Isabelle Guyon and André Elisseeff. 'An Introduction to Variable and Feature Selection'. In: *J. Mach. Learn. Res.* 3 (2003), pp. 1157–1182. URL: http://jmlr.org/papers/v3/guyon03a.html (pages 7, 17).

[Geb+14] Kareem El Gebaly, Parag Agrawal, Lukasz Golab, Flip Korn and Divesh Srivastava. 'Interpretable and Informative Explanations of Outcomes'. In: *PVLDB* 8.1 (2014), pp. 61–72. DOI: 10.14778/2735461.2735467 (page 7).

[Geb+18] Kareem El Gebaly, Guoyao Feng, Lukasz Golab, Flip Korn and Divesh Srivastava. 'Explanation Tables'. In: *IEEE Data Eng. Bull.* 41.3 (2018), pp. 43–51. URL: http://sites.computer.org/debull/A18sept/p43.pdf (page 7).

[GH05] Sudipto Guha and Boulos Harb. 'Wavelet Synopsis for Data Streams: Minimizing Non-Euclidean Error'. In: *KDD*. ACM, 2005, pp. 88–97. DOI: 10.1145/1081870.1081884 (page 22).

[GLH15] Salvador García, Julián Luengo and Francisco Herrera. *Data Preprocessing in Data Mining*. Vol. 72. Intelligent Systems Reference Library. Springer, 2015 (page 18).

[GM08] Aurélien Garivier and Eric Moulines. *On Upper-Confidence Bound Policies for Non-Stationary Bandit Problems*. 2008. arXiv: 0805.3415 [math.ST] (pages 22, 78, 135).

[GM11] Aurélien Garivier and Eric Moulines. 'On Upper-Confidence Bound Policies for Switching Bandit Problems'. In: *ALT*. Vol. 6925. Lecture Notes in Computer Science. Springer, 2011, pp. 174–188. DOI: 10.1007/978-3-642-24412-4_16 (pages 23, 78).

[Gou+19] Jianping Gou, Hongxing Ma, Weihua Ou, Shaoning Zeng, Yunbo Rao and Hebiao Yang. 'A generalized mean distance-based k-nearest neighbor classifier'. In: *Expert Syst. Appl.* 115 (2019), pp. 356–372. DOI: 10.1016/j.eswa.2018.08.021 (page 111).

[Gre+07] Arthur Gretton, Kenji Fukumizu, Choon Hui Teo, Le Song, Bernhard Schölkopf and Alexander J. Smola. 'A Kernel Statistical Test of Independence'. In: *NIPS*. Curran Associates, Inc., 2007, pp. 585–592. URL: http://papers.nips.cc/paper/3201-a-kernel-statistical-test-of-independence (page 21).

[Gup+14] Manish Gupta, Jing Gao, Charu C. Aggarwal and Jiawei Han. 'Outlier Detection for Temporal Data: A Survey'. In: *IEEE Trans. Knowl. Data Eng.* 26.9 (2014), pp. 2250–2267. DOI: 10.1109/TKDE.2013.184 (pages 23, 24).

[Has11] H. M. Hashemian. 'State-of-the-Art Predictive Maintenance Techniques'. In: *IEEE Trans. Instrumentation and Measurement* 60.1 (2011), pp. 226–236. DOI: 10.1109/TIM.2010.2047662 (page 8).

[Hec+00] David Heckerman, David Maxwell Chickering, Christopher Meek, Robert Rounthwaite and Carl Myers Kadie. 'Dependency Networks for Inference, Collaborative Filtering, and Data Visualization'. In: *J. Mach. Learn. Res.* 1 (2000), pp. 49–75. URL: http://jmlr.org/papers/v1/heckerman00a.html (page 130).

[HH03] Mark A. Hall and Geoffrey Holmes. 'Benchmarking Attribute Selection Techniques for Discrete Class Data Mining'. In: *IEEE Trans. Knowl. Data Eng.* 15.6 (2003), pp. 1437–1447. DOI: 10.1109/TKDE.2003.1245283 (pages 7, 11).

[HKP11] Jiawei Han, Micheline Kamber and Jian Pei. *Data Mining: Concepts and Techniques*. 3rd ed. Morgan Kaufmann, 2011. ISBN: 978-0123814791 (pages 3, 4).

[HLN99] Jiawei Han, Laks V. S. Lakshmanan and Raymond T. Ng. 'Constraint-Based Multidimensional Data Mining'. In: *Computer* 32.8 (1999), pp. 46–50. DOI: 10.1109/2.781634 (page 130).

[Hoe63] Wassily Hoeffding. 'Probability Inequalities for Sums of Bounded Random Variables'. In: *J. Amer. Statist. Assoc.* 58.301 (1963), pp. 13–30. DOI: 10.1080/01621459.1963.10500830 (pages 35, 36).

[HS04] K. Hechenbichler and K. Schliep. *Weighted k-Nearest-Neighbor Techniques and Ordinal Classification*. Discussion Paper 399, SFB 386, Ludwigs Maximilians University Munich. 2004. DOI: 10.5282/ubm/epub.1769 (page 111).

[HTF09] Trevor Hastie, Robert Tibshirani and Jerome H. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd ed. Springer Series in Statistics. Springer, 2009. DOI: 10.1007/978-0-387-84858-7 (pages 5, 31).

[Joe89] Harry Joe. 'Relative Entropy Measures of Multivariate Dependence'. In: *J. Amer. Statist. Assoc.* 84.405 (1989), pp. 157–164. ISSN: 01621459. DOI: 10.2307/2289859 (page 92).

[KA14] Justin B. Kinney and Gurinder S. Atwal. 'Equitability, mutual information, and the maximal information coefficient'. In: *Proceedings of the National Academy of Sciences* 111.9 (2014), pp. 3354–3359. ISSN: 0027-8424. DOI: 10.1073/pnas.1309933111 (pages 12, 46).

[Kai+03] Karin Kailing, Hans-Peter Kriegel, Peer Kröger and Stefanie Wanka. 'Ranking Interesting Subspaces for Clustering High Dimensional Data'. In: *PKDD*. Vol. 2838. Lecture Notes in Computer Science. Springer, 2003, pp. 241–252. DOI: 10.1007/978-3-540-39804-2_23 (page 17).

[Kan+17] Ramakrishnan Kannan, Hyenkyun Woo, Charu C. Aggarwal and Haesun Park. 'Outlier Detection for Text Data'. In: *SDM*. SIAM, 2017, pp. 489–497. DOI: 10.1137/1.9781611974973.55 (pages 24, 114, 135).

[Kan19] Mehmed Kantardzic. *Data Mining: Concepts, Models, Methods, and Algorithms*. 3rd ed. John Wiley & Sons, 2019. ISBN: 978-1-119-51604-0 (page 3).

[Kar+06] Hillol Kargupta, Vasundhara Puttagunta, Martin Klein and Kakali Sarkar. 'On-board Vehicle Data Stream Monitoring Using MineFleet and Fast Resource Constrained Monitoring of Correlation Matrices'. In: *New Generation Comput.* 25.1 (2006), pp. 5–32. DOI: 10.1007/s00354-006-0002-4 (page 22).

[Kel15] Fabian Keller. 'Attribute Relationship Analysis in Outlier Mining and Stream Processing'. PhD thesis. Karlsruhe Institute of Technology, 2015. DOI: 10.5445/IR/1000048790 (pages ii, 17).

[Ken38] M. G. Kendall. 'A New Measure of Rank Correlation'. In: *Biometrika* 30.1/2 (1938), pp. 81–93. ISSN: 00063444. DOI: 10.2307/2332226 (page 21).

[KHN15] Junpei Komiyama, Junya Honda and Hiroshi Nakagawa. 'Optimal Regret Analysis of Thompson Sampling in Stochastic Multi-armed Bandit Problem with Multiple Plays'. In: *ICML*. Vol. 37. JMLR Workshop and Conference Proceedings. JMLR.org, 2015, pp. 1152–1161. URL: http://proceedings.mlr.press/v37/komiyama15.html (pages 15, 23, 66, 70, 73, 74, 78, 84, 127, 134).

[Kim14] Yoon Kim. 'Convolutional Neural Networks for Sentence Classification'. In: *EMNLP*. ACL, 2014, pp. 1746–1751. DOI: 10.3115/v1/d14-1181 (pages 25, 114, 135).

[KKM12] Emilie Kaufmann, Nathaniel Korda and Rémi Munos. 'Thompson Sampling: An Asymptotically Optimal Finite-Time Analysis'. In: *ALT*. Vol. 7568. Lecture Notes in Computer Science. Springer, 2012, pp. 199–213. DOI: 10.1007/978-3-642-34106-9_18 (page 22).

[KKZ09] Hans-Peter Kriegel, Peer Kröger and Arthur Zimek. 'Clustering High-Dimensional Data: A Survey on Subspace Clustering, Pattern-Based Clustering, and Correlation Clustering'. In: *TKDD* 3.1 (2009), 1:1–1:58. DOI: 10.1145/1497577.1497578 (page 24).

[Kle06] Robert D. Kleinberg. 'Anytime Algorithms for Multi-Armed Bandit Problems'. In: *SODA*. ACM, 2006, pp. 928–936. ISBN: 0898716055. URL: http://dl.acm.org/citation.cfm?id=1109557.1109659 (page 23).

[Kli04] Ralf Klinkenberg. 'Learning drifting concepts: Example selection vs. example weighting'. In: *Intell. Data Anal.* 8.3 (2004), pp. 281–300. DOI: 10.3233/IDA-2004-8305 (page 22).

[KMB12] Fabian Keller, Emmanuel Müller and Klemens Böhm. 'HiCS: High Contrast Subspaces for Density-Based Outlier Ranking'. In: *ICDE*. IEEE Computer Society, 2012, pp. 1037–1048. DOI: 10.1109/ICDE.2012.88 (pages 5, 7, 11, 17, 21, 24, 31, 32, 40, 89, 95, 97–99, 133).

[KMB15] Fabian Keller, Emmanuel Müller and Klemens Böhm. 'Estimating Mutual Information on Data Streams'. In: *SSDBM*. ACM, 2015, 3:1–3:12. DOI: 10.1145/2791347.2791348 (pages 22, 134).

[KN98] Edwin M. Knorr and Raymond T. Ng. 'Algorithms for Mining Distance-Based Outliers in Large Datasets'. In: *VLDB*. Morgan Kaufmann, 1998, pp. 392–403. URL: http://www.vldb.org/conf/1998/p392.pdf (page 24).

[Koy00] Ivan Koychev. 'Gradual Forgetting for Adaptation to Concept Drift'. In: *In Proceedings of ECAI 2000 Workshop Current Issues in Spatio-Temporal Reasoning*. 2000, pp. 101–106 (page 22).

[KP14] Volodymyr Kuleshov and Doina Precup. *Algorithms for multi-armed bandit problems*. 2014. arXiv: 1402.6028 [cs.AI] (page 67).

[KPM06] Maria Kontaki, Apostolos N. Papadopoulos and Yannis Manolopoulos. 'Efficient Incremental Subspace Clustering in Data Streams'. In: *IDEAS*. IEEE Computer Society, 2006, pp. 53–60. DOI: 10.1109/IDEAS.2006.19 (pages 24, 95).

[Kri+11] Hans-Peter Kriegel, Peer Kröger, Irene Ntoutsi and Arthur Zimek. 'Density Based Subspace Clustering over Dynamic Data'. In: *SSDBM*. Vol. 6809. Lecture Notes in Computer Science. Springer, 2011, pp. 387–404. DOI: 10.1007/978-3-642-22351-8_24 (page 23).

[Kri+12] Hans-Peter Kriegel, Peer Kröger, Erich Schubert and Arthur Zimek. 'Outlier Detection in Arbitrarily Oriented Subspaces'. In: *ICDM*. IEEE Computer Society, 2012, pp. 379–388. DOI: 10.1109/ICDM.2012.21 (page 24).

[KS12] JooSeuk Kim and Clayton D. Scott. 'Robust Kernel Density Estimation'. In: *J. Mach. Learn. Res.* 13 (2012), pp. 2529–2565. URL: http://dl.acm.org/citation.cfm?id=2503323 (page 24).

[KSG04] Alexander Kraskov, Harald Stögbauer and Peter Grassberger. 'Estimating Mutual Information'. In: *Phys. Rev. E* 69 (6 June 2004), p. 066138. DOI: 10.1103/PhysRevE.69.066138 (pages 81, 84).

[KSZ08] Hans-Peter Kriegel, Matthias Schubert and Arthur Zimek. 'Angle-Based Outlier Detection in High-dimensional Data'. In: *KDD*. ACM, 2008, pp. 444–452. DOI: 10.1145/1401890.1401946 (page 24).

[Lai+15] Siwei Lai, Liheng Xu, Kang Liu and Jun Zhao. 'Recurrent Convolutional Neural Networks for Text Classification'. In: *AAAI*. AAAI Press, 2015, pp. 2267–2273. URL: http://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/view/9745 (pages 25, 114, 134).

[LHS13] David Lopez-Paz, Philipp Hennig and Bernhard Schölkopf. 'The Randomized Dependence Coefficient'. In: *NIPS*. 2013, pp. 1–9. URL: http://papers.nips.cc/paper/5138-the-randomized-dependence-coefficient (page 21).

[Li+10] Lihong Li, Wei Chu, John Langford and Robert E. Schapire. 'A Contextual-Bandit Approach to Personalized News Article Recommendation'. In: *WWW*. ACM, 2010, pp. 661–670. DOI: 10.1145/1772690.1772758 (page 23).

[Li+18] Tian Li, Jie Zhong, Ji Liu, Wentao Wu and Ce Zhang. 'Ease.ml: Towards Multi-tenant Resource Sharing for Machine Learning Workloads'. In: *PVLDB* 11.5 (2018), pp. 607–620. DOI: 10.1145/3187009.3177737 (page 23).

[Liu+09] Huawen Liu, Jigui Sun, Lei Liu and Huijie Zhang. 'Feature selection with dynamic mutual information'. In: *Pattern Recognit.* 42.7 (2009), pp. 1330–1339. DOI: 10.1016/j.patcog.2008.10.028 (page 7).

[LK05] Aleksandar Lazarevic and Vipin Kumar. 'Feature Bagging for Outlier Detection'. In: *KDD*. ACM, 2005, pp. 157–166. DOI: 10.1145/1081870.1081891 (pages 24, 95).

[Lop16] David Lopez-Paz. 'From Dependence to Causation'. PhD thesis. University of Cambridge and Max Planck Institute for Intelligent Systems, 2016. arXiv: 1607.03300 [stat.ML] (page 130).

[LR05] Erich L. Lehmann and Joseph P. Romano. *Testing Statistical Hypotheses*. 3rd ed. Springer, 2005. ISBN: 978-0-387-98864-1. DOI: 10.1007/0-387-27605-X (page 40).

[LS20] Tor Lattimore and Csaba Szepesvári. *Bandit Algorithms*. Cambridge University Press, 2020. ISBN: 9781108571401 (page 23).

[Mai17] Odalric-Ambrym Maillard. 'Boundary Crossing for General Exponential Families'. In: *ALT*. Vol. 76. Proceedings of Machine Learning Research. PMLR, 2017, pp. 151–184. URL: http://proceedings.mlr.press/v76/maillard17a.html (pages 66, 73, 77).

[McG54] William J. McGill. 'Multivariate Information Transmission'. In: *Trans. of the IRE Professional Group on Information Theory (TIT)* 4 (1954), pp. 93–111. DOI: 10.1109/TIT.1954.1057469 (pages 21, 51, 133).

[Men+18] Yu Meng, Jiaming Shen, Chao Zhang and Jiawei Han. 'Weakly-Supervised Neural Text Classification'. In: *CIKM*. ACM, 2018, pp. 983–992. DOI: 10.1145/3269206.3271737 (page 130).

[Men+19] Yu Meng, Jiaxin Huang, Guangyuan Wang, Chao Zhang, Honglei Zhuang, Lance M. Kaplan and Jiawei Han. 'Spherical Text Embedding'. In: *NeurIPS*. 2019, pp. 8206–8215. URL: http://papers.nips.cc/paper/9031-spherical-text-embedding (pages 19, 107, 108, 115, 133).

[MLA18] Emaad Manzoor, Hemank Lamba and Leman Akoglu. 'xStream: Outlier Detection in Feature-Evolving Data Streams'. In: *KDD*. ACM, 2018, pp. 1963–1972. DOI: 10.1145/3219819.3220107 (pages 24, 99).

[MMM14] Ben Murrell, Daniel Murrell and Hugh Murrell. 'R2-equitability is satisfiable'. In: *Proceedings of the National Academy of Sciences* 111.21 (2014), E2160–E2160. DOI: 10.1073/pnas.1403623111 (page 12).

[MNP10] Chris Mayfield, Jennifer Neville and Sunil Prabhakar. 'ERACER: A Database Approach for Statistical Inferenceand Data Cleaning'. In: *SIGMOD Conference*. ACM, 2010, pp. 75–86. DOI: 10.1145/1807167.1807178 (page 7).

[MR10] Oded Maimon and Lior Rokach, eds. *The Data Mining and Knowledge Discovery Handbook*. 2nd ed. Springer, 2010. ISBN: 978-0-387-09822-7. DOI: 10.1007/978-0-387-09823-4 (page 3).

[Mül+12] Emmanuel Müller, Ira Assent, Patricia Iglesias Sánchez, Yvonne Mülle and Klemens Böhm. 'Outlier Ranking via Subspace Analysis in Multiple Views of the Data'. In: *ICDM*. IEEE Computer Society, 2012, pp. 529–538. DOI: 10.1109/ICDM.2012.112 (page 24).

[MW47] H. B. Mann and D. R. Whitney. 'On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other'. In: *The Annals of Mathematical Statistics* 18.1 (1947), pp. 50–60. URL: http://www.jstor.org/stable/2236101 (page 37).

[Nan16] Preetam Nandy. 'Causal Learning from High-dimensional Observational Data'. PhD thesis. ETH Zurich, 2016. DOI: 10.3929/ethz-a-010710937 (page 130).

[Ngu+13] Hoang Vu Nguyen, Emmanuel Müller, Jilles Vreeken, Fabian Keller and Klemens Böhm. 'CMI: An Information-Theoretic Contrast Measure for Enhancing Subspace Cluster and Outlier Detection'. In: *SDM*. SIAM, 2013, pp. 198–206. DOI: 10.1137/1.9781611972832.22 (pages 5, 7, 11, 21, 24, 51, 133).

[Ngu+14a] Hoang Vu Nguyen, Emmanuel Müller, Periklis Andritsos and Klemens Böhm. 'Detecting Correlated Columns in Relational Databases with Mixed Data Types'. In: *SSDBM*. ACM, 2014, 30:1–30:12. DOI: 10.1145/2618243.2618251 (page 22).

[Ngu+14b] Hoang Vu Nguyen, Emmanuel Müller, Jilles Vreeken, Pavel Efros and Klemens Böhm. 'Multivariate Maximal Correlation Analysis'. In: *ICML*. Vol. 32. JMLR Workshop and Conference Proceedings. JMLR.org, 2014, pp. 775–783. URL: http://proceedings.mlr.press/v32/nguyenc14.html (pages 21, 51, 134).

[Ngu+16] Xuan Vinh Nguyen, Jeffrey Chan, Simone Romano, James Bailey, Christopher Leckie, Kotagiri Ramamohanarao and Jian Pei. 'Discovering outlying aspects in large datasets'. In: *Data Min. Knowl. Discov.* 30.6 (2016), pp. 1520–1555. DOI: 10.1007/s10618-016-0453-2 (page 95).

[Ngu15] Hoang Vu Nguyen. 'Non-parametric Methods for Correlation Analysis in Multivariate Data with Applications in Data Mining'. PhD thesis. Karlsruhe Institute of Technology, 2015. DOI: 10.5445/IR/1000049548 (page 89).

[NMB13] Hoang Vu Nguyen, Emmanuel Müller and Klemens Böhm. '4S: Scalable Subspace Search Scheme. Overcoming Traditional Apriori Processing'. In: *BigData*. IEEE Computer Society, 2013, pp. 359–367. DOI: 10.1109/BigData.2013.6691596 (pages 5, 7, 24).

[NMV16] Hoang Vu Nguyen, Panagiotis Mandros and Jilles Vreeken. 'Universal Dependency Analysis'. In: *SDM*. SIAM, 2016, pp. 792–800. DOI: 10.1137/1.9781611974348.89 (pages 21, 24, 46, 51, 95, 135).

[Nto+12] Irene Ntoutsi, Arthur Zimek, Themis Palpanas, Peer Kröger and Hans-Peter Kriegel. 'Density-based Projected Clustering over High Dimensional Data Streams'. In: *SDM*. SIAM, 2012, pp. 987–998. DOI: 10.1137/1.9781611972825.85 (pages 4, 23).

[NV16] Hoang Vu Nguyen and Jilles Vreeken. 'Linear-time Detection of Non-linear Changes in Massively High Dimensional Time Series'. In: *SDM*. SIAM, 2016, pp. 828–836. DOI: 10.1137/1.9781611974348.93 (page 22).

[Pea09] Judea Pearl. 'Causal inference in statistics: An overview'. In: *Statistics Surveys* 3 (Jan. 2009), pp. 96–146. DOI: 10.1214/09-SS057 (page 130).

[Pea19] Judea Pearl. 'The Seven Tools of Causal Inference with Reflections on Machine Learning'. In: *Commun. ACM* 62.3 (2019), pp. 54–60. DOI: 10.1145/3241036 (page 130).

[Pfi+16] Cornelius Pfitzer, Nicolaus Dahmen, Nicole Tröger, Friedhelm Weirich, Jörg Sauer, Armin Günther and Matthias Müller-Hagedorn. 'Fast Pyrolysis of Wheat Straw in the Bioliq Pilot Plant'. In: *Energy & Fuels / Special Issue* 30.10 (2016). 35.13.04; LK 01, pp. 8047–8054. ISSN: 0887-0624, 1520-5029. DOI: 10.1021/acs.energyfuels.6b01412 (pages 8, 10, 59).

[PHL04] Lance Parsons, Ehtesham Haque and Huan Liu. 'Subspace Clustering for High Dimensional Data: A Review'. In: *SIGKDD Explorations* 6.1 (2004), pp. 90–105. DOI: 10.1145/1007730.1007731 (pages 7, 24).

[PL07] Nam Hun Park and Won Suk Lee. 'Grid-based Subspace Clustering over Data Streams'. In: *CIKM*. ACM, 2007, pp. 801–810. DOI: 10.1145/1321440.1321551 (pages 17, 95).

[PLD05] Hanchuan Peng, Fuhui Long and Chris H. Q. Ding. 'Feature Selection Based on Mutual Information: Criteria of Max-Dependency, Max-Relevance, and Min-Redundancy'. In: *IEEE Trans. Pattern Anal. Mach. Intell.* 27.8 (2005), pp. 1226–1238. DOI: 10.1109/TPAMI.2005.159 (pages 7, 11).

[PLL07] Dragoljub Pokrajac, Aleksandar Lazarevic and Longin Jan Latecki. 'Incremental Local Outlier Detection for Data Streams'. In: *CIDM*. IEEE, 2007, pp. 504–515. DOI: 10.1109/CIDM.2007.368917 (page 95).

[Pro+02] Cecilia Magdalena Procopiuc, Michael Jones, Pankaj K. Agarwal and T. M. Murali. 'A Monte Carlo Algorithm for Fast Projective Clustering'. In: *SIGMOD Conference*. ACM, 2002, pp. 418–427. DOI: 10.1145/564691.564739 (page 17).

[R C19] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria, 2019. URL: https://www.R-project.org/ (page 40).

[Ram+17] Sergio Ramírez-Gallego, Bartosz Krawczyk, Salvador García, Michal Wozniak and Francisco Herrera. 'A survey on Data Preprocessing for Data Stream Mining: Current status and future directions'. In: *Neurocomputing* 239 (2017), pp. 39–57. DOI: 10.1016/j.neucom.2017.01.078 (page 4).

[Rei+16] Denis Moreira dos Reis, Peter A. Flach, Stan Matwin and Gustavo E. A. P. A. Batista. 'Fast Unsupervised Online Drift Detection Using Incremental Kolmogorov-Smirnov Test'. In: *KDD*. ACM, 2016, pp. 1545–1554. DOI: 10.1145/2939672.2939836 (page 22).

[Rén59] Alfréd Rényi. 'On measures of dependence'. In: *Acta Mathematica Academiae Scientiarum Hungaricae* 10.3-4 (1959), pp. 441–451. DOI: 10.1007/BF02024507 (page 12).

[Res+11] David N. Reshef, Yakir A. Reshef, Hilary K. Finucane, Sharon R. Grossman, Gilean McVean, Peter J. Turnbaugh, Eric S. Lander, Michael Mitzenmacher and Pardis C. Sabeti. 'Detecting Novel Associations in Large Data Sets'. In: *Science* 334.6062 (2011), pp. 1518–1524. ISSN: 0036-8075. DOI: 10.1126/science.1205438 (pages 12, 21, 46).

[RK17] Vishnu Raj and Sheetal Kalyani. 'Taming Non-stationary Bandits: A Bayesian Approach'. In: (2017). arXiv: 1707.09727 [stat.ML] (pages 78, 133).

[Rom+16] Simone Romano, Oussama Chelly, Vinh Nguyen, James Bailey and Michael E. Houle. 'Measuring Dependency via Intrinsic Dimensionality'. In: *ICPR*. IEEE Computer Society, 2016, pp. 1207–1212. DOI: 10.1109/ICPR.2016.7899801 (pages 21, 134).

[RRS00] Sridhar Ramaswamy, Rajeev Rastogi and Kyuseok Shim. 'Efficient Algorithms for Mining Outliers from Large Data Sets'. In: *SIGMOD Conference*. ACM, 2000, pp. 427–438. DOI: 10.1145/342009.335437 (page 24).

[RS12] Attila Reiss and Didier Stricker. 'Introducing a New Benchmarked Dataset for Activity Monitoring'. In: *ISWC*. IEEE Computer Society, 2012, pp. 108–109. DOI: 10.1109/ISWC.2012.13 (page 97).

[Ruf+19] Lukas Ruff, Yury Zemlyanskiy, Robert A. Vandermeulen, Thomas Schnake and Marius Kloft. 'Self-Attentive, Multi-Context One-Class Classification for Unsupervised Anomaly Detection on Text'. In: *ACL (1)*. Association for Computational Linguistics, 2019, pp. 4061–4071. DOI: 10.18653/v1/p19-1398 (pages 24, 114, 133).

[SA16] Saket Sathe and Charu C. Aggarwal. 'Subspace Outlier Detection in Linear Time with Randomized Hashing'. In: *ICDM*. IEEE Computer Society, 2016, pp. 459–468. DOI: 10.1109/ICDM.2016.0057 (pages 4, 24, 99, 114, 134).

[SA18] Saket Sathe and Charu C. Aggarwal. 'Subspace histograms for outlier detection in linear time'. In: *Knowl. Inf. Syst.* 56.3 (2018), pp. 691–715. DOI: 10.1007/s10115-017-1148-8 (pages 24, 97, 99, 134).

[SB18] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning – An Introduction*. 2nd ed. Adaptive Computation and Machine Learning series. MIT Press, 2018. ISBN: 978-0262039246 (pages 67, 78, 133).

[SC88] S. Siegel and N.J. Castellan. *Nonparametric Statistics for the Behavioral Sciences*. 2nd ed. McGraw-Hill international editions. Statistics series. McGraw-Hill, 1988. ISBN: 9780070573574 (page 37).

[Sch+15] Erich Schubert, Alexander Koos, Tobias Emrich, Andreas Züfle, Klaus Arthur Schmid and Arthur Zimek. 'A Framework for Clustering Uncertain Data'. In: *PVLDB* 8.12 (2015), pp. 1976–1979. DOI: 10.14778/2824032.2824115 (pages 99, 114).

[Sch+16] Oliver Schulte, Zhensong Qian, Arthur E. Kirkpatrick, Xiaoqian Yin and Yan Sun. 'Fast learning of relational dependency networks'. In: *Mach. Learn.* 103.3 (2016), pp. 377–406. DOI: 10.1007/s10994-016-5557-9 (page 130).

[Sel+14] Aleka Seliniotaki, George Tzagkarakis, Vassilis Christophides and Panagiotis Tsakalides. 'Stream Correlation Monitoring for Uncertainty-Aware Data Processing Systems'. In: *IISA*. IEEE Computer Society, 2014, pp. 342–347. DOI: 10.1109/IISA.2014.6878827 (page 22).

[SG18] Erich Schubert and Michael Gertz. 'Numerically Stable Parallel Computation of (Co-)Variance'. In: *SSDBM*. ACM, 2018, 10:1–10:12. DOI: 10.1145/3221269.3223036 (page 22).

[Sha+18] Jingbo Shang, Jialu Liu, Meng Jiang, Xiang Ren, Clare R. Voss and Jiawei Han. 'Automated Phrase Mining from Massive Text Corpora'. In: *IEEE Trans. Knowl. Data Eng.* 30.10 (2018), pp. 1825–1837. DOI: 10.1109/TKDE.2018.2812203 (pages 108, 115).

[Sha48] Claude E. Shannon. 'A Mathematical Theory of Communication'. In: *Bell Syst. Tech. J.* 27.3 (1948), pp. 379–423. DOI: 10.1002/j.1538-7305.1948.tb01338.x (pages 7, 11).

[Sil+13] Jonathan A. Silva, Elaine R. Faria, Rodrigo C. Barros, Eduardo R. Hruschka, André C. P. L. F. de Carvalho and João Gama. 'Data Stream Clustering: A Survey'. In: *ACM Comput. Surv.* 46.1 (2013), 13:1–13:31. DOI: 10.1145/2522968.2522981 (page 4).

[Spe04] C. Spearman. 'The Proof and Measurement of Association between Two Things'. In: *The American Journal of Psychology* 15.1 (1904), pp. 72–101. ISSN: 00029556. DOI: 10.2307/1422689 (page 21).

[SR09] Gábor J. Székely and Maria L. Rizzo. 'Brownian Distance Covariance'. In: *Ann. Appl. Stat.* 3.4 (2009), pp. 1236–1265. ISSN: 19326157. DOI: 10.1214/09-AOAS312 (page 21).

[SR18] Mahsa Salehi and Lida Rashidi. 'A Survey on Anomaly detection in Evolving Data: [with Application to Forest Fire Risk Prediction]'. In: *SIGKDD Explorations* 20.1 (2018), pp. 13–23. DOI: 10.1145/3229329.3229332 (page 4).

[SRL14] Vaibhav Srivastava, Paul Reverdy and Naomi E. Leonard. 'Surveillance in an Abruptly Changing World via Multiarmed Bandits'. In: *CDC*. IEEE Computer Society, 2014, pp. 692–697. DOI: 10.1109/CDC.2014.7039462 (page 23).

[SS07] Friedrich Schmid and Rafael Schmidt. 'Multivariate extensions of Spearman's rho and related statistics'. In: *Statistics & Probability Letters* 77.4 (2007), pp. 407–416. ISSN: 0167-7152. DOI: https://doi.org/10.1016/j.spl.2006.08.007 (pages 21, 51, 134).

[Ste20] Georg Steinbuss. 'Calibration and evaluation of outlier detection with generated data'. PhD thesis. Karlsruhe Institute of Technology, 2020. DOI: 10.5445/IR/1000120534 (page ii).

[SU08] Aleksandrs Slivkins and Eli Upfal. 'Adapting to a Changing Environment: the Brownian Restless Bandits'. In: *COLT*. Omnipress, 2008, pp. 343–354. URL: http://colt2008.cs.helsinki.fi/papers/45-Slivkins.pdf (page 23).

[Tao+16] Fangbo Tao, Honglei Zhuang, Chi Wang Yu, Qi Wang, Taylor Cassidy, Lance M. Kaplan, Clare R. Voss and Jiawei Han. 'Multi-Dimensional, Phrase-Based Summarization in Text Cubes'. In: *IEEE Data Eng. Bull.* 39.3 (2016), pp. 74–84. URL: http://sites.computer.org/debull/A16sept/p74.pdf (page 108).

[Tat+12] Andrada Tatu, Fabian Maass, Ines Färber, Enrico Bertini, Tobias Schreck, Thomas Seidl and Daniel Keim. 'Subspace Search and Visualization to Make Sense of Alternative Clusterings in High-Dimensional Data'. In: *IEEE VAST*. IEEE Computer Society, 2012, pp. 63–72. DOI: 10.1109/VAST.2012.6400488 (pages 7, 17).

[TB19] Holger Trittenbach and Klemens Böhm. 'Dimension-based subspace search for outlier detection'. In: *Int. J. Data Sci. Anal.* 7.2 (2019), pp. 87–101. DOI: 10.1007/s41060-018-0137-7 (pages 5, 7, 17, 24, 87–89, 95, 97, 133).

[TB99] Michael E. Tipping and Christopher M. Bishop. 'Probabilistic Principal Component Analysis'. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 61.3 (1999), pp. 611–622. DOI: 10.1111/1467-9868.00196 (page 24).

[Tho33] William R. Thompson. 'On the Likelihood that One Unknown Probability Exceeds Another in View of the Evidence of Two Samples'. In: *Biometrika* 25.3/4 (1933), pp. 285–294. ISSN: 00063444. DOI: 10.2307/2332286 (pages 9, 22, 65, 99, 102, 135).

[Tim+14] Nicholas Timme, Wesley Alford, Benjamin Flecker and John M. Beggs. 'Synergy, redundancy, and multivariate information measures: an experimentalist's perspective'. In: *Journal of Computational Neuroscience* 36.2 (2014), pp. 119–140. DOI: 10.1007/s10827-013-0458-4 (page 21).

[Tor03] Kari Torkkola. 'Feature Extraction by Non-Parametric Mutual Information Maximization'. In: *J. Mach. Learn. Res.* 3 (2003), pp. 1415–1438. URL: http://jmlr.org/papers/v3/torkkola03a.html (page 7).

[Tra+10] Long Tran-Thanh, Archie C. Chapman, Enrique Munoz de Cote, Alex Rogers and Nicholas R. Jennings. 'Epsilon-First Policies for Budget-Limited Multi-Armed Bandits'. In: *AAAI*. AAAI Press, 2010. URL: http://www.aaai.org/ocs/index.php/AAAI/AAAI10/paper/view/1817 (page 23).

[Tri20] Holger Trittenbach. 'User-Centric Active Learning for Outlier Detection'. PhD thesis. Karlsruhe Institute of Technology, Germany, 2020. DOI: 10.5445/IR/1000117443 (page ii).

[TSK19] Pang-Ning Tan, Michael S. Steinbach and Vipin Kumar. *Introduction to Data Mining*. 2nd ed. Pearson, 2019. ISBN: 9780134545943 (page 4).

[TTL11] Swee Chuan Tan, Kai Ming Ting and Fei Tony Liu. 'Fast Anomaly Detection for Streaming Data'. In: *IJCAI*. IJCAI/AAAI, 2011, pp. 1511–1516. DOI: 10.5591/978-1-57735-516-8/IJCAI11-254 (page 4).

[UNK10] Taishi Uchiya, Atsuyoshi Nakamura and Mineichi Kudo. 'Algorithms for Adversarial Bandit Problems with Multiple Plays'. In: *ALT*. Vol. 6331. Lecture Notes in Computer Science. Springer, 2010, pp. 375–389. DOI: 10.1007/978-3-642-16108-7_30 (pages 23, 70, 78, 94, 133).

[Van+12] Andrei Vanea, Emmanuel Müller, Fabian Keller and Klemens Böhm. 'Instant Selection of High Contrast Projections in Multi-Dimensional Data Streams'. In: *Proceedings of the Workshop on Instant Interactive Data Mining (IID 2012) in conjunction with ECML PKDD*. 2012 (page 24).

[VB19] Michael Vollmer and Klemens Böhm. 'Iterative Estimation of Mutual Information with Error Bounds'. In: *EDBT*. OpenProceedings.org, 2019, pp. 73–84. DOI: 10.5441/002/edbt.2019.08 (pages 22, 61, 129).

[Vol+19a] Michael Vollmer, Adrian Englhardt, Holger Trittenbach, Pawel Bielski, Shahab Karrari and Klemens Böhm. 'Energy Time-Series Features for Emerging Applications on the Basis of Human-Readable Machine Descriptions'. In: *e-Energy*. ACM, 2019, pp. 474–481. DOI: 10.1145/3307772.3331022 (page 7).

[Vol+19b] Michael Vollmer, Lukasz Golab, Klemens Böhm and Divesh Srivastava. 'Informative Summarization of Numeric Data'. In: *SSDBM*. ACM, 2019, pp. 97–108. DOI: 10.1145/3335783.3335797 (page 7).

[Vol20] Michael Vollmer. 'Time-Efficient Analysis of Complex Dependencies'. PhD thesis. Karlsruhe Institute of Technology, 2020. DOI: 10.5445/IR/1000105784 (page ii).

[VRB18] Michael Vollmer, Ignaz Rutter and Klemens Böhm. 'On Complexity and Efficiency of Mutual Information Estimation on Static and Dynamic Data'. In: *EDBT*. OpenProceedings.org, 2018, pp. 49–60. DOI: 10.5441/002/edbt.2018.06 (page 22).

[Wan+17] Yisen Wang, Simone Romano, Vinh Nguyen, James Bailey, Xingjun Ma and Shu-Tao Xia. 'Unbiased Multivariate Correlation Analysis'. In: *AAAI*. AAAI Press, 2017, pp. 2754–2760. URL: http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14214 (pages 4, 21, 24, 88, 89, 95, 135).

[Wat60] Michael Satosi Watanabe. 'Information Theoretical Analysis of Multivariate Correlation'. In: *IBM Journal of Research and Development* 4.1 (1960), pp. 66–82. DOI: 10.1147/rd.41.0066 (pages 21, 51, 135).

[WLH12] Xindong Wu, Pei-Pei Li and Xuegang Hu. 'Learning from Concept Drifting Data Streams with Unlabeled Data'. In: *Neurocomputing* 92 (2012), pp. 145–155. DOI: 10.1016/j.neucom.2011.08.041 (page 4).

[Xia+16] Yingce Xia, Tao Qin, Weidong Ma, Nenghai Yu and Tie-Yan Liu. 'Budgeted Multi-Armed Bandits with Multiple Plays'. In: *IJCAI*. IJCAI/AAAI Press, 2016, pp. 2210–2216. URL: http://www.ijcai.org/Abstract/16/315 (page 23).

[Yan+16] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alexander J. Smola and Eduard H. Hovy. 'Hierarchical Attention Networks for Document Classification'. In: *NAACL HLT*. Association for Computational Linguistics, 2016, pp. 1480–1489. DOI: 10.18653/v1/n16-1174 (pages 114, 133).

[YH09] Xiaotong Yuan and Bao-Gang Hu. 'Robust Feature Extraction via Information Theoretic Learning'. In: *ICML*. Vol. 382. ACM, 2009, pp. 1193–1200. DOI: 10.1145/1553374.1553526 (page 7).

[YZ06] Chunyu Yang and Jie Zhou. 'HClustream: A Novel Approach for Clustering Evolving Heterogeneous Data Stream'. In: *ICDM Workshops*. IEEE Computer Society, 2006, pp. 682–688. DOI: 10.1109/ICDMW.2006.89 (pages 6, 11).

[ZGW08] Ji Zhang, Qigang Gao and Hai H. Wang. 'SPOT: A System for Detecting Projected Outliers From High-dimensional Data Streams'. In: *ICDE*. IEEE Computer Society, 2008, pp. 1628–1631. DOI: 10.1109/ICDE.2008.4497638 (pages 4, 17, 23, 24).

[ZH19] Chao Zhang and Jiawei Han. *Multidimensional Mining of Massive Text Data*. Synthesis Lectures on Data Mining and Knowledge Discovery. Morgan & Claypool Publishers, 2019. DOI: 10.2200/S00903ED1V01Y201902DMK017 (page 108).

[Zho+16] Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao and Bo Xu. 'Attention-Based Bidirectional Long Short-Term Memory Networks for Relation Classification'. In: *ACL (2)*. Association for Computational Linguistics, 2016. DOI: 10.18653/v1/p16-2034 (page 25).

[Zho17] Zhi-Hua Zhou. 'A brief introduction to weakly supervised learning'. In: *National Science Review* 5.1 (Aug. 2017), pp. 44–53. ISSN: 2095-5138. DOI: 10.1093/nsr/nwx106 (page 130).

[Zhu+17] Honglei Zhuang, Chi Wang, Fangbo Tao, Lance M. Kaplan and Jiawei Han. 'Identifying Semantically Deviating Outlier Documents'. In: *EMNLP*. Association for Computational Linguistics, 2017, pp. 2748–2757. DOI: 10.18653/v1/d17-1291 (pages 24, 112, 114, 135).

[Zim03] Donald W. Zimmerman. 'A Warning About the Large-Sample Wilcoxon-Mann-Whitney Test'. In: *Understanding Statistics* 2.4 (2003), pp. 267–280. DOI: 10.1207/S15328031US0204_03 (page 37).

[ZLW07] Qi Zhang, Jinze Liu and Wei Wang. 'Incremental Subspace Clustering over Multiple Data Streams'. In: *ICDM*. IEEE Computer Society, 2007, pp. 727–732. DOI: 10.1109/ICDM.2007.100 (pages 17, 24, 95).

[ZS02] Yunyue Zhu and Dennis E. Shasha. 'StatStream: Statistical Monitoring of Thousands of Data Streams in Real Time'. In: *VLDB*. Morgan Kaufmann, 2002, pp. 358–369. DOI: 10.1016/B978-155860869-6/50039-1 (pages 7, 11, 22).

[ZSK12] Arthur Zimek, Erich Schubert and Hans-Peter Kriegel. 'A Survey on Unsupervised Outlier Detection in High-Dimensional Numerical Data'. In: *Statistical Analysis and Data Mining* 5.5 (2012), pp. 363–387. DOI: 10.1002/sam.11161 (pages 4, 5, 24).

[ZX08] Wenjun Zhou and Hui Xiong. 'Volatile Correlation Computation: A Checkpoint View'. In: *KDD*. ACM, 2008, pp. 848–856. DOI: 10.1145/1401890.1401991 (page 22).