



UNIL | Université de Lausanne

Unicentre

CH-1015 Lausanne

<http://serval.unil.ch>

---

*Year : 2023*

## Advances in Interpretable Machine Learning : Applications, Analytics, Robustness

Ajalloeian Ahmad

Ajalloeian Ahmad, 2023, Advances in Interpretable Machine Learning : Applications,  
Analytics, Robustness

Originally published at : Thesis, University of Lausanne

Posted at the University of Lausanne Open Archive <http://serval.unil.ch>

Document URN : urn:nbn:ch:serval-BIB\_BD485D7880371

### **Droits d'auteur**

L'Université de Lausanne attire expressément l'attention des utilisateurs sur le fait que tous les documents publiés dans l'Archive SERVAL sont protégés par le droit d'auteur, conformément à la loi fédérale sur le droit d'auteur et les droits voisins (LDA). A ce titre, il est indispensable d'obtenir le consentement préalable de l'auteur et/ou de l'éditeur avant toute utilisation d'une oeuvre ou d'une partie d'une oeuvre ne relevant pas d'une utilisation à des fins personnelles au sens de la LDA (art. 19, al. 1 lettre a). A défaut, tout contrevenant s'expose aux sanctions prévues par cette loi. Nous déclinons toute responsabilité en la matière.

### **Copyright**

The University of Lausanne expressly draws the attention of users to the fact that all documents published in the SERVAL Archive are protected by copyright in accordance with federal law on copyright and similar rights (LDA). Accordingly it is indispensable to obtain prior consent from the author and/or publisher before any use of a work or part of a work for purposes other than personal use within the meaning of LDA (art. 19, para. 1 letter a). Failure to do so will expose offenders to the sanctions laid down by this law. We accept no liability in this respect.



UNIL | Université de Lausanne

---

FACULTÉ DES HAUTES ÉTUDES COMMERCIALES  
DÉPARTEMENT DES SYSTÈMES D'INFORMATION

**Advances in Interpretable Machine Learning:  
Applications, Analytics, Robustness**

THÈSE DE DOCTORAT

présentée à la

Faculté des Hautes Études Commerciales  
de l'Université de Lausanne

pour l'obtention du grade de  
Doctorat en systèmes d'information

par

Ahmad AJALLOEIAN

Directeur de thèse  
Prof. Michalis Vlachos

Jury

Prof. Boris Nikolov, Président  
Prof. Yash Raj Shrestha, expert interne  
Prof. Grigorios Tsoumakas, expert externe  
Dre Jasmina Bogojenska, experte externe  
Prof. Johannes Schneider, expert externe

LAUSANNE  
2023





UNIL | Université de Lausanne

---

FACULTÉ DES HAUTES ÉTUDES COMMERCIALES  
DÉPARTEMENT DES SYSTÈMES D'INFORMATION

**Advances in Interpretable Machine Learning:  
Applications, Analytics, Robustness**

THÈSE DE DOCTORAT

présentée à la

Faculté des Hautes Études Commerciales  
de l'Université de Lausanne

pour l'obtention du grade de  
Doctorat en systèmes d'information

par

Ahmad AJALLOEIAN

Directeur de thèse  
Prof. Michalis Vlachos

Jury

Prof. Boris Nikolov, Président  
Prof. Yash Raj Shrestha, expert interne  
Prof. Grigorios Tsoumakas, expert externe  
Dre Jasmina Bogojeska, experte externe  
Prof. Johannes Schneider, expert externe

LAUSANNE  
2023



# IMPRIMATUR

La Faculté des hautes études commerciales de l'Université de Lausanne autorise l'impression de la thèse de doctorat rédigée par

**Ahmad AJALLOEIAN**

intitulée

*Advances in Interpretable Machine Learning: Applications, Analytics, Robustness*

sans se prononcer sur les opinions exprimées dans cette thèse.

Lausanne, le 23.11.2023

Professeure Marianne Schmid Mast, Doyenne



# Thesis Committee

Prof. Michalis Vlachos  
Full Professor, HEC, University of Lausanne  
Thesis supervisor

Prof. Yash Raj Shrestha  
Assistant Professor, HEC, University of Lausanne  
Internal expert

Prof. Grigorios Tsoumakas  
Associate Professor, Aristotle University of Thessaloniki (AUTH)  
External expert

Dr. Jasmina Bogojeska  
AI Scientist, Senior Lecturer and Group Leader, Zurich University of Applied  
Sciences (ZHAW)  
External expert

Prof. Johannes Schneider  
Associate Professor, University of Liechtenstein  
External expert



University of Lausanne  
Faculty of Business and Economics

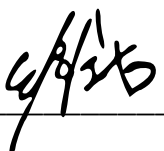
PhD in Information Systems

I hereby certify that I have examined the doctoral thesis of

**Ahmad AJALLOEIAN**

and have found it to meet the requirements for a doctoral thesis.

All revisions that I or committee members  
made during the doctoral colloquium  
have been addressed to my entire satisfaction.

Signature:  \_\_\_\_\_ Date: 31.10.2023

Prof. Michalis VLACHOS  
Thesis supervisor





University of Lausanne  
Faculty of Business and Economics


PhD in Information Systems

I hereby certify that I have examined the doctoral thesis of

**Ahmad AJALLOEIAN**

and have found it to meet the requirements for a doctoral thesis.

All revisions that I or committee members  
made during the doctoral colloquium  
have been addressed to my entire satisfaction.

Signature: \_\_\_\_\_  \_\_\_\_\_ Date: 27.10.2023

Prof. Yash Raj SHRESTHA  
Internal expert



University of Lausanne  
Faculty of Business and Economics

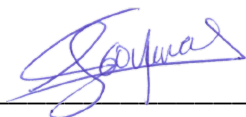
PhD in Information Systems

I hereby certify that I have examined the doctoral thesis of

**Ahmad AJALLOEIAN**

and have found it to meet the requirements for a doctoral thesis.

All revisions that I or committee members  
made during the doctoral colloquium  
have been addressed to my entire satisfaction.

Signature:  \_\_\_\_\_ Date: October 21, 2023

Prof. Grigorios TSOUMAKAS  
External expert



University of Lausanne  
Faculty of Business and Economics

PhD in Information Systems

I hereby certify that I have examined the doctoral thesis of

**Ahmad AJALLOEIAN**

and have found it to meet the requirements for a doctoral thesis.

All revisions that I or committee members  
made during the doctoral colloquium  
have been addressed to my entire satisfaction.

Signature: *Jasmina Bogojeska* Date: 24.10.2023

Dr. Jasmina BOGOJESKA  
External expert



University of Lausanne  
Faculty of Business and Economics

PhD in Information Systems

I hereby certify that I have examined the doctoral thesis of

**Ahmad AJALLOEIAN**

and have found it to meet the requirements for a doctoral thesis.

All revisions that I or committee members  
made during the doctoral colloquium  
have been addressed to my entire satisfaction.

Signature: Johannes Schneider Date: 22. 10. 23

Prof. Johannes SCHNEIDER  
External expert





# Acknowledgements

Thank God that I could finish my PhD thesis.

I express my sincere appreciation to Prof. Michalis Vlachos for his guidance and support throughout my Ph.D. journey and for the trust and the freedom he gave me in exploring research directions that I was interested in. I am truly grateful for his patience, especially during the most challenging moments of my Ph.D.

I extend my sincerest thanks to my thesis committee members—Jasmina Bogojeska, Johannes Schneider, Yash Raj Shrestha, and Grigorios Tsoumakas. Their insightful comments on the thesis draft and engaging discussions during the private defense session enriched the quality of this work.

Special acknowledgment to the administrative staff of DESI, Sarah Duplan, and Caroline Kleinheny, for their efficient handling of paperwork and administrative matters.

I would also like to express my gratitude to all my friends within the Iranian community in Lausanne, whose friendship was a blessing throughout my years away from home.

Most importantly, I express deep gratitude to my beloved parents. Their unwavering love and unconditional support have been the cornerstone of my journey. I am profoundly thankful, knowing that I can never fully repay the love and sacrifices they have made for me.



## Abstract

In recent years, there has been a notable surge in the adoption of complex and inherently opaque Machine Learning (ML) models across diverse domains and industries. This trend has extended to high-stakes sectors like healthcare, loan eligibility, hiring, criminal risk assessment, and self-driving vehicles. However, incorporating ML-driven decisions in such critical contexts bears profound implications for human lives. The opacity inherent in complex ML model outputs raises substantial concerns encompassing security, ethics, robustness, and comprehensibility from a scientific perspective. As complex models such as deep neural networks (DNN) become instrumental in automated decision-making directly affecting individuals, it becomes imperative to cultivate tools and methodologies that facilitate a comprehensive grasp of these models' functioning. This is essential to prevent undesirable outcomes, including propagating societal biases and engendering erroneous predictions. In response to these imperatives, the domain of **Interpretable Machine Learning (IML)** emerges, aiming to address the aforementioned concerns associated with ML models. It seeks to explain both the average behavior and specific predictions of ML models, equipping researchers and practitioners with insights into the complex mechanisms governing model predictions. In this thesis, we investigate interpretability methods both in the context of classical ML and for explaining the prediction of deep neural networks.

In this thesis, we begin by focusing on data dimensionality reduction techniques, which are vital for understanding large-scale high-dimensional datasets, and introduce MoDE, a novel technique that generates interpretable, low-dimensional visualizations for big datasets. Going beyond preserving inter-data point distances, MoDE also maintains correlations and objects' ordinal scores. This unique ability, preserving ordinal scores, enables MoDE to offer interpretable visualizations of high-dimensional data within reduced dimensions, enhancing data understanding. Furthermore, our comprehensive analysis and empirical assessments highlight MoDE's computational efficiency, confirming its practicality.

We proceed by examining explanation methods employed for interpreting deep neural networks (DNNs). Our analysis assesses how these methods align with established desired explanation properties. Through various evaluation protocols, we compare distinct explanation methods, revealing that no single method outperforms all others universally. Optimal selection depends on the task requiring the explanation. Additionally, we inspect common explanation methods' robustness against

adversarial attacks. Our findings demonstrate that explanation methods that are claimed to be robust can be manipulated by crafting attacks involving non-additive perturbations. Moreover, we propose a novel class of attacks using sparse perturbations to manipulate explanations, showcasing their potential to mitigate bias in models. These findings hold significant implications for enhancing the robustness of explanation methods.

Finally, as an application of IML, we have devised a recommender system utilizing graph neural networks (GNNs). This recommender system excels in recommendation accuracy while also promoting recommendation interpretability. By leveraging user interaction subgraphs, we present in-depth interpretations for recommendations, highlighting the key users and items that influenced specific suggestions. This endeavor is significant as explainability research is scarce in the area of recommender systems. Moreover, interpretability can potentially help business owners to improve recommender systems in their platforms.

To conclude, we provide a comprehensive discussion on the significance of explainability methods and strategies for enhancing our understanding of their mechanisms. We also discuss future directions to improve ML explanations. We believe that interpretability is a pivotal concern that will become more prominent in the years ahead. With this work, we strive to contribute to the ongoing journey towards a more transparent and accountable Machine Learning.

# Contents

|          |   |           |
|----------|---|-----------|
| <b>1</b> | <b>Introduction</b>   | <b>1</b>  |
| 1.1      | Interpretable Machine Learning (IML) . . . . .                      | 2         |
| 1.2      | Definitions of interpretability . . . . .                           | 3         |
| 1.2.1    | Interpretability from social sciences standpoint . . . . .          | 4         |
| 1.2.2    | Interpretability from the ML standpoint . . . . .                   | 5         |
| 1.3      | Importance of IML . . . . .   | 7         |
| 1.3.1    | Why is there a need for IML methods? . . . . .                      | 7         |
| 1.3.2    | Who needs IML methods? . . . . .                                    | 9         |
| 1.4      | Desired properties for IML methods . . . . .                        | 10        |
| 1.5      | Categories of IML methods . . . . .                                 | 11        |
| 1.5.1    | Inherently interpretable models . . . . .                           | 12        |
| 1.5.2    | Post-hoc interpretability . . . . .                                 | 12        |
| 1.6      | Contributions of the thesis . . . . .                               | 14        |
| 1.6.1    | Publications . . . . .  | 16        |
| <b>2</b> | <b>Interpretable Embedding and Visualization of Compressed Data</b> | <b>18</b> |
| 2.1      | Introduction . . . . .  | 18        |
| 2.1.1    | Notation . . . . .  | 21        |
| 2.2      | Related Work . . . . .  | 21        |
| 2.3      | Problem Formulation . . . . .                                       | 23        |
| 2.4      | Algorithm . . . . .   | 27        |
| 2.5      | Extension to $p > 2$ dimensions . . . . .                           | 30        |
| 2.6      | Experiments . . . . .   | 32        |
| 2.6.1    | Embedding Quality . . . . .   | 33        |
| 2.6.2    | Classification Accuracy . . . . .                                   | 36        |
| 2.6.3    | Additional experiments for MoDE . . . . .                           | 38        |
| 2.6.4    | Embedding in more than $p > 2$ dimensions . . . . .                 | 40        |
| 2.7      | Discussion . . . . .  | 43        |

|          |  |           |
|----------|--|-----------|
| 2.8      | Conclusion . . . . .   | 43        |
| <b>3</b> | <b>On Smoothed Explanations: Quality and Robustness</b>  | <b>45</b> |
| 3.1      | Introduction . . . . .   | 46        |
| 3.2      | Background . . . . .   | 49        |
|          | 3.2.1 Attacks to manipulate explanations . . . . .   | 50        |
|          | 3.2.2 Robustness of explanations . . . . .   | 51        |
| 3.3      | Evaluation of post-hoc approaches . . . . .  | 52        |
|          | 3.3.1 Quality of explanations of post-hoc approaches . . . . .   | 52        |
|          | 3.3.2 Robustness of explanations of post-hoc approaches . . . . .  | 57        |
| 3.4      | Evaluation of ad-hoc approaches . . . . .  | 60        |
|          | 3.4.1 Quality of explanations of ad-hoc approaches . . . . .   | 61        |
|          | 3.4.2 Robustness of explanations of ad-hoc approaches . . . . .  | 63        |
| 3.5      | Discussion . . . . .   | 64        |
| 3.6      | Conclusion . . . . .   | 66        |
| <b>4</b> | <b>Sparse Attacks for Manipulating Explanations in Deep Neural Network Models</b>  | <b>67</b> |
| 4.1      | Introduction . . . . .   | 68        |
| 4.2      | Background . . . . .   | 70        |
| 4.3      | Baseline sparse attacks . . . . .  | 72        |
|          | 4.3.1 Single-step attack . . . . .   | 72        |
|          | 4.3.2 PGD attack with $\ell_0$ box constraint . . . . .  | 73        |
| 4.4      | GrID attack: Greedy Increasing and Decreasing the perturbation density . . . . .   | 74        |
| 4.5      | Experiments . . . . .  | 78        |
|          | 4.5.1 Evaluating explanation manipulation and sparsity . . . . .   | 79        |
|          | 4.5.2 Ablation study . . . . .   | 83        |
|          | 4.5.3 Case study: fair-washing a gender-biased model . . . . .   | 85        |
| 4.6      | Discussion . . . . .   | 87        |
| 4.7      | Related work . . . . .   | 87        |
| 4.8      | Conclusion . . . . .   | 88        |
| <b>5</b> | <b>Using Neural and Graph Neural Recommender Systems to Overcome Choice Overload: Evidence from a Music Education Platform</b> | <b>90</b> |
| 5.1      | Introduction . . . . .   | 91        |
| 5.2      | The Tomplay app . . . . .  | 93        |
| 5.3      | Data characteristics . . . . .   | 94        |
| 5.4      | Neural Networks with Entity Resolution . . . . .   | 97        |

---

|          |  |            |
|----------|--|------------|
| 5.4.1    | Architecture of the Neural Network model . . . . . | 100        |
| 5.4.2    | Training the model . . . . .                       | 101        |
| 5.5      | Using Graph Neural Networks . . . . .              | 101        |
| 5.5.1    | Architecture of the GNN model . . . . .            | 102        |
| 5.5.2    | Preprocessing the graph . . . . .                  | 104        |
| 5.5.3    | Handling new users . . . . .                       | 105        |
| 5.6      | Results . . . . .                                  | 107        |
| 5.6.1    | Accuracy of predictions . . . . .                  | 107        |
| 5.6.2    | Examples of recommendations . . . . .              | 110        |
| 5.6.3    | Tuning the GNN recommender system . . . . .        | 112        |
| 5.6.4    | Cold start evaluation . . . . .                    | 114        |
| 5.6.5    | Explaining the recommendations . . . . .           | 118        |
| 5.7      | Discussion . . . . .                               | 121        |
| 5.8      | Related Work . . . . .                             | 121        |
| 5.9      | Conclusion . . . . .                               | 124        |
| <b>6</b> | <b>Conclusion</b>                                  | <b>126</b> |
| 6.1      | Review of main contributions . . . . .             | 127        |
| 6.2      | Future directions . . . . .                        | 128        |
|          | <b>Bibliography</b>                                | <b>130</b> |





# Chapter 1

## Introduction

In the field of machine learning research, the terms "interpretability" and "explainability"<sup>1</sup> have gained popularity over the past few years. However, these concepts are by no means novel; they have deep-rooted origins that date back to ancient times. Philosophers throughout history have always sought to answer existential questions, seeking explanations for the reasons behind events. Similarly, social scientists and psychologists have devoted years to understanding how humans communicate explanations to one another.

Today, we aim to apply these fundamental ideas to mathematical models, making them not only intelligent and accurate but also understandable to non-experts. This becomes especially important when these models are used in critical areas, where machine decisions can greatly impact human lives. Under the concept of Explainable Artificial Intelligence (XAI), our ultimate goal is to comprehend better how complex machine learning models operate. This has significant implications for **security**, **ethics**, **robustness**, and **scientific understanding**.

By striving to bridge the gap between complex machine learning models and human interpretability, we seek to create a more transparent and responsible AI environment. This fosters trust in AI systems, empowers users to make informed decisions, and ensures accountability when deploying these models for crucial tasks. Ultimately, our work in XAI can revolutionize how we interact with and rely on artificial intelligence in various domains, leading to greater clarity and ethical use.

---

<sup>1</sup>Throughout this thesis we use the terms "interpretability" and "explainability" interchangeably.

## 1.1 Interpretable Machine Learning (IML)

As machine learning (ML) increasingly becomes indispensable for automation and decision-making, its applications span various scientific domains, including finance & banking [1, 2, 3], ecology [4, 5], medicine [6, 7], and the social sciences [8, 9]. For instance, Jurgovsky et al. [3] employed the LSTM (a type of recurrent neural network) to detect credit card fraud and find that LSTM can enhance detection accuracy. While ML focuses on optimizing loss functions for unseen data and often achieves superior performance, its inherent complexity poses challenges to interpretability.

In contrast to ML's emphasis on predictive performance, classical statistical modeling prioritizes insights into the data-generating process. These considerations of the data-generating process are what make the statistical modeling approach often more interpretable: model parameters are usually connected to a concept of the data-generating process. For instance, coefficients in linear regression can be directly mapped to individual features or the terminal node of a decision tree gives us a list of binary decisions that lead to a certain prediction, both allowing for clear interpretations. However, as ML models, like neural networks and gradient-boosted trees, become more structurally flexible and optimized, understanding the relationships between model parameters and features becomes less straightforward.

With ML playing a vital role in regulated industries and science, interpretability becomes paramount for various reasons. Regulated sectors, such as banking and healthcare, require auditability to understand model operations and ensure fairness and accuracy. Moreover, ML models can unknowingly encode social biases such as gender bias [10] and learn non-causal relationships [11, 12], emphasizing the need for interpretability to detect such issues. For example, in the realm of law and justice, there have been cases where incorrect data fed into a black box model have made the model biased, leading to unfairly long prison sentences (e.g., a prisoner was denied parole due to an incorrect COMPAS score, [13]). ML has also been used increasingly in scientific endeavors; here interpretability becomes crucial as explaining phenomena lies at the heart of scientific exploration. All of the above-mentioned needs (auditability, fairness, knowledge generation, etc.) can be addressed by making the ML models more interpretable.

As we encounter more real-life scenarios involving ML, the issue of interpretability becomes more prominent, especially in high-stakes decision-making. Instances, where incorrect data inputs have led to unjust outcomes, underscore the importance of transparency in ML models. For instance, the healthcare sector also demands models that can be reasoned and understood by medical professionals to prioritize

patient treatment effectively. The U.S. Food and Drug Administration (FDA) now requires all clinical decision support systems based on ML to provide “the rationale or support for the recommendation” [14]. Consequently, calls for interpretable, human-understandable ML models have been growing, advocating for “opening the black box” to address these concerns [15, 16, 17, 18].

Interpretable Machine Learning (IML) emerges as a dedicated research field focused on extracting knowledge from ML models and explaining individual predictions. The IML field encompasses three main areas: inherently interpretable models, modifications to increase interpretability in complex models, and post-hoc interpretation methods. By advancing IML techniques, we can foster greater trust and accountability in ML models, ultimately paving the way for responsible and transparent AI implementations in diverse domains.

## 1.2 Definitions of interpretability

While explanations can be broadly understood as means to provide reasons, causes, or logical relationships for events, there exists no consensus on a formal definition. Extensive research in psychology, philosophy, and social sciences has explored this topic from various perspectives. Some authors view explanations as conveying information about causal history [19], while others define them as sequences of logical consequences or mechanisms responsible for specific phenomena [20]. However, within the context of artificial intelligence, a precise formal definition of explanation remains elusive [21, 22, 23].

Similarly, Explainable Artificial Intelligence lacks a universally accepted technical definition, often referring loosely to an explanatory agent that reveals the underlying decision-making causes to a human [23]. Although the notion of interpretability in machine learning dates back to the advent of expert systems in the 1970s, there still exists no formal and widely agreed-upon definition. Much of the recent work in this area is based on personal views of what constitutes a good explanation, largely overlooking the wealth of studies on human explanations in sociology, psychology, and philosophy. While interpretability is inherently subjective, disregarding the vast body of research on how humans define, communicate, and evaluate explanations would be counterproductive.

Therefore, we start by reviewing some insights from social sciences literature, so as to understand what people generally expect from explanations. Then we continue by providing a more rigorous definition of interpretability from the machine learning standpoint.

### 1.2.1 Interpretability from social sciences standpoint

In philosophical terms, explaining an event involves providing information about its causal history [19]. Explanations are both a product and a process, serving as answers to *why* questions and involving cognitive derivation. Philosophers and psychologists agree that explanations relate to causes [24, 25]<sup>2</sup>. However, there are also other important properties to consider when discussing explanations. Notably, many explanations are contrastive, addressing why event  $E_1$  occurred instead of event  $E_2$ . Explanations are selective, focusing on one or two significant causes. Additionally, explanations can be social, involving knowledge transfer through conversations. Understanding these attributes is vital in defining interpretability for machine learning, leading to AI systems that align better with human cognition and expectations.

Mita [26] provides more insight on different aspects of explanations from the social sciences standpoint:

- **Interpretable explanations.** In [27], interpretability is defined as the extent to which an observer can comprehend the reasons behind a particular decision. This implies that a concept used to explain a decision can be conveyed in various forms and complexities, tailored to different listeners' levels of understanding.
- **The reasons behind an explanation.** Seeking an explanation is a natural way for humans to understand the reasons behind an event. Moreover, explanations can serve the purpose of persuading someone that a particular decision is correct [28]. In such cases, the focus may shift from providing the true reason to emphasizing the persuasive aspect of the explanation.
- **The structure of explanations.** Different questions require different explanations. Aristotle's Four Causes model, one of the oldest in philosophy, suggests that understanding something involves identifying its causes: material and formal causes for its composition, and efficient and final causes for its origin and purpose. For example, in a statue, the material cause is the marble, the formal cause is its shape, the efficient cause is the sculptor's scalpel, and the final cause is the sculptor's intention. Each of these causes can be a valid explanation for answering *why* questions.
- **How people explain behavior.** Research on social attribution, studying how people explain behavior to others, underpins much of the work on explanations,

---

<sup>2</sup>In this thesis we solely focus on causal explanations, that is, those that answer *why* questions.

including AI explainability. [29] showed that humans attribute folk psychological concepts to objects, perceiving intentions in their movements. [30] later proposed a framework distinguishing between intentional and unintentional behaviors in explanations, considering mental states and desires for intentional actions. Social norms and morals also shape explanations, as individuals may include personal judgments when explaining immoral behaviors or actions against accepted norms. Incorporating these aspects into AI explanations remains challenging due to the complexity of human behavior and social beliefs.

### 1.2.2 Interpretability from the ML standpoint

The main criticism that IML faces is that the term "interpretability" is not well defined [21, 22]. Miller [23] attempts to define interpretability as the degree to which a human can understand the cause of a decision. However, this definition raises the challenge of precisely defining "human understanding," making it still somewhat vague. Another proposed definition based on simulatability suggests that interpretability is the degree to which a human can consistently predict the model's result [31]. Although this definition allows for the quantification of interpretability, it still has limitations. For instance, certain measures of feature importance might not directly predict the model's results but still offer valuable insights into its workings.

Conducting research on IML becomes complex when the field lacks an accurate mathematical definition of interpretability. When using interpretability as a measurable quantity, the absence of ground truth for comparison makes it challenging to benchmark new approaches against existing ones in terms of interpretability. Without a proper reference point, proving one approach to be more interpretable than another becomes difficult. This issue significantly impacts the scientific evaluation of IML methods, emphasizing the need for a clear and consistent definition of interpretability for meaningful progress in the field.

There is no single conclusive, mathematical definition of interpretability for ML models. Previous works [32, 33] argued that interpretability can be viewed from multiple aspects rather than relying on a single definition. These measurable aspects can be categorized into **human-based evaluations** on proxy tasks or real applications, and function-level or **mathematical evaluations** [22].

Human-based evaluations often use proxy tasks due to the difficulty of real application evaluation. They include measuring how interpretations enhance a human's task performance [34, 35, 36] to reproduce model outputs [37, 38], and to predict changes in predictions given feature modifications [38]. User response

times, answer confidence, and the ability to detect model errors are also evaluated. These assessments demonstrate the diverse nature of interpretability evaluation and highlight the influence of the target audience’s prior knowledge on the outcomes. Research in social sciences [23] tells us that many properties that constitute a good explanation are in conflict with each other. This is another argument that even for humans no single best-performing explanation exists, but interpretability is more of a multi-objective problem.

Function-level evaluations, on the other hand, are derived analytically from the model itself. Some examples are examining sparsity [39], linearity, and monotonicity in feature effects. Model size, such as the number of non-zero coefficients or length of decision rules, is another example that is a model class-dependent criterion for interpretability [40, 41, 42]. Other measures include fidelity [43], which assesses how well an explanation predicts the model outcome. These quantifiable dimensions of interpretability enable model optimization not only for predictive performance but also for interpretability. For example, optimizing for sparsity in used features, monotonicity in feature effects, and sparsity in explanations allows for a trade-off between interpretability and performance.

### **limitations of popular definitions**

Recent works have proposed different definitions for interpretable machine learning (IML). For instance, Ribeiro et al. [12] defines explaining a prediction as:

*Presenting textual or visual artifacts that provide a qualitative understanding of the relationship between the instance’s components and the model’s prediction, [...] as a solution to the trusting a prediction problem.*

Several similar definitions have been provided in the literature [22, 44, 45, 46]. Such definitions often have several drawbacks in common:

1. **Vagueness:** They lack clarity and specificity, making them challenging to apply consistently.
2. **Specific to Explanation Type:** Certain definitions are limited to particular explanation formats, such as textual or visual, overlooking other potential alternatives.
3. **Subjective Nature of ”Understanding”:** Introducing the concept of ”understanding” ties interpretability to the individual seeking the explanation, leading to subjective interpretations.

4. Subset of Desired Properties: Some definitions focus solely on specific properties like informativeness and trust, disregarding other essential aspects of interpretability.

## 1.3 Importance of IML

In this section, we delve deeper into the significance of interpretable machine learning (IML) methods by addressing two main questions: 1) Why is there a need for IML methods? 2) Who needs IML methods?

### 1.3.1 Why is there a need for IML methods?

Recent research [47] highlights that the necessity to explain decisions arises due to either 1) incomplete knowledge of the data or 2) incomplete understanding of the training process. When models are trained on data gathered from human activities (e.g., professional records, social-network interactions, purchases, movements, etc.), there is a concern about capturing cultural stereotypes, historical prejudices, or individual biases embedded in the data during the training process. On the other hand, even if data quality requirements are met, the resulting model might not behave as expected on edge cases that are not adequately represented in the training data. Addressing these questions is at the core of the interpretable machine learning (IML) investigation.

In the following, we explore the need for interpretability from four different standpoints: ethics, safety, regulations, and scientific discovery. In each of these areas, the ability to understand and explain model decisions becomes paramount to ensure fairness, reliability, compliance, and meaningful insights.

#### **Ethics**

Fairness and unbiasedness are essential in machine learning systems, especially when their decisions have real-life consequences, such as loan approvals, job applications, or the risk of crime reiteration. Discrimination based on sensitive attributes like race, religion, or gender must be avoided, aligning with legislation that prohibits such biases [48]. The challenge arises when historical datasets, containing inherent prejudices, are used blindly in the training process. If models are optimized purely for predictive accuracy, they may unknowingly capture and perpetuate these biases in the form of discriminative rules.



Even if the model's predictions do not directly affect individuals, there can be negative outcomes. For instance, Google faced criticism when its image classification algorithm mislabeled people of color as "gorillas" due to the under-representation of certain samples in the training data [49]. Similarly, word embeddings, common representations of text data as numerical vectors, have shown gender stereotypes [50], which can affect applications like Google Translate [51].

Data de-biasing is an active research field, but transparent models can aid in detecting and correcting biases. By highlighting the training data patterns, transparent models provide insights into where biases originate, helping researchers and developers address these issues effectively.

### **Safety**

Transparency in machine learning models is crucial for identifying and rectifying errors before deploying them in safety-critical domains. The lack of transparency is a major barrier to adopting ML models in several applications such as the medical domain, where there is a demand for models that are both well-performing and trustworthy and explainable [52, 53].

Artifacts in the training data can cause the models to be biased. Such models can often be well-performing on the training data, however, they may make erroneous predictions on the more generalized test data. For instance, [12] demonstrated that a classifier trained to distinguish between photos of wolves and Eskimo dogs achieved high accuracy by focusing solely on the background of the pictures, labeling samples with snow as "wolf," regardless of the animal's actual features. Similarly, [54] discussed a model that learned to classify images based on a watermark rather than the subject itself. In these examples, the model has learned to make a correct prediction based on incorrect reasons. Understanding the features the model relies on can help researchers anticipate potential failures before deploying the model.

### **Regulations**

The implementation of more stringent regulations, such as the General Data Protection Regulation (GDPR) [55] in the European Union, highlights the growing importance of interpretable machine learning. The concept of the "right to explanation" [56] included in the GDPR emphasizes the need for transparency and accountability in machine learning systems, aiming to protect user rights in the era of machine learning dominance. Interpretable machine learning research plays a crucial role in achieving these objectives.

## Scientific understanding

Machine learning is not limited to data science but is gaining prominence across various research domains. As scientists integrate machine learning to optimize and enhance their research outcomes, the need for explainability becomes essential to ensure the scientific validity of these results [57]. In natural sciences, leveraging vast datasets from experiments and observations offers the potential for profound scientific insights and discoveries. However, to extract meaningful knowledge, having explainable data and outcomes is crucial, and this is possible via leveraging domain knowledge and model transparency.

### 1.3.2 Who needs IML methods?

After discussing the key justifications for interpretable machine learning methods, it is essential to identify the diverse stakeholders potentially interested in IML approaches. Understanding these stakeholders helps us grasp the demand for IML methods, enabling us to tailor these techniques more effectively to their specific needs and requirements. By considering the interests and perspectives of various stakeholders, we can enhance the applicability and relevance of IML methods in real-world scenarios. [58] mentions five different stakeholders interested in IML methods, namely: creators, operators, executors, decision subjects and examiners.

Ensuring precise control over model performance is undeniably crucial for model **creators**, yet challenges can still arise despite achieving satisfactory test results. For instance, predictions might be influenced by non-causal artifacts, as discussed in section 1.3.1. These artifacts might remain undetected during performance evaluation, but become apparent when using IML methods. By employing IML methods, model creators gain the power to effectively debug their models and compare feature importance values against expert knowledge. **Operators** work with the model's output, while **executors** act based on the model's predictions. Lack of proper model interpretability can complicate communication between operators and executors. This may lead to operators using their own interpretations to convey information, potentially resulting in unintended consequences due to inaccuracies. The individuals affected by the model's decisions are referred to as **decision subjects**. These individuals could be individuals whose loan applications were rejected or who were diagnosed by a data-driven algorithm with a disease. The decisions made by the model can have varying impacts on their lives, ranging from negligible to significant. Interpretable models can empower decision subjects to understand the rationale behind the decisions and make informed choices. Moreover, interpretability can be crucial for challenging the model decisions. **Examiners** are responsible for testing,

auditing, or investigating the model's performance. In regulated industries like finance and healthcare, algorithms often require thorough auditing. ML interpretation methods serve as important tools in an examiner's toolbox.

## 1.4 Desired properties for IML methods

Recent research on interpretable machine learning has identified several properties that are desirable for an interpretable explanation. In the domain of machine learning, we seek explanations that can answer the question, "why does the model give that output?" However, this question can lead to various side-goals depending on the specific task and application area, which ultimately determine the true reason why an explanation is needed. In the following, we will briefly introduce some of the properties that have been identified as desirable for interpretable machine learning methods in the existing literature.

- **Informativeness.** Interpretable machine learning aims to enhance the information obtained from a machine learning model's output. By interpreting the internal mechanisms of the model or the output itself, we can gain additional insights that can assist human decision makers in making better decisions. The goal of IML is to make this "additional information" more understandable for the explainee and the specific task at hand [26].
- **Trustworthiness.** Interpretability and trust in machine learning models are closely related concepts [12, 21]. Trustworthiness refers to the confidence that a model will perform as expected, but it does not guarantee interpretability. Even a perfect model that consistently delivers accurate results may not be completely trustworthy, especially in scenarios where humans also make mistakes. Trust and interpretability are intertwined, but distinct aspects, when evaluating machine learning models.
- **Fairness.** Interpretability is seen as a practical approach to enhance fairness in machine learning models. Since data often reflects societal biases, machine learning algorithms can inadvertently amplify these biases, especially in critical applications where decisions directly impact people's lives and ethical considerations are paramount. The "right to explanation" outlined in the GDPR mandates that explanations provided by models should not only ensure fairness but should also be verifiable and open to challenge. This requirement arises in situations involving personal data usage, making it crucial to address fairness concerns and ensure transparency in decision-making processes.

Furthermore, another important property for an explanation is that it should present consistent quality among different subgroups of the population [59]. For instance, it should not be the case that explanations associated with instances belonging to women, are less accurate than those associated with other genders.

- **Fidelity.** In the context of interpretable machine learning, fidelity refers to how well an interpretable model or explanation approximates the behavior of the original, often complex, black-box model it is meant to explain. It assesses the accuracy and faithfulness of the explanation in capturing the decision-making process of the original model for different inputs. A faithful explanation should provide insights into how the black-box model arrives at its predictions, highlighting the key features and patterns considered by the model. This helps users understand the reasoning behind the model's decisions and builds trust in its predictions. In Chapter 3 we explore this property in more detail for a specific class of explanation methods.
- **Robustness.** In the context of interpretable machine learning, robustness refers to the stability of the explanations across different variations or perturbations in the input data or model. A robust interpretability method should produce reliable and consistent explanations even when there are slight changes in the data or model configuration. Recent research has demonstrated attacks to craft adversarial examples against explanations [60, 61], and also defenses to make interpretability methods more robust [62]. The robustness of interpretability methods ensures that the explanations are not overly sensitive to minor changes, thus providing users with more confidence in the reliability and validity of the explanations. Robust interpretability methods can withstand variations and uncertainties in the data and model, making them more suitable for real-world applications where the data may be noisy or subject to fluctuations. In Chapters 3 and 4 we will discuss the robustness of the explanations of deep neural networks in more detail.

## 1.5 Categories of IML methods

In the literature, a clear distinction is made between two types of machine learning models based on their interpretability. The first type is referred to as "inherently interpretable models", which are designed in a way that their internal workings and decision-making process are transparent and easy to understand. These models

are explicitly constructed to be interpretable, and their structure allows users to directly interpret the reasons behind their decisions without the need for additional techniques. Linear/Logistic Regression, Decision Trees, and Naive Bayes classifier are some examples of such models.

On the other hand, the second type is known as "post-hoc interpretability" and it applies to black-box or opaque models, such as neural networks. These models are inherently complex and understanding the meaning of their parameters or how they reach specific decisions is not straightforward. To make such models interpretable, external techniques and tools are applied after the model has been trained. These post-hoc interpretability methods aim to shed light on the model's decision-making process and provide insights into how it arrives at certain outcomes.

### 1.5.1 Inherently interpretable models

In the field of inherently interpretable models, not all models offer the same level of interpretability. According to [26], three different levels of transparency are established for these models: simulatability, decomposability, and algorithmic transparency. Each level encompasses the next one in the mentioned order, with simulatability being the highest level of interpretability. However, since inherently interpretable models are not the main focus of this thesis, we will not delve further into this topic. Readers interested in exploring this subject in depth are encouraged to refer to the main work [26] for more comprehensive information on these different levels of transparency in inherently interpretable models.

### 1.5.2 Post-hoc interpretability

As mentioned earlier, black-box models require external techniques to be interpreted. We refer to such techniques as post-hoc interpretability methods. These methods can be classified according to three different levels.

- **Local vs. global.** The difference between local and global explanations lies in the scope of the interpretability provided. Local explanations focus on understanding the model's predictions for individual instances or data points. They aim to provide insights into why a specific prediction was made by the model for a particular input. Local explanations are instance-specific and help to understand the model's decision-making process for a single data point. Techniques like LIME [12] and SHAP [63] or gradient-based explanations like Integrated Gradients [64] are commonly used to generate local explanations. Global explanations, on the other hand, aim to provide a broader understanding

of the model's behavior across the entire dataset or a significant portion of it. They offer insights into the overall patterns, trends, and feature importance that the model considers when making predictions on the entire dataset. Global explanations help users understand the general behavior of the model and identify which features are most influential in its decision-making process. An example of global explanations are rule-based models that use a series of if-then rules to represent the decision boundaries of the original model.

- **Model-specific vs model-agnostic.** Model-specific explanations are tailored to a particular machine learning model. They are designed to provide insights into how a specific model makes predictions by leveraging the internal mechanisms and parameters of that model. Gradient-based explanations like Integrated Gradients [64] and DeepLIFT [65] are examples of such explanations. Model-agnostic explanations, on the other hand, are applicable to a wide range of machine learning models, regardless of their architecture or complexity. They do not rely on the internal details of a specific model and are based on the input-output behavior of the model. Model-agnostic explanations use techniques that are independent of the model's structure, making them more versatile and applicable to various models, including black-box models like deep neural networks. Techniques like LIME [12] and SHAP [63] are among these types of explanations.
- **Explanation type.** There are 4 types of explanations based on how they present and convey information [26]. 1) Text Explanations: Text explanations provide explanations in textual format. This includes models that generate captions from images, where the generated text explains the content of the image. 2) Visual Explanations: Visual explanations aim to explain what a model has learned by using visualizations. Some methods use dimensionality reduction techniques to represent input observations in two or three dimensions, allowing for visual exploration and understanding of the data. Methods from the computer vision community highlight the specific portions of an image or video that the target model focuses on to generate its output, providing insights into the model's decision process. 3) Explanations by Example: Explanations by example rely on the assumption that a model should behave similarly when processing similar input observations. In this approach, the decision of a model is explained by using one or more similar examples or proxies, helping to understand the reasoning behind the model's predictions. 4) Feature Relevance Methods: Feature relevance methods clarify the inner mechanisms of a model by computing a relevance score for each input feature. The higher

the score, the greater the impact of that feature on the model’s output. These methods help to identify the most influential features in the decision-making process and provide insights into the model’s behavior.

Note that each of these explanation types serves a specific purpose and can be useful in different scenarios, depending on the nature of the data, the complexity of the model, and the interpretability requirements of the task at hand.

## 1.6 Contributions of the thesis

In response to the remarkable successes of Machine Learning models like deep neural networks and recognizing the paramount significance of interpretability methods highlighted in preceding sections, this thesis undertakes an exhaustive exploration of diverse aspects within the area of interpretable machine learning (IML). Organized into three parts, each segment casts light on different aspects of IML, encompassing theoretical advancements and practical applications.

The overarching theme unifying these segments is the pursuit of interpretability at various stages within the machine learning pipeline, aiming to construct a machine learning pipeline that not only delivers high predictive accuracy but also maintains transparency and trustworthiness. The journey begins with a deep dive into the sphere of interpretability within data dimensionality reduction techniques which are essential for comprehending big high-dimensional datasets. Addressing the common challenge of non-interpretable embedding spaces resulting from these techniques, we propose a novel approach that generates interpretable, low-dimensional visualizations for big datasets.

The exploration continues by inspecting post-hoc attribution methods utilized to explain predictions from deep neural networks. In this part, we thoroughly evaluate several attribution methods in terms of satisfying established properties desired for explanations. Our results offer interesting insights into the task-specific efficacy of attribution methods. We also investigate the robustness of these methods against adversarial attacks, revealing novel attack strategies that challenge robust techniques previously introduced in the literature. These findings hold important implications for developing more trustworthy explanations.

In the final part, we demonstrate a practical application of IML in the recommender systems domain. We develop a graph neural network-based recommender system that not only excels in recommendation accuracy but also promotes interpretability. By Seamlessly merging these two attributes, the system provides transparent insights into recommendation rationales.

These three parts are interconnected by their shared mission of bringing interpretability into various stages of the machine learning pipeline. Each addresses the interpretability challenge in its own unique context, contributing to the broader goal of enhancing transparency, trustworthiness, and user understanding in machine learning models and applications. In the following, we present the main contributions in each part in more detail:

**Interpretable Embedding and Visualization of Compressed Data.** Dimensionality reduction techniques convert high-dimensional data into lower dimensions to speed up machine learning operations while preserving original data relationships. Common methods include MDS, ISOMAP, LLE, t-SNE, spectral embedding, and UMAP. These techniques rely on precise distance information between data pairs. We introduce MoDE, an efficient embedding technique capable of handling inexact distance estimates between object pairs using lower and upper bounds. This adaptability is especially useful for large datasets subjected to lossy compression. MoDE effectively captures multiple aspects of data relationships—correlations, distances, and object importance rankings—with remarkable fidelity. Notably, by preserving orders of object importance, if available in the dataset, MoDE enhances the interpretability of the resulting embedding. Additionally, our theoretical analysis establishes the linear convergence of MoDE, providing robust evidence of its computational efficiency.

**On Smoothed Explanations: Quality and Robustness.** Recent works have proposed several attribution methods based on computing the gradient of the output with respect to input to explain deep neural network predictions. However, these explanations are vulnerable to adversarial perturbations that shift the focus of the explanation towards irrelevant features. In response to such attacks, various defense mechanisms have been proposed which include making the decision boundary of neural networks smoother. However, there has not been a thorough study in the IML literature attempting to assess the robustness of these approaches against different types of attacks. We present attacks based on combining additive and non-additive perturbations and show that this class of attacks is indeed a threat to the smoothed explanations. Additionally, through various quality tests, we assess the quality of these explanations in terms of sensitivity to model parameters, discriminating the features related to the predicted class, fidelity to the predictor function, and conciseness (sparsity). Our finding reveals no single smoothed explanation method is universally superior to the rest and the optimal selection of a method depends on the task requiring the explanation.

**Sparse Attacks for Manipulating Explanations in Deep Neural Network Models.** This chapter delves into the utilization of sparse perturbations for ex-



planation manipulation in deep neural networks. Sparse attacks aim to modify a minimal set of input features. However, due to the NP-hard nature of the corresponding  $\ell_0$  minimization problem, finding such perturbations is challenging. We propose GrID, an innovative algorithm for efficiently computing sparse perturbations that alter explanations while preserving model predictions. GrID showcases enhanced effectiveness in manipulating explanations with fewer feature perturbations compared to the PGD with  $\ell_0$  projection method. Empirical results on image and tabular datasets underscore the efficacy of our approach. These insights indicate how realistic perturbations can erode trustworthiness in systems based on deep neural networks.

**Using Neural and Graph Neural Recommender Systems to Overcome Choice Overload: Evidence from a Music Education Platform.** In this chapter, we explore the advantages of integrating recommendation technologies into educational platforms, focusing on music education using Tomplay as a case study. Tomplay offers sheet music and audio recordings for learning music. We analyze interaction patterns on this educational platform, highlighting its distinct user behaviors compared to other recommendation datasets. Educational platforms like Tomplay exhibit sparse interactions, as users often concentrate on specific content for learning. To address data sparsity, we augment a neural network (NN) recommendation model with entity resolution. Additionally, we enhance this model by utilizing graph neural networks (GNNs), that provide superior predictive accuracy. In addition to accuracy, we show that by leveraging user interaction subgraphs, we can produce explanations for recommendations by highlighting the key users and items that influenced the recommendation. Interpretability can help business owners understand user behaviors on their platforms.

### 1.6.1 Publications

This thesis comprises six publication works accepted or under review in peer-reviewed scientific conferences and journals. Chapter 2 is based on our published work [66] and its extension [67]. Chapter 3 is based on our published work [68] in CIKM 2022 and chapter 4 is based on a work under review for the ICDM 2023 conference. Finally, chapter 5 is based on a published paper [69] and its extension which is under review for the ACM Transactions on Information Systems journal. A complete list of these papers is given below.

- **“An Interpretable Data Embedding under Uncertain Distance Information”**. Nikolaos M. Freris, Michalis Vlachos, and **Ahmad Ajalloeian**. *Proceedings of the 20th IEEE International Conference on Data Mining, ICDM*

2020.

- **“Interpretable Embedding and Visualization of Compressed Data”**. Nikolaos M. Freris, **Ahmad Ajalloeian**, and Michalis Vlachos. *ACM Transactions on Knowledge Discovery from Data*.
- **“On Smoothed Explanations: Quality and Robustness”**. **Ahmad Ajalloeian**, Seyed-Mohsen Moosavi-Dezfooli, Michalis Vlachos, and Pascal Frossard. *Proceedings of the 31st ACM International Conference on Information & Knowledge Management, CIKM 2022*.
- **“Sparse Attacks for Manipulating Explanations in Deep Neural Network Models”**. **Ahmad Ajalloeian**, Seyed-Mohsen Moosavi-Dezfooli, Michalis Vlachos, and Pascal Frossard. *Proceedings of the 23rd IEEE International Conference on Data Mining, ICDM 2023*
- **“A Case Study in Educational Recommenders: Recommending Music Partitures at Tomplay”**. **Ahmad Ajalloeian**, Michalis Vlachos, Johannes Schneider, and Alexis Steinmann. *Proceedings of the 31st ACM International Conference on Information & Knowledge Management, CIKM 2022*.
- **“Using Neural and Graph Neural Recommender Systems to Overcome Choice Overload: Evidence from a Music Education Platform”**. Hédi Razgallah, Michalis Vlachos, **Ahmad Ajalloeian**, Ninghao Liu, Johannes Schneider, and Alexis Steinmann. *Under revision for the ACM Transactions on Information Systems at the time of the submission of this thesis*.



# Chapter 2

## Interpretable Embedding and Visualization of Compressed Data

**Abstract.** Traditional embedding methodologies, also known as dimensionality reduction techniques, assume the availability of exact pairwise distances between the high-dimensional objects that will be embedded in a lower dimensionality. In this paper, we propose an embedding that overcomes this limitation and can operate on pairwise distances that are represented as a range of lower and upper bounds. Such bounds are typically estimated when objects are compressed in a lossy manner, so our approach is highly applicable in the case of big compressed datasets. Our methodology can preserve multiple aspects of the original data relationships: distances, correlations, and object scores/ranks, whereas existing techniques typically preserve only distances. Comparative experiments with prevalent embedding methodologies (ISOMAP, t-SNE, MDS, UMAP) illustrate that our approach can provide faithful preservation of multiple object relationships, even in the presence of inexact distance information. Our visualization method is also easily interpretable.

### 2.1 Introduction

Dimensionality reduction is instrumental for processing large volumes of data. Embedding algorithms map high-dimensional data into low-dimensional representations so as to alleviate the run-time of machine learning operations, while aiming to retain object relations between the original data. This work presents an efficient embedding, that is, the mapping of a high-dimensional dataset onto a Euclidean space while attempting to preserve not only distances, but also correlations and

object ranks/importances (if the last are available). First, we show how to create the 2-dimensional embedding, which can be used for data visualizing. Then, we generalize the method presented to project the original data on any embedding dimension.

The literature comprises a wide range of dimensionality reduction methods: Multi-Dimensional Scaling (MDS) [70], ISOMAP [71], Locally Linear Embedding (LLE) [72], t-distributed Stochastic Neighbor Embedding (t-SNE) [73], random projections [74], spectral embedding [75], and Uniform Manifold Approximation and Projection (UMAP) [76] are some of the most commonly used techniques. All of these typically operate using *exact* distance information between data pairs.

A distinct trait of the proposed embedding method is that it can accommodate *inexact* distance estimates between object pairs, as captured by lower and upper bounds. Specifically, our method has access only to distance *ranges* between pairs of objects. Such distance estimates can be derived, for example, when data are represented and compressed in a *lossy* manner using orthonormal transforms (Fourier, wavelets, etc.) [77], or compressed sensing methodologies [78]. In such scenarios, exact distances between pairs of objects are no longer computable, but upper/lower bounds can be estimated instead; see Fig. 2.1 for a visual illustration.

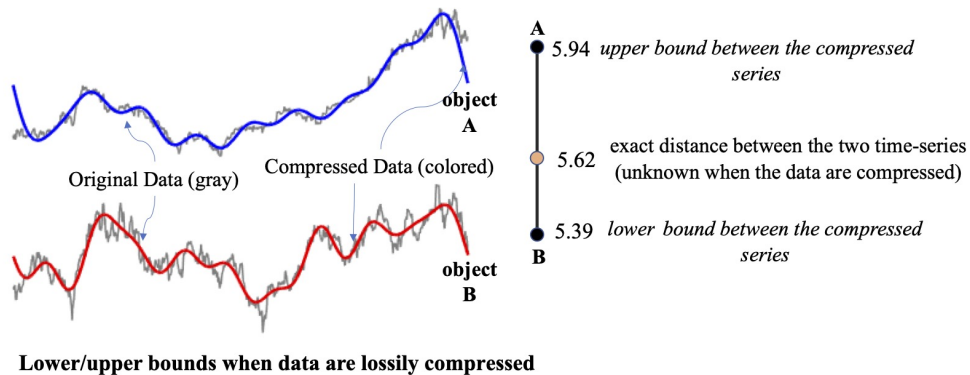


Figure 2.1: The proposed embedding does not require exact distances but only ranges, *i.e.*, lower/upper distance estimates. One case where such estimates are available is when operating on lossily compressed data (*e.g.*, Fourier, Wavelet, Chebyshev compression).

Our data embedding method, called *Multi-objective Data Embedding* (MoDE), can successfully capture, with high fidelity, multiple facets of the data relationships: correlations, distances, and orders or importance rankings. In particular, MoDE may serve as an effective big-data visualization tool in applications where the amount

of collected data is so large that it must be compressed in a lossy manner, in which case only bounds on distances can be deduced such as in [77, 79, 80, 81].

MoDE organizes the dataset as a graph, across the edges of which (*i.e.*, between pairs of data objects) lower/upper bounds on pairwise distances are obtained. The method leverages these bounds to cast the 2D embedding problem as a set of *linear inequalities* also exploiting the (partial) ordering of objects. It solves the inequality system in a least-squares sense to obtain the angular values of the embedded data points. That is, our objective function involves optimizing the angular values of the data points in 2D such that more important data points (data points with higher rank) are placed higher in the 2D visualization. Figure 2.2 shows an example of MoDE embeddings for a time series dataset of stocks. Thanks to the MoDE algorithm, we are able to incorporate the market capitalization of each stock (as importance ranking) in the process of learning the embeddings. As a result, the embedded data points will have a crescent shape in 2D where the more important data points are placed at a higher angular position.

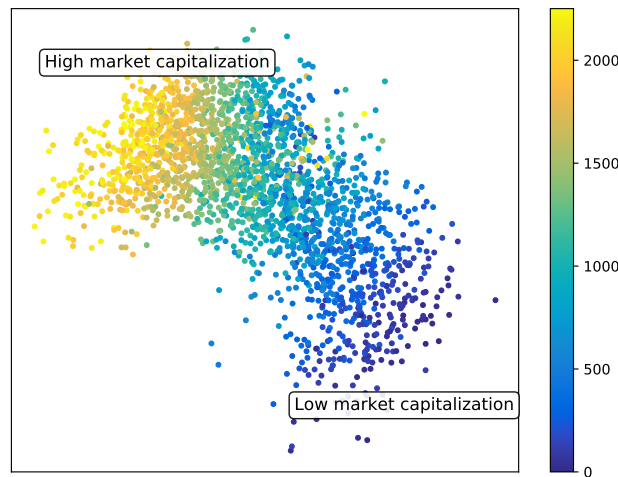


Figure 2.2: MoDE promotes more interpretable embeddings by placing more important data points at higher angular positions in the 2D space.

Our **contributions** enlist: (1) an iterative, anytime [82] embedding algorithm that operates on distance ranges; the anytime nature provides early embeddings even before complete execution of the algorithm; (2) the preservation of distance/correlation relations with competitive accuracy to the state-of-the-art; (3) the preservation of (partial) orders, that is, object importances, (if present in the dataset), thus promoting the interpretability of the embedding outcome (see Remark 1); and (4) a theoretical

analysis establishing linear convergence (*i.e.*, exponential decay of the approximation error with the number of iterations) despite a lack of strong convexity in the objective.

This work represents an extension of [66]. A significant extension of the present work is to provide a generalization of MoDE so that it embeds not only on 2-dimensions, but on *any number of dimensions*. This general embedding uses a recursive methodology that operates on the hyper-spherical coordinate representation. We provide related experiments about the performance of MoDE for an increasing number of embedding dimensions and compare it with several other embedding techniques. We also include experimental comparisons with UMAP [76]. Moreover, we provide experiments that highlight that MoDE is an efficient embedding even when object orders/scores are not present. Finally, the current manuscript has several additional figures to better explain the operation and the performance of the new embedding.

To the best of our knowledge, this is the first embedding approach that can accommodate **uncertain distance information** in the form of distance ranges.

### 2.1.1 Notation

For  $n \in \mathbb{Z}_+$ , we denote  $[n] := \{1, \dots, n\}$ . We use boldface lower-case variables for vectors (represented as column vectors) and denote matrices with upper-case variables. The standard inner product in  $\mathbb{C}^d$  is denoted by  $\langle \cdot, \cdot \rangle$ , whereas  $\|\cdot\|$  denotes the Euclidean norm. For a matrix  $A$ , we use  $\mathbf{a}_i$ ,  $A^\dagger$ ,  $\text{Range}(A)$ , and  $\|A\| = \sigma_{\max}(A)$  for the  $i$ -th column, Moore–Penrose pseudoinverse, range (column space), and maximum singular value, respectively. For  $\ell < u$ , we define  $(x)_\ell^u := \min(\max(x, \ell), u)$ , the projection of  $x$  to the interval  $[\ell, u]$ . We extend the notation for vectors, with  $\ell \leq \mathbf{u}$ ,  $[\ell, \mathbf{u}]$ , and  $(\mathbf{x})_\ell^{\mathbf{u}} := ((x_i)_{\ell_i}^{u_i})$  meant entry-wise.

## 2.2 Related Work

With the development of advanced data collection techniques, researchers and practitioners typically encounter large volumes of data which are often high-dimensional. Studying and analyzing such datasets is challenging and requires techniques that reduce the dimensionality of the data while being faithful to the original dataset as much as possible. A class of such techniques, termed linear dimensionality reduction, produce a low-dimensional linear mapping of the original high-dimensional data that preserves some features of interest in the data [83]. Notable examples

include Principal Component Analysis (PCA) [84] and Multi-Dimensional Scaling (MDS) [70]: PCA finds the directions of maximum variance, and MDS attempts to preserve the squared Euclidean distance between pairs of points. However, the high-dimensional data often reside on a low-dimensional manifold and linear dimensionality reduction methods cannot account for this intrinsic geometry [85]. For instance, the Euclidean distance used in MDS quantifies the distance between points in the high-dimensional space rather than the actual distance on the manifold, and therefore MDS has difficulties in inferring a faithful low-dimensional embedding of nonlinear data.

Non-linear dimensionality reduction techniques, also referred to as manifold learning, have the ability to account for the intrinsic geometry of the data [86]. These techniques typically use graphs to represent the manifold topology of the data, alongside with metrics such as *geodesic distance* to compute the distance between data points. ISOMAP [71], t-SNE [73], and UMAP [76] are examples of such techniques, that output low-dimensional representations that are usually most suitable to visualize the data in 2D or 3D. Apart from manifold learning methods, there are also other types of non-linear dimensionality reduction techniques. For instance, [87] uses deep generative networks to capture and visualize the low-dimensional structures in single-cell gene expression data by learning a probabilistic parametric mapping. As another example, kernel dimensionality reduction (KDR) techniques find a low-dimensional representation of the original data by optimizing kernel dependency measures that are capable of capturing non-linear relationships [88, 89]. Another class of dimensionality reduction techniques try to keep the meaning of the original data features by using feature selection techniques or selecting a linear combination of the features. For instance, [90] introduces a general approach to reduce the dimensionality of the data based on feature transformation or feature selection methods.

Our method, MoDE, is more similar to manifold learning methods because it organizes the data as a graph. Moreover, to offer more meaningful embeddings we leverage object scores or rankings that may be present in the dataset so as to accommodate better interpretability. Such data rankings/scores are typically present in many datasets and can often be identified by domain experts in different applications [91].

The performance of dimensionality reduction algorithms often degrades in the presence of noisy or compressed data. To face this challenge, the authors in [92] propose the Smooth Geodesic Embedding (GSE), which is similar to ISOMAP in the sense that it uses a geodesic metric to represent pairwise distances between points, but it can also accommodate the embedding of noisy data. In ISOMAP, the



geodesic distance between two points is defined as the shortest path in the data graph and hence it is generally piecewise linear. GSE replaces the piecewise linear ISOMAP geodesic by a smoothing spline and considers the length of the spline as the estimation of the manifold distance between points. In our method, however, we consider compressed data for which the exact pairwise distances cannot be computed and we only have access to upper and lower bounds on pairwise distances. In such a scenario, most dimensionality reduction techniques can only work with one pairwise distance matrix, for instance, we can use the average of lower and upper bounds on the pairwise distances. Our method, MoDE, is able to *directly* use upper and lower bounds on distances and provide low-dimensional embeddings with fidelitous preservation of distances, correlations and object ranks. In general, merits of MoDE include:

1. It can visualize object relationships even in the presence of in-exact distances (ranges of lower and upper bounds between pairs of objects), which would be the case when the objects are compressed, or collected from multiple sources (therefore one could use the *range* of values to represent each measurement).
2. MoDE has a favorable computational cost that allows it to scale to very large datasets, particularly when compared to very computationally demanding embedding methods such as MDS and ISOMAP.
3. MoDE also provides a very consistent and deterministic way of embedding data (typically leading to a form of a crescent moon on 2D), whereas many embedding techniques customarily result in different embedding outcomes across different runs. Having a consistent way of visualizing data works in favor of the end-user, who is not “surprised” by different embedding outcomes for the same data. We believe that consistent visualization and embedding significantly promotes higher interpretability.

## 2.3 Problem Formulation

In this section, we provide an embedding onto two dimensions. Later, we will generalize the approach to work on any projected dimensionality using the algorithms presented in this section.

**Objectives.** Given a dataset  $X \in \mathbb{C}^{d \times n}$ , where  $n$  is the number of data points and  $d$  their dimension, a primary goal of the proposed 2D embedding is to generate  $X_{2d} \in \mathbb{R}^{2 \times n}$  aiming to preserve (as accurately as possible) the distance structure, that is:

- **Objective 1:**  $\|\mathbf{x}_i - \mathbf{x}_j\| \approx \|\mathbf{x}_{2d,i} - \mathbf{x}_{2d,j}\|$ .

In our setting, *direct access to  $X$  is assumed unavailable*, and the only information accessible encompasses *ranges of distances (i.e., lower and upper bounds)* between a subset of object pairs. For example, this is the case when data are lossily compressed using orthonormal transforms or they are indexed, see Fig. 2.1. Nonetheless, norms of original points are assumed known—one scalar value per object is stored—and MoDE is *norm-preserving* in that it satisfies  $\|\mathbf{x}_i\| = \|\mathbf{x}_{2d,i}\|$  for all  $i \in [n]$ . We express the embedding in polar coordinates using  $\theta_i$  to denote the angle of  $\mathbf{x}_{2d,i}$ , which are the decision variables in our method (one scalar value per object).

In the following, we leverage pairwise *correlations* to obtain linear constraints on angular values  $\boldsymbol{\theta} \in \mathbb{R}^n$ . For simplicity, we assume for now that exact pairwise distances are known, and showcase how to utilize them for formulating the embedding problem as a linear system. Next, we illustrate how to incorporate bounds in obtaining a set of linear inequalities (i.e., a linear system with ranges in the entries of the measurement vector).

**Relative angle from correlation.** From the basic relation

$$\|\mathbf{x}_i - \mathbf{x}_j\|^2 = \|\mathbf{x}_i\|^2 + \|\mathbf{x}_j\|^2 - 2c_{\mathbf{x}_i, \mathbf{x}_j} \|\mathbf{x}_i\| \|\mathbf{x}_j\|, \quad (2.1)$$

where  $c_{\mathbf{x}_i, \mathbf{x}_j} := \frac{\text{Re}\langle \mathbf{x}_i, \mathbf{x}_j \rangle}{\|\mathbf{x}_i\| \|\mathbf{x}_j\|} \in [-1, 1]$  is the *correlation (coefficient)*, along with the fact that MoDE is norm-preserving, it becomes apparent that *distance preservation is equivalent to preservation of pairwise correlations*, that is,

- **Objective 2:**  $c_{\mathbf{x}_i, \mathbf{x}_j} \approx c_{\mathbf{x}_{2d,i}, \mathbf{x}_{2d,j}}$ .

Note that for a single pair  $(\mathbf{x}_i, \mathbf{x}_j)$ , objectives 1 & 2 can be achieved *perfectly* when  $c_{\mathbf{x}_i, \mathbf{x}_j}$  is known, by setting:

$$\theta_j - \theta_i = \pm \arccos(c_{\mathbf{x}_i, \mathbf{x}_j}), \quad (2.2)$$

where the sign indeterminacy is due to the fact that  $\cos(\cdot)$  is an even function. A distinctive attribute of MoDE, not present in other embedding techniques, is that it further seeks to *preserve an order* on data points.

**Definition 1.** (*Partial order*) A strict partial order is a binary relation  $\prec$  that is non-symmetric ( $x \prec y$  implies  $y \not\prec x$ ) and transitive ( $x \prec y$  and  $y \prec z$  imply  $x \prec z$ ). This generalizes the notion of (total) ordering because for  $x, y$  it may hold that neither  $x \prec y$  nor  $y \prec x$ , i.e., two points may not be comparable. A non-strict partial order  $\preceq$  differs in that it does not exclude both  $x \preceq y$  and  $y \preceq x$  for  $x \neq y$ . A set equipped with a partial order is called a partially ordered set (poset).

Given a (strict) partial order on  $[n]$ , the distinct objective of MoDE is to plot points so that:

- **Objective 3:**

$$i \prec j \implies \theta_i < \theta_j. \quad (2.3)$$

MoDE embodies this objective by means of adopting  $\theta_j - \theta_i = \arccos(c_{\mathbf{x}_i, \mathbf{x}_j}) \in [0, \pi]$ , when  $i \prec j$ . Our method organizes the dataset as a (directed) data graph  $G = (V, E)$ , where  $V$  is the set of vertices (points) and  $E$  is the set of edges. The ordered pair  $(i, j) \in E$  if and only if  $i \prec j$ . We let  $A \in \mathbb{R}^{m \times n}$  denote the incidence matrix ( $m = |E|, n = |V|$ ), where each row corresponds to a directed edge  $(i, j) \in E$  and takes values  $-1, 1$  at the  $i$ -th and  $j$ -th entry, respectively, and is zero elsewhere. Consequently, it follows that (2.2), (2.3) can be written compactly as a linear system:

$$\mathbf{y} = A\boldsymbol{\theta}, \quad (2.4)$$

where  $\mathbf{y} \in \mathbb{R}^m$  stacks the values  $\{\arccos(c_{\mathbf{x}_i, \mathbf{x}_j})\}_{(i,j) \in E}$  computed across the edges of the data graph  $G$ . The following remark discusses means of constructing the data graph, when not defined *a priori* alongside a strict partial order.

**Remark 1** (Ordering the dataset). A typical embedding will try to preserve only the pairwise distances in the lower dimensionality. However, in many applications, the notion of *importance ranking* of a data point is explicitly given by means of a *score function*. For example, in a recommender system application for movies, in addition to the movie similarities, it would be advantageous to be able to somehow highlight the importance of the movie. In this case, the average user rating can be used as a proxy for highlighting the importance of a movie, in addition to other features such as the director, the actors, etc. Similarly, when visualizing similarities in financial time-series data, one can use the market capitalization of a company to highlight its importance. The question that we address here is how to incorporate in our embedding this importance of a data point.

The score mechanism incurs a total *ordering* in  $V$ , that is, a non-strict partial order on  $V \times V$ . MoDE aims to plot more “important” points at higher angles (see (2.3)), thus yielding an *interpretable* embedding outcome. In practice, it may be beneficial to consider only a subset of relations, *e.g.*, based on the  $K$ -Nearest Neighbors (K-NN) of each point, for the sake of computational savings (this is the option we adopt in our experiments), which incurs a partial order. MoDE does not *require* a score per object, but if available it is used to create a more interpretable embedding. In the absence of scores, *random scores are used*. Our experiments reveal that preservation of the objectives of MoDE are not compromised in the absence of object score values.

**Operating on distance ranges.** When the original dataset is unavailable, exact distance/correlation information between a pair of points in the original space is lost. We restrict our attention to cases for which it is possible to infer *lower/upper bounds* on pairwise distances/correlations between original points. Moreover, note that one may further use these in constructing a K-NN graph (K-NNG), for example consider the graph incurred by the average of lower/upper distance bounds across points. This was the case for our experiments, where we assumed that only the compressed data are available.

Obtaining such lower and upper bounds can be regarded as a pre-processing step, during which, for every pair of points  $(i, j) \in E$ , one deduces a correlation uncertainty of the form  $c_{ij} \in [\underline{c}_{ij}, \bar{c}_{ij}]$ , where  $c_{ij} \equiv c_{\mathbf{x}_i, \mathbf{x}_j}$  is the correlation of original points  $\mathbf{x}_i, \mathbf{x}_j$  (this can be inferred from distance bounds immediately from (2.1), given knowledge of norms). These bounds directly yield lower/upper bounds of  $u_{ij}$  and  $\ell_{ij}$  on the angular difference  $\theta_j - \theta_i$ , namely  $u_{ij} := \arccos(\underline{c}_{ij})$  and  $\ell_{ij} := \arccos(\bar{c}_{ij})$ , in light of the fact that  $\arccos(\cdot) : [-1, 1] \rightarrow [0, \pi]$  is a decreasing function. To conclude, we stack the lower/upper bounds in vectors  $\boldsymbol{\ell} \leq \mathbf{u}$  with  $\boldsymbol{\ell}, \mathbf{u} \in [0, \pi]^m$  to capture the uncertainty in correlation via a linear system of inequalities on the angular domain:

$$\boldsymbol{\ell} \leq \mathbf{A}\boldsymbol{\theta} \leq \mathbf{u}. \quad (2.5)$$

Given that the embedding is from a high-dimensional space  $\mathbb{R}^d$  to the 2D plane, it is plausible that the inequality system is infeasible, *i.e.*,  $\{\boldsymbol{\theta} | \boldsymbol{\ell} \leq \mathbf{A}\boldsymbol{\theta} \leq \mathbf{u}\} = \emptyset$ . For this reason, we consider a solution in the least-squares sense, that is:

$$\underset{\boldsymbol{\theta}}{\text{minimize}} \quad f(\boldsymbol{\theta}) := \frac{1}{2} \|\mathbf{A}\boldsymbol{\theta} - (\mathbf{A}\boldsymbol{\theta})_{\boldsymbol{\ell}}^{\mathbf{u}}\|^2. \quad (2.6)$$

This is equivalent to computing  $\text{dist}(\text{Range}(\mathbf{A}); [\boldsymbol{\ell}, \mathbf{u}])$ , *i.e.*, the closest point to  $[\boldsymbol{\ell}, \mathbf{u}]$  that belongs in the range space (*i.e.*, the linear span of columns) of  $\mathbf{A}$ . In the next section, we devise an efficient algorithm to solve this problem.

We highlight that the scheme may as well operate on exact distances (or combinations of exact and inexact distances for object pairs) by letting  $\ell_{(i,j)} = u_{(i,j)} \equiv \arccos(c_{ij})$ . Additionally, when a lower (upper) bound is not available, it can simply be replaced with  $-\infty$  ( $+\infty$ ).

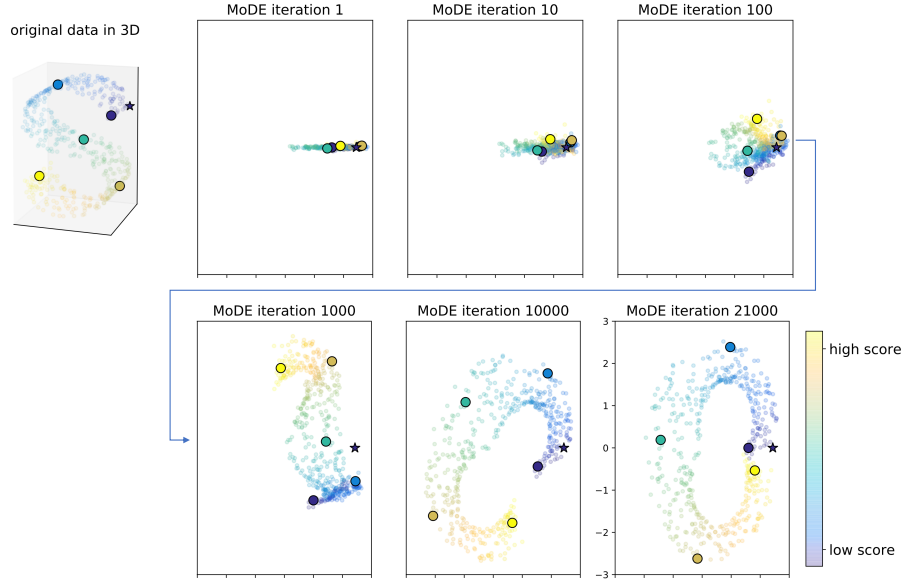


Figure 2.3: Progress of Algorithm 2 (DAE) across iterations. We ran MoDE on a sample of the S-shaped dataset with 500 points. We highlight 5 randomly selected points with a larger marker size and a darker color. The DAE algorithm optimizes the angular values of the data points such that *more important* points have larger angular values on 2D. In our implementation, the DAE algorithm does not optimize the angular value of the point that has the least importance (indicated by a star marker) and keeps the initial angular value of this point.

## 2.4 Algorithm

We invoke the gradient method for solving (2.6); see the Appendix for an analysis. The iterates ( $k \in \mathbb{Z}_+$ ) are:

$$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} - \gamma \mathbf{A}^\top \left( \mathbf{A} \boldsymbol{\theta}^{(k)} - (\mathbf{A} \boldsymbol{\theta}^{(k)})_{\ell}^{\mathbf{u}} \right). \quad (2.7)$$

The algorithmic description of MoDE is illustrated in Alg. 1. In Steps 1 & 2, respectively, the algorithm constructs a data graph  $G$  and computes the relevant lower/upper correlation bounds (in case the data graph is available, step 1 is omitted). In accordance with our experiments, we focus on the special case when the dataset is not organized a priori as a strict poset; in such a case, one may compute lower/upper distance bounds for each pair of points and use the average of these bounds to construct a K-NNG.

The direction of each edge is then determined based on a partial order, for example, score values, that also constitute an input to the algorithm. If no score

**Algorithm 1** Multi-objective Data Embedding (MoDE)

---

**Input:**  $X, \mathbf{s}, K$  ▷  $X$ : ‘inexact’ dataset;  $\mathbf{s}$ : scores;  $K$ : # NNs

- 1: Construct  $G$  (incidence matrix  $A$ )
- 2: Obtain  $\ell, \mathbf{u}$  in (2.5) ▷ Lower/Upper correlation bounds
- 3:  $\boldsymbol{\theta} \leftarrow DAE(A, \ell, \mathbf{u}, \epsilon)$  ▷ Compute angular values
- 4: **for**  $i = 1, \dots, n$  **do**
- 5:      $\mathbf{x}_{2d,i} \leftarrow (\|\mathbf{x}_i\| \cos \theta_i, \|\mathbf{x}_i\| \sin \theta_i)$  ▷ Embedding
- 6: **end for**
- 7: **Output:**  $X_{2d}$

---

**Algorithm 2** Distributed Angle Estimator (DAE)

---

**Input:**  $A, \ell, \mathbf{u}, \epsilon$  ▷  $A$ : incidence matrix;  $\ell, \mathbf{u}$ : lower/upper bounds;  $\epsilon$ : tolerance

- 1:  $\boldsymbol{\theta}^{(0)} \leftarrow \mathbf{0}; k \leftarrow 0$  ▷ Initialization
- 2:  $\mathbf{d} \leftarrow \text{diag}(A^\top A)$  ▷ Degrees
- 3:  $d_{\max} \leftarrow \max_{i \in [n]} d_i$  ▷ Maximum degree
- 4:  $\gamma \leftarrow \frac{1}{2d_{\max}}$  ▷ Step size
- 5: **repeat**
- 6:      $k \leftarrow k + 1$
- 7:     **for**  $i = 1, \dots, n$  **do** ▷ Updates
- 8:          $\phi_i^{(k+1)} \leftarrow \sum_{j \in \mathcal{N}_i^{in}} \left[ \theta_j^{(k)} + \left( \theta_i^{(k)} - \theta_j^{(k)} \right) \frac{u_{ij}}{\ell_{ij}} \right] +$
- 9:          $\sum_{j \in \mathcal{N}_i^{out}} \left[ \theta_j^{(k)} - \left( \theta_j^{(k)} - \theta_i^{(k)} \right) \frac{u_{ij}}{\ell_{ij}} \right]$
- 10:          $\theta_i^{(k+1)} \leftarrow (1 - \gamma d_i) \theta_i^{(k)} + \gamma \phi_i^{(k+1)}$
- 11:     **end for**
- 12: **until**  $\|\boldsymbol{\theta}^{(k+1)} - \boldsymbol{\theta}^{(k)}\| \leq \epsilon$  ▷ Termination criterion
- 13: **Output:**  $\boldsymbol{\theta}^{(k+1)}$

---

exists for the dataset objects, a random score assignment is used. Step 3 uses gradient descent (Alg. 2) to compute the angular values, while Steps 4–6 produce the (norm-preserving) embedding outcome.

The gradient method translates to Alg. 2, referred to as *Distributed Angle Estimator* (DAE). The update equations for each point’s angular value are *distributed* in the sense that a point uses solely angular values pertaining to its neighbors to update its own, see Steps 8-9 (we define  $\mathcal{N}_i^{in} := \{j | (j, i) \in E\}$ ,  $\mathcal{N}_i^{out} := \{j | (i, j) \in E\}$ , and  $d_i = |\mathcal{N}_i^{in}| + |\mathcal{N}_i^{out}|$  as the in-/out neighborhood, and (total) degree of point

$i$ , respectively). We deem this a favorable attribute of MoDE, in that updating a single point’s angular estimate does not require processing the entire dataset, but rather a subset of neighboring points. The algorithm terminates when the norm of the gradient of the objective in (2.6) falls below a given tolerance (Step 11; see proof of Theorem 2 in the appendix for the equivalence).

Fig. 2.3 shows the progress of Algorithm 2 (DAE) while optimizing the angular values of each data point in 2D. Note that, in practice, after the initialization step (step 1) in Algorithm 2, we remove the column corresponding to the lowest ranked point in the incidence matrix. Therefore, the angular value for the lowest ranked point will not be updated during the optimization and will be equal to the initial value (which is equal to zero). One can observe that in Fig. 2.3, the lowest ranked point, indicated by a star, is always placed at the same coordinates during the optimization. As the algorithm progresses through iterations, it optimizes the angular values (of the rest of the points) such that the points with higher ranks (the points with lighter colors) have larger angular values.

Moreover, because of its iterative implementation, MoDE is an **anytime algorithm** that provides an embedding of all the points (which progressively improves), even before the full execution of the algorithm, that is, when the termination criterion is reached. We provide an example of the anytime nature in the experiments.

The following theorem establishes linear convergence for Alg. 2. The proof is given in the appendix.

**Theorem 2** (Convergence of Alg. 2). *For tolerance  $\epsilon > 0$ , Algorithm 2 takes  $\mathcal{O}\left(d_{\max}^2 \eta^2 \log\left(\frac{1}{\epsilon}\right)\right)$  iterations and outputs an  $\epsilon'$ -optimal solution where  $\eta > 0$  is the regularity constant of  $\nabla f$ ,  $d_{\max}$  is the maximum degree of the graph, and  $\epsilon' := 2d_{\max} \cdot \eta \cdot \epsilon$ .*

In the following, we discuss the scalability of MoDE in terms of dataset size.

**Remark 3** (Computational Complexity). MoDE comprises four main steps. The cost of constructing the K-NN graph  $G$  which is  $\mathcal{O}(m) = \mathcal{O}(Kn)$  when data are already indexed, or  $\mathcal{O}(Kn + n \log n)$  when a  $K$ -NN data graph is constructed on-the-fly. The aforementioned complexity is typically shared by most embedding algorithms, as a standard pre-processing step. Then, the computation of lower/upper bounds over the edges of  $G$  is  $\mathcal{O}(m)$ . The cost of producing the embedding given angular values obtained from DAE (Steps 4-9) is  $\mathcal{O}(n)$ . We next discuss the complexity of DAE (Alg. 2). One iteration of Alg. 2 takes  $\mathcal{O}(m)$  computations. The expected number of iterations, for chosen tolerance, is given in Theorem 2. Characterizing the regularity constant with respect to singular values of  $A$  is a challenging task. A

notable exception is when  $\ell = \mathbf{u} \equiv \mathbf{y}$ , for example, when exact distance information is available. In such a case, problem (2.6) boils down to ordinary least-squares and Alg. 2 computes the min-norm LS solution  $\mathbf{A}^\dagger \mathbf{y}$ . Assume with no loss in generality a connected graph: it holds that  $\eta = \lambda_2^{-1}(\mathbf{L})$ , the smallest non-zero eigenvalue of the Laplacian also known as *Fiedler value*. This value quantifies the connectivity of the graph, and can be bounded by Cheeger’s inequality [93, Thm. 2.4]. Among  $k$ -regular graphs, Ramanujan graphs [93, Sec. 5.3] feature optimal expansion, *i.e.*,  $\lambda_2(G) = \Omega(k - \sqrt{k})$ , and the degree can be selected as  $k = \mathcal{O}(1)$  [93, Thm. 5.12]. For such a choice, Alg. 2 requires  $\mathcal{O}(n \log \frac{1}{\epsilon})$  operations to compute an  $\epsilon$ -optimal solution, that is, it features *linear* scalability with respect to dataset size.

## 2.5 Extension to $p > 2$ dimensions

In the prequel, we have presented a mechanism for embedding high-dimensional data to  $p = 2$  dimensions. In this section, we generalize MoDE to embed data in an arbitrary number of dimensions, by capitalizing on the *hyper-spherical* coordinate representation of  $\mathbb{R}^p$ .

The embedding is obtained *recursively*, that is, an embedding to  $3 \leq p' + 1 \leq p$  dimensions is obtained by applying DAE (see Alg. 2, which optimizes over only one scalar value per data point) after having obtained the embedding to  $p'$  dimensions. To illustrate how this works, let us assume that exact distance/correlation is available (that is, lower and upper bounds coincide, *i.e.*, this is the case for uncompressed data), an assumption that we shall relax in the end of this section. For such a case, denote the resulting correlation of a pair of embedded data points  $(i, j)$  by  $c_{ij}^{p'}$  and the true one by  $c_{ij}$ . The idea is to now compute the  $(p' + 1)$ -th angular coordinate by applying DAE to a certain function of the previously computed hyper-spherical coordinates along with the correlation error  $(c_{ij} - c_{ij}^{p'})$ , for each pair of points  $(i, j) \in E$ . In effect, this method successively incorporates the correlation in an increasing number of coordinates, thus yielding a progressively better embedding and allowing for trading-off accuracy (in terms of distance/correlation preservation) for data compression efficiency (the dimension of the data embedding). The technical motivation and implementation details follow.

Recall that we assume that the norms of the data points are known (and perfectly preserved by our mechanism), thus an embedding of an object to  $p$  coordinates is parameterized by its norm  $r$  and angular values  $(\phi_1, \phi_2, \dots, \phi_{p-1})$ . In addition, note that the correlation of two embedded points in  $p' + 1$  dimensions can also be



recursively computed using:

$$c_{ij}^{p'+1} = c_{ij}^{p'} + \prod_{k=1}^{p'-1} (\sin(\phi_k^i) \sin(\phi_k^j)) \cdot [\cos(\phi_{p'}^j - \phi_{p'}^i) - 1], \quad (2.8)$$

where  $\phi_k^i$  represents the  $k$ -th hyper-spherical coordinate of the  $i$ -th datapoint. This recursion is an immediate consequence of the transformation from hyper-spherical to Cartesian coordinates  $(x_1, x_2, \dots, x_p)$ , which is efficiently computed by:

$$\begin{aligned} x_1 &= r \cos(\phi_1) \\ x_2 &= r \sin(\phi_1) \cos(\phi_2) \\ x_3 &= r \sin(\phi_1) \sin(\phi_2) \cos(\phi_3) \\ &\vdots \\ x_{p-1} &= r \sin(\phi_1) \cdots \sin(\phi_{p-2}) \cos(\phi_{p-1}) \\ x_p &= r \sin(\phi_1) \cdots \sin(\phi_{p-2}) \sin(\phi_{p-1}) \end{aligned}$$

To conclude, the embedding to  $p' + 1$  coordinates can be obtained by targeting for  $c_{ij}^{p'+1} \approx c_{ij}$ , which in view of (2.8) boils down to a system of linear equations:

$$\phi_{p'}^j - \phi_{p'}^i = \pm \arccos \left( \left( 1 + \frac{c_{ij} - c_{ij}^{p'}}{\prod_{k=1}^{p'-1} (\sin(\phi_k^i) \sin(\phi_k^j))} \right)_{-1}^1 \right) \quad (2.9)$$

that is to be solved in a least-squares sense. We note two technical points in the above equation: 1) a truncation to  $[-1, 1]$  is needed to make for a meaningful problem, and 2) in order for (2.9) to be well-defined, none of the already computed hyper-spherical coordinates can be equal to zero (for the denominator to be non-zero); the latter can be ensured by simple shifting in view of the fact that each linear smoothing problem has (at least) one degree of freedom (since the null space contains the all-one vector). Once again, the signs can be selected aiming to preserve a partial order relation (in such a case,  $p - 1$  distinct such partial orders can be preserved for an embedding in  $\mathbb{R}^p$ ). The extension to lower/upper bounds is straightforward (by replacing  $c_{ij}$  with a range  $[\underline{c}_{ij}, \bar{c}_{ij}]$ ).

The algorithmic description of MoDE in  $p$  dimensions is described in Algorithm 3. Note that we need to run a single instance of DAE (Algorithm 2) for computing each of the angular coordinates. Therefore our method requires running  $p$  instances of DAE for computing embeddings in  $p + 1$  dimensions. Before running DAE for each angular coordinate, we need to obtain the lower and upper bound vectors in

(2.5). We call these vectors  $\ell_{p'}$ , and  $\mathbf{u}_{p'}$  (for  $p'$ -th angular coordinate), and they can be obtained by computing (2.9) for  $\bar{c}_{ij}$ , and  $\underline{c}_{ij}$  respectively. The proposed method requires running  $p$  instances of DAE, thus yielding a total complexity of  $\mathcal{O}(np)$  for embedding in  $p$  dimensions (for an expander data graph, see Remark 3).

---

**Algorithm 3** Multi-objective Data Embedding (MoDE) - extension to  $p$  dimensions

---

**Input:**  $X, \mathbf{s}, K, p \triangleright X$ : ‘inexact’ dataset;  $\mathbf{s}$ : scores;  $K$ : # NNs;  $p$ : dimensionality of the embeddings

- 1: Construct  $G$  (incidence matrix  $A$ )
  - 2: **for**  $k = 1, \dots, p$  **do**
  - 3:     Obtain  $\ell_k, \mathbf{u}_k$  in (2.9)
  - 4:      $\theta_k \leftarrow DAE(A, \ell_k, \mathbf{u}_k, \epsilon)$   $\triangleright$  Compute angular values
  - 5: **end for**
  - 6: **for**  $i = 1, \dots, n$  **do**
  - 7:     Compute  $\mathbf{x}_{pd,i}$  by transforming the hyper-spherical coordinates to Cartesian
  - 8: **end for**
  - 9: **Output:**  $X_{pd}$
- 

## 2.6 Experiments

We compare the quality of our embedding methodology with several widely-used embedding techniques, namely ISOMAP, MDS, t-SNE, and UMAP. The comparisons are in terms of:

1. Quality of embedding
2. Classification accuracy
3. Quality of embedding in  $p > 2$  dimensions

Moreover, for MoDE alone we highlight its ability to provide an accurate embedding even in the absence of object orders/scores in the dataset, its anytime nature, and its linear scalability. The code and datasets used are available at <https://github.com/ahmadajal/MoDE>. All experiments have been conducted on a 2.5 GHz 14-Core Intel Xeon W with 256 GB of RAM.

**Experimental setup.** Our method assumes as input lower and upper bounds on the Euclidean distance for a given pair of objects. Any methodology that provides

such bounds can be used. Here, we derive such bounds by lossily compressing time series using the approach of [77], which has been proven to compute optimally tight bounds. First, the data is compressed by keeping only a small subset of the *high energy coefficients* of an orthonormal data transform, *e.g.*, the Fourier transform. Consequently, tight bounds on the Euclidean distances are derived using the method described in [77]. Therefore, for each pair of objects, we do not compute the exact distance (as this information is lost in compression), but rather a lower bound  $\ell$  and an upper bound  $u$ . Our methodology uses both of these bounds, see (2.6). The techniques with which we compare our methodology assume an exact distance, and for those we use the mid-point  $\frac{1}{2}(\ell + u)$  as a surrogate. For techniques that work on the K-NN graph (all except MDS),  $K = 20$  was chosen. MDS, which is equivalent to *Principal Component Analysis* (PCA), is a global technique and uses all pairwise distances.

### 2.6.1 Embedding Quality

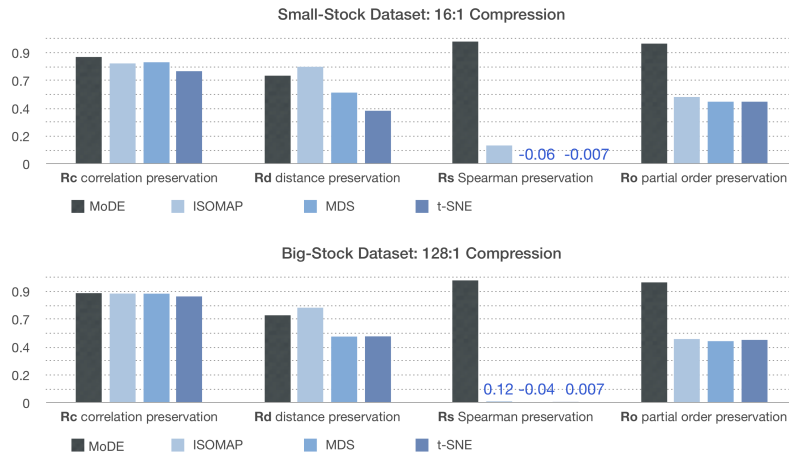


Figure 2.4: Comparison of MoDE with other embedding techniques. Our method lags behind ISOMAP only in distance preservation ( $R_d$ ), whereas it is superior in the preservation of correlations ( $R_c$ ). MoDE performs better than t-SNE and MDS across all evaluated metrics. The metrics  $R_o$  and  $R_s$  are depicted for other methods for the sake of completeness, because scores are not considered by techniques other than MoDE. For example, notice that the order preservation across each object’s neighborhood  $R_o$  is consistently at almost 100% for MoDE, while it is at around 50% (*i.e.*, random) for all other techniques.

**Comparison Metrics.** We evaluate the embedding quality on 2D of various techniques across a variety of metrics that highlight the quality in distance, correlation,

and score order preservation. We define and evaluate metrics on the K-NNG with respect to *original* distances on uncompressed data so as to simultaneously assess the impact of both compression and embedding on a lower-dimensional space (2D) on the retention of relations. We define the generic formula:

$$R := 1 - \frac{1}{m} \sum_{(i,j) \in E} C_{ij},$$

where  $C_{ij}$  is the cost of preservation accuracy on the pair  $(i, j) \in E$ . In all cases, a higher metric value implies a more accurate preservation, with 1 corresponding to perfect preservation. Specifically, we introduce:

1. The distance preservation metric  $R_d \in [0, 1]$ , by setting  $C_{ij} \equiv \frac{|d_{ij} - \hat{d}_{ij}|}{d_{ij} + \hat{d}_{ij}}$ , where  $d_{ij} := \|\mathbf{x}_i - \mathbf{x}_j\|$  denotes the original distance on the high-dimensional data, and  $\hat{d}_{ij} := \|\mathbf{x}_{2d,i} - \mathbf{x}_{2d,j}\|$  denotes the distance between corresponding embeddings in 2D.
2. The correlation preservation metric  $R_c \in [-1, 1]$ , by setting  $C_{ij} \equiv |c_{ij} - \hat{c}_{ij}|$ , where  $c_{ij} := c_{\mathbf{x}_i, \mathbf{x}_j}$ ,  $\hat{c}_{ij} := c_{\mathbf{x}_{2d,i}, \mathbf{x}_{2d,j}}$  refer to the correlation between original and embedded points, respectively.
3. The order preservation metric  $R_o \in [0, 1]$ , by setting  $C_{ij} \equiv 1$  when  $i \prec j$  and  $\theta_i > \theta_j$ , that is, when order is not preserved, and 0 otherwise; in other words, we compute the fraction of preserved order relations. Finally, for tests involving scored datasets (*i.e.*, a total ordering) we further assess the *Spearman* correlation metric [94, p. 508]  $R_s \in [-1, 1]$ , which assesses how well the total order is preserved in the K-nearest neighborhoods of data points.

Understandably, metrics involving order preservation are mainly used to capture the additional objective of our methodology and are meant to highlight that our approach can indeed satisfy its multi-objective desiderata. To this end, the corresponding values for the baseline methods are only presented for the sake of completeness. Finally, we emphasize that ***none of the tested methods directly optimizes any of the used quality metrics*** (for example, MDS/ISOMAP consider the square of the difference between original and embedded distances, while an absolute value was taken in our case). This allows us to ascertain as fair a comparison as possible, which was the main motivation in introducing the aforementioned metrics.



Figure 2.5: [Big-Stock dataset] Comparative visualization of our method (MoDE) with ISOMAP, t-SNE, and MDS. Color coding demonstrates objects' score values. MoDE excels in the preservation of partial orders (scores are the market capitalization of each stock), as indicated by the smooth color coding of the points. MoDE outperforms t-SNE and MDS across all metrics evaluated.

**Datasets.** For the first set of experiments, we used two time-series datasets that we compiled ourselves: `Small-Stock` (436 stocks of length 128) and `Big-Stock` (2252 stocks of length 1024). The time series nature of these datasets allows us to try small and large compression ratios using Fourier coefficients (by dropping low-energy coefficients). The datasets were created using historical prices from equities in the NASDAQ stock index, from which data were extracted using the Investors Exchange (IEX) API (<https://iextrading.com/developer/docs>), and were compressed in the Fourier basis (by storing the highest magnitude Fourier coefficients). For “score,” we use the market capitalization of each stock. Therefore, stocks with greater capitalization are aimed to be placed “higher” on a 2D plot, which provides an additional attribute of *interpretability* not present in any of the other baseline methods. We compress the `Small-Stock` dataset using a 16:1 compression ratio, and the `Big-Stock` dataset using a 128:1 compression ratio.

We assess the performance of the embedding methods using the aforementioned quality metrics. Fig. 2.4 depicts the values for each of those metrics for the four techniques we compared. For t-SNE, because of its randomized implementation, we report results averaged over 10 runs. We observe that MoDE (darkest bar) depicts the highest preservation across all metrics and across all techniques, with the singular exception of ISOMAP which features more accurate distance preservation ( $R_d$  metric), but lags behind MoDE for all other metrics. It should be noted that ISOMAP incurs a higher computational cost than many other techniques, because after the K-NN graph computation it builds a minimum spanning tree based on distances. The metrics  $R_o$  and  $R_s$  are depicted for other methods only for the sake of completeness: they are not meant to claim any superiority of MoDE, as it is the only method with an objective of maintaining partial orders. Note that for baseline methods  $R_o$  and  $R_s$  take values very close to 0.5 and 0, respectively, which reveals no order preservation, in full agreement with the fact that only MoDE considers score information in its embedding process. This fact is further illustrated on the comparative visualization plot for the `Big-Stock` dataset in Fig. 2.5. The two-dimensional plot reveals that MoDE succeeds in preserving score orders very accurately, as captured by the smooth transition of color-coded points. In this example, the scores/colors encode the market capitalization for each stock.

## 2.6.2 Classification Accuracy

We evaluate the embedding quality of MoDE in the context of classification tasks. In particular, we assess the prediction accuracy when using MoDE as a pre-processing step to reduce the dimensionality of the data. We compare with t-SNE, ISOMAP, and

parametric t-SNE (an extension to t-SNE that learns a parametric mapping between the high-dimensional data space and the latent space [95]). In this experiment, all techniques use the exact distances between the object and not approximate distance information. We re-iterate that this is very easy to accommodate in MoDE by simply providing the same lower and upper bound with value equal to the exact distance between any two objects.

To conduct this experiment, we split the dataset into training and test (80-20 split). We map the *training* points onto 2D using the previously mentioned techniques and build a classifier on 2D using the dimensionality-reduced training dataset. Subsequently, we map the test points on 2D and use the classifier built to predict their class. We evaluate on two classification methods:

- Multinomial Logistic Regression (LR), that generalizes logistic regression to multi-class problems. We used the implementation of scikit-learn [96]. The regularization hyper-parameter was tuned for each dataset and for each of the embedding methods, separately.
- K-Nearest Neighbor classifier (KNN), which classifies an object by majority vote across its nearest neighbors. In the voting phase, we weigh the nearest-neighbor points by their inverse distance to the query point to give more importance to closer points. The number of neighbors for this method is set equal to the number of nearest-neighbors used for the embedding methods (MoDE, t-SNE, parametric t-SNE, ISOMAP, and UMAP).

For each of the datasets and embedding methods, we report the classification accuracy, that is, the number of correctly classified data points divided by the total number of data points. These datasets do not come with inherent scores, so for MoDE we use the actual class labels as score values for each of the data points. This does not make the comparison to the other methods unfair, because the scores are only used to embed the training set. The test set only uses the original object features.

For MoDE and t-SNE, we embed the test data points in the 2D space by considering the nearest neighbors in the original space and using their location in the new space to map the new points. We take the weighted average of their nearest neighbor embeddings with weights set proportional to the inverse of the distance to these neighbors, that is, given a point  $i$  in the test data, we compute its embedding  $\mathbf{x}_{2d,i}$  as follows:

$$\mathbf{x}_{2d,i} = \frac{1}{\sum_{j \in \mathcal{N}_i} d_{ij}^{-1}} \cdot \sum_{j \in \mathcal{N}_i} d_{ij}^{-1} \mathbf{x}_{2d,j}, \quad (2.10)$$

where  $\mathcal{N}_i$  is the set of  $K$ -nearest neighbors of  $i$  in the training set and  $d_{ij}$  is the distance between points  $i$  and  $j$ . For ISOMAP, given that it depends on geodesic neighbors, we first find the nearest neighbors with respect to the geodesic distance of each test data point in the training set. Then, we construct a kernel by computing the shortest geodesic distances from each test point to another in the training set. To conclude, the embedding of the test set is taken using this kernel on the embedded vectors of the training set.

Table 2.1: Classification accuracy using Logistic Regression (LR) and K-Nearest Neighbors (KNN). Datasets were split using an 80–20 training/test split. Reported accuracy is for the test set.

| Dataset (# classes)   | n     | dim  | original data |       | MoDE         |             | t-SNE |       | parametric t-SNE |              | ISOMAP |             | UMAP         |              |
|-----------------------|-------|------|---------------|-------|--------------|-------------|-------|-------|------------------|--------------|--------|-------------|--------------|--------------|
|                       |       |      | LR            | KNN   | LR           | KNN         | LR    | KNN   | LR               | KNN          | LR     | KNN         | LR           | KNN          |
| Arrow (5)             | 500   | 1024 | 0.93          | 0.87  | 0.48         | 0.51        | 0.72  | 0.79  | 0.22             | 0.37         | 0.69   | 0.71        | <b>0.82</b>  | <b>0.86</b>  |
| Wafer (2)             | 1000  | 128  | 0.955         | 0.985 | <b>0.915</b> | 0.93        | 0.905 | 0.985 | 0.905            | 0.945        | 0.905  | <b>0.99</b> | 0.905        | 0.98         |
| Breast Cancer (2)     | 569   | 30   | 0.982         | 0.982 | 0.719        | 0.807       | 0.964 | 0.964 | 0.964            | 0.956        | 0.947  | 0.956       | <b>0.99</b>  | <b>0.98</b>  |
| Heart Beat (2)        | 14545 | 188  | 1.0           | 0.996 | 0.92         | 0.932       | 0.81  | 0.958 | <b>1.0</b>       | <b>1.0</b>   | 0.867  | 0.898       | 0.98         | 0.98         |
| Madelon (2)           | 2600  | 500  | 0.561         | 0.586 | 0.586        | 0.526       | 0.463 | 0.501 | <b>0.592</b>     | <b>0.613</b> | 0.517  | 0.532       | 0.53         | 0.515        |
| EEG eye state (2)     | 11853 | 14   | 0.682         | 0.925 | <b>0.822</b> | <b>0.84</b> | 0.536 | 0.742 | 0.545            | 0.587        | 0.569  | 0.581       | 0.562        | <b>0.84</b>  |
| Wine quality (3)      | 3961  | 11   | 0.606         | 0.602 | <b>0.523</b> | 0.537       | 0.484 | 0.499 | 0.442            | 0.496        | 0.479  | 0.503       | 0.5          | <b>0.56</b>  |
| Phishing websites (2) | 5425  | 68   | 0.938         | 0.941 | 0.748        | 0.757       | 0.712 | 0.884 | 0.727            | 0.84         | 0.536  | 0.682       | <b>0.757</b> | <b>0.92</b>  |
| cifar-10 (10)         | 10000 | 3072 | 0.392         | 0.278 | 0.106        | 0.111       | 0.208 | 0.189 | <b>0.224</b>     | <b>0.213</b> | 0.221  | 0.205       | <b>0.227</b> | <b>0.223</b> |

Table 2.2: Embedding quality on datasets without object orders.

| Dataset           | MoDE         |              | ISOMAP       |              | MDS   |              | t-SNE       |       | UMAP  |       |
|-------------------|--------------|--------------|--------------|--------------|-------|--------------|-------------|-------|-------|-------|
|                   | $R_d$        | $R_c$        | $R_d$        | $R_c$        | $R_d$ | $R_c$        | $R_d$       | $R_c$ | $R_d$ | $R_c$ |
| Arrow             | 0.68         | <b>0.88</b>  | 0.72         | 0.88         | 0.56  | 0.85         | <b>0.73</b> | 0.84  | 0.098 | 0.81  |
| Wafer             | <b>0.63</b>  | 0.88         | 0.61         | 0.91         | 0.41  | <b>0.92</b>  | 0.57        | 0.90  | 0.25  | 0.87  |
| Breast Cancer     | 0.75         | <b>0.789</b> | <b>0.757</b> | 0.787        | 0.6   | 0.785        | 0.401       | 0.707 | 0.39  | 0.67  |
| Heart Beat        | <b>0.64</b>  | 0.872        | 0.624        | 0.94         | 0.492 | <b>0.95</b>  | 0.008       | 0.029 | 0.38  | 0.88  |
| Madelon           | 0.32         | 0.155        | <b>0.59</b>  | <b>0.364</b> | 0.193 | 0.295        | 0.091       | 0.358 | 0.08  | 0.18  |
| EEG eye state     | 0.73         | 0.858        | <b>0.779</b> | 0.871        | 0.614 | <b>0.873</b> | 0.325       | 0.793 | 0.41  | 0.8   |
| Wine quality      | 0.739        | <b>0.827</b> | <b>0.744</b> | 0.79         | 0.647 | 0.8          | 0.264       | 0.633 | 0.477 | 0.826 |
| Phishing websites | 0.667        | <b>0.862</b> | <b>0.743</b> | 0.677        | 0.396 | 0.677        | 0.223       | 0.609 | 0.35  | 0.8   |
| cifar-10          | <b>0.696</b> | <b>0.939</b> | 0.665        | 0.836        | 0.394 | 0.844        | 0.068       | 0.063 | 0.11  | 0.91  |

Table 2.1 shows the test accuracy for datasets from the UCR time series archive and the UCI machine learning repository [97]. These results highlight that training machine learning models on MoDE embeddings typically yields superior accuracy than when using the other methods for dimensionality reduction with the exception of UMAP.

### 2.6.3 Additional experiments for MoDE

We conducted experiments to further elaborate on several desirable traits of MoDE.



**Embedding without object scores:** One might be prompt to consider that a limitation of MoDE is the fact that it asserts the existence of a score function, but this *is not the case*. In the absence of scores that dictate an ordering of points, *random* orders can be used without affecting the preservation of the given objectives. This is because the points will still be mapped so that distances and correlations are (approximately) preserved, by also considering (possibly random) orders. Therefore the absence of object scores does not affect the performance of MoDE in terms of distance/correlation preservation. Fig. 2.6 depicts this for the `Small-Stock` dataset, where we use both the original and shuffled market capitalization of the stocks, without observing any deterioration in the metrics evaluated.

To further support this argument, we provide additional experiments with the previously used classification datasets that come with no ordering of objects. The preservation of distances ( $R_d$ ) and correlations ( $R_c$ ) for all techniques is shown in Table 2.2.

Although the absence of object orders does not affect MoDE’s performance regarding distance and correlation preservation, it is important to recognize that randomly assigning orders can occasionally produce unhelpful or misleading visualizations. Particularly this may happen when no meaningful order information can be derived from the data. For example, in 2D word embedding visualizations, MoDE with completely random orders may not capture the semantic similarity between nearby points, unlike t-SNE, which maintains local clusters and, therefore, reflects semantic relationships. It’s crucial to understand that this is not a limitation of MoDE but rather highlights the need to choose the most suitable dimensionality reduction method based on the specific application’s requirements.

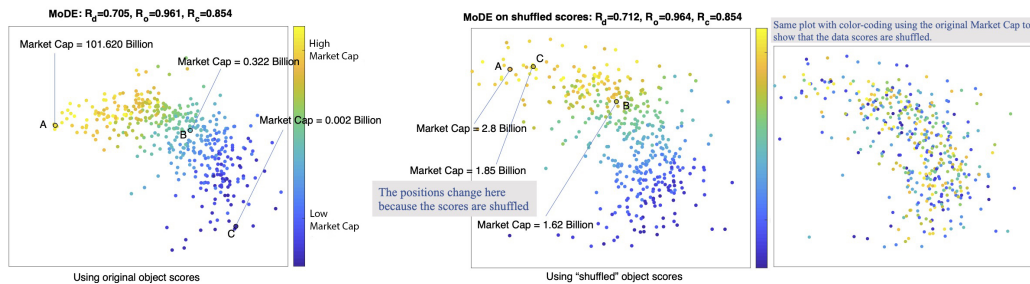


Figure 2.6: [`Small-Stock` dataset] MoDE operates effectively even without object scores. In their absence, random scores can be used without affecting the embedding quality.

**Anytime nature:** An important feature of MoDE is its iterative nature which creates a powerful *anytime* embedding algorithm, whose outcome progressively improves, and can be portrayed at any time of the execution. We illustrate this aspect of MoDE

in Fig. 2.7. With the exception of t-SNE and UMAP, the other embedding algorithms that we compare operate in a batch fashion, so the embedding outcome is only given at the end of the algorithm execution (which may be quite time-consuming).

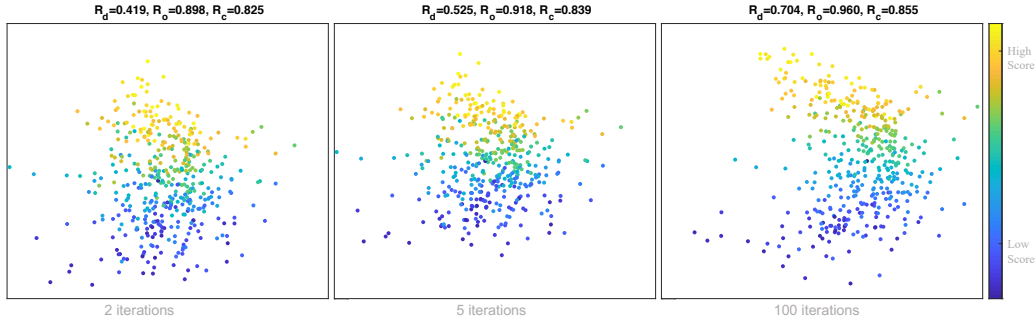


Figure 2.7: [Small-Stock dataset] Anytime nature of MoDE: embedding can be visualized at any time, even before the full execution of the algorithm is completed, with progressively improving results.

### 2.6.4 Embedding in more than $p > 2$ dimensions

We evaluate the quality of  $p$ -dimensional embeddings in terms of the preservation of distances and correlations between data points for different embedding methods. Figs. 2.8 & 2.9 show the progress of distance and correlation metrics respectively as we embed the datasets in higher dimensions using different embedding methods. We can observe that as we embed a dataset in higher dimensions using MoDE, we progressively obtain better distance and correlation preservation. As explained in Section 2.5, this is due to the fact that MoDE successively incorporates the correlation in an increasing number of coordinates, thus yielding a progressively better embedding. However, for the other dimensionality reduction methods, except for MDS, we do not observe such a consistent behavior (always getting a better distance and correlation preservation by embedding in higher dimensions). MDS seeks a low-dimensional representation of the data in which the distances respect the distances in the original high-dimensional space. Therefore, by embedding the data with MDS in higher dimensions we will always get a better distance preservation. As expected, when the dimensionality of the embedding space is equal to the dimensionality of the data, we almost get perfect distance preservation for MDS (recall that the distance metric used in this paper is local, meaning that it is computed for the  $K$  nearest neighbors of each point). Additionally, we should note that MDS has a cubic complexity in  $n$  (number of data points), whereas MoDE has a quadratic

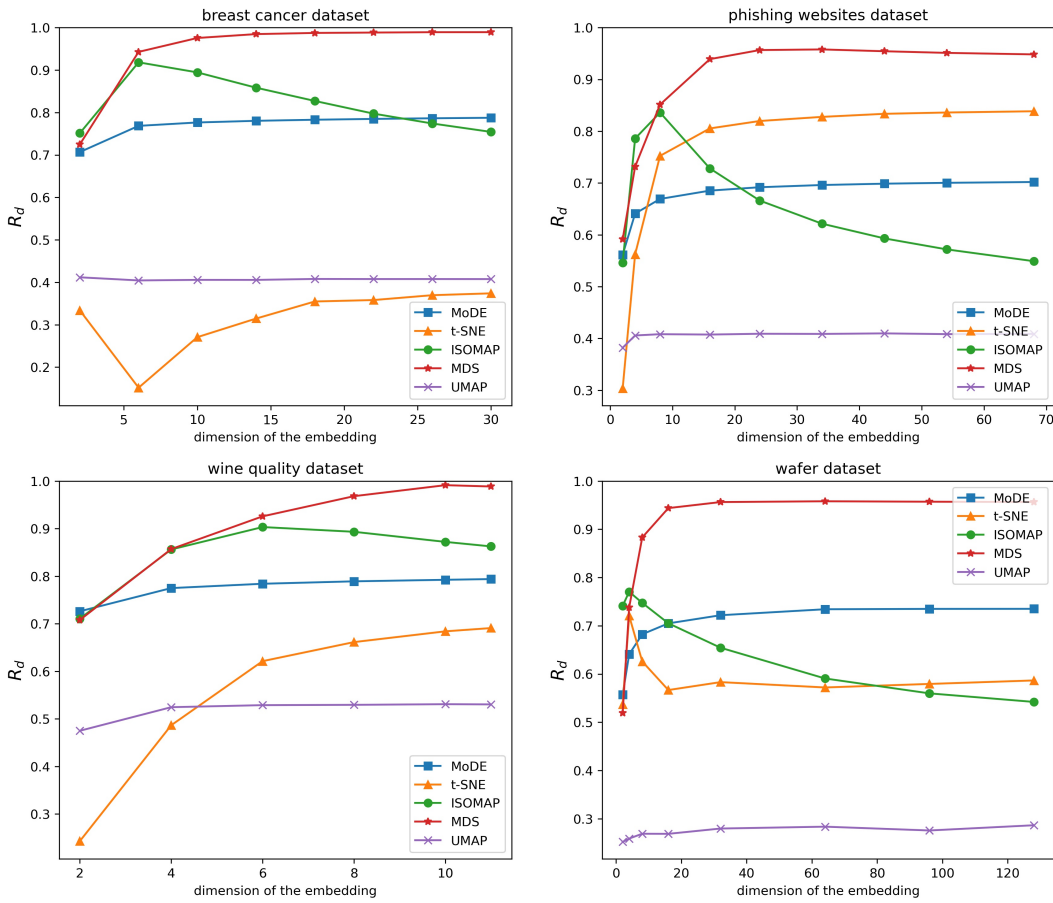


Figure 2.8: Evolution of the distance metric  $R_d$ , while embedding the data in higher dimensions. Observe that we consistently get a better distance preservation only for MoDE and MDS by increasing the dimensionality of the embeddings.

complexity in  $n$  after constructing the KNN graph. This could also be observed from Figure 2.10, MoDE has a much smaller running time compared to MDS. Furthermore, as suggested by [73], using t-SNE for embedding a dataset in  $p > 3$  dimensions requires setting the degrees of freedom for the Student-t distribution to be  $p - 1$ . However, the results suggest that the distance and correlation metrics are not really meaningful for t-SNE embeddings in  $p > 3$  dimensions, as already attested in the original publication [73].

These experiments highlight that among the methods compared MDS, ISOMAP and MoDE feature the best preservation of relationships. However, MoDE has the advantage of the lowest computational complexity among them. In Fig. 2.10, we

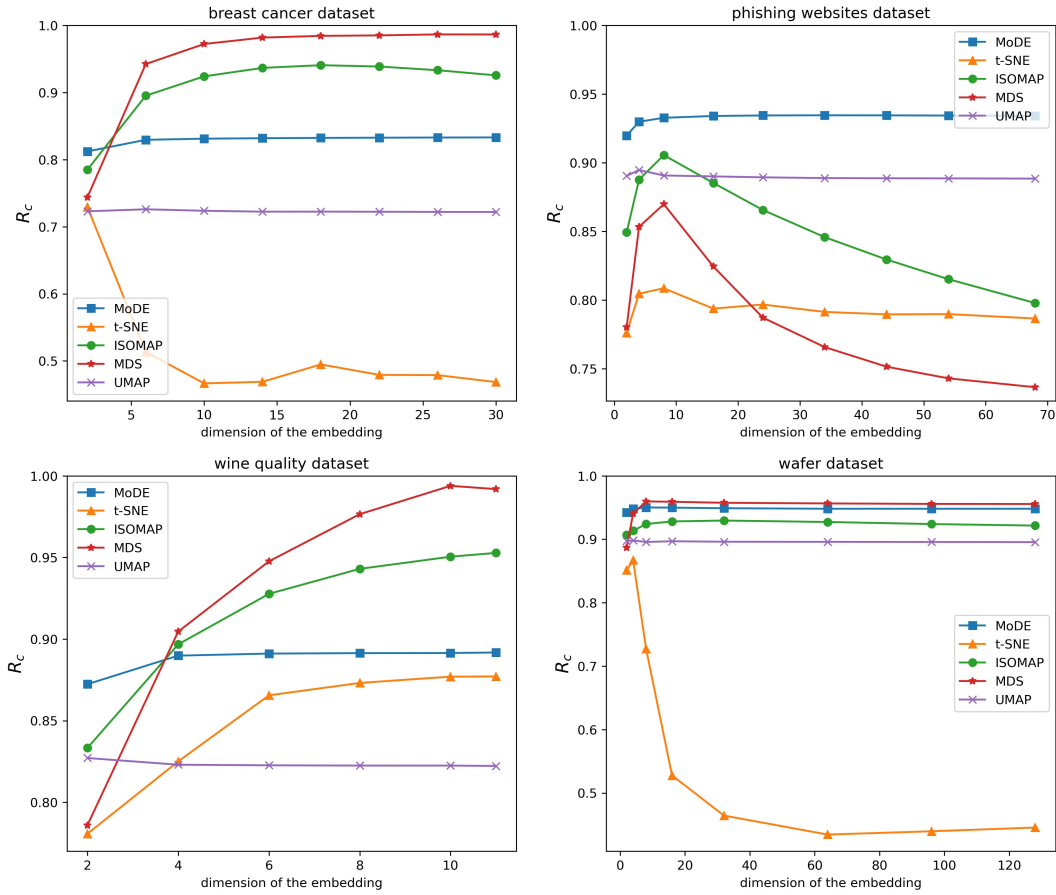


Figure 2.9: Evolution of the correlation metric  $R_c$ , while embedding the data in higher dimensions. We consistently observe an increasing preservation of the correlation for each additional projected dimension only for MoDE.

compare the running time of MoDE with ISOMAP and MDS. To strictly capture the embedding part of each algorithm, we only report the running time of the embedding process, that is, *after* the computation of the neighborhood graph. For this experiment, we use the EEG eye state dataset, with increasing dataset sizes ranging between 2,000 and 11,000 objects.

The results of Fig. 2.10 support the fact that our embedding algorithm offers a competitive embedding quality at a lower computational cost.

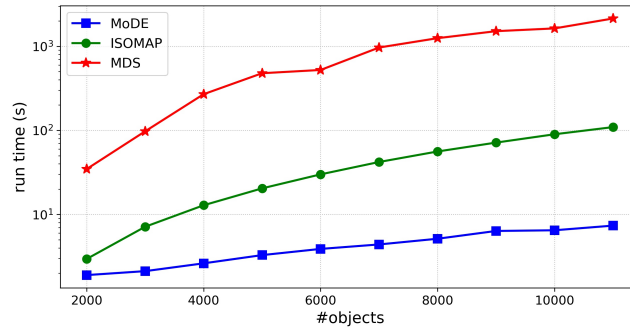


Figure 2.10: Running time comparison for the three techniques that produced the best quality of embeddings: MoDE, ISOMAP, and MDS. Note that the  $y$ -axis (time) is in logarithmic scale, highlighting that MoDE not only offers an embedding of high quality, but it can accomplish it in a much more efficient manner.

## 2.7 Discussion

As explained in Section 2.6 in the absence of object scores, random scores could be used to run the MoDE algorithm and the resulting embedding would still hold a high distance and correlation preservation. However, in some cases, randomly assigning orders can produce unhelpful or misleading visualizations. Therefore it is important to choose the most suitable dimensionality reduction method based on the specific application's requirements. A limitation of this work is the lack of an analysis on how to choose the number of nearest neighbors  $k$  in MoDE. Larger datasets usually require a larger  $k$  value; however, a larger  $k$  value would cause the computational complexity of MoDE to increase. An empirical analysis of how to choose  $k$  with respect to the size of the dataset and how different values of  $k$  affect the resulting visualization can be a subject of future extension of this work.

## 2.8 Conclusion

We presented, to the best of our knowledge, the first embedding approach that is able to operate on inexact distance information, expressed as ranges of upper and lower bounds. This can be encountered when operating on compressed data, which was the scenario considered in this work. However, our research can also be applied to other scenarios, such as when collecting data from multiple sensors measuring the same quantity, which would also yield value ranges. Given that our approach can accommodate visualization of compressed data, we believe that it is a suitable choice for the dimensionality reduction and visualization of very large datasets.

Our experiments have shown that our method offers competitive embedding quality and classification accuracy vis-à-vis other prevalent embedding approaches, albeit at a lower computational cost. Moreover, when the embedding technique uses two-dimensions, the placement of objects and the resulting visualization is easily interpretable by humans.



## Chapter 3

# On Smoothed Explanations: Quality and Robustness

**Abstract.** Explanation methods highlight the importance of the input features in taking a predictive decision, and represent a solution to increase the transparency and trustworthiness in machine learning and deep neural networks (DNNs). However, explanation methods can be easily manipulated generating misleading explanations particularly under visually imperceptible adversarial perturbations. Recent work has identified the decision surface geometry of DNNs as the main cause of this phenomenon. To make explanation methods more robust against adversarially crafted perturbations, recent research has promoted several *smoothing* approaches. These approaches smooth either the explanation map or the decision surface.

In this work, we initiate a very thorough evaluation of the **quality** and **robustness** of the explanations offered by smoothing approaches. Different properties are evaluated. We present settings in which the smoothed explanations are both better, and worse, than the explanations derived by the commonly-used (non-smoothed) Gradient explanation method. By making the connection with the literature on adversarial attacks, we demonstrate that such smoothed explanations are robust primarily against additive attacks. However, a combination of additive and non-additive attacks can still manipulate these explanations, revealing important shortcomings in their robustness properties.



## 3.1 Introduction

Explanation methods attribute a numerical value to each data feature in order to quantify its relative importance towards the model’s prediction. Such attributions help to better understand and trust complex models like deep neural networks (DNNs). In safety-critical tasks, such an understanding is a prerequisite to the deployment of DNNs, because a domain expert will never make important decisions based on a model’s prediction unless that model is trustworthy. Moreover, explanations can help to understand the reasons behind a model’s decisions, and when it comes to model debugging, they can reveal the presence of any spurious data correlations that may lead to faulty predictions during inference [12].

In the context of image classification with deep neural networks, several explanation methods have been proposed based on the gradient of the output with respect to input, also called gradient-based explanations [98, 99, 100, 64, 101]. The explanation generated by these methods, *a saliency map*, highlights the parts of the image that contributed to the prediction. Recent work has shown that gradient-based explanations of neural networks can be fragile and can be easily manipulated via adversarially perturbed inputs [102, 60, 103, 104, 105]. That is, one can find a small-norm perturbation to be added to an input (often imperceptible), such that the focus of the explanation changes towards irrelevant features while the model’s output remains unchanged. This, in turn, can make explanations inappropriate to help end-users gain trust in a model’s prediction.

The large curvature of the decision surface of neural networks has been identified as one of the causes of fragility for gradient-based explanations [102, 60, 62]. To make explanation methodologies more robust, a class of approaches were proposed that made the decision surface of neural networks smoother [62, 60, 106]. We refer to these approaches as *smoothing approaches*. Figure 3.1 shows a sketch of explanation attacks and a smoothing approach, the smooth gradient method, which can make the explanations more robust against additive perturbations. Although, the perturbation applied to manipulate the smooth gradient explanation is stronger (it is more visible in the perturbed image in the bottom row), the resulting explanation is still similar to the original explanation (not the target explanation). It is worth mentioning that similar methods have been proposed in the context of adversarial robustness, with the aim of flattening the decision surface of neural networks to reach more robust predictions [107, 108].

Here, we shed light on the *quality* and *robustness* of the explanations made robust by different smoothing approaches. Because quality can be measured in numerous ways, we employ various tests to assess the quality of explanations. Each

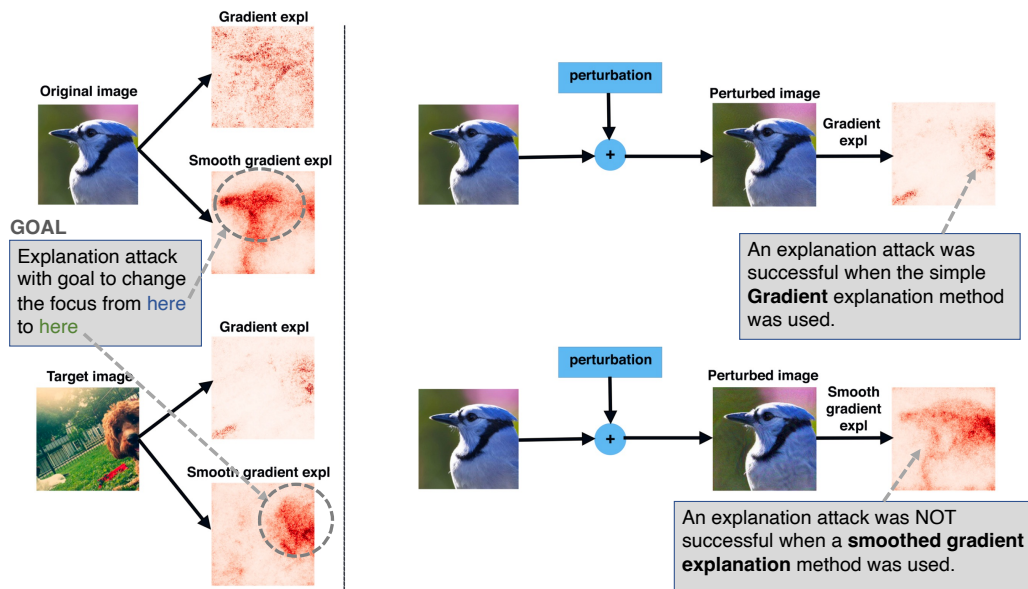


Figure 3.1: An illustration of explanation attacks. The left column top row shows the original image and its corresponding explanation. The goal of the attack is to manipulate the original explanation such that it resembles a given target explanation (left column bottom row). In the right column, the result of the attacks against the gradient and smooth gradient explanations are shown.

test evaluates a desirable property for explanations, including sensitivity to changes in the model, fidelity to the predictor function, etc. Robustness evaluates how stable or sensitive are the explanations with respect to changes in the input. We show that explanations derived by smoothing approaches only provide robustness against additive  $\ell_p$  norm attacks. This is the first work which highlights that attacks based on the combination of spatial transformation [109] and/or color transformation [110] together with additive perturbations are more effective in manipulating the outcome of, even smoothed, explanation methods. Our contributions can be summarized as follows:

- We study the effectiveness of smoothing approaches to achieve robust explanations. We present results on evaluating both the quality and robustness properties of smoothed explanations. We show our results by conducting extensive experiments that required several days to execute on GPU-accelerated hardware.
- We assess the quality of smoothed explanations by presenting the results of a battery of quality tests. Our tests evaluate explanation quality with respect to the following aspects: sensitivity to model parameters, the ability to discriminate the features related to the predicted class, fidelity to the predictor function, and conciseness (sparsity) of the explanation. Our results demonstrate that the smoothing approaches based on adversarial training, which penalize the large curvature of the decision surface during training, are superior to the Gradient explanation in terms of conciseness and fidelity to the predictor function. However, smoothing approaches based on post processing, for example, Smooth Gradient, are inferior to the Gradient method in terms of desirable qualities such as conciseness and class discriminativeness.
- Until now, the effectiveness of smoothing methodologies has been studied only under additive attacks. We present experiments that *combine* additive and non-additive attacks. We show that such attacks are able to manipulate the explanations derived by smoothing approaches more successfully. Moreover, combinatory attacks typically result in a less perceptible perturbation of the input data thus making them more effective. To the best of our knowledge, this is the first time that such attacks are used in the context of explanations.

**Related work.** There have been several works aiming to make explanation methods more robust. These works mostly focused on either modifying the explanation

method itself or modifying the predictor model to achieve robust explanations. Wang et al. [62] introduced the Uniform Gradient, which is similar to the Smooth Gradient, but it uses uniform noise. They showed that it can hardly be manipulated by additive attacks. Dombrowski et al. [60] showed that a network with soft-plus activations has a more robust Gradient explanation compared to a ReLU network, given that the parameter  $\beta$  of the soft-plus function is chosen to be sufficiently small. Consequently, they proposed the  $\beta$ -smoothing approach in which they substitute the ReLU activations of a trained network by soft-plus functions with a small  $\beta$  parameter. Wang et al. [62] introduced a regularization term called *Smooth Surface Regularization (SSR)* for the training objective of a DNN. This training objective penalizes the large curvature of a DNN by regularizing the eigenvalue of the input hessian with the maximum absolute value. Moreover, they showed that adversarial training [111] also leads to more robust explanations. This fact can also be deduced from the results of [107] as they showed that adversarial training leads to a significant decrease in the curvature of the loss surface with respect to inputs. Anders et al. [61] proposed an attack in which they adversarially manipulate the model instead of the input in order to manipulate the explanation. Then they propose a modification to the existing explanation methods to make them more robust against such manipulated models. Lakkaraju et al. [112] proposed a framework for generating robust and stable black box explanations based on adversarial training. Chen et al. [113] introduced a regularization term to the training objective of neural networks to achieve robust Integrated Gradient explanations. Finally, Dombrowski et al. [114] developed a theoretical framework to derive bounds on the maximum manipulability of explanations and proposed three different techniques to boost the robustness of explanations.

In this work, we show that the robustness of smoothed explanations *can* be affected by a combination of additive and non-additive attacks. Furthermore, we present a thorough evaluation of the different quality aspects of smoothed explanations.

## 3.2 Background

First, we provide the definition of an explanation map and then briefly describe the explanation methods we used in this paper. Then we continue by introducing the attacks against explanations and the smoothing approaches we are going to study in this paper.

Consider a model  $f : \mathbb{R}^d \rightarrow \mathbb{R}^K$  which classifies an input  $\mathbf{x} \in \mathbb{R}^d$  into one of

the  $K$  classes. *The explanation* of an input  $\mathbf{x}$ , denoted by  $h_f(\mathbf{x}) \in \mathbb{R}^d$ , associates a score to each feature of the input indicating the relevance of that feature towards the model’s prediction. For instance, in the context of image classification, saliency maps associate a score to each pixel of the input image resulting in a heatmap that highlights important regions of the image leading to the model prediction. In this work, we focus on the gradient-based explanations and mainly on the Gradient method. Given a model  $f$  and an input  $\mathbf{x}$ , the Gradient explanation is defined as  $\nabla_{\mathbf{x}}f(\mathbf{x})$ , the gradient of the output with respect to the input data. Because several gradient-based explanation methods make use of the gradient with respect to input, our results could be extended to those explanation methods as well. We will also consider two methodologies for smoothing the explanations: the Smooth-Gradient [115] and Uniform-Gradient [62] methods. The Smooth-Gradient is computed by averaging the Gradient explanations of noisy versions of  $\mathbf{x}$ , sampled from a Gaussian distribution:  $\mathbb{E}_{\mathbf{z} \sim \mathcal{N}(\mathbf{x}, \sigma^2 I)} \nabla_{\mathbf{z}}f(\mathbf{z})$ . The Uniform-Gradient is computed in a similar way, with the exception that the noise distribution is uniform and not Gaussian:  $\mathbb{E}_{\mathbf{u} \sim \mathcal{U}(\mathbf{x}-\sigma, \mathbf{x}+\sigma)} \nabla_{\mathbf{u}}f(\mathbf{u})$ .

### 3.2.1 Attacks to manipulate explanations

Similarly to common adversarial attacks [116, 117, 118], recent work has shown that explanations can also be manipulated by adding a small and almost imperceptible perturbation to the input [102, 60]. We refer to this class of attacks as *explanation attacks*. There have been various formulations for explanation attacks [102, 60]. In this work, we will use the formulation introduced by Dombrowski et al. [60]. In this attack, the attacker tries to find a perturbed input for which the explanation is manipulated to be very similar to a given target explanation map while the output of the model remains approximately unchanged, i.e., the model prediction is the same for the original and the perturbed input. Keeping the prediction unchanged makes the explanation attack more effective, that is, the model is making a correct prediction for a seemingly legitimate input, but the explanation is highlighting irrelevant features as important towards this prediction and hence is misleading. The target map used in this attack could be any heatmap in general. However, we used the explanation of a target image as a target map in this work. Below, we will give a formal definition of this attack.

**Definition 2** (Targeted manipulation attack). *For an input  $\mathbf{x}$ , an explanation  $h_f(\mathbf{x})$  of a model  $f(\mathbf{x})$  is vulnerable to attack if there exists a perturbed/adversarial input  $\mathbf{x}_{adv}$ , such that  $h_f(\mathbf{x}_{adv})$  is similar to a given target map  $h^t$  but the model’s output remains unchanged. An attacker finds  $\mathbf{x}_{adv}$  by minimizing the following objective*

function:

$$\mathcal{L} = \underbrace{\|h_f(\mathbf{x}_{adv}) - h^t\|^2}_{\text{manipulate the expl}} + \underbrace{\gamma_1 \|f(\mathbf{x}_{adv}) - f(\mathbf{x})\|^2}_{\text{prediction unchanged}} + \underbrace{\gamma_2 \mathcal{L}_{reg}(\mathbf{x}, \mathbf{x}_{adv})}_{\text{perceptual similarity}} \quad (3.1)$$

The first term of the equation ensures the similarity between the manipulated explanation and the target map. The second term ensures the similarity between the model output for the original and perturbed inputs. The third term regularizes the perturbation to ensure perceptual similarity between the original and perturbed inputs. Note that  $\mathcal{L}_{reg}$  is defined by the attacker according to the type of the perturbation. For instance, in an *additive attack*, that is, when  $\mathbf{x}_{adv} = \mathbf{x} + \delta$ ,  $\mathcal{L}_{reg}$  could be some  $\ell_p$  norm of  $\delta$ , for example  $\|\delta\|_\infty$ . The relative weighting of the terms in equation (3.1) is controlled by the hyper-parameters  $\gamma_1$  and  $\gamma_2$ .

### 3.2.2 Robustness of explanations

Recent work has tried to define the robustness of explanations in terms of the sensitivity of input gradients to changes in the input data [62, 60]. Wang et al. [62] define the robustness of explanations by the Lipschitz continuity coefficient of the input gradients; a smaller coefficient means that the explanation is less sensitive to the changes in the input and hence more robust. In this regard, a class of approaches to generate robust explanations has been proposed in the recent works. These approaches are either based on smoothing out the explanation maps or flattening the decision boundary of the model itself. Broadly, these approaches can be classified into two categories:

1. *Post-hoc approaches* do not require retraining of the network and can be applied as a post-processing step.
2. *Ad-hoc approaches* to robust explanations require retraining of the network and hence are more costly.

The post-hoc approaches we use in our experiments are Smooth-Gradient [115], Uniform-Gradient [62], and  $\beta$ -smoothing [60]. The first two methods smooth the explanation map and the third smooths the decision surface of the model. All three approaches act on pre-trained models, hence the characterization "post-hoc". There exist two well-known ad-hoc methods, CURE [107] that uses networks trained with curvature regularization, and SSR [62]. In our experiments we use only CURE, because with the publicly available code of SSR we were not able to reproduce the results in the authors' original publication.

### 3.3 Evaluation of post-hoc approaches

We commence by evaluating the *quality* of explanations derived by post-hoc approaches. These do not require retraining of the network. Then, we evaluate the *robustness* of these explanations by presenting results on effective non-additive attacks to manipulate the explanations. For all of the experiments in this section, we used a VGG-16 style network trained on the ImageNet dataset [119].

#### 3.3.1 Quality of explanations of post-hoc approaches

To evaluate the explanation quality, there exist several quality tests in the literature. In general, assessing the quality of an explanation is a challenging task and each quality test only evaluates a particular aspect of an explanation. Therefore the assemblage of quality tests helps understand which aspects of the explanations are improved and which are deteriorated by the smoothing approaches.

**Cascade randomization of model parameters.** Adebayo et al. [120] argued that it is desired for an explanation to be sensitive to the changes in the model parameters. They proposed a model parameter randomization test to assess this sensitivity. In this test, the parameters of a model are progressively randomized (re-initialized) from the top layer (logits) to the bottom layers. In each randomization step, the explanation from the resulting model is compared against the explanation from the original model. Randomizing (re-initializing) the model parameters means losing what the model has learned from the data during training and hence leads to a quick drop in the prediction accuracy of the model. Therefore, we expect a "good" explanation to change significantly as the parameters of the model are being substituted with random values. However, if an explanation is insensitive to the randomization of the model parameters, then it is not deemed appropriate for debugging the model under erroneous predictions [120].

The visual results of this test for Gradient explanation and post-hoc approaches are shown in Figure 3.2. One can observe that the explanations derived from post-hoc approaches show less sensitivity to the randomization of model parameters than compared to the Gradient method. This can also be verified by the Spearman rank correlation between the original and randomized explanations shown in Figure 3.3. We observe that for the smoothed explanation methods, the original and randomized explanations have a high rank correlation after the randomization of the top layers of the network. *These results highlight that using Smooth Gradient, Uniform Gradient, and  $\beta$ -smoothing to achieve a more robust explanation can come at the expense of*

having explanations that are less sensitive to model parameters.

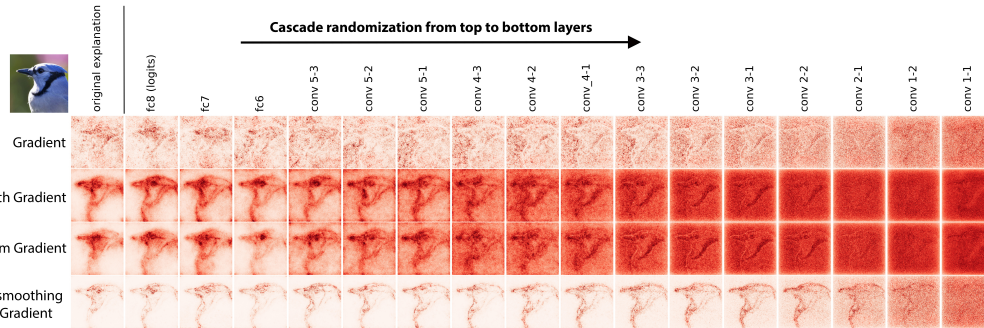


Figure 3.2: Cascade randomization of the VGG-16 (ImageNet) layers. The first column shows the original explanation for the image "Jay" bird. Each subsequent column shows the effect of the randomization of the parameters of the network up to that layer (inclusive) on the explanations. Smooth Gradient, Uniform Gradient, and  $\beta$ -smoothing explanations remain almost visually intact after randomizing the top layers of the network.

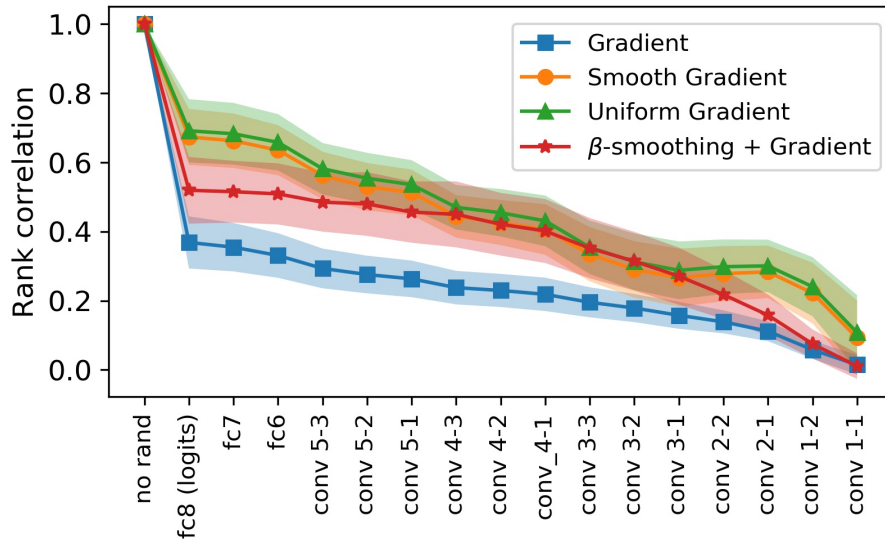


Figure 3.3: Spearman rank correlation between the original and randomized explanation derived for randomization up to the layer indicated by the x-axis. A higher rank correlation value indicates a higher similarity, i.e., *the higher the curve of an explanation method, the less sensitive it is to model parameters*. The results are averaged over 1000 Images from Imagenet and the shaded area around each curve indicates the standard deviation.



**Class sensitivity of explanations.** A good visual explanation should be able to localize the image regions relevant to the target category, that is, it should be *class discriminative* [100]. This is particularly significant when dealing with images containing more than one object. To assess the class discriminativeness of an explanation we used a quality test equivalent to the pointing game [121]. We sampled images from the MS COCO dataset [122], containing two objects that are also present among the ImageNet class labels. For this test we only keep the samples for which one of the objects in the image is the top predicted class by the network and the other object is among the top 20 predicted classes by the network. We compute the explanation maps for each of the class labels corresponding to the objects, that is, we compute the gradient of the output logit corresponding to each object with respect to the input. Using the segmentation mask of the objects provided in the dataset as ground truth, we compute the percentage of the top-20 values in the explanation maps generated for each target category that are inside the corresponding segmentation masks. The results of this test are shown in table 3.1 and a visual depiction of this test is given in Figure 3.4. These results indicate that the smoothed explanation methods are less discriminatory when generated for the target class label that has a lower probability. *This suggests that in terms of the class discriminativeness of explanations, the post-hoc smoothing approaches investigated in this paper are inferior to the Gradient method.*

Table 3.1: Ratio of the explanation top-20 values included in the segmentation mask of each object. Note that the columns "obj 1" and "obj 2" are for the explanations computed for the top predicted class and the class corresponding to the second object in the image respectively. The results are averaged over 60 samples.

| Explanation Method            | obj 1       | obj 2       |
|-------------------------------|-------------|-------------|
| Gradient                      | 0.6         | <b>0.49</b> |
| Smooth Gradient               | 0.54        | 0.37        |
| Uniform Gradient              | 0.57        | 0.38        |
| $\beta$ -smoothing + Gradient | <b>0.62</b> | 0.45        |

**Sparseness of explanations.** To create explanations that are human-accessible, it is advantageous to have a *sparse* explanation map [123], that is, only the features that are truly predictive of the model output should have significant contributions, and irrelevant features should have negligible contributions. Sparse explanations are more concise as they focus more on the features which had a significant contribution

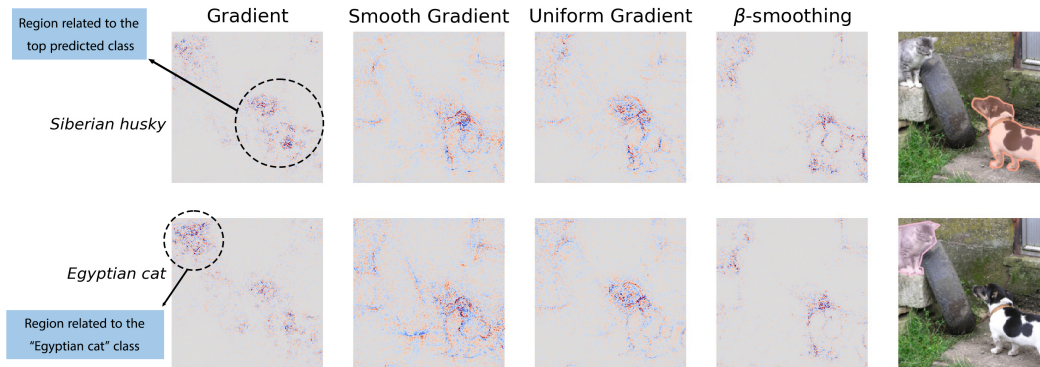


Figure 3.4: Class sensitivity of explanation methods. The top and bottom rows show the explanations for the top-predicted class (“Siberian husky”) and the “Egyptian cat” class respectively. We can observe that the post-hoc smoothed explanations for the class with lower probability (bottom row) have less intersection with the segmentation mask of the object (right-most column).

to model prediction. Therefore, sparser explanations can be more helpful for end-users to understand the reasons for a specific prediction of the model [124]. This is mainly important in applications where potentially new knowledge can be extracted from explanations, for instance in medical domain, an expert can potentially learn about features in an image that could lead to a specific diagnosis. Therefore, it is important for the explanations to be concise and to highlight only relevant features.

To measure the sparseness of an explanation map, we applied the Gini Index on the absolute value of the flattened explanation maps. The Gini Index is a metric that measures the sparseness of a vector with non-negative values [125]. By definition, the Gini Index takes values in  $[0, 1]$  with higher values indicating more sparseness. Table 3.2 shows the average Gini Index of the Gradient, Smooth Gradient, Uniform Gradient, and  $\beta$ -smoothing computed for 1000 randomly sampled images from ImageNet. *The results show that compared to the Gradient method, Smooth Gradient and Uniform Gradient provide less concise explanations, whereas  $\beta$ -smoothing actually improves the sparseness of the explanations as compared to the Gradient method.* This means that explanations generated by smooth and uniform gradient methods can be less comprehensive as they can potentially present large attribution values for irrelevant features. This can also be visually seen in Figure 3.5.

**Explanation Infidelity.** Introduced in Yeh et al. [43], this metric captures how the predictor function changes in response to significant perturbations to the input. For a predictor function  $f$ , explanation method  $h_f$ , and a random variable  $I \sim \mu_I$

Table 3.2: Average Gini Index for the explanations of a VGG-16 network (averaged over 1000 samples). A **larger** Gini Index indicates more sparseness and hence a more concise explanation. Notation: SG~Smooth Gradient, UG~Uniform Gradient,  $\beta$ -s~ $\beta$ -smoothing

|      | Gradient         | SG               | UG               | $\beta$ -s                         |
|------|------------------|------------------|------------------|------------------------------------|
| Gini | $0.56 \pm 0.038$ | $0.34 \pm 0.053$ | $0.35 \pm 0.058$ | <b><math>0.65 \pm 0.067</math></b> |

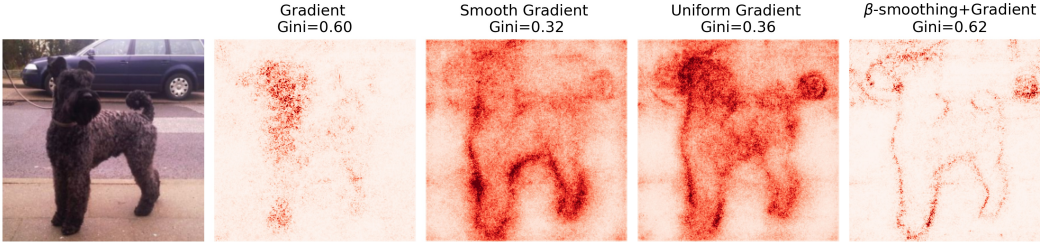


Figure 3.5: Sparser explanations (higher gini index) are more concise and do not contain attributions irrelevant to the predicted class.

which represents a meaningful perturbation, the explanation infidelity for an input  $\mathbf{x}$  is defined as:

$$\text{INFID}(h_f, f, \mathbf{x}) = \mathbb{E}_{\mathbf{I} \sim \mu_{\mathbf{I}}} \left[ (\mathbf{I}^T h(f, \mathbf{x}) - (f(\mathbf{x}) - f(\mathbf{x} - \mathbf{I})))^2 \right] \quad (3.2)$$

This metric generalizes the completeness axiom [65, 64] because it allows for different types of perturbations, which could be of interest depending on the problem and the dataset. For instance, if we set  $I = \mathbf{x} - \mathbf{x}_0$ , that is, the difference between input  $\mathbf{x}$  and some baseline  $\mathbf{x}_0$ , then the explanation that optimizes the infidelity for this deterministic perturbation  $I$ , satisfies the axiom of completeness [43].

We use the infidelity metric to compare the effect of post-hoc smoothing approaches on the fidelity of explanations to the predictor function. As suggested in [43], we used the square removal perturbation to compute the infidelity of explanations for randomly selected images from ImageNet. Table 3.3 shows the results for the post-hoc approaches. A *lower infidelity* value indicates better fidelity of the explanation to the predictor function. *The results suggest that the degree of smoothing used to make explanations more robust, also improves their fidelity to the predictor function.* Therefore, with respect to the Infidelity metric, all of the smoothed explanations investigated in this section are superior to the Gradient method. This finding is also in line with the results of [43], that is, that modest smoothing improves the infidelity of explanations. Note that in general, post-hoc smoothing of an explanation reduces its sensitivity to input changes, which can be seen as enhancing

Table 3.3: Average Infidelity for the explanations of a VGG-16 network (averaged over 200 samples). A **lower** Infidelity value indicates better fidelity of the explanation to the predictor function.

|            | Gradient       | SG              | UG             | $\beta$ -s                       |
|------------|----------------|-----------------|----------------|----------------------------------|
| Infidelity | $2.26 \pm 2.8$ | $2.21 \pm 2.71$ | $2.2 \pm 2.64$ | <b><math>1.84 \pm 1.7</math></b> |

its "robustness" [43]. You can achieve lower sensitivity by applying more aggressive smoothing, like increasing the standard deviation of Gaussian noise in the smooth gradient method. However, it's crucial to understand that lower sensitivity doesn't necessarily equate to a better explanation. For instance, consider an extreme case of a constant explanation with the same value for every input. This explanation has zero sensitivity but lacks fidelity to the model. So, it's essential to strike a balance and apply a reasonable amount of smoothing to maintain a faithful explanation.

### Discussion of the results

We compared different quality aspects of the post-hoc smoothed explanations with the Gradient method. Our results show that none of the investigated post-hoc methods are constantly better or worse than the Gradient explanation in all quality aspects. However, investigating different quality aspects can help us understand whether using a post-hoc smoothed explanation can be harmful in a certain application or not. For instance, if we intend to analyze erroneous predictions of a model, then smoothed explanations *cannot be helpful as they show less sensitivity to model parameters*. On the other hand, our results with the infidelity metric showed that modest amount of smoothing helps to achieve explanations which are closer to satisfying the axiom of completeness. Table 3.4 shows a summary of our results on the different quality aspects of post-hoc smoothed explanations.

### 3.3.2 Robustness of explanations of post-hoc approaches

We employ different kinds of explanation attacks to assess the robustness of Smooth Gradient, Uniform Gradient, and  $\beta$ -smoothing explanations. Mainly, we present attacks composed of additive and non-additive perturbations, and show that they are more effective than additive attacks to manipulate explanations. The non-additive attacks we use are spatial transformation attacks [109], and recoloring attacks [110]. In the rest of this paper, we refer to the additive attack as *Delta*, spatial transformation attack as *StAdv*, and recoloring attack as *Recolor*.

Table 3.4: Comparison between different quality aspects of post-hoc smoothed explanations and the Gradient method. In each cell, the symbol  $\times$ ( $\checkmark$ ) indicates that the smoothed explanation is worse (better) than the Gradient method in this quality aspect.

|                                 | SG           | UG           | $\beta$ -s   |
|---------------------------------|--------------|--------------|--------------|
| Sensitivity to model parameters | $\times$     | $\times$     | $\times$     |
| Class sensitivity               | $\times$     | $\times$     | $\times$     |
| Sparseness                      | $\times$     | $\times$     | $\checkmark$ |
| Fidelity                        | $\checkmark$ | $\checkmark$ | $\checkmark$ |

We used the projected gradient descent (PGD) algorithm to optimize the objective function (3.1)<sup>1</sup>. In our experiments, we evaluate three combinations of attacks, namely Delta, Delta+StAdv, and Delta+StAdv+Recolor, against the explanation of a VGG-16 network trained on ImageNet [119].

To evaluate the results of the attacks, we use two metrics: (1) The **Cosine Distance metric (cosd)** to evaluate the similarity between the target and manipulated explanations [62]. A lower cosine distance corresponds to a lower  $\ell_2$  distance between the target and manipulated explanations, which shows a higher similarity between these explanations and hence a more successful attack. The range of the values for cosd is between 0 and 1. (2) The **LPIPS** metric [126] for quantifying the *perceptual* similarity between the original and perturbed images. A lower LPIPS value indicates higher perceptual similarity. In an explanation attack, the attacker aims to generate a perturbed image perceptually similar to the original image that can change the explanation significantly. Hence a lower LPIPS value is desirable for a successful explanation attack. We specifically use these two metrics in our evaluation to make sure that the comparison between the attacks is fair. While combining different types of perturbations will naturally make an attack stronger, it will also cause the perturbation to be easily detectable in the perturbed image. We show that by carefully setting the parameters of the non-additive perturbations, combining additive and non-additive attacks can lead to better manipulation of the explanation (lower cosd value) while having *equally or even less perceptible perturbation* (lower LPIPS value) as compared to the additive attacks.

Figure 3.6 shows the cosine distance between the target and manipulated expla-

<sup>1</sup>As discussed in [102, 60], to avoid zero-valued gradients when optimizing (3.1), we have to replace the ReLU activation with its smooth approximation. In this work, we used a soft-plus function with a sufficiently large  $\beta$ .

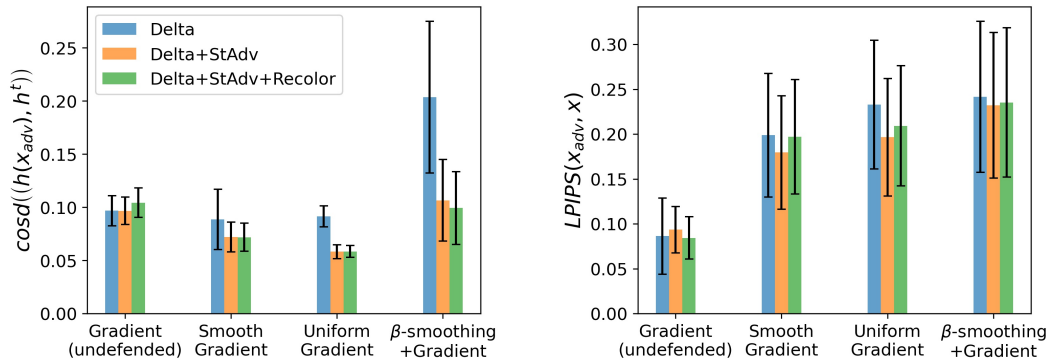


Figure 3.6: Evaluation with  $\text{cosd}$  (left) and LPIPS metrics for attacks against post-hoc approaches. For both  $\text{cosd}$  and LPIPS, smaller values indicate higher similarity. The results show that the combination of additive and non-additive attacks can manipulate the  $\beta$ -smoothing explanations significantly (lower  $\text{cosd}$  value) while resulting in equally or even less perceptible perturbations (lower LPIPS value). All results are averaged over 320 images from ImageNet. The black bars indicate standard deviation.

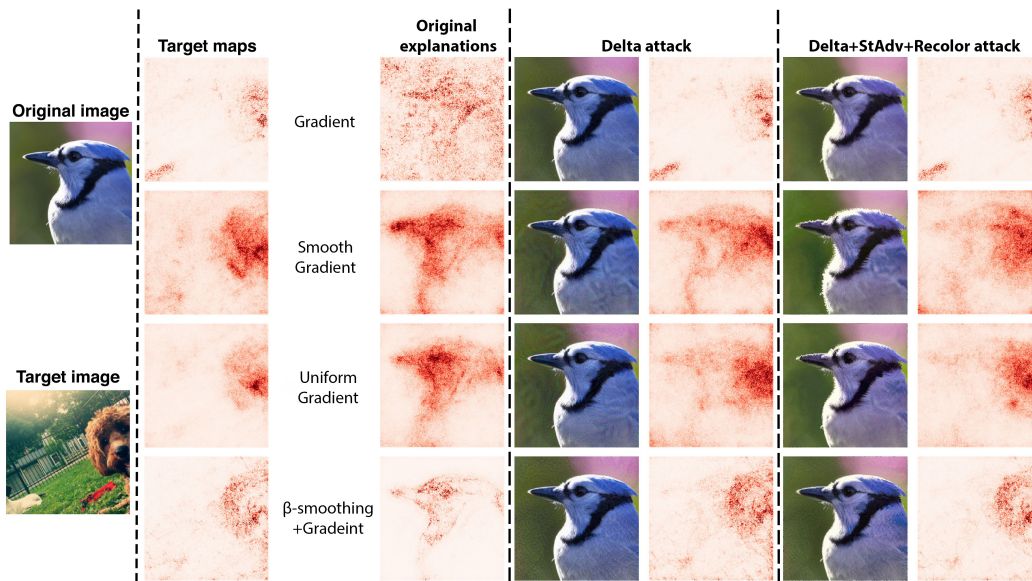


Figure 3.7: A visual example of explanation attacks against different methods. Under each attack setting, the left column shows the perturbed image and the right column shows the corresponding perturbed explanation.

nations, and the perceptual similarity (LPIPS) between the perturbed and original images for each attack. We can observe that Delta+StAdv, and Delta+StAdv+Recolor attacks are more effective than Delta attacks to manipulate  $\beta$ -smoothing explana-

tions, that is, with a less perceptible perturbation (lower LPIPS value), we can reach a cosd value between manipulated and target explanations very close to the cosd value when attacking the Gradient method. Note that the Smooth Gradient and Uniform Gradient methods present a weaker robustness as compared to the  $\beta$ -smoothing method, that is, under the additive attack setting the cosd value for these two methods is very close to the cosd value for the Gradient method (Figure 3.6 left). However, for reaching the same level of explanation manipulation, a more perceptible perturbation is required when attacking Smooth Gradient and Uniform Gradient methods (Figure 3.6 right). The effect of the non-additive attacks is less significant on the Smooth and Uniform Gradient methods. However we can still observe that the non-additive attacks are more successful against these methods, that is, a smaller cosd value can be reached for an equal or even smaller LPIPS value compared to the additive attacks. Taken together, these results show that the  $\beta$ -smoothing explanations show better robustness to additive attacks as compared to Smooth Gradient and Uniform Gradient explanations, however, they can be manipulated significantly by combining additive and non-additive attacks. Moreover, Smooth Gradient and Uniform Gradient explanations are also more vulnerable to non-additive attacks and hence such attacks should be considered as a threat to the robustness of these methods. As an example, we visually compare the effectiveness of Delta and Delta+StAdv+Recolor attacks against different explanation methods in Figure 3.7.

### 3.4 Evaluation of ad-hoc approaches

Here, we recreate the experiments of Section 3.3 for the ad-hoc approaches. We study the explanations of networks trained with curvature regularization (CURE) [107], and adversarial training [111]. Training with CURE, regularizes the eigenvalue of the input hessian with maximum absolute value and is similar to SSR, which was shown to improve the robustness of explanations against additive attacks [62]. Adversarial training also smooths the decision surface and can provide more robust explanations.

For the experiments in this section, we used a ResNet-18 network trained with CURE and an adversarially trained ResNet-18 network trained on adversarial examples with  $\ell_\infty$  norm of the perturbations upper bounded by  $8/255$  [127]. Both networks are trained on CIFAR-10 dataset [128].

### 3.4.1 Quality of explanations of ad-hoc approaches

**Cascade randomization of model parameters.** We evaluate the sensitivity of explanations of the networks trained with CURE and adversarial training using the cascade randomization of model parameters test.

The visual results of this test for CURE and adversarial training are shown in Figure 3.8. We can observe that the explanations from both of these networks show less sensitivity to the model parameters compared to the explanations of a network trained in a standard way. Specifically, the explanation of an adversarially trained network is highly invariant to model parameters. This can also be observed in the Spearman rank correlation between the original and randomized explanations shown in Figure 3.9. These Results show that the explanation of an adversarially trained network is less sensitive to model parameters. *This suggests that the explanation of an adversarially trained network cannot provide helpful visual evidence to debug a model when it is making a wrong prediction.*

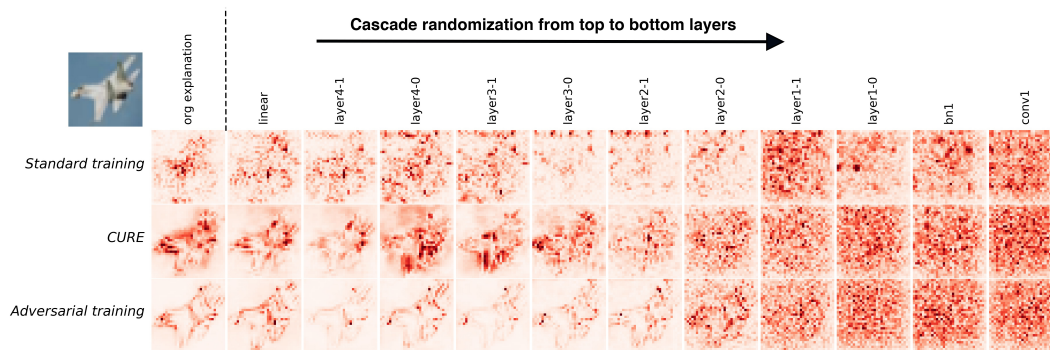


Figure 3.8: Cascade randomization of ResNet-18 layers. The first column shows the Gradient explanation for the image ”plane”. Each of the other columns show the effect of randomization of the network parameters up to that layer (inclusive) on Gradient explanations. The explanations of the adversarially trained network do not visually degrade after the randomization of the top layers of the network.

**Sparseness of explanations.** We compare the sparseness of the explanations derived by ad-hoc approaches, using the Gini Index metric. Table 3.5 compares the Gini Index for the explanations of networks trained with different training objectives. These results show that adversarial training helps improve the sparseness of explanations as compared to standard training. Hence the explanations of an adversarially trained network are more *concise*. This is also in line with the results of [124]. However, the results in Table 3.5 indicate that training a network with



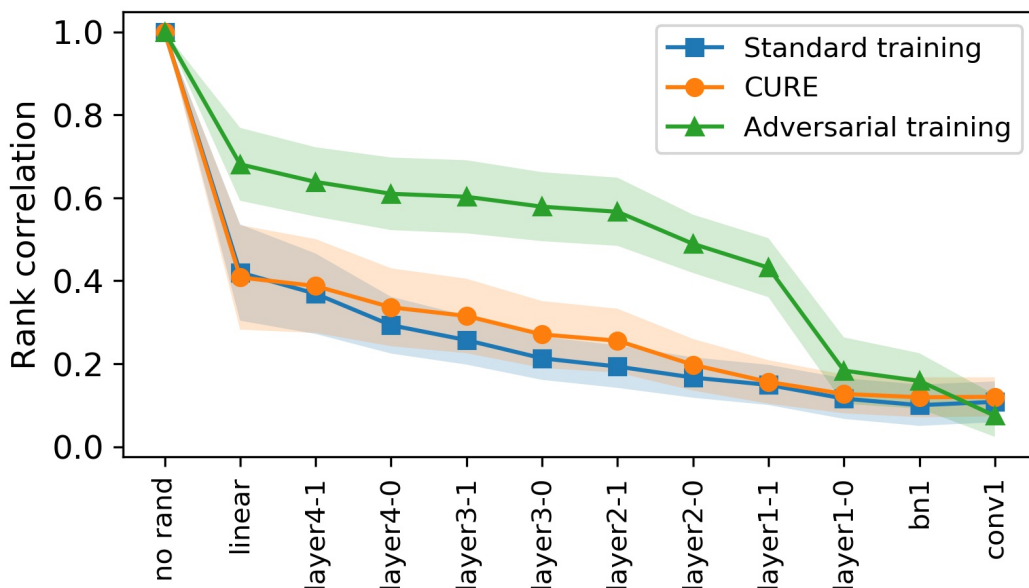


Figure 3.9: Spearman rank correlation between the original and the randomized explanations (averaged over 1000 Images from CIFAR-10). A higher rank correlation value indicates a higher similarity, i.e., *the higher the curve of an explanation method, the less sensitive it is to model parameters.*

CURE does not help to improve the sparseness of explanations as compared to standard training.

Table 3.5: Gini Index for the explanations of a ResNet-18 network (averaged over 1000 images from CIFAR-10). A **larger** Gini Index suggests a more concise explanation.

|            | Standard training | CURE             | Adversarial training               |
|------------|-------------------|------------------|------------------------------------|
| Gini Index | $0.54 \pm 0.035$  | $0.54 \pm 0.045$ | <b><math>0.71 \pm 0.054</math></b> |

**Explanation Infidelity.** To compare the fidelity of explanations derived by ad-hoc approaches to the predictor function, we use the Infidelity metric with square perturbation [43]. Table 3.6 shows the results for randomly selected images from CIFAR-10. *A lower infidelity value indicates better fidelity of the explanation to the predictor function.* From these results, we can observe that the explanation of the networks trained with CURE and adversarial training have a higher fidelity to the predictor model.

Table 3.6: Infidelity of the explanations of a ResNet-18 network (averaged over 1000 images from CIFAR-10). A **lower** Infidelity value is better.

|            | Standard training | CURE                              | Adversarial training |
|------------|-------------------|-----------------------------------|----------------------|
| Infidelity | $5.69 \pm 4.44$   | <b><math>0.59 \pm 0.34</math></b> | $1.56 \pm 0.76$      |

### Discussion of the results

We compared different quality aspects of the explanations of ad-hoc approaches with the Gradient method. We did not evaluate the class-sensitivity for the ad-hoc approaches as there are no segmentation masks available for CIFAR-10 images. Table 3.7 shows a summary of our results on the different quality aspects of ad-hoc smoothed explanations.

Table 3.7: Comparison between different quality aspects of ad-hoc smoothed explanations and the Gradient method. In each cell, the symbol "×(✓)" indicates that the smoothed explanation is worse (better) than the Gradient method in this quality aspect. The symbol "-" shows that there is no significant difference.

|                                 | CURE | adv-train |
|---------------------------------|------|-----------|
| Sensitivity to model parameters | -    | ×         |
| Sparseness                      | -    | ✓         |
| Fidelity                        | ✓    | ✓         |

### 3.4.2 Robustness of explanations of ad-hoc approaches

We evaluate the robustness of the explanations of ad-hoc approaches against explanation attacks composed of different types of perturbations. We present results for Delta, Delta+StAdv, and Delta+StAdv +Recolor attacks against the explanation of the networks trained with CURE and adversarial training. Figure 3.10 (left) shows the cosine distance (cosd) between the original and the manipulated explanation and Figure 3.10 (right) shows the perceptual similarity between the original and the perturbed image as a result of these attacks. For the adversarially trained network, we can observe that non-additive attacks can more effectively manipulate explanations compared to the additive attacks, that is, a lower cosd value can be reached with equally or less perceptible perturbations as compared to the additive

attacks (Figure 3.10, right). However, even with the strongest attack setting, that is, the Delta+StAdv+Recolor attack, we still cannot get close to the cosd value reached by attacking the explanation of the network trained in a standard way. This shows that adversarial training is indeed an effective method in making explanations more robust and it can show some levels of robustness even against combinations of additive and non-additive attacks. Explanations of the network trained with CURE are in general not as robust as the adversarially trained network, that is, a smaller cosd value is reached when attacking the CURE explanations with an additive attack. However, non-additive attacks have a less significant effect in manipulating the explanation of a CURE network, that is, only a small decrease can be seen in the cosd value compared to the additive attack while the perceptual similarity is also not significantly better. In Figure 3.11, we can also visually compare the result of Delta and Delta+StAdv+Recolor attacks against the explanation of a sample image from CIFAR-10. We can observe that, the additive attack (Delta) almost cannot visually change the explanation of the adversarially trained network while the Delta+StAdv+Recolor attack can manipulate the explanation of this network to be visually very similar to the target explanation (Figure 3.11, last row).

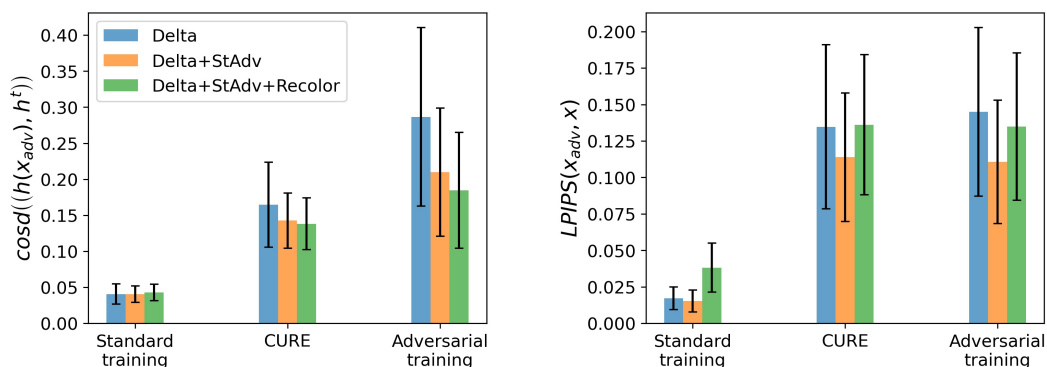


Figure 3.10: Evaluation with cosd (left) and LPIPS (right) metrics for the attacks against ad-hoc approaches. For both cosd and LPIPS, smaller values indicate higher similarity. The results show that non-additive attacks are more effective to manipulate explanations of CURE and adversarially trained networks. All results are averaged over 1024 images from CIFAR-10 (vertical bars indicate standard deviation)

## 3.5 Discussion

Investigating the smoothed explanations is an important topic, since the majority of the efforts in the explainability literature to make robust explanations consists of some

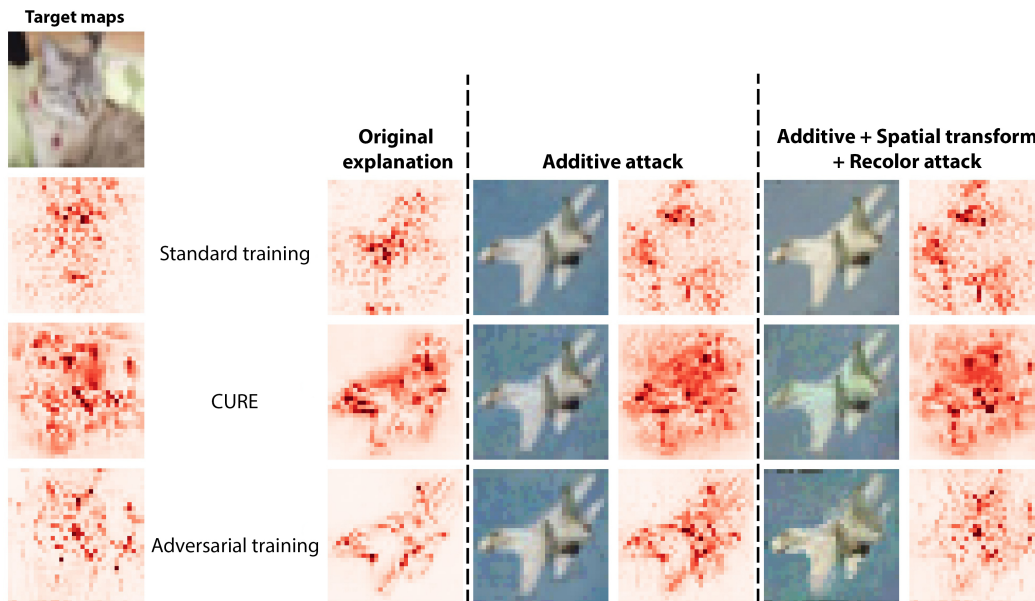


Figure 3.11: Results of explanation attacks against ad-hoc methods. Left: the target image and the corresponding target explanation maps used to manipulate each explanation. Right: Perturbed images together with the manipulated explanations for Delta and Delta+StAdv+Recolor attacks.

form of smoothing [60, 62, 114, 129, 130]. Therefore it is important to understand the advantages and disadvantages of the smoothed explanations in terms of usefulness and the quality of the explanation. The advantage of post-hoc smoothing approaches is that they have a very low computational cost, that is, they only need to post process the explanations and in the case of the Smooth and Uniform Gradient methods they need to compute a few forward passes. However, we showed that explanations derived by these approaches fall short in providing concise explanations that focus only on the object related to the predicted class and discriminating it from the other irrelevant objects in the image. On the other hand the ad-hoc approaches are very costly and require retraining of the network, but they can provide explanations which are superior to the Gradient explanations in terms of conciseness and fidelity to the predictor model. Moreover, the attack results indicate that the explanations by ad-hoc approaches present stronger robustness as compared to post-hoc approaches. Furthermore, it is harder to manipulate ad-hoc explanations with strong attacks that combine additive and non-additive perturbations. Taken together, our results show that explanations derived by the ad-hoc approaches not only provide better robustness compared to post-hoc approaches, but they also possess certain desirable qualities which makes them superior to Gradient explanation.

## 3.6 Conclusion

Explanation techniques should be able to increase the transparency of black-box systems such as Neural Networks, by highlighting which of the input features contributed to the decision. In this work we highlighted the shortcomings and strengths of "smoothed explanations".

We have evaluated two aspects of smoothed explanations: a) explanation quality, and b) explanation robustness. In terms of explanation quality, we performed a thorough evaluation of four quality aspects: sensitivity to model parameters, class discriminativeness, sparseness, and Infidelity. Our results show that the smoothed explanations investigated in this paper perform worse than those of the Gradient method in terms of sensitivity to model parameters and class discriminativeness. Moreover, with the exception of explanations of the adversarially trained models, smoothed explanation cannot improve the explanation sparseness. On the other hand, we show that using such smoothing methods helps to improve explanation Infidelity.

We further looked at the robustness of explanations, when inputs are perturbed by a combination of additive and non-additive attacks. To the best of our knowledge, this is the first time such combinations of attacks are used to manipulate explanations. Our experimental results highlighted the fact that non-additive attacks are still a threat for explanation methods, including the smoothed ones. These results also indicate that many problems in explanation robustness can be addressed by making analogies with the domain of prediction robustness. As these two domains are closely related, the solutions already explored in prediction robustness can be potentially helpful to study explanation robustness. This will be the focus of our future work.

## Chapter 4

# Sparse Attacks for Manipulating Explanations in Deep Neural Network Models

**Abstract.** We investigate methods for manipulating classifier explanations while keeping the predictions unchanged. Our focus is on using a sparse attack, which seeks to alter only a minimal number of input features. However, finding sparse adversarial perturbations that manipulate explanations can be difficult due to the NP-hardness of the corresponding  $\ell_0$  minimization problem. Our work presents an efficient and novel algorithm for computing sparse perturbations that alter the explanations, but keep the predictions unaffected. We demonstrate that our approach leads to more effective explanation attacks using fewer features than projected gradient descent with  $\ell_0$  projection. Additionally, our algorithm generates sparser perturbations than existing methods while resulting in greater discrepancies between original and manipulated explanations. It is also possible to conceal the attribution of the  $k$  most significant features in the original explanation by perturbing fewer than  $k$  features of the input data. We present results for both image and tabular datasets, and emphasize the significance of sparse perturbation-based attacks for trustworthy model building in high-stakes applications. Our research reveals important vulnerabilities in explanation methods that should be taken into account when developing reliable explanation methods. Code can be found at <https://github.com/ahmadajal/sparse-expl-attacks>

## 4.1 Introduction

Deep neural networks (DNNs) have achieved exceptional accuracy in classification tasks for image [131, 132], textual [133, 134], and tabular data [135]. However, understanding the reasoning behind DNN predictions remains a difficult task, making them unreliable in critical fields such as finance, law, and medicine where transparency and interpretability are vital. To address this concern, a range of explanation methods have been developed to attribute numerical values to each data feature, indicating its contribution to the model's prediction [98, 136, 65, 115, 100, 12]. This understanding is crucial for high-stakes tasks in which domain experts make decisions based on the model's predictions. For example, regulators in the banking sector may need to ensure that the model used to determine loan eligibility is fair and unbiased. Thus, explanations are essential for building transparency and trust between users and the model [21]. They also aid in debugging the model by revealing potential spurious data correlations that may lead to incorrect predictions during inference [12].

A widely used class of explanation methods, called gradient-based explanations, calculates the gradient of the output logit corresponding to the predicted class with respect to the input [98, 136, 65, 115]. These methods produce an attribution vector that highlights the importance of each data feature in the prediction. However, recent research has shown that gradient-based explanations of neural networks can be easily manipulated through adversarial inputs [102, 60, 103, 104, 105]. By adding small-norm perturbations to the input, an attacker can shift the focus of the explanation towards irrelevant features, while keeping the model's prediction unchanged. This undermines the ability of the explanation to help end-users gain trust in a model's prediction.

For example, by manipulating an explanation, an attacker could potentially mislead end-users into believing that a model does not rely on sensitive features, such as gender or race, when it actually does [61, 137, 138, 139]. Another example is depicted in Figure 4.1. A medical expert may use a model trained on X-ray images of hands to diagnose bone age. However, if the model was trained on a biased dataset, it may rely on a spurious signal, such as the tag with the letter "L" in the image, when predicting the "mid-puberty" class. An appropriate explanation can reveal this reliance on spurious signals and prevent the expert from using an unreliable model. However, a malicious model owner can embed an adversarial noise in the input layer of the model to cause manipulated explanations that could conceal the biasedness of the model. As a result, the medical expert will be misled into thinking that the model relies on appropriate features to make predictions. Therefore, the expert may

trust this model since the explanations seem reasonable and the model has a high accuracy as the attack does not change the predictions!

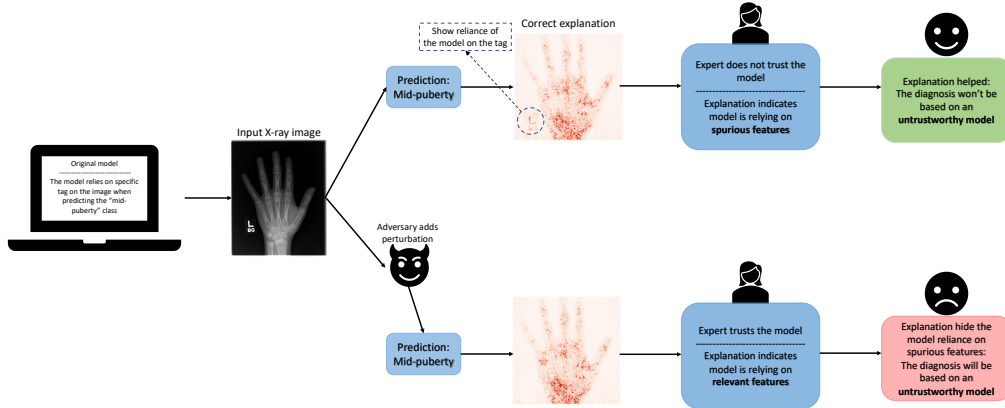


Figure 4.1: An example illustrating how manipulated explanations can lead to the usage of an untrustworthy model in a real-world medical application. This figure is inspired from [59].

Explanation attacks are a type of method in which an attacker aims to manipulate the output of an explanation method while keeping the model’s prediction unchanged. This can be achieved by either altering the input to the model, such as by adding a perturbation to the input [60, 102, 68], or by manipulating the model’s weights [61]. In this work, we primarily focus on the scenario where an attacker crafts an imperceptible perturbation to be added to the input in order to manipulate the explanation. Additionally, we consider explanation methods that are based on calculating the gradient of the output logit with respect to the input, also known as gradient-based explanations [98, 136, 65, 100].

Most recent works on explanation attacks only consider  $\ell_\infty$  constraints on the input perturbation to create an imperceptible perturbation [62, 102]. They use a Projected Gradient Descent (PGD) attack [111] in which the attacker projects the perturbed input onto the  $\ell_\infty$  ball with some radius  $\delta$  around the original input. In general, PGD attacks work very well in both prediction attack and explanation attack scenarios. However, the perturbations resulting from these attacks are very dense and they change almost every feature of the original input. Such perturbations are less likely to occur in real-world scenarios. As a result, several *sparse attacks* have been proposed in recent works to manipulate a classifier’s prediction by perturbing only a few features [140, 141, 142, 143].

In explanation attacks, sparse perturbations can create a more realistic scenario by only altering a small number of features. For example, hiding the importance of



sensitive features like gender or race by only changing a small fraction of categorical or numerical features is a more realistic scenario than changing a large number of features.

In many applications, the goal of an explanation attack is to decrease the importance of a specific set of features in the explanation, such as in a top- $k$  attack. A naive solution in such cases is to directly zero out the information corresponding to that set of features. However, such a naive method does not guarantee that the model’s prediction will remain unchanged. A more effective solution is to use a carefully chosen sparse perturbation that alters the minimum number of features necessary to reduce the importance of the targeted set of features while maintaining the prediction accuracy.

To address the aforementioned scenarios in explanation attacks, we introduce *sparse explanation attacks*, which manipulate explanations by only altering a small fraction of input features. There are two main challenges in finding perturbations that affect the least number of features to manipulate an explanation:

1. Respecting the dynamic range of the data. (for example,  $[0,1]$  for images)
2. Maintaining the prediction of the model after applying the sparse perturbation.

Figure 4.2 illustrates an example of a sparse explanation attack, where altering only 0.3% of the image pixels changes the focus of the explanation.

Our main **contributions** are :

- We propose a novel greedy-based attack called **GrID** to manipulate explanations using sparse perturbations while respecting the valid range of data and keeping the model’s prediction unchanged.
- We present the outcome of our attacks in both **image** and **tabular** data and under various gradient-based explanation methods and demonstrate that our attack can successfully manipulate explanations by perturbing fewer features compared to baseline attacks.
- Finally, we explain in detail an example case that uses sparse attacks to hide the bias of a model.

## 4.2 Background

Explanation attacks aim to change the focus of an explanation away from the important features identified by the original explanation. For example, in the context

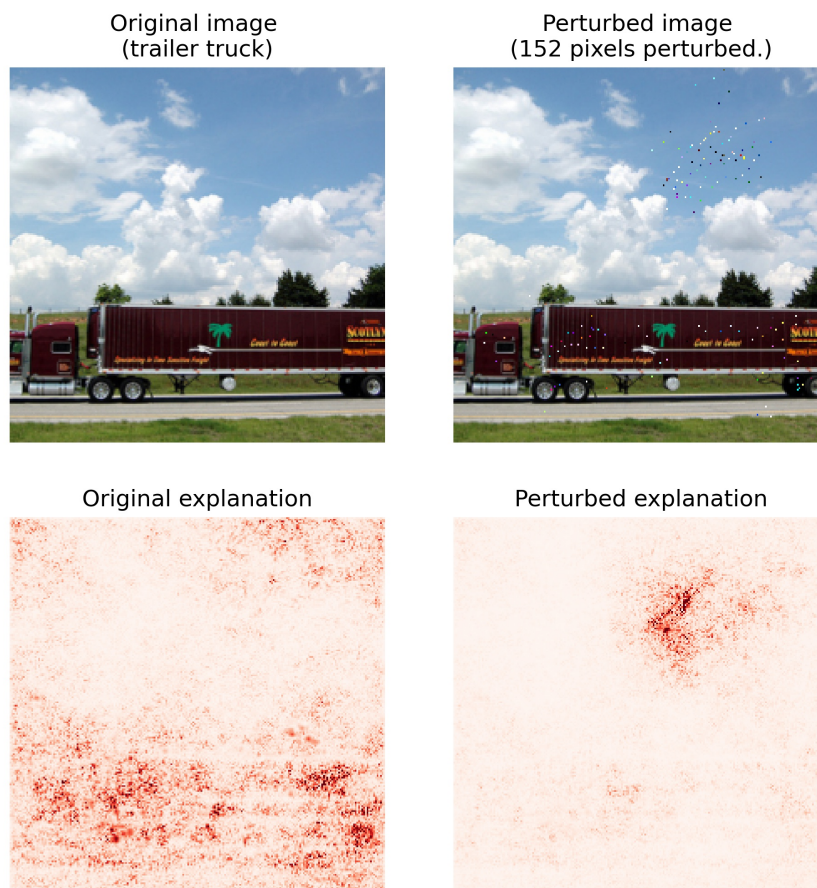


Figure 4.2: An example of a sparse explanation attack (top- $k$  attack with  $k = 1000$ ). By only perturbing 0.3% of the pixels, the focus of the explanation shifts from the object corresponding to the predicted class (trailer truck) to the background.

of image classifiers, this could mean reducing the importance of the region related to the predicted class in the explanation heatmap, and highlighting irrelevant regions as important instead (e.g., Figure 4.2).

To alter an explanation, we use the concept of the top- $k$  attack [102]. In this attack, the goal is to find a perturbed input such that the top- $k$  features in the perturbed explanation have the least intersection with the top- $k$  features in the original explanation. For a classifier  $f(\cdot)$  and an explanation function  $h(\cdot)$ , the perturbed input  $\mathbf{x}_{adv}$  can be computed from an input  $\mathbf{x}$  by minimizing the following objective function:

$$\min_{\mathbf{x}_{adv}} \sum_{i \in \mathcal{B}} |h(\mathbf{x}_{adv})_i| + \gamma (\|f(\mathbf{x}_{adv}) - f(\mathbf{x})\|^2) \quad (4.1)$$

where  $\mathcal{B}$  indicates the set of the  $k$  largest values in the original explanation  $h(\mathbf{x})$ . The first term of (4.1) aims to reduce the importance of the features in the manipulated explanation that correspond to the top- $k$  values in the original explanation. The second term ensures that the model’s prediction for the perturbed input remains the same as for the original input. The choice of  $k$  in the top- $k$  attack is dependent on the dimensionality of the data. To create a manipulated explanation that is significantly different from the original,  $k$  should be large enough in relation to the dimensionality of the data.

It’s worth noting that the concept of a top- $k$  attack can be extended to cases where the attacker aims to hide the importance of a specific set of features in the explanation, rather than only high-ranked features. In this case, a new set  $\mathcal{C}$  is defined, consisting of the targeted features, and the first term of (4.1) is replaced with  $\sum_{i \in \mathcal{C}} |h(\mathbf{x}_{adv})_i|$ . For instance, if the goal is to make a gender/race discriminative model appear fair in a loan eligibility prediction scenario, an attacker can choose the set  $\mathcal{C} = \{\text{gender, race}\}$  to reduce the importance of these features in the explanation.

## 4.3 Baseline sparse attacks

One might expect that reducing the top- $k$  attribution values of an explanation can be done by carefully selecting  $k$  features of the input to manipulate. However, there are two main challenges with this approach: 1) How to find the appropriate amount of perturbation applied to each selected feature such that the top- $k$  attributions change significantly, but the prediction of the model remains the same. 2) How to find a valid perturbed input with respect to the valid range of the data. Therefore, one cannot arbitrarily select  $k$  features of the input and perturb them with a large value, as this will likely violate the valid range of the data or change the prediction of the model. In the following sections, we introduce two simple methods to find sparse perturbations that both respect the valid range of the data and preserve the model prediction. These methods will be used as baseline sparse explanation attacks.

### 4.3.1 Single-step attack

The objective of the top- $k$  attack is to reduce the importance of the top- $k$  features in the original explanation, which naturally motivates a sparse solution. One approach

to this problem is to only perturb the  $k$  features corresponding to the top- $k$  attribution values. To compute such a perturbation, it is sufficient to compute the direction in which the objective (4.1) can be reduced the most, which is the opposite direction of the gradient of (4.1). Then, only the dimensions of the gradient corresponding to the top- $k$  attribution values are used to update the original input  $\mathbf{x}$  and derive  $\mathbf{x}_{adv}$ . This attack is called a *Single-step attack* because it only performs a single step of gradient descent. The attack process is shown in Algorithm 4.

---

**Algorithm 4** Single-step attack
 

---

- 1: **Input:** original input  $\mathbf{x}$ , predictor function  $f$ , explanation function  $h$
  - 2: **Parameter:** top- $k$  attack parameter  $k$ , step size  $\eta$ , prediction loss coefficient  $\gamma$
  - 3: **Initialize:**  $\mathbf{m} \leftarrow \mathbf{0}$ ,  $\mathcal{B} \leftarrow \text{argtop}_k(h(\mathbf{x}))$ ,  $\mathbf{x}_{adv} \leftarrow \mathbf{x}$
  - 4:  $\mathcal{L} = \sum_{i \in \mathcal{B}} h(\mathbf{x}_{adv})_i + \gamma (\|f(\mathbf{x}_{adv}) - f(\mathbf{x})\|^2)$
  - 5:  $\mathbf{m}_{\{d|d \in \mathcal{B}\}} = 1$
  - 6:  $\mathbf{x}_{adv} = \text{Clip}(\mathbf{x}_{adv} - \eta \cdot \mathbf{m} \cdot \nabla_{\mathbf{x}_{adv}} \mathcal{L})$
  - 7: **Return:**  $\mathbf{x}_{adv}$
- 

We define a mask  $\mathbf{m}$  to indicate the features of the gradient vector that will be used to derive  $\mathbf{x}_{adv}$  from  $\mathbf{x}$ . Initially, we have  $\mathbf{x}_{adv} = \mathbf{x}$ , so the gradient of the second term of  $\mathcal{L}$  is zero. As a result, by choosing a large  $\eta$  in the Single-step attack, the first term of  $\mathcal{L}$  (the explanation loss) can be reduced arbitrarily. However, choosing a large step size  $\eta$  will cause the model prediction to change in the Single-step attack, as the update direction increases the output loss (second term in (4.1)). Therefore, the step size  $\eta$  in this attack should be small enough such that the model prediction does not change.

### 4.3.2 PGD attack with $\ell_0$ box constraint

Recent explanation attack research has primarily focused on using a Projected Gradient Descent (PGD) attack with an  $\ell_\infty$  constraint [102, 62]. This attack ensures that the perturbation applied to each feature is always less than a certain value  $\delta$ , the  $\ell_\infty$  ball radius. To create *sparse* perturbations, a similar attack can be modified to use an  $\ell_0$  constraint instead. The  $\ell_0$  ball of radius  $k$  around  $\mathbf{x}$  includes all points that differ from  $\mathbf{x}$  in no more than  $k$  dimensions. By applying an  $\ell_0$  projection after each step of gradient descent, we can create a PGD method with  $\ell_0$  constraint that results in a sparse perturbation. Formally, the  $\ell_0$  projection is defined as:

**Definition 3** ( $\ell_0$  projection [141]). *Given an input  $\mathbf{x} \in [0, 1]^d$ , we wish to project a*

point  $\mathbf{y} \in \mathbb{R}^d$  onto the set

$$C(\mathbf{x}) = \left\{ \mathbf{z} \in \mathbb{R}^d \mid \sum_{i=1}^d \mathbf{1}_{|z_i - x_i| > 0} \leq k, 0 \leq z_i \leq 1 \right\}$$

Or, in other words, the set of points  $\mathbf{z}$ , which respect the valid range of the data, i.e.,  $\mathbf{z} \in [0, 1]^d$  and differ from  $\mathbf{x}$  in at most  $k$  dimensions.

To compute this projection, it suffices to solve the following constrained minimization problem:

$$\begin{aligned} \min_{\mathbf{z}} \sum_{i=1}^d (\mathbf{y}_i - \mathbf{z}_i)^2 \\ \text{s. th. } 0 \leq \mathbf{z}_i \leq 1, \quad i = 1, \dots, d, \quad \sum_{i=1}^d \mathbf{1}_{|z_i - x_i| > 0} \leq k \end{aligned} \tag{4.2}$$

The projection discovers the closest point to  $\mathbf{y}$  in euclidean distance, that respects the valid range of the data and differs from  $\mathbf{x}$  in only  $k$  dimensions or less. Croce et al [141] introduced an efficient solution for the above projection that scales well to data dimensionality. We refer the readers to the main paper [141] for the details of this solution.

In the rest of this paper, we denote the PGD attack with  $\ell_0$  box constraint as PGD<sub>0</sub> attack. Algorithm 5 summarizes the PGD<sub>0</sub> explanation attack. In this algorithm, the radius of the  $\ell_0$  ball is set to be  $k$  in the top- $k$  attack. In other words, we are interested in manipulating the top- $k$  attributions by only perturbing, at most,  $k$  features.

The main drawback of the PGD<sub>0</sub> attack is that it always creates perturbations with exactly  $k$  non-zero elements. In other words, the number of perturbed features is defined before the attack and these features must be perturbed in the attack. As a result, this attack may perturb many redundant features and therefore may not result in the minimum number of perturbed features required to manipulate the explanation.

## 4.4 GrID attack: Greedy Increasing and Decreasing the perturbation density

To address the pre-defined number of perturbed features in the PGD<sub>0</sub> attack, we propose a novel greedy-based attack called GrID (Greedy Increasing and Decreasing

**Algorithm 5** PGD<sub>0</sub> attack

- 
- 1: **Input:** original input  $\mathbf{x}$ , predictor function  $f$ , explanation function  $h$
  - 2: **Parameter:** top- $k$  attack parameter  $k$ , step size  $\eta$ , prediction loss coefficient  $\gamma$ , number of iterations  $N$
  - 3: Initialize:  $\mathcal{B} \leftarrow \text{argtop}_k(h(\mathbf{x}))$ ,  $\mathbf{x}_{adv} \leftarrow \mathbf{x}$ ,  $j \leftarrow 0$
  - 4: **while**  $j < N$  **do**
  - 5:      $\mathcal{L} = \sum_{i \in \mathcal{B}} h(\mathbf{x}_{adv})_i + \gamma (\|f(\mathbf{x}_{adv}) - f(\mathbf{x})\|^2)$
  - 6:      $\mathbf{x}_{adv} = \mathbf{x}_{adv} - \eta \cdot \nabla_{\mathbf{x}_{adv}} \mathcal{L}$
  - 7:      $\mathbf{x}_{adv} = \text{project-}\ell_0(\mathbf{x}_{adv}, \mathbf{x}, k)$
  - 8:      $j = j + 1$
  - 9: **end while**
  - 10: **Return:**  $\mathbf{x}_{adv}$
- 

the perturbation density). This attack consists of two stages. In the first stage, we iteratively select the  $n$  most important features to perturb based on the gradient of the loss (4.1) with respect to the input until a successful attack, i.e. until the top- $k$  intersection between the original and the manipulated explanation is zero. In the second stage, we perform a greedy search to drop as many less important features as possible to further improve the sparsity.

We denote  $\mathbf{x}$  as the original input and  $y$  as its corresponding ground-truth label. Let  $f$  be the predictor model and  $h$  be the function being used to compute the explanation. Moreover, we denote the set of top- $k$  coordinates of the original and manipulated explanations as  $\mathcal{B}$  and  $\mathcal{B}_{adv}$  respectively. In the first stage of the attack, we iteratively increase the density of the perturbation until we find an adversarial input  $\mathbf{x}_{adv}$  for which  $\mathcal{B}_{adv} \cap \mathcal{B} = \emptyset$ , i.e. there is no intersection between the top- $k$  attributions of the original and manipulated explanations. For clarity, we define a mask  $\mathbf{m}$  indicating the features of the input that have been already perturbed and initialize it with all zeros. In each iteration, we compute the gradient of the attack loss (4.1) with respect to the current adversarial input  $\mathbf{x}_{adv}$ . The features corresponding to the highest values of the gradient have a more significant effect in manipulating the explanation. Therefore, in each iteration, we update the mask  $\mathbf{m}$  by adding the features with the highest gradient value that have not been already selected.

$$\begin{aligned}
 \mathbf{g} &= \nabla_{\mathbf{x}_{adv}} \mathcal{L} \cdot (1 - \mathbf{m}) \\
 d_1, d_2, \dots, d_n &\leftarrow \text{argtop}_n(|\mathbf{g}|), \quad \mathbf{m}_{d_1, d_2, \dots, d_n} = 1
 \end{aligned} \tag{4.3}$$

Then we update the adversarial input as follows:

$$\mathbf{x}_{adv} = \text{Clip}(\mathbf{x}_{adv} - \eta \cdot \mathbf{m} \cdot \nabla_{\mathbf{x}_{adv}} \mathcal{L}) \quad (4.4)$$

where  $\eta$  is the step size and the Clip operator is used to make sure that the computed adversarial input stays in the valid range of the data. This stage of the attack ends when we reach a zero top- $k$  intersection between the original and manipulated explanations. However, in practice, we can also have a *perturbation budget*, i.e. the maximum allowed number of features to perturb. In such cases, we can end this stage either when we reach a zero top- $k$  intersection or when the number of perturbed features reaches the perturbation budget.

The first stage of the GrID attack searches for sparse perturbations in a greedy way by exploiting the latest updated gradient. This ensures that the selected features to perturb in each iteration are the best choice and avoids selecting redundant features to perturb contrary to the PGD<sub>0</sub> attack. Moreover, in each iteration, we update  $\mathbf{x}_{adv}$  in the direction of all of the so far selected features. This further helps to achieve an adversarial input with zero top- $k$  intersection faster.

Previous works that used sparse perturbations to change the prediction of a model did not incorporate a step size in their attacks ([141, 142]). They used the sign of the gradient vector to update  $\mathbf{x}_{adv}$ , i.e. in each iteration they perturb a feature to the minimum or maximum value of the data valid range, depending on the direction of the gradient vector. However, for explanation attacks, we found that this approach often leads to a change in prediction after the attack. Increasing the coefficient  $\gamma$  in (4.1) does not solve this issue, as it causes the explanation loss term to increase. Therefore, we decided to include a step size in the sparse explanation attacks to have better control over the output loss term and prevent changes in prediction after the attack.

The first stage of the GrID attack achieves an adversarial input that provides zero top- $k$  intersection, However, some of the selected perturbed features can be redundant due to the greedy nature of their selection process. Therefore, the GrID attack also incorporates a second reducing stage to further improve the sparsity of the adversarial input.

Formally, we denote  $\mathbf{r} = \mathbf{x}_{adv} - \mathbf{x}$  as the adversarial noise and define a set  $S$  that contains the non-zero coordinates of  $\mathbf{r}$ . We hypothesize that the features with smaller noise values have a less significant contribution to creating a successful adversarial input. Therefore in each iteration, we drop the noise component with the minimum magnitude. Then we check whether the top- $k$  intersection is still zero and the model prediction is still intact after dropping the noise component. If this is the case then this shows that the selected noise component is redundant and we update

the adversarial noise and input by removing this noise component. However, if after removing the noise component either the top- $k$  intersection becomes non-zero or the prediction changes then this shows that the selected component was a necessary adversarial component and hence we keep it. In any case, at the end of each iteration, we remove the selected component from the set  $S$  as we do not want to check this component anymore. The whole process of the GrID attack is shown in Algorithm 6.

---

**Algorithm 6** GrID attack
 

---

```

1: Input: original input  $\mathbf{x}$ , true label  $y$ , predictor function  $f$ , explanation function  $h$ 
2: Parameter: top-k attack parameter  $k$ , step size  $\eta$ , prediction loss coefficient  $\gamma$ , select number  $n$ 
3: Initialize:  $\mathbf{m} \leftarrow \mathbf{0}$ ,  $\mathcal{B} \leftarrow \text{argtop}_k(h(\mathbf{x}))$ ,  $\mathcal{B}_{adv} \leftarrow \mathcal{B}$ 
4: Initialize:  $\mathbf{x}_{adv} \leftarrow \mathbf{x}$ 
5: while  $\mathcal{B}_{adv} \cap \mathcal{B} \neq \emptyset$  do
6:    $\mathcal{L} = \sum_{i \in \mathcal{B}} h(\mathbf{x}_{adv})_i + \gamma (\|f(\mathbf{x}_{adv}) - f(\mathbf{x})\|^2)$ 
7:    $\mathbf{g} \leftarrow \nabla_{\mathbf{x}_{adv}} \mathcal{L} \cdot (1 - \mathbf{m})$ 
8:    $d_1, d_2, \dots, d_n \leftarrow \text{argtop}_n(|\mathbf{g}|)$ 
9:    $\mathbf{m}_{d_1, d_2, \dots, d_n} = 1$ 
10:   $\mathbf{x}_{adv} = \text{Clip}(\mathbf{x}_{adv} - \eta \cdot \mathbf{m} \cdot \nabla_{\mathbf{x}_{adv}} \mathcal{L})$ 
11:   $\mathcal{B}_{adv} \leftarrow \text{argtop}_k(h(\mathbf{x}_{adv}))$ 
12: end while
13:
14:  $\mathbf{r} \leftarrow \mathbf{x}_{adv} - \mathbf{x}$ ,  $S \leftarrow \{d | d \in \mathbb{N}, \mathbf{r}_d \neq 0\}$ 
15: while  $S \neq \emptyset$  do
16:    $d \leftarrow \underset{d \in S}{\text{argmin}} |\mathbf{r}_d|$ 
17:    $\mathbf{r}' \leftarrow \mathbf{r}$ ,  $\mathbf{r}'_d \leftarrow 0$ 
18:    $\mathbf{x}'_{adv} \leftarrow \mathbf{x} + \mathbf{r}'$ ,  $\mathcal{B}'_{adv} \leftarrow \text{argtop}_k(h(\mathbf{x}'_{adv}))$ 
19:   if  $\mathcal{B}'_{adv} \cap \mathcal{B} = \emptyset$  and  $\text{argmax}(f(\mathbf{x}'_{adv})) = y$  then
20:      $\mathbf{r} \leftarrow \mathbf{r}'$ ,  $\mathbf{x}_{adv} \leftarrow \mathbf{x}'_{adv}$ 
21:   end if
22:    $S \leftarrow S - \{d\}$ 
23: end while
24: Return:  $\mathbf{x}_{adv}$ 

```

---

Note that in practice, if we apply a restriction on the perturbation budget, then for some samples we may not reach a zero top- $k$  intersection at the end of the first stage. Therefore, we can alter the condition in the second stage to check if the number



of elements in the top- $k$  intersection set does not increase after removing a noise component, i.e.  $|\mathcal{B}'_{adv} \cap \mathcal{B}| \leq |\mathcal{B}_{adv} \cap \mathcal{B}|$ . This way we can make sure that we can have at least the same explanation manipulation as at the end of the first stage but with a better sparsity.

## 4.5 Experiments

To demonstrate the effectiveness of the GrID attack in altering the explanations provided by deep neural network architectures, we conduct extensive experiments and investigate the following research questions:

- **RQ1:** How does the GrID attack compare to the baseline sparse explanation attacks in terms of manipulating the explanation and the sparsity of the perturbation?
- **RQ2:** How do the top- $k$  intersection and the sparsity of the perturbation change as we increase the  $k$  in the top- $k$  attack?
- **RQ3:** What is the effect of the decreasing stage in the GrID attack in improving the sparsity of the perturbation?
- **RQ4:** How well does the GrID attack perform in fair-washing a biased model?

We present our results on both image and tabular datasets to demonstrate the generality of the techniques presented. To the best of our knowledge, this is the first work that presents attacks on both types of datasets.

To evaluate the success of an attack in manipulating the explanation, we define the top- $k$  intersection loss.

$$\text{top-}k \text{ intersection loss} = \frac{|\mathcal{B}_{adv} \cap \mathcal{B}|}{k} \quad (4.5)$$

where  $\mathcal{B}_{adv} = \text{argtop}_k(h(\mathbf{x}_{adv}), \mathcal{B} = \text{argtop}_k(h(\mathbf{x}))$

A lower value for the top- $k$  intersection loss indicates that the explanation has been manipulated to a greater degree, and therefore the attack was more successful. In the extreme cases where  $|\mathcal{B}_{adv} \cap \mathcal{B}| = k$ , and  $|\mathcal{B}_{adv} \cap \mathcal{B}| = 0$ , this metric equals one, and zero respectively.

To provide a complete analysis, we also calculate the total number of features perturbed. For color images with 3 channels (red, green, blue), this value is computed

with respect to the total pixels of the image (not individual features). For example, for images in the CIFAR-10 dataset with size  $3 \times 32 \times 32$ , we compute the percentage of the total 1024 pixels that are perturbed. Moreover, we also report the accuracy of the model for the perturbed samples to show how successful an attack is in terms of keeping the prediction intact while manipulating the explanation.

### 4.5.1 Evaluating explanation manipulation and sparsity

We evaluate the sparse attacks on the explanations provided by four gradient-based explainability methods: Gradient [98], Input times Gradient [65], Integrated Gradients [136], and DeepLIFT [65]. Across all attack settings, *we set the perturbation budget to be equal to  $k$  in top- $k$  loss* because we want to observe whether the sparse attack can manipulate the top- $k$  attributions by altering at most  $k$  features of the data.

#### Attacks in image datasets

We evaluate the sparse explanation attacks on the explanations of a ResNet-18 [131] network trained on the CIFAR-10 [128] dataset and a VGG-16 network trained on the Imagenet [119] dataset. For each attack and explanation method, we run the attack on several images and report the average top- $k$  intersection loss, the average number of perturbed pixels, and the accuracy of the model on the perturbed images. For the top- $k$  attack, we set  $k = 20$  for CIFAR-10 and  $k = 1000$  for Imagenet datasets, in line with past work on image datasets [62]. These values of  $k$  are sufficient to manipulate the explanations of CIFAR-10 and Imagenet images. For all cases, we set the pixel perturbation budget to be  $k$ . Moreover, for the GrID attack, we set  $n = 1$  when attacking the CIFAR-10 explanations. When attacking Imagenet explanations, since these images are high dimensional, perturbing only one pixel in each iteration of the GrID’s attack first stage would have made the attack very slow. Therefore, we decided to perturb  $n = 5$  pixels in each iteration of the first stage.

Tables 4.1 and 4.2 show the results of the sparse attacks on the explanations of networks trained on CIFAR-10 and Imagenet datasets, respectively. The single-step attack performs worse than the other attacks because it only applies one step of optimization. Since in the single optimization step of this attack, only the explanation loss term gets reduced, this attack also performs worse than the other attacks in terms of keeping the model prediction intact and its adversarial samples have lower accuracy. The PGD<sub>0</sub> and the GrID attacks can successfully reach a top- $k$  intersection loss close to zero and manipulate the explanation significantly. Moreover, both of these attacks have an adversarial accuracy very close to 100% meaning that they

can successfully preserve the model prediction while manipulating the explanation. For the CIFAR-10 dataset, our GrID attack can reach a lower top- $k$  intersection loss compared to PGD<sub>0</sub> attack while perturbing a significantly less number of pixels. For the Imagenet dataset, however, the top- $k$  intersection loss for the GrID attack is slightly larger than the PGD<sub>0</sub> attack. This could be due to the fact that we need to perform more optimization steps to manipulate Imagenet explanations as these images are high dimensional. Therefore the PGD<sub>0</sub> perturbation gets more updated during the PGD<sub>0</sub> attack which leads to the top- $k$  intersection loss being slightly smaller.

Table 4.1: Evaluation of the sparse attacks against the explanations of a ResNet-18 trained on CIFAR-10. We choose  $k = 20$  in the top- $k$  attack and set the perturbation budget to 20. Results are averaged over 1000 images. (Grad~Gradient, IG~Integrated Gradients, top- $k$  ints~ average top- $k$  intersection loss, # perturb~ average number of perturbed pixels)

|                        |                                | Single-step | PGD <sub>0</sub> | GrID         |
|------------------------|--------------------------------|-------------|------------------|--------------|
| <b>Grad</b>            | <b>top-<math>k</math> ints</b> | 0.22        | 0.06             | <b>0.017</b> |
|                        | <b># perturb</b>               | 20          | 20               | <b>12.15</b> |
|                        | <b>Accuracy (%)</b>            | 92.83       | 99.79            | <b>99.79</b> |
| <b>IG</b>              | <b>top-<math>k</math> ints</b> | 0.28        | 0.049            | <b>0.019</b> |
|                        | <b># perturb</b>               | 20          | 20               | <b>14.71</b> |
|                        | <b>Accuracy (%)</b>            | 92.94       | 99.83            | <b>100</b>   |
| <b>Grad×<br/>input</b> | <b>top-<math>k</math> ints</b> | 0.22        | 0.052            | <b>0.021</b> |
|                        | <b># perturb</b>               | 20          | 20               | <b>13.15</b> |
|                        | <b>Accuracy (%)</b>            | 90.86       | 99.27            | <b>100</b>   |
| <b>DeepLift</b>        | <b>top-<math>k</math> ints</b> | 0.22        | 0.054            | <b>0.02</b>  |
|                        | <b># perturb</b>               | 20          | 20               | <b>13.3</b>  |
|                        | <b>Accuracy (%)</b>            | 90.76       | 99.27            | <b>99.72</b> |

Figure 4.3 provides a visual comparison of the results of the sparse attacks against the Gradient explanation for a sample image from the Imagenet dataset. By comparing the visual heatmaps of the original and manipulated explanations for GrID and PGD<sub>0</sub> attacks, we can see that features with high attribution values in the manipulated explanation are not relevant to the predicted class (trailer truck). It can also be observed that the sparse perturbation resulting from the single-step attack is less noticeable. This is because the single-step attack performs only one optimization step with a small step-size to keep the prediction unchanged. However,

Table 4.2: Evaluation of the sparse attacks against the explanations of a VGG-16 trained on Imagenet. We choose  $k = 1000$  in the top- $k$  attack and set the perturbation budget to 1000. Results are averaged over 500 samples from Imagenet.

|  |                                | Single-step | PGD <sub>0</sub> | GrID          |
|--|--------------------------------|-------------|------------------|---------------|
| <b>Grad</b>                              | <b>top-<math>k</math> ints</b> | 0.16        | 0.029            | <b>0.023</b>  |
|  | <b># perturb</b>               | 1000        | 1000             | <b>505.6</b>  |
|  | <b>Accuracy (%)</b>            | 78.19       | <b>99.13</b>     | 97.38         |
| <b>IG</b>                                | <b>top-<math>k</math> ints</b> | 0.23        | <b>0.048</b>     | 0.06          |
|  | <b># perturb</b>               | 1000        | 1000             | <b>422.80</b> |
|  | <b>Accuracy (%)</b>            | 79.21       | <b>100</b>       | 95.05         |
| <b>Grad<math>\times</math><br/>input</b> | <b>top-<math>k</math> ints</b> | 0.19        | <b>0.036</b>     | 0.044         |
|  | <b># perturb</b>               | 1000        | 1000             | <b>500.89</b> |
|  | <b>Accuracy (%)</b>            | 81.69       | <b>99.71</b>     | 97.1          |
| <b>DeepLift</b>                          | <b>top-<math>k</math> ints</b> | 0.19        | <b>0.034</b>     | 0.042         |
|  | <b># perturb</b>               | 1000        | 1000             | <b>497.65</b> |
|  | <b>Accuracy (%)</b>            | 80.52       | <b>99.42</b>     | 98.25         |

this attack is not successful and the resulting manipulated explanation is hardly different from the original explanation. Furthermore, the GrID attack can reach a manipulated explanation with a much sparser perturbation compared to the PGD<sub>0</sub> attack. The sparser perturbation from the GrID attack also results in a less perceptible perturbation compared to the PGD<sub>0</sub> attack.

Note that in Figure 4.3, although the GrID attack provides a very sparse perturbation, still some of the perturbed pixels are visually perceptible in the image. This is due to the fact that in the GrID attack, we only pursue the objective of achieving a perturbation that is as sparse as possible and we do not have any restriction on the maximum magnitude of the perturbation. Therefore, in some cases, the perturbed pixels may be visible due to the high magnitude of the perturbation. However, our algorithm can be easily adapted to also accept an  $\ell_\infty$  norm restriction to reach a less visible perturbation. Note that with such additional restriction, there will be a trade-off between sparsity of the perturbation and imperceptibility.

### Attacks in tabular datasets

We evaluate sparse explanation attacks against the explanation of tabular datasets. For training DNNs on tabular data, we used the benchmark models from [135]. We

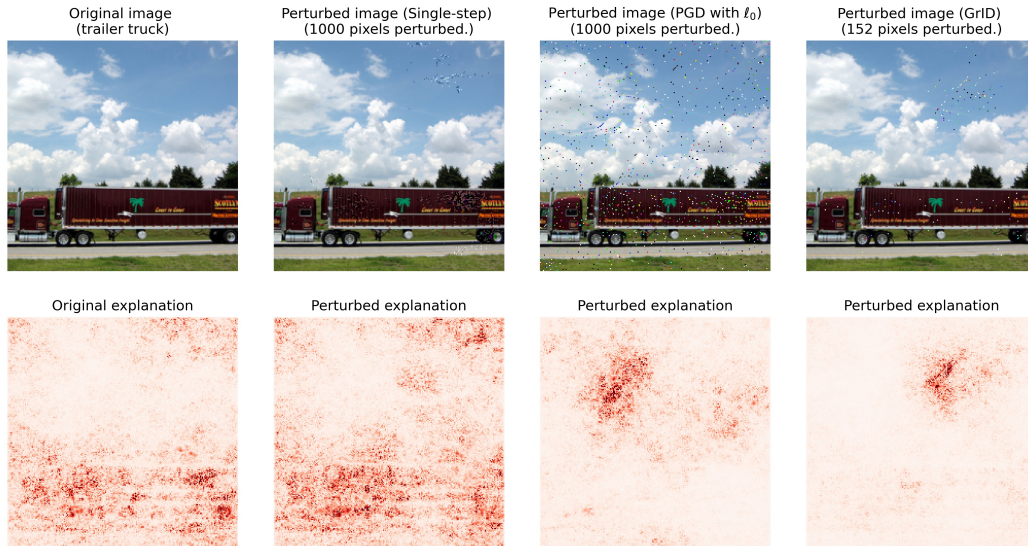


Figure 4.3: Perturbed images and corresponding perturbed explanations resulting from sparse attacks against the explanation of a VGG-16 network for an image sample from Imagenet.

used the MLP and ResNet architectures trained on tabular datasets. For implementing these architectures, we used the `rtdl` Python library<sup>1</sup> and for performing neural architecture search and tuning the hyper-parameters we used the `optuna` [144] Python package. We used two synthetic tabular datasets for the experiments of this section. The characteristics of these datasets are summarized in Table 4.3.

Table 4.3: Tabular datasets

|                   | Epsilon | Yahoo  |
|-------------------|---------|--------|
| <b># objects</b>  | 400000  | 709877 |
| <b># features</b> | 2000    | 519    |
| <b># classes</b>  | 2       | 5      |

The results in Tables 4.4 (a) and (b) for the Epsilon and Yahoo datasets show that the GrID attack is more successful in manipulating the explanations compared to the other attacks, as it achieves a smaller top- $k$  intersection loss with a sparser perturbation proving the superiority of this attack.

<sup>1</sup><https://yura52.github.io/rtdl/stable/index.html>

Table 4.4: Evaluation of the sparse attacks against the explanations of neural networks trained on the a) Epsilon and b) Yahoo dataset. We choose  $k = 20$  in the top- $k$  attack and set the perturbation budget to 20. Results are averaged over 1000 samples. (Grad~Gradient, IG~Integrated Gradients, top- $k$  ints~ average top- $k$  intersection loss, # perturb~ average number of perturbed pixels).

(a) Epsilon dataset

| Expl. method | ResNet model  |                  |      | MLP model    |                  |      |              |
|--------------|---------------|------------------|------|--------------|------------------|------|--------------|
|              | Single-step   | PGD <sub>0</sub> | GrID | Single-step  | PGD <sub>0</sub> | GrID |              |
| IG           | top- $k$ ints | 0.59             | 0.18 | <b>0.11</b>  | 0.58             | 0.18 | <b>0.11</b>  |
|              | # perturb     | 20               | 20   | <b>18.97</b> | 20               | 20   | <b>19.0</b>  |
| Grad× input  | top- $k$ ints | 0.63             | 0.25 | <b>0.11</b>  | 0.58             | 0.17 | <b>0.11</b>  |
|              | # perturb     | 20               | 20   | <b>18.95</b> | 20               | 20   | <b>19.04</b> |

(b) Yahoo! dataset

| Expl. method | ResNet model  |                  |      | MLP model    |                  |      |              |
|--------------|---------------|------------------|------|--------------|------------------|------|--------------|
|              | Single-step   | PGD <sub>0</sub> | GrID | Single-step  | PGD <sub>0</sub> | GrID |              |
| IG           | top- $k$ ints | 0.86             | 0.29 | <b>0.26</b>  | 0.76             | 0.24 | <b>0.14</b>  |
|              | # perturb     | 20               | 20   | <b>15.68</b> | 20               | 20   | <b>15.06</b> |
| Grad× input  | top- $k$ ints | 0.72             | 0.18 | <b>0.088</b> | 0.63             | 0.24 | <b>0.13</b>  |
|              | # perturb     | 20               | 20   | <b>13.80</b> | 20               | 20   | <b>12.08</b> |

## 4.5.2 Ablation study

### Effect of the decreasing stage in the GrID attack

We investigate the effect of the decreasing stage in the GrID attack on the sparsity of the adversarial input. In what follows, we denote the first stage of the GrID attack as "greedy increase", and the second stage as "greedy decrease". For this experiment, we perform top- $k$  attack with  $k = 20$  against the gradient explanation of a ResNet-18 network trained on CIFAR-10 dataset and we set the perturbation budget to 20.

The results shown in Table 4.5 show that the first (increasing) stage of the GrID attack can successfully manipulate the explanation, i.e. it can reach an average top- $k$  intersection loss close to zero. Moreover, the increasing stage of the GrID attack is already better than the PGD<sub>0</sub> attack in terms of the perturbation sparsity as it can reach a zero top- $k$  intersection loss without using the whole perturbation budget. By adding the decreasing stage, which forms the GrID attack, we can observe that the average number of perturbed pixels reduces from 14.55 to 12.15. This proves that

the reducing stage is indeed effective in improving the sparsity of the adversarial input. Furthermore, we also applied the decreasing stage of the GrID attack to the result of the PGD<sub>0</sub> attack to improve the sparsity of the resulting adversarial input. The results shown in Table 4.5 indicate that the perturbation resulting from the GrID attack is sparser compared to the perturbation resulting from the PGD<sub>0</sub> + greedy decrease attack. This proves that the choice of the pixels to perturb in the increasing stage of the GrID attack is more effective compared to the PGD<sub>0</sub> attack.

Table 4.5: Effect of the decreasing stage in the GrID attack. Here top- $k$  ints indicate average top- $k$  intersection loss and # perturbation indicates average number of perturbed pixels. Results are averaged over 1000 images.

|  | top- $k$ ints | # perturbation |
|--|---------------|----------------|
| <b>greedy increase</b>                   | 0.021         | 14.55          |
| <b>PGD<sub>0</sub> + greedy decrease</b> | 0.053         | 15.73          |
| <b>GrID</b>                              | <b>0.017</b>  | <b>12.15</b>   |

### Effect of $k$ in the top- $k$ explanation attack

We evaluate the sparse explanation attacks under different values of  $k$ . To do this, we run the sparse attacks against the gradient explanations of CIFAR-10 images for increasing values of  $k$  in the top- $k$  attack objective (4.1). For all attacks, we set the pixel perturbation budget to be  $k$ .

Figure 4.4 shows the results of these attacks. As we can see, for all values of  $k$ , our GrID attack reaches a lower top- $k$  intersection loss and performs a more successful attack (Figure 4.4, left). It is important to note that as we increase  $k$ , the top- $k$  intersection loss initially decreases, but then starts to increase again. This is expected, as for small values of  $k$ , the perturbation is more restricted due to the small number of pixels that can be modified. As we increase  $k$ , we have a larger perturbation budget and hence the attack can reach a lower top- $k$  intersection loss. However, for large values of  $k$  that are close to the dimensionality of the image, changing all of the top- $k$  features of the explanation becomes a difficult task, resulting in a larger top- $k$  intersection loss. However, this does not mean that the explanation gets less manipulated for very large values of  $k$ . By looking at the cosine distance (cosd) values for different values of  $k$  (Figure 4.4, right), we can see that as we increase  $k$ , the cosine distance between the original and the manipulated explanations increases, indicating that the manipulated explanation becomes increasingly different from the original. The number of pixels perturbed

by the  $\text{PGD}_0$  attack is always equal to  $k$  since the  $\ell_0$  projection always results in a perturbed image different from the original image in exactly  $k$  pixels. This can be seen in Figure 4.4 (middle column). On the other hand, the GrID attack can manipulate the explanation with sparser perturbations. Moreover, we can observe that for very large values of  $k$ , reducing the top- $k$  intersection loss to zero becomes more challenging, and hence the GrID attack requires perturbing a larger fraction of the perturbation budget.

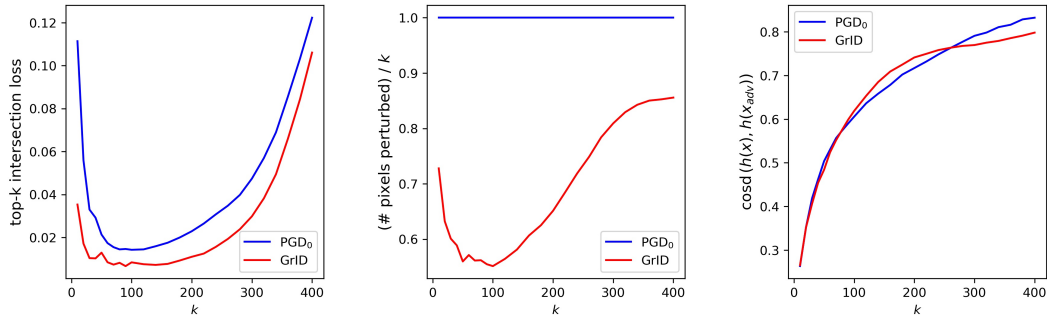


Figure 4.4: Comparing the performance of the sparse attacks for different values of  $k$  in the top- $k$  attack. **left:** top- $k$  intersection loss vs  $k$ , **middle:** number of perturbed pixels divided by  $k$  vs  $k$ , **right:** cosine distance between the manipulated and target explanations vs  $k$ . Each attack setting is run over 500 samples from CIFAR-10 and the average results are shown in the figure.

### 4.5.3 Case study: fair-washing a gender-biased model

As mentioned in Section 4.1, explanations can help to reveal the dependence of a model on spurious signals. This is particularly important in applications in the areas of medicine, finance or law where provision of explanations is an important component to make sure biases or unfair treatment of certain classes can be avoided. Particularly, dependence on sensitive features such as gender or race can lead to unfair models which are gender or race discriminative.

Although explanations can help to reveal the dependence of a model decision on sensitive features, they can also be used to hide such fact and hence mislead a user into believing that the model is fair (when it is actually not). Recent works refer to this (mis)use of explanations as *fair washing* [137, 145, 138, 59]. In this experiment, using a sparse explanation attack we seek to hide the fact that a model is gender-biased, in specific, that it is discriminative against females. To conduct this experiment, we used the *adult income* dataset [146], which has been widely used in the fairness literature and is well-known to have a gender bias. The classification



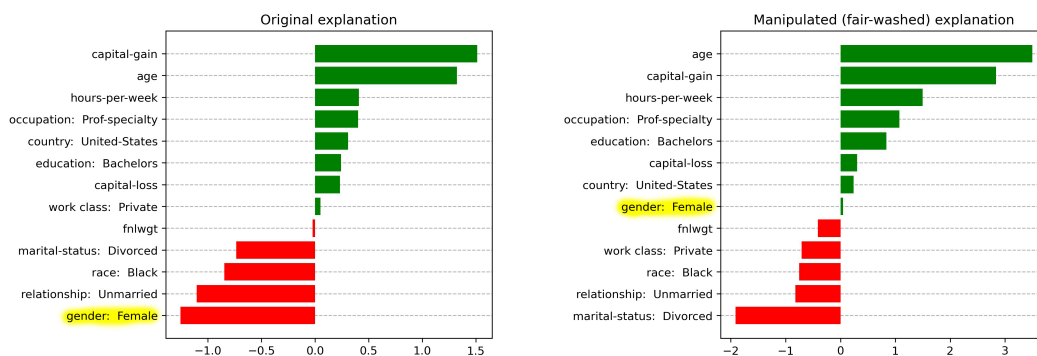


Figure 4.5: Explanation for a data point from the adult income dataset. The scores show the contribution of each feature towards predicting the positive class (high salary). The original explanation shows high negative importance for the `gender=female` feature (highlighted in yellow), i.e., the model depends heavily on the gender being female when predicting a low salary. The manipulated explanation hides this gender-discriminative behavior of the model and shows that being a female has no effect on model’s decision when predicting a low salary.

task for this dataset is to predict whether an adult earns more or less than \$50K annual salary. To create a predictive model, we trained a ResNet architecture on this dataset. We trained a model on a subset of this dataset to strengthen the dependence of the model on `gender=female` when predicting `salary≤$50K`. We can also observe this dependence in Figure 4.5 (left) which shows the Integrated Gradients explanation of model prediction for a female sample. This explanation shows that being a female has the highest negative contribution towards predicting a high salary, which clearly shows the discrimination of the model against females.

We perform an explanation attack to fair-wash the above-mentioned model and hide the fact that it discriminates against females. For the samples for which `gender=female` and `salary≤$50K`, we wish to reduce the importance of the gender feature in the explanation while keeping the prediction unchanged. Using our GrID attack, we are able to do so by only perturbing a few features of the data. To perform the attack in this setting we do not use the top- $k$  loss in (4.1). Instead, we choose the set  $\mathcal{B} = \{\text{gender=female}\}$  in the first term of (4.1) as we intend to reduce the importance of this feature in the explanation. We do not perturb the `gender=female` and `gender=male` features in this attack, i.e., we do not change the gender of the data point when perturbing it. Figure 4.5 (right) shows the resulting manipulated explanation after the attack for a sample in the adult income dataset. We can observe that in this explanation being a female is not a decisive feature for predicting a low or high salary. Compared to the original explanation, the ranking of the remaining features has not been drastically modified which enhances

the truthlikeness of the manipulated explanation.

## 4.6 Discussion

We introduce an innovative and efficient algorithm named GrID, designed for generating sparse perturbations that effectively manipulate the explanations provided by graph neural networks. Our method demonstrates remarkable efficacy in achieving substantial explanation manipulation using notably sparser perturbations compared to PGD attacks with  $\ell_0$  constraints. In some scenarios, as illustrated in Figure 4.2, the GrID attack may produce perceptible perturbations. By integrating an  $\ell_\infty$  constraint, we can achieve perturbations that are less conspicuous yet less sparse. Furthermore, to enhance the imperceptibility of sparse perturbations, an enhancement approach involves the identification of pixels in the image that are less susceptible to visible perturbations. Subsequently, the algorithm assigns a higher probability to perturbing these less noticeable pixels. This has the potential for further refinement and optimization in future iterations of the GrID attack.

## 4.7 Related work

**Explanation attacks.** Past work has shown that explanations can be manipulated. This could be done by adding a small-norm perturbation to the input data such that the explanation changes significantly, but the model prediction remains unchanged. Such perturbations can be found by optimizing an appropriate objective function with the gradient descent method. Ghorbani, et al [102] showed that gradient-based explanations of deep neural network (DNN) image classifiers are fragile and defined three different types of explanation attacks. Dombrowski, et al [60] gave an analysis showing that the explanations of DNNs are malleable due to the high curvature of the decision surface of DNNs. In line with this analysis, Wang et al [62] defined the robustness of gradient-based explanations around a data point  $\mathbf{x}$  as the local Lipschitz Continuity of the explanation function around point  $\mathbf{x}$ . To robustify explanations, they introduced a regularization term called *Smooth Surface Regularization (SSR)* for the training objective of a DNN, which penalizes the large curvature in a DNN. [107] and [62] showed that adversarial training [111] leads to a significant decrease in the curvature of the decision surface of a DNN and hence to more robust explanations. Finally, Dombrowski, et al [114] developed a theoretical framework to derive bounds on the maximum manipulability of explanations and proposed three different techniques to boost the robustness of explanations. Besides

input perturbation, another approach to achieve manipulated explanations is via adversarial model manipulation, which was explored in [61, 103].

**Sparse perturbations.** Sparse perturbations have been explored in the context of adversarial attacks to change the decision of image classifiers. [141] proposed a black-box technique to craft sparse adversarial perturbations. Their method is based on sampling progressively more pixels of an image and perturbing each selected pixel to one of the 8 corners of the unit cube in the RGB space which decreases the corresponding logit of the true label the most. Although this method can reach the minimum number of perturbed pixels required to change the prediction, it is a very costly method, and applying it to high-dimensional data is not practical. As a white-box and cheap technique to find adversarial examples with small  $\ell_0$  distance to the original input, [141] proposed a method to compute projection onto the  $\ell_0$  ball. Using this projection in an adversarial attack leads to finding an adversarial input with a small  $\ell_0$  distance to the original input (and hence a sparse perturbation). They called such an attack as PGD<sub>0</sub>. As another method to craft adversarial examples with sparse perturbations, [142] proposed sparsefool. This method is a white box method which can find sparse perturbations very fast and can scale well to high-dimensional data. This method is based on linearizing the decision surface around a data point and then solving an  $\ell_1$  constrained linear optimization problem, which has a sparse solution. Finally, Dong, et al [143] proposed GreedyFool, a distortion-aware greedy-based sparse attack to change model prediction. This attack uses a Generative Adversarial Net (GAN) to learn a precise distortion map which acts as an invisibility constraint that helps the attack to achieve better invisibility.

## 4.8 Conclusion

Computing sparse adversarial perturbations that can manipulate the explanations is a challenging task. This is mainly due to the NP-hardness of the  $\ell_0$  minimization problem, which is equivalent to finding the most sparse solution. One needs also to make sure that the valid range of the data is respected and that the prediction does not change after the attack. Here, we explored these challenges and proposed an effective algorithm that computes sparse perturbations. We showed that our approach works better than performing a PGD attack with  $\ell_0$  projection. It also leads to higher dissimilarity between the original and manipulated explanations and a sparser perturbation. We demonstrated that it is possible to completely hide the attribution of the  $k$  most important features in the original explanation by perturbing less than  $k$  features of the input data. Additionally, we highlighted the potential implications

of these attacks on building trustworthy models in high-stake applications, where explanations are crucial for ensuring fairness and avoiding bias. Our results reveal new vulnerabilities in the current explanation methods and call for the development of more robust explanation methods.



## Chapter 5

# Using Neural and Graph Neural Recommender Systems to Overcome Choice Overload: Evidence from a Music Education Platform

**Abstract.** The application of recommendation technologies has been crucial in the promotion of physical and digital content across numerous global platforms such as Amazon, Apple, and Netflix. Our study aims to investigate the advantages of employing recommendation technologies on educational platforms, with a particular focus on an educational platform for learning and practicing music.

Our research is based on data from Tomplay, a music platform that offers sheet music with professional audio recordings, enabling users to discover and practice music content at varying levels of difficulty. Through our analysis, we emphasize the distinct interaction patterns on educational platforms like Tomplay, which we compare with other commonly used recommendation datasets. We find that interactions are comparatively sparse on educational platforms, with users often focusing on specific content as they learn, rather than interacting with a broader range of material. Therefore, our primary goal is to address the issue of data sparsity. We achieve this through entity resolution principles and propose a neural network (NN) based recommendation model. Further, we improve this model by utilizing graph neural networks (GNNs), which provide superior predictive accuracy compared to NNs. Moreover, we demonstrate that by leveraging the computational subgraph of a node in a GNN, we can generate an explanation for a recommendation by highlighting

the influential users and items. Finally, our study demonstrates that GNNs are highly effective even for users with little or no historical preferences (cold-start problem).

Our cold-start experiments also provide valuable insights into an independent issue, namely the number of historical interactions needed by a recommendation model to gain a comprehensive understanding of a user. Our findings demonstrate that a platform acquires a solid knowledge of a user's general preferences and characteristics with 50 past interactions. Overall, our study makes significant contributions to information systems research on business analytics and prescriptive analytics. Moreover, our framework and evaluation results offer implications for various stakeholders, including online educational institutions, education policymakers, and learning platform users.

## 5.1 Introduction

E-commerce sites and digital distribution platforms offer a vast amount of physical and digital content, which can make it overwhelming for a user to discover content that aligns with their personal preferences. As a result, users are often faced with a choice overload [147]. Recommender systems fall under the category of business analytics that can alleviate these challenges by discovering content that aligns with the user's interests based on their historical preferences [148]. Such analytics have been shown to increase user satisfaction by reducing the time required to search for relevant content [149]. The presence of a recommendation system is equally advantageous for the business implementing it [150, 151], as it extends the users' overall engagement on the platform, resulting in interaction with new content or leading to new purchases [151].

Numerous studies have demonstrated the advantages of recommender systems for both users and businesses [152]. Most research has focused, however, on e-commerce and multimedia content providers and platforms including Amazon, Netflix, Spotify, and YouTube, where users are regarded as quick "consumers" of physical or digital content. Such platforms benefit from increased user engagement with existing content to gain a better understanding of users and provide tailored and personalized recommendations. Conversely, in an educational or learning system, users typically interact with fewer items due to the inherent learning curve associated with comprehending or assimilating presented content. Our study examines a case that generally displays different interaction patterns than typical multimedia and e-commerce platforms.

In collaboration with Tomplay, we studied the potential benefits of using recommendation technologies on their educational platform. Tomplay is an app that offers interactive music sheets, or partitures. The music sheets for any type of music are paired with high-quality recordings. Noteworthy is that the same content is created and is available at different degrees of difficulty and for different instruments. Musicians can download music sheets that match their proficiency level and play along with a professional multi-track recording of the piece. This allows them to enjoy the experience of playing with a band or an orchestra while practicing the music.

We examined the users' past preferences to understand their musical tastes and predict music sheets that they would be interested in. While building the recommendation engine, we discovered interesting characteristics of user interactions on this educational platform. We found that the majority of users interact with fewer items compared to traditional recommender systems. We hypothesize this to be a characteristic of educational platforms, where the goal of users is to "learn" some content (such as how to play a song in this case) rather than to consume content (e.g., a movie or a song). As a result, users interact with fewer items but access patterns exhibit a more repetitive behavior, as users come back to the same item multiple times. We posit that to be another important characteristic of an educational or learning platform, where users revisit the same content multiple times as part of the learning process.

We begin by conducting a comparative analysis of the data characteristics of this educational platform compared to other recommendation datasets for movies, songs, and books. We highlight the high degree of data sparsity in this educational dataset and discuss the implications for applying recommendation technologies. In the case of Tomplay, we take advantage of the availability of music sheets for multiple musical instruments to create additional associations between users and items (music sheets). We achieve this by implementing a lightweight entity resolution system based on neural embeddings. By combining the embedded music sheets with user-item interactions, we create a neural recommendation engine. We demonstrate that our approach can improve accuracy compared to matrix factorization and neural network-based recommender systems. Subsequently, we build a recommendation engine using graph neural networks and present various design choices. We show that this approach provides even better recommendation accuracy. **Contributions** of this work include:

- A presentation and comparative study of the interaction characteristics of an educational platform compared to more traditional content-providing platforms



for books, songs, and videos. To the best of our knowledge, this is the first time such findings and comparisons have been reported.

- We take advantage of the "multi-instance" of music sheets for different instruments and at multiple difficulty levels to alleviate some of the data sparsity issues for user-item interactions. We use entity resolution concepts based on embeddings to alleviate the multi-instrument and multi-difficulty instances of the same music, thus reducing the interaction sparsity.
- We show the improvement in predictive power that we can achieve with the use of a neural architecture and an architecture based on graph neural networks. We also compare our approach to other prevalent batch and sequential neural recommendation algorithms.
- We demonstrate that by leveraging the computation sub-graph used by the GNN architecture, we can generate an explanation for a recommendation. This explanation highlights the most influential users and items toward a specific recommendation, making the recommender system decision process more transparent to business owners.
- We investigate the number of past user interactions that a predictive model must access in order to sufficiently understand a new user and generate effective recommendations. To study this, we conduct a cold-start experiment where we progressively increase the recommender model's knowledge of a user's historical preferences and calculate the recall and precision of the predictive system. We discover that with approximately 50 historical interactions, the recommender model has good knowledge of the user.

In what follows, first, we describe the type of data that we use. Then, we present a neural recommendation system, and afterward, one based on graph neural networks. We show how to properly tune the graph model and how to address cold-start situations. Later, we present our experiments. We delegate the discussion of the related literature until after the experiments so that the reader becomes more acquainted with the contributions of this work. We conclude with a discussion and further applications and extensions of this research.

## 5.2 The Tomplay app

To give the reader an understanding of the type of data collected, we will start by briefly describing some of the functionalities of the Tomplay app. The app offers

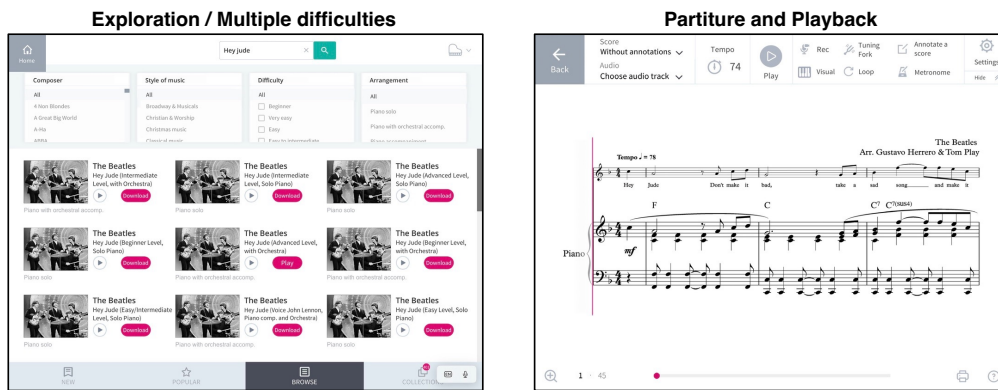


Figure 5.1: Two major functionalities offered by the Tomplay app related to this paper: exploration and playback. The Beatles song "Hey Jude" (left panel) has multiple instances: different difficulty levels, different accompaniments (with or without orchestra, etc) and offered to various instruments.

two main modes, as shown in Fig. 5.1:

1. **Exploration** of available music sheets through the Tomplay store or already downloaded content. Users can filter the music sheets based on instrument, song difficulty, or perform a text search. There are more than 20 instruments for which music content exists, including piano, violin, saxophone, clarinet, drums, etc.
2. **Playback** of already downloaded content. Users can accompany the playback of the music performance with their own instrument. When viewing the actual music partiture, there are several other functionalities offered (annotation, partial playback, etc.), but they are not relevant to this work.

The data we have available is as follows: during every user playback of a music sheet, a log event is recorded as {song ID, user ID, timestamp}. Therefore, the dataset can be viewed as a graph where nodes are the music sheets and the edges represent the time sequence of the playback. We provide examples of the data graph for three users in Fig. 5.2.

## 5.3 Data characteristics

The dataset that we examine has several interesting characteristics that are different from traditional recommendation datasets.

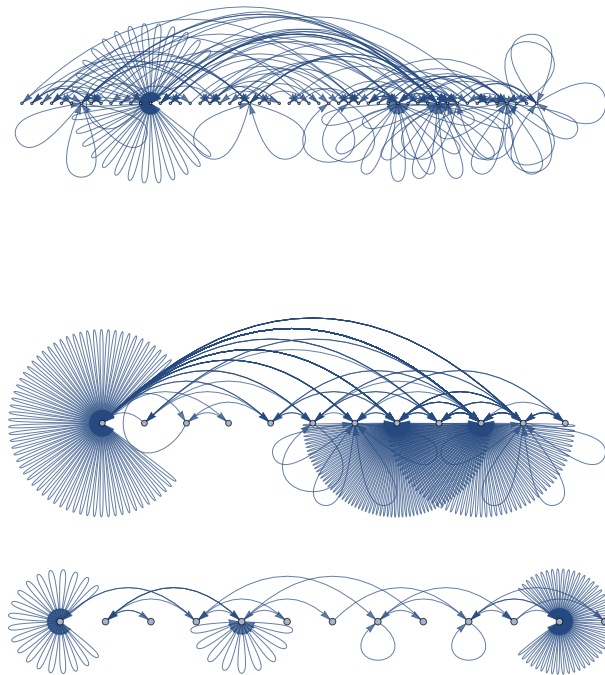


Figure 5.2: Playback patterns for three different users (from top to bottom). Each node is a music sheet/song, and each arc indicates the sequence of interactions in time. The user behavior is highly repetitive; users visit the same song several times.

1. **Multiple instantiations** of the same content (Fig. 5.5). In our scenario, there are multiple versions of the same song: a) at different levels of difficulty, b) for various instruments, and c) with different accompaniments or orchestrations (e.g., single instrument recording, duet, with/without orchestra, etc.). In Table 5.1, we see an example of the same musical piece by Bach, offered for different instruments (violin and cello), at different difficulty levels (easy and intermediate), and with diverse accompaniments (duet or single instrument). This is important to note because we have multiple versions of the same piece but under different IDs. These IDs may be consecutive in many cases, but not always, as different versions of the same song may have been created at different points in time. The main challenge is that if we treat each version with a separate ID as a separate item, we may miss important content relationships, for example, similar musical content accessed by users playing different instruments. All these songs would have completely different

IDs because they represent separate entities.

**2. Extremely sparse interactions.** The previous observation suggests that the interaction between users and items in the Tomplay dataset is even more sparse than typical recommendation platforms that offer a single version of the same content. Fig. 5.3 compares the sparsity of interactions of the Tomplay dataset to other well-known recommendation datasets for books (Goodreads), movies (Netflix and MovieLens), and songs (LastFM). We observe that for Tomplay, users have on average much fewer interactions than for all the other datasets. This is likely because each item (music sheet) may require learning time, and thus users interact with fewer items in total.

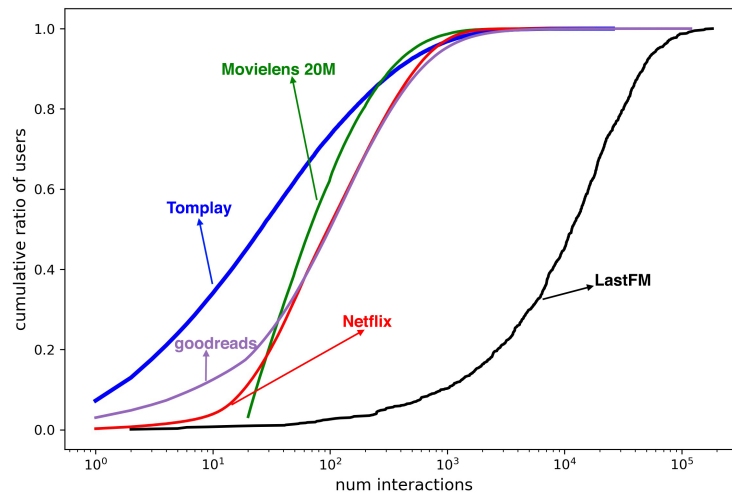


Figure 5.3: Comparison of Tomplay dataset with other benchmark recommender datasets in terms of interaction sparsity. We see that in an educational setup, such as Tomplay, users interact with fewer items because of the learning curve involved in properly appreciating and learning an item on the platform.

**3. Multiple visits.** Items (music sheets) are commonly visited multiple times by the same user (Fig. 5.2). We hypothesize that, in an educational platform, content must be repeated several times for it to be learned. In comparison, one rarely reads a book more than once, and it is quite infrequent to watch a movie several times. These observations are also discernible in Fig. 5.4, where we provide a histogram comparing the average in-degrees of user graphs, as shown in Figure 5.2, for 100 random users in the Tomplay and LastFM datasets. For LastFM, nodes have lower in-degrees because users usually listen to a song a few times. In contrast, in the Tomplay dataset, it is not uncommon to have nodes (music sheets) that are visited 10, 20, or more times by a user.

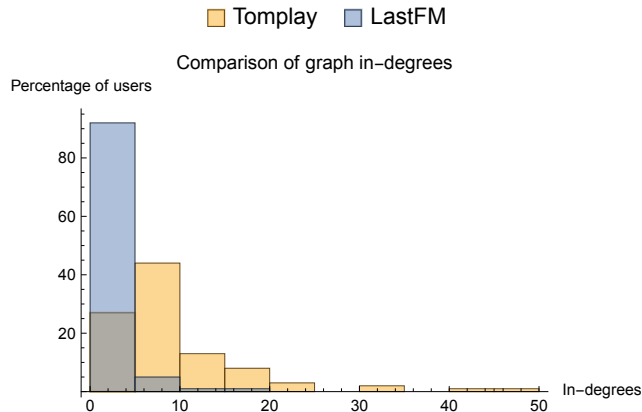


Figure 5.4: Comparing the average user graph in-degrees for 100 random users of the Tomplay dataset and the LastFM dataset (users stream music songs). We see that in the Tomplay dataset, many nodes (music sheets) have higher in-degrees because they are visited multiple times.

We believe that the characteristics mentioned above are generic to educational platforms, and are not typically exhibited in non-educational platforms or recommendation datasets. In non-educational settings, a user typically "consumes" content and interacts with it a limited number of times, as there is no learning curve present as in educational settings.

Based on the previous discussion, we can provide recommendations by leveraging the collective knowledge of users with similar musical preferences. The musical taste would be captured by a high overlap in their preferences for music items, regardless of the instrument, difficulty, or accompaniment of those items.

## 5.4 Neural Networks with Entity Resolution

The previous observations have highlighted that to uncover the underlying music preferences of users, one should emphasize the higher-level characteristics of an item (e.g., the title and the composer) and underemphasize the importance of the specific instantiation of a song: difficulty, instrument, accompaniment.

To address this challenge, we decided to perform a lightweight entity resolution phase in our recommendation engine that would allow us to "group" (but in a soft, probabilistic manner) the multiple instantiations of a musical item. The main idea of our approach is to first extract representations of song names from a powerful language model and then combine them with the song embeddings learned during the training of the recommender system. The song title embeddings provide us

Table 5.1: A sample of the data that we have at our disposal. The same song (a song by Bach in this case) is available for different instruments, at different levels of expertise, and across various recordings (e.g., single instrument or with orchestra, etc.).

| ID    | Title   | Composer     | Style      | Difficulty   | Instrument | Accompaniment         |
|-------|---|--------------|------------|--------------|------------|-----------------------|
| 1001  | Erbarne dich, mein Gott, BWV 244 (level easy)         | Bach         | Classical  | Easy         | Violin     | Without Accompaniment |
| 1002  | Erbarne dich, mein Gott, BWV 244 (level intermediate) | Bach         | Classical  | Intermediate | Violin     | Without Accompaniment |
| 3035  | Erbarne dich, mein Gott, BWV 244 (level easy)         | Bach         | Classical  | Easy         | Cello      | Duet                  |
| 3036  | Erbarne dich, mein Gott, BWV 244 (level intermediate) | Bach         | Classical  | Intermediate | Cello      | With Orchestra        |
| 6032  | The Pink Panther                                      | Mancini      | Film Music | Progressive  | Clarinet   | With Orchestra        |
| 7455  | Ma rendi pur contento - SOPRANO                       | Bellini      | Classical  | Intermediate | Singing    | With Orchestra        |
| 12704 | The Logical Song (level difficult, sax soprano)       | Supertramp   | Pop/rock   | Difficult    | Saxophone  | With Orchestra        |
| 21047 | Braveheart  | James Horner | Film Music | Easy         | Violin     | With Orchestra        |
| 84176 | Braveheart (level beginner, with orchestra)           | James Horner | Film Music | Beginner     | Piano      | With Orchestra        |

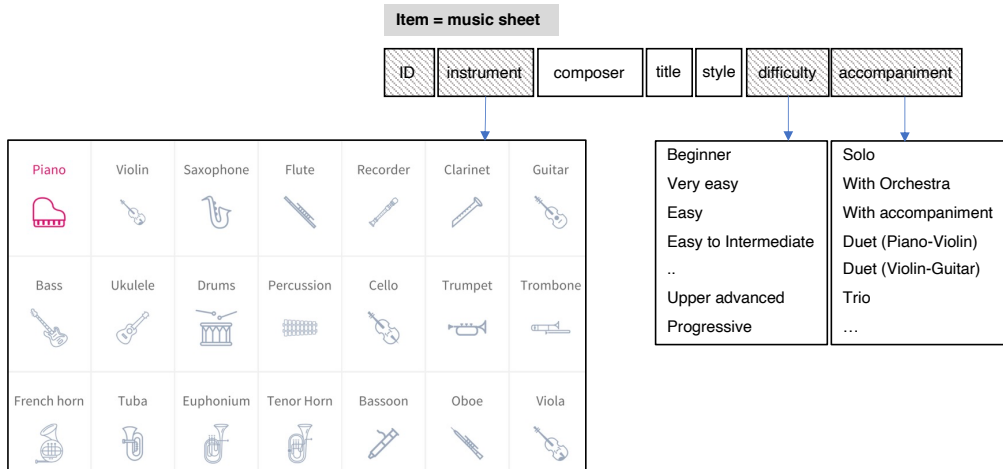


Figure 5.5: The database can contain the same song with multiple IDs, instruments, difficulties, and accompaniments, because of the multi-instantiations of a song. The attributes that do not necessarily have a unique value for the same song are striped.

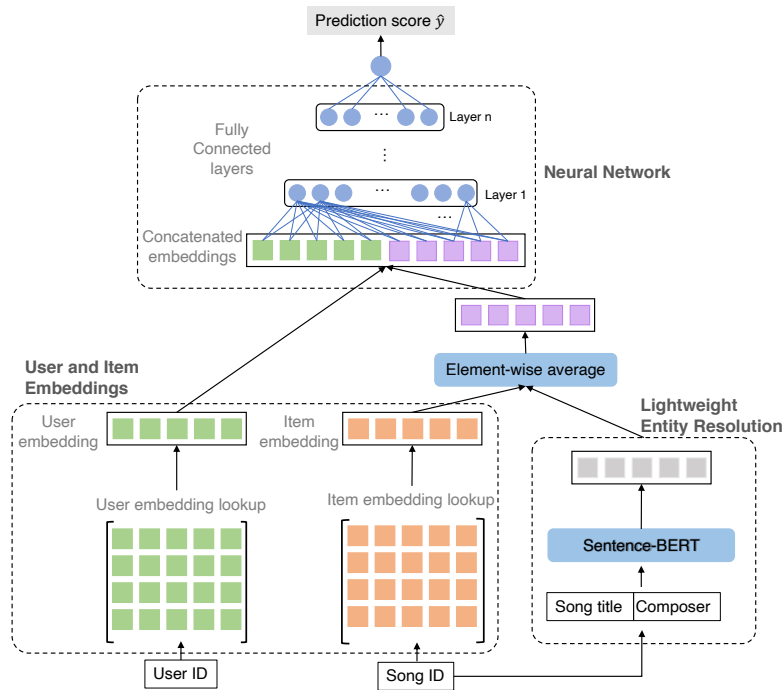


Figure 5.6: Our neural recommendation architecture uses a lightweight entity resolution process based on text embeddings to enforce the “soft” grouping of multiple instantiations of the same song.

with embedding vectors that place the same songs (with potentially different levels, instruments, etc.) close to each other. On the other hand, the song (item) embedding learned by the recommender system places similar songs (according to users’ preferences) close to each other in the embedding space. Therefore, combining these two embeddings is expected to give us the best of the two cases.

More precisely, for each song we append the composer’s name to the song title and extract the representation of the resulting text from the sentence-BERT model [153]. We used the pre-trained multi-lingual model from huggingface<sup>1</sup>. Supporting multiple languages is necessary as the song titles in our dataset are in English, French, and German. After extracting the text representation, the resulting vector is averaged with the item embedding vector. Although averaging the two embedding vectors may lead to losing some information specific to the dimensions of the two embedding spaces, in practice we found averaging more effective than concatenating the two vectors. Moreover, by avoiding concatenation, the resulting model will have fewer parameters and hence will be lighter. The resulting vector can

<sup>1</sup><https://huggingface.co/sentence-transformers/paraphrase-xlm-r-multilingual-v1>

be treated as a new item embedding which is fed to the recommender architecture. Our approach can be applied to any recommender model that learns item embeddings, for example, neural recommender systems. Figure 5.6 provides the details of our architecture. Similar to conventional recommender systems, the item embeddings capture different features of the items. In our case, they capture the features of the songs such as the instrument and difficulty. By combining the item embeddings with the user embeddings, the recommender system learns about users who have the same level of expertise and instrument of choice and have similar musical tastes. Moreover, the sentence-BERT model embeddings place instances of the same song with varying difficulty levels, instruments, and accompaniment close to each other in the latent space. By combining the sentence-BERT embeddings and the item embeddings, the model can learn about users with similar musical tastes regardless of their level of expertise and preferred instrument.

### 5.4.1 Architecture of the Neural Network model

The input layer of the model consists of user and item identifiers together with the item features used by the model. On top of the input layer is the embedding layer which projects sparse representations, i.e., user (item) identifiers, into dense vectors called user (item) embedding. These embeddings are trained such that similar users (items) are placed close to each other in the users' (items') latent space. To perform the entity resolution task for the same songs with different levels or instruments, the model contains another embedding lookup which is initialized with the embeddings of the concatenated song and composer names extracted from the pre-trained sentence-BERT language model.

**Notation** Consider a recommendation problem consisting of  $M$  users and  $N$  items. The matrices  $\mathbf{E}_U^{M \times K}$  and  $\mathbf{E}_I^{N \times K}$  are the users and items embedding matrices respectively, where  $K$  is the dimensionality of the users (items) latent space. The embedding vectors for a user  $u$  and an item  $i$  are denoted by  $\mathbf{v}_u^U$  and  $\mathbf{v}_i^I$  respectively. The matrix  $\mathbf{E}_T^{N \times K}$  contains the textual embeddings of song names and composers.<sup>2</sup> For an item  $i$ , its textual embedding vector is denoted by  $\mathbf{v}_i^T$ .

For a user  $u$  and item  $i$ , the input to the fully connected network is the concatenation of the vectors  $\mathbf{v}_u^U$  and  $\mathbf{v}_i^{avg}$ , where  $\mathbf{v}_i^{avg} = (\mathbf{v}_i^I + \mathbf{v}_i^T) / 2$ . The fully connected network can include several fully connected layers (also known as hidden layers).

<sup>2</sup>Note that this matrix has the same dimensions as the item embedding matrix  $\mathbf{E}_I^{N \times K}$



The output of each layer serves as the input to the next layer:

$$\mathbf{a}^{l+1} = \text{ReLU}(\mathbf{W}^l \mathbf{a}^l + \mathbf{b}^l), \quad (5.1)$$

where  $\mathbf{a}^l$  is the vector of the neuron values in layer  $l$ ,  $\mathbf{b}^l$  is the bias of the layer,  $\mathbf{W}^l$  is the weight matrix of the layer. A Rectified Linear Unit (ReLU) function is used as the activation function of the network. Finally the prediction of the model for a user  $u$  and an item  $i$  can be formulated as:

$$\hat{y}_{u,i} = f(\mathbf{v}_u^U, \mathbf{v}_i^I, \mathbf{v}_i^T \mid \mathbf{E}_U, \mathbf{E}_I, \mathbf{E}_T, \Theta_f), \quad (5.2)$$

where  $\Theta_f$  denotes the learnable parameters of the network  $f$ . Note that the textual embeddings of song names and composers, extracted from the sentence-BERT pre-trained model, can also be updated during the training. That is why we included the matrix  $\mathbf{E}_T$  as part of the model parameters in (5.2).

### 5.4.2 Training the model

The output of the model,  $\hat{y}_{u,i}$  represents how likely it is that user  $u$  interacts with item  $i$ . To have a probabilistic form of this likelihood,  $\hat{y}_{u,i}$  is constrained to be in the range  $[0, 1]$  by applying a sigmoid function  $\sigma(\cdot)$  on the output neuron. To learn the model parameters, the binary cross-entropy loss function is minimized:

$$L = -\sum_{(u,i) \in \mathcal{Y}^+} \log \sigma(\hat{y}_{u,i}) + \sum_{(u,i) \in \mathcal{Y}^-} \log(1 - \sigma(\hat{y}_{u,i})) \quad (5.3)$$

where  $\mathcal{Y}^+$  is the set of all user-item interactions in the dataset. Since we are dealing with an implicit feedback problem, that is, we have access only to the positive interactions, we need to create a set of negative interactions  $\mathcal{Y}^-$  as well. For each positive interaction  $(u, i)$  several items are uniformly sampled which are not among the items the user  $u$  interacted with. The set  $\mathcal{Y}^-$  in (5.3) is built in this way.

## 5.5 Using Graph Neural Networks

In this section, we extend and tune the previous neural network into a graph neural network (GNN) architecture. GNNs have been shown to outperform deep neural networks in many setups where the problem can be more naturally posed and represented as an instance of a graph problem. Moreover, GNNs are designed to handle complex relationships and interactions by learning from the graph structure of the data, making them well-suited for recommendation tasks.

There exist several GNN architectures, which one can use to build their model. As our GNN architecture, we adopt the LightGCN [154] architecture because it is simple and lightweight. We modified and augmented this architecture in the following ways : (1) we employ the same lightweight entity resolution as the previous approach for the item embeddings, (2) we intelligently assign weights to the user-item interaction graph's edges to capture the time dependency of interactions, (3) we augment the LightGCN architecture to address the cold start problem – that is, how to predict future items for new users for whom we do not have access to any of their historical preferences. The last part is particularly important because it allows us to make predictions for users without (or with only a few) historical preferences.

### 5.5.1 Architecture of the GNN model

Recommendation of items to users can be formulated as a link prediction problem in the user-item interaction graph. We represent the interactions between users and items as a bipartite graph  $\mathcal{G} = \{\mathcal{V}_U \cup \mathcal{V}_I, \mathcal{E}\}$  where  $\mathcal{V}_U$  is the set of users,  $\mathcal{V}_I$  is the set of items, and  $\mathcal{E}$  is the set of edges between them. In our case, we have a multigraph with one edge for each log event  $\{u, i, \text{timestamp}_{ui}\} \in \mathcal{V}_U \times \mathcal{V}_I \times \mathbb{R}$ .

As mentioned, we use LightGCN as the backbone of our GNN model, which captures node features by averaging the features of its neighbors. The GNN architecture that we use is presented in Figure 5.7.

Starting with the initial embedding  $\mathbf{e}_u^{(0)}$  for each user  $u$  and  $\mathbf{e}_i^{(0)}$  for each item  $i$ , embeddings at each layer of the GNN are propagated to the next layer according to the following equations:

$$\begin{aligned} \mathbf{e}_u^{(k+1)} &= \sum_{i \in \mathcal{N}(u)} \frac{\mathbf{e}_i^{(k)}}{\sqrt{|\mathcal{N}(u)|} \sqrt{|\mathcal{N}(i)|}} \quad \forall u \in \mathcal{V}_U \\ \mathbf{e}_i^{(k+1)} &= \sum_{u \in \mathcal{N}(i)} \frac{\mathbf{e}_u^{(k)}}{\sqrt{|\mathcal{N}(u)|} \sqrt{|\mathcal{N}(i)|}} \quad \forall i \in \mathcal{V}_I \end{aligned} \quad (5.4)$$

where  $\mathcal{N}(u)$  is the set of neighbors of user  $u$  in the graph,  $\mathcal{N}(i)$  is the set of neighbors of item  $i$ , and the superscript  $(k)$  indicates the layer number. The final embedding of a user/item is computed as the average of the embeddings across all layers and the initial embeddings:

$$\mathbf{e}_n = \frac{1}{K+1} \sum_{k=0}^K \mathbf{e}_n^{(k)} \quad \forall n \in \mathcal{V}_U \cup \mathcal{V}_I, \quad (5.5)$$

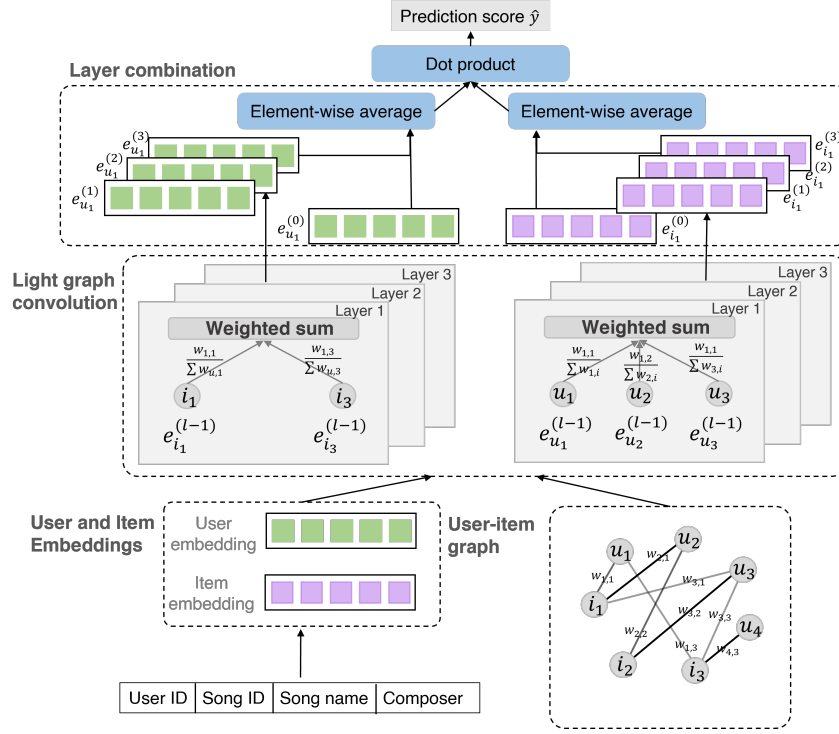


Figure 5.7: The Graph Neural Network architecture used in this work.

where  $K$  is the number of GNN layers. The predicted score of an interaction between user  $u$  and item  $i$  is then computed as the dot product of their embeddings:

$$y_{ui} = \mathbf{e}_u^T \mathbf{e}_i. \quad (5.6)$$

We denote the adjacency matrix of the graph by  $\mathbf{A}$ , the degree matrix by  $\mathbf{D}$ , the normalized adjacency matrix by  $\tilde{\mathbf{A}} = \mathbf{D}^{-\frac{1}{2}} \mathbf{A} \mathbf{D}^{-\frac{1}{2}}$ , and the initial and final embedding matrices as  $\mathbf{E}^{(0)}$  and  $\mathbf{E}$ , respectively. The users are represented in the first  $M$  rows of  $\mathbf{A}$  and the items are represented in the next  $N$  rows. The adjacency matrix is given by:

$$\mathbf{A} = \begin{pmatrix} \mathbf{0} & \mathbf{R} \\ \mathbf{R}^T & \mathbf{0} \end{pmatrix},$$

where  $\mathbf{R}$  is an  $M \times N$  matrix that represents the interactions between users and items. As we have a multigraph,  $\mathbf{R} \in \mathbb{N}^{(M+N) \times (M+N)}$ , we can rewrite the equations (5.4)-

(5.6) into simplified matrix operations:

$$\mathbf{E}^{(k+1)} = \tilde{\mathbf{A}}\mathbf{E}^{(k)}, \quad (5.7)$$

$$\mathbf{E} = \frac{1}{K+1} \sum_{k=0}^K \mathbf{E}^{(k)}. \quad (5.8)$$

where  $\mathbf{E}^{(k)}$  is the matrix of embeddings at layer  $k$ . We compute the scores of all interactions as:

$$\mathbf{Y} = \mathbf{E}_U^T \mathbf{E}_I,$$

where  $\mathbf{E}_U$  is the matrix containing the embeddings of all users and  $\mathbf{E}_I$  is the matrix containing the embeddings of all items (i.e. the first  $M$  and the last  $N$  rows of  $\mathbf{E}$ ). The embeddings are initialized in the same way as the neural recommender system introduced in Section 5.4, i.e., we also use the item name and composer embeddings from the sentence-BERT model.

### 5.5.2 Preprocessing the graph

Here, we describe pre-processing steps that we considered for the user-item interactions graph, to make it more amenable for use by a GNN-based predictive model. In particular, we considered (1) adding edges between items to highlight the connection between multiple instantiations of the same content/song in various difficulty levels, and, (2) assigning weights on the edges of the graph in order to differentiate between interactions based on their timestamp. Below, we describe these two pre-processing steps in more detail.

**Adding edges.** As discussed in Section 5.3, users tend to interact with only a few items, resulting in a sparse graph. However, in our case, many items are similar, e.g., instances of the same song with different difficulty levels or different instruments. Therefore, one may expect that connecting such items with edges can be advantageous for the GNN-based recommender system. We proposed to connect “similar” item nodes based on the various instantiations of a song. Adding extra edges between such item nodes increases the density of the graph and can potentially enhance the performance of the GNN. We investigated various strategies for adding edges, namely, adding edges between items with the same title and similar difficulty, adding edges between items with the same title and same instrument, and connecting the items with the same composer, style, or instrument. However, none of these strategies proved to be helpful to achieve better recommendation accuracy.

**Adding edge weights.** To consider the time dependency of the interactions, we assigned weights to the graph edges using the timestamp of each interaction. Specifically, we assigned weights such that more importance is given to more recent interactions. For an interaction between user  $u$  and item  $i$ , we compute its weight based on the difference between the timestamp of the interaction and the timestamp of the most recent interaction of user  $u$ . More precisely, the edge weights are computed as follows:

$$w_{ui} = \exp\left(-\gamma \cdot \frac{\max_{i' \in \mathcal{N}(u)}(\text{timestamp}_{ui'}) - \text{timestamp}_{ui}}{C}\right)$$

where  $\gamma$  is a hyperparameter that controls the decay of edge weights over time.  $\gamma = 0$  means that the edge weights are constant and equal to one, while  $\gamma \rightarrow \infty$  means that only the most recent interactions are considered.  $C$  is a normalization constant that can be selected to be in the order of the cardinality of interactions in a recommendation dataset. With edge weights, equations (5.4) can be rewritten as:

$$\begin{aligned} \mathbf{e}_u^{(k+1)} &= \sum_{i \in \mathcal{N}(u)} \frac{w_{ui} \mathbf{e}_i^{(k)}}{\sqrt{\sum_{i' \in \mathcal{N}(u)} w_{ui'}} \sqrt{\sum_{u' \in \mathcal{N}(i)} w_{u'i}}} \quad \forall u \in \mathcal{V}_U, \\ \mathbf{e}_i^{(k+1)} &= \sum_{u \in \mathcal{N}(i)} \frac{w_{ui} \mathbf{e}_u^{(k)}}{\sqrt{\sum_{i' \in \mathcal{N}(u)} w_{ui'}} \sqrt{\sum_{u' \in \mathcal{N}(i)} w_{u'i}}} \quad \forall i \in \mathcal{V}_I. \end{aligned} \quad (5.9)$$

Adding edge weights improved the model performance, indicating that the time dependency of interactions is important for capturing user preferences. Results are shown in Figure 5.9 and discussed in Section 5.6.

### 5.5.3 Handling new users

In this section, we explain how the graph neural network architecture can be adapted to handle new users for whom there are no historical preferences.

#### Reformulation of the problem

We divide the set of users into two groups: existing users  $\mathcal{V}_{U,t}$ , and new users (cold-start)  $\mathcal{V}_{U,c}$  such that  $\mathcal{V}_{U,t} \cup \mathcal{V}_{U,c} = \mathcal{V}_U$ ,  $\mathcal{V}_{U,t} \cap \mathcal{V}_{U,c} = \emptyset$ . We assume that we have a model trained on the existing users and their interactions. The goal is to make recommendations to new users. Our goal is to understand the amount of information (past interactions) we need to know about a new user in order to reach a

good understanding of the user's preference. To achieve this goal, we examine the recommendation performance for the new users when varying the number of known interactions of the new users. We denote the number of known interactions of the new users by  $m$ . In this case, we can represent the adjacency matrix of the graph as follows:

$$\mathbf{A} = \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{R}_t \\ \mathbf{0} & \mathbf{0} & \mathbf{R}_c \\ \mathbf{R}_t^T & \mathbf{R}_c^T & \mathbf{0} \end{pmatrix},$$

where  $\mathbf{R}_t$  represents the interactions of the existing users and  $\mathbf{R}_c$  represents the already known interactions of the new users. Note that in the case where we do not know any past interactions of the new users, i.e.,  $m = 0$ , the matrix  $\mathbf{R}_c$  will not have any non-zero element. We also define the matrices  $\mathbf{A}_t$  and  $\mathbf{A}_c$  as the interactions of the existing users and the known interactions of the new users respectively:

$$\mathbf{A}_t = \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{R}_t \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{R}_t^T & \mathbf{0} & \mathbf{0} \end{pmatrix}, \quad \mathbf{A}_c = \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{R}_c \\ \mathbf{0} & \mathbf{R}_c^T & \mathbf{0} \end{pmatrix}$$

Having a model trained on the adjacency matrix  $\mathbf{A}_t$ , the goal is to predict the next interactions of the new users given the information we already know about their first  $m$  interactions, i.e., the information represented in the matrix  $\mathbf{A}_c$ .

### GNN forward pass for the new users

The GNN forward pass to compute the node embeddings is based on the Equations 5.4 and 5.7. Since the model has been trained only on the existing users' interactions, that is, the matrix  $\mathbf{A}_t$ , we only have the embedding of the items and existing users and do not have the representation of the new users in the embedding space. By leveraging the GNN forward pass together with the  $m$  interactions we already know about the new users, we can compute the embedding of the new users.

To compute the embedding of the new users, we mainly face two challenges: 1) How to normalize the message aggregation in the forward pass, that is, equation (5.4), for the new users? 2) How to initialize the embedding of the new users?

In the standard forward pass of the GNN, we normalize the message aggregation by the square root of the degree of the nodes that are at both ends of the message passing (equation (5.4)). However, we empirically observed that when computing the embedding of a new user, normalizing only by the square root of the user node's

degree leads to a better cold start accuracy. Suppose that  $u_c$  is a new (cold start) user. Then its embedding at the  $(k + 1)^{\text{th}}$  layer of the GNN can be computed as follows:

$$\mathbf{e}_{u_c}^{(k+1)} = \sum_{i \in \mathcal{N}(u_c)} \frac{1}{\sqrt{|\mathcal{N}(u)|}} \mathbf{e}_i^{(k)}, \quad (5.10)$$

To initialize the embeddings of the new users, for instance in the above case  $\mathbf{e}_{u_c}^{(0)}$ , we decided to initialize them with an all-zero vector. In our experiments, we observed that a zero vector initialization led to a better cold start accuracy compared to using the average embedding of the users with the same level and instrument as the cold start user.

## 5.6 Results

In this section we present the results of our experiments and address the following research questions:

- **RQ1:** How does our proposed approach perform in comparison with the state-of-the-art classical recommender systems and neural recommender systems (batch or sequential)?
- **RQ2:** Do the recommendations of our approach make sense considering the user’s expertise, instrument of choice, and musical taste?
- **RQ3:** Can the item embeddings learnt by our architecture capture the similarities between music sheets?
- **RQ4:** Under a cold-start setting, how many historical interactions do we need to know for a new user to make good recommendations?

In what follows, we present the experimental settings together with the answers to the above research questions.

### 5.6.1 Accuracy of predictions

**Dataset.** The Tomplay dataset that we used for the experiments of this section has in total 7,510,000 interactions from 35,028 users practicing 33,397 music partitures (items).

**Baseline methods.** We compare our GNN approach to various methods, including both neural network (NN) based recommender systems and classic recommender systems such as matrix factorization:

- **Matrix Factorization.** This method uses the classic matrix factorization [155] approach to derive latent vectors for users and items. The dot product of these latent vectors gives the predicted score for a user-item pair. For training this model, we used the Bayesian Personalized Ranking (BPR) [156] loss function.
- **Factorization Machine** [157]. This method learns user and item vector representations and uses both first-order and second-order user/item feature interactions in order to predict recommendations.
- **Neural Factorization Machine** [158]. This method combines the linearity of factorization machines and the non-linearity of neural networks to model higher-order feature interactions.
- **Neural Collaborative Filtering** [159]. This model learns embedding vectors to represent users and items. To predict recommendations, this model uses a multi-layer perceptron (MLP) network to learn the user-item interaction function.
- **TiSASRec** [160]. This is a sequential recommendation approach that not only models the time order of interactions using self-attention layers, but also models the *actual time interval* between interactions.
- **Wide & Deep** [161]. This method jointly trains wide linear models and deep neural networks to combine the benefits of memorization and generalization for recommender systems.
- **Neural Network + entity resolution** [69]: Our neural recommender system as described in Section 5.4.

**Parameters settings** We implemented the models in PyTorch. The embedding size is tuned among  $\{64, 128, 256, 512\}$  for all baselines and fixed to 512 for LightGCN. We used the Adam optimizer with a batch size fixed to 1024. The learning rate is tuned among  $\{0.001, 0.0001\}$  across all models. For our GNN architecture, the number of layers is tuned among the set of values  $\{1, 2, 3, 4\}$ .



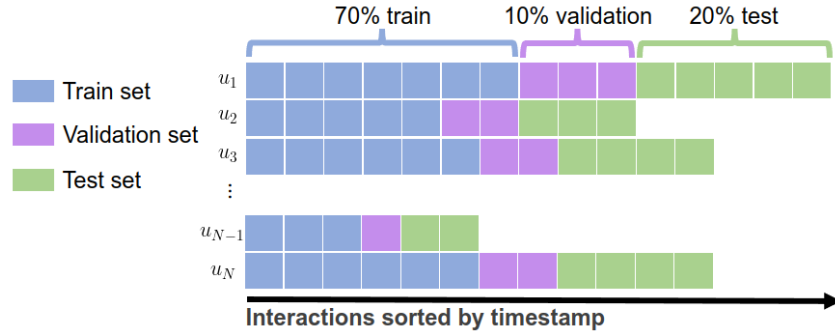


Figure 5.8: Evaluation process. Split: The first 70% of each user interactions are in the training set, the next 10% in the validation set, and the last 20% in the test set. When training the GNN model, the edges corresponding to validation and test interactions are not included in the graph. In the test phase, the validation edges are added to the graph.

**Evaluation.** To evaluate the performance of item recommendation, we re-organized each user’s interactions based on time, and kept 20% of the most recent interactions in the test set, the last 10% of the interactions before the test interactions in the validation set, and the rest in the training set (see Figure 5.8). For evaluation metrics, we used: Recall@25, Precision@25, HR@10, and NDCG@10. Recall@25 shows how much of the ground truth items in the test set are among the top 25 recommendations for a user, whereas Precision@25 computes the fraction of the top 25 recommended items that are included in the test set. To compute HR@10 and NDCG@10, we use a method similar to [159]. For each user, we considered the oldest item in the test set as the test item. Then we sampled 99 items with which the user has not interacted in the past, and ranked the test item among these 100 items. HR@10 computes the existence of the test item among the top 10 recommendations, while NDCG@10 also considers the rank of the test item among the top 10 recommendations and assigns higher values to higher ranks. To assess the statistical significance of our results, we repeated the evaluation phase multiple times and each time we sub-sampled 50% of the test set randomly. Also, we trained the models multiple times with a random initialization of the networks. In this way, we assess robustness with respect to variation of data and network initialization. Finally, we performed a t-test to determine whether the difference in performance between our approaches and the comparative approaches is statistically significant. We can observe that the majority of the reported results between our approach and all other techniques are statistically significant for a  $p$ -value  $< 0.001$ .

Table 5.2 presents the performance of different recommender models for the

Table 5.2: Comparison of various batch and sequential recommendation approaches on the Tomplay dataset. All reported results between our approach and the other methods are statistically significant for  $p$ -value  $< .001$ .

| Model                                      | F1-score     | Recall@25    | Precision@25 | HR@10        | NDCG@10        |
|--|--------------|--------------|--------------|--------------|----------------|
| Matrix Factorization [155]                 | 0.09***      | 0.14***      | 0.07***      | 0.93***      | 0.74***        |
| Factorization Machine [157]                | 0.184***     | 0.337***     | 0.127***     | 0.967***     | 0.864*         |
| Neural Factorization Machine [158]         | 0.185***     | 0.322***     | 0.130***     | 0.943***     | 0.847***       |
| Neural Collaborative Filtering (NCF) [159] | 0.19***      | 0.33***      | 0.13***      | 0.97***      | <b>0.87***</b> |
| TiSASRec [160]                             | 0.182***     | 0.226***     | 0.152***     | 0.954***     | 0.863***       |
| Wide & Deep [161]                          | 0.167***     | 0.319***     | 0.131***     | 0.959***     | 0.861          |
| Deep & Cross [162]                         | 0.191***     | 0.333***     | 0.134***     | 0.964***     | 0.869***       |
| Neural Network + entity resolution [69]    | 0.205***     | 0.353***     | 0.145***     | 0.998        | 0.810***       |
| This work: GNN + entity resolution         | <b>0.239</b> | <b>0.382</b> | <b>0.174</b> | <b>0.996</b> | 0.86           |

\*\*\*:  $p$ -value  $< 0.001$ , \*\*:  $p$ -value  $< 0.01$ , \*:  $p$ -value  $< 0.1$

Tomplay dataset. These results show the superiority of our approach across a variety of metrics. Our method performs particularly well with respect to recall and precision metrics. This means that our approach performs better in recommending items that will be in the interest of the user for future interactions. This is due to the fact that our approach alleviates the problem of having multiple instances of the same song, and recommends items that both match the musical taste of the user and her level of expertise and instrument of choice. Our approach is only marginally inferior to some of the other baseline models in terms of the NDCG metric. The NDCG metric has a higher value when the test item has a higher rank among the recommendations. However, for our application, it is the F1-score, recall, and precision metrics that are of more interest to the user. They weigh equally the global basket of future recommendations, and this is exactly what the user needs to see in this application: all the potential music sheets to explore based on past preferences.

### 5.6.2 Examples of recommendations

Here we showcase some examples of recommendations generated by our recommender system. We show the actual top-10 recommended music sheets for random users in Tables 5.3 and 5.4. From these results, we can observe that even for the recommended music sheets that were not included in the user’s test set (the rows that are not colored green), they are still compatible with the user’s difficulty level and instrument. Moreover, the recommended items closely capture a user’s musical taste as their style and composer reflect similar music genres.

In Table 5.5 we observe recommendations for a sampled user given by our neural network (Section 5.4) and GNN (Section 5.5) recommender systems. We observe that their predictions are quite similar and the recommendations from the GNN-based

Table 5.3: Recommended songs for a sample user with 101 interactions in the training set. Highlighted green rows indicate the undeniably "correct" recommendations, i.e., those which were present in the user's test set.

| Title  | Composer       | Style           | Level                | Instrument |
|--|----------------|-----------------|----------------------|------------|
| The Four Seasons<br>'Spring' - I. Allegro      | Vivaldi        | Classic         | Very easy            | Flute      |
| The Twelve Days of Christmas                   | Traditional    | Christmas music | Easy to intermediate | Flute      |
| Titanic - My Heart Will Go On<br>(level easy)  | Horner (James) | Film music      | Easy                 | Flute      |
| Marche nuptiale                                | Wagner         | Classic         | Very easy            | Flute      |
| Pirates of the Caribbean                       | Zimmer (Hans)  | Film music      | Intermediate         | Flute      |
| We Wish You a Merry Christmas                  | Traditional    | Christmas music | Very easy            | Flute      |
| The Nutcracker - Dance of the Sugar-Plum Fairy | Tchaikovsky    | Classic         | Very easy            | Flute      |
| Overture 1812, Opus 49 - Andante               | Tchaikovsky    | Classic         | Easy                 | Flute      |
| Ochi Chernye                                   | Traditional    | World music     | Easy to intermediate | Flute      |
| The Four Seasons<br>'Spring' - I. Allegro      | Vivaldi        | Classic         | Difficult            | Flute      |

Table 5.4: Recommended songs for a sample user with 81 interactions in the training set. Highlighted green rows indicate the undeniably "correct" recommendations, i.e., those which were present in the user's test set.

| Title   | Composer        | Style       | Difficulty                | Instrument |
|---|-----------------|-------------|---------------------------|------------|
| Chasing Cars (level intermediate)   | Snow Patrol     | Pop/rock    | Intermediate              | Violin     |
| Honeysuckle Rose  | Fats Waller     | Jazz        | Easy to intermediate      | Violin     |
| The Sound of Safed (Klezmer)  | Sher            | World music | Easy to intermediate      | Violin     |
| Summertime (level intermediate)   | Gershwin        | Jazz        | Intermediate              | Violin     |
| Ki Mitziyon (Klezmer)   | Burstyn         | World music | Easy                      | Violin     |
| Tico Tico no Fuba   | Traditionnel    | World music | Intermediate              | Violin     |
| What a Wonderful World<br>(level intermediate/difficult)                    | Louis Armstrong | Jazz        | Intermediate to difficult | Violin     |
| Tango Habanera  | Nazareth        | World music | Intermediate              | Violin     |
| Concerto pour deux violons en ré mineur,<br>BWV 1043 – I. Vivace (Violon 1) | Bach            | Classic     | Intermediate              | Violin     |
| Pirates des Caraïbes (level intermediate)                                   | Zimmer (Hans)   | Film music  | Intermediate              | Violin     |

Table 5.5: Comparison of recommended songs between our Neural recommender system (above double horizontal lines), and by our GNN recommender systems (below double horizontal line) for a user who had 79 interactions in the training set. Highlighted green rows indicate the undeniably ”correct” recommendations, i.e., those which were present in the user’s test set.

| Title  | Composer       | Style            | Level       | Instrument |
|--|----------------|------------------|-------------|------------|
| Titanic - My Heart Will Go On (niveau débutant)    | Horner (James) | Musique de film  | Beginner    | Guitar     |
| Sur le pont d’Avignon                              | Traditionnel   | Enfants          | Beginner    | Guitar     |
| Au clair de la lune                                | Traditionnel   | Enfants          | Beginner    | Guitar     |
| A la claire fontaine                               | Traditionnel   | Enfants          | Beginner    | Guitar     |
| Alouette   | Traditionnel   | Enfants          | Very Easy   | Guitar     |
| Cadet Rouselle                                     | Traditionnel   | Enfants          | Very Easy   | Guitar     |
| L’eau vive   | Béart          | Musique du monde | Très Facile | Guitar     |
| Vive le vent (niveau facile)                       | Pierpont       | Musique de Noël  | Easy        | Guitar     |
| 1492 : Christophe Colomb - Conquest of Paradise    | Vangelis       | Musique de film  | Beginner    | Guitar     |
| Ton invitation (niveau facile, guitare acoustique) | Louise Attaque | Pop/rock         | Easy        | Guitar     |
| Titanic - My Heart Will Go On (niveau débutant)    | Horner (James) | Musique de film  | Beginner    | Guitar     |
| Old MacDonald Had a Farm                           | Traditionnel   | Enfants          | Beginner    | Guitar     |
| Sur le pont d’Avignon                              | Traditionnel   | Enfants          | Beginner    | Guitar     |
| Frère Jacques                                      | Traditionnel   | Enfants          | Beginner    | Guitar     |
| Au clair de la lune                                | Traditionnel   | Enfants          | Beginner    | Guitar     |
| A la claire fontaine                               | Traditionnel   | Enfants          | Beginner    | Guitar     |
| Joyeux anniversaire                                | Traditionnel   | Enfants          | Beginner    | Guitar     |
| Une souris verte                                   | Traditionnel   | Enfants          | Beginner    | Guitar     |
| Alouette   | Traditionnel   | Enfants          | Very Easy   | Guitar     |
| Le Parrain (niveau très facile, guitare seule)     | Nino Rota      | Musique de film  | Very Easy   | Guitar     |

model are slightly more accurate.

Another way to use the recommendation model is for generating ”more like this” recommendations, when a user visits a particular music-sheet. Therefore, the recommendations now are not necessarily tailored to a particular user, but capture global similarities between songs. This can be achieved by computing the embedding of each item using our architecture, and then finding the k-Nearest-Neighbors (kNNs) of that item in the embedding space. We show an example of this for the ”We wish you a Merry Christmas” song for the violin in Table 5.6. We see that the 10 most similar items are related to Christmas music. The second example in Table 5.7 is for a rock song, ”She Said” at intermediate level for Drums. The kNNs of that item are also music-sheets for drums in the Pop/rock style and at intermediate or difficult levels.

### 5.6.3 Tuning the GNN recommender system

Our GNN recommender system has several hyperparameters that can influence its performance, such as the embedding size, the learning rate, the number of layers, and the hyperparameter  $\gamma$  related to edge weights (see Section 5.5.2). Our hyper-

Table 5.6: Top-10 similar items to the first item highlighted in grey. The item similarities are derived by computing the euclidean distance between the trained song embeddings.

| Title   | Composer     | Style           | Difficulty           | Instrument |
|---|--------------|-----------------|----------------------|------------|
| We Wish you a Merry Christmas<br>(Quatuor - Violin 1)             | Traditionnel | Christmas music | Intermediate         | Violin     |
| Deck the Halls (Violin 1)   | Traditionnel | Christmas music | Intermediate         | Violin     |
| Vive le vent (Violon 1)   | Pierpont     | Christmas music | Easy                 | Violin     |
| Away in a Manger (Là-bas dans une étable)<br>(Quatuor - Violin 1) | Traditionnel | Christmas music | Easy                 | Violin     |
| Les Douze Jours de Noël<br>(The Twelve Days of Christmas)         | Traditionnel | Christmas music | Easy                 | Violin     |
| The First Noël (Quatuor, Violin 1)                                | Traditionnel | Christmas music | Intermediate         | Violin     |
| Deck the Halls (Violin 2)   | Traditionnel | Christmas music | Intermediate         | Violin     |
| Carol of the Bells (level easy)                                   | Traditionnel | Christmas music | Easy                 | Violin     |
| Les anges dans nos campagnes                                      | Traditionnel | Christmas music | Very Easy            | Violin     |
| Les gammes Tomplay, Vol. 3 – N° 23 Si majeur                      | Cherubini    | Classic         | Easy to intermediate | Violin     |
| Douce nuit, sainte nuit (level very easy, Duo)                    | Gruber       | Christmas music | Very easy            | Violin     |

Table 5.7: Top-10 similar items to the first item highlighted in grey. The item similarities are derived by computing the euclidean distance between the trained song embeddings.

| Title  | Composer       | Style    | Difficulty                | Instrument |
|--|----------------|----------|---------------------------|------------|
| She Said - Version original<br>(level intermediate)          | Plan B         | Pop/rock | Intermediate              | Drums      |
| Don't Speak (level intermediate/difficult)                   | No Doubt       | Pop/rock | Intermediate to difficult | Drums      |
| Shotgun - Version original<br>(level intermediate/difficult) | George Ezra    | Pop/rock | Intermediate to difficult | Drums      |
| Do I Wanna Know? (level intermediate)                        | Arctic Monkeys | Pop/rock | Intermediate              | Drums      |
| Should I Stay Or Should I Go<br>(level Intermediate)         | The Clash      | Pop/rock | Intermediate              | Drums      |
| Uptown Girl - Version original<br>(level Intermediate)       | Billy Joel     | Pop/rock | Intermediate              | Drums      |
| Sweet Home Alabama (level intermediate)                      | Lynyrd Skynyrd | Pop/rock | Intermediate              | Trumpet    |
| Mr. Jones (level intermediate/difficult)                     | Counting Crows | Pop/rock | Intermediate to difficult | Drums      |
| Sweet Child O' Mine (level intermediate)                     | Guns N' Roses  | Pop/rock | Intermediate              | Drums      |
| The Reason (level difficult)                                 | Hoobastank     | Pop/rock | Difficult                 | Drums      |
| Take on Me (level intermediate)                              | A-Ha           | Pop/rock | Intermediate              | Drums      |

parameter tuning experiments revealed that the values 0.0001 for the learning rate and 512 for the embedding size lead to the best performance for the GNN recommender system.

Table 5.8 demonstrates the performance of our GNN recommender system with respect to the increasing number of layers. Each model had the lightweight entity resolution component and was trained on a graph with edge weights as presented in Section 5.5.2 and with  $\gamma = 2$ . The best performance was attained when using only one layer of GNN. We argue that this is mainly due to the repetitive interactions of users with the same item which makes the diameter of the graph smaller. Therefore, using more graph convolution layers may lead to over-smoothing, degrading the recommendation performance.

Table 5.8: Performance of our GNN-based model for different number of layers.

| Number of layers | F1-score     | Recall@25    | Precision@25 |
|------------------|--------------|--------------|--------------|
| <b>1</b>         | <b>0.239</b> | <b>0.382</b> | <b>0.174</b> |
| 2                | 0.232        | 0.376        | 0.168        |
| 3                | 0.218        | 0.357        | 0.157        |

In Figure 5.9 we can observe the performance of our GNN-based recommender system when using the edge weights described in Section 5.5.2 with different values of  $\gamma$ . The value of  $\gamma$  determines the extent of the importance we give to the most recent interactions. The curve of Recall@25 as a function of  $\gamma$  exhibits an inverted "U" shape, indicating that up to a certain point, giving more importance to the most recent interactions improves the model performance. These results were obtained by training a GNN model with an embedding size of 64. With an embedding size of 512, between  $\gamma = 0$  and  $\gamma = 2$  we had a 2.5% increase in Recall@25 (p-value of  $5.10^{-5}$ ).

#### 5.6.4 Cold start evaluation

In this section, we analyze the ability of our models to recommend items to new users. A new user is one for whom there are no past interactions. Therefore, for that user, there is no trained embedding vector and we compute its embedding based on what we discussed in Section 5.5.3. For this experiment, we take 20% of the users as new (cold start) users and use the remaining 80% of the users in the training process.

The results of the cold-start experiments also aim to answer an important question, which is, *how many interactions do we need in order to know a new user well enough to make good recommendations?* To answer this question, we evaluate the

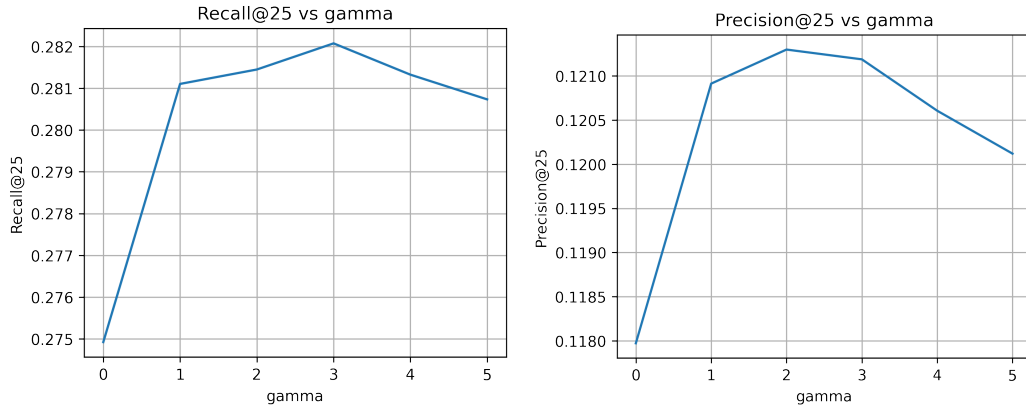


Figure 5.9: Performance of GNN-based recommender system for different values of gamma for the edge weights. Here the models are trained with an embedding dimension of 64 and have only one layer.

recommendation accuracy for the new users while increasing the value  $m$ , i.e., the number of known interactions of the new users. In this experiment, we vary  $m$  in the range of 0 to 300 and evaluate the Recall@25 and Precision@25 for the next 25 interactions. In each step of the experiment, we exclude the users in the set of new users who had less than  $m + 25$  interactions in total. The experimental setup is depicted in Figure 5.10.

**Cold-start evaluation baselines.** We used the following strategies as baselines to compare with our cold-start recommendation strategy:

- **Random:** In this approach, we randomly assign a score to each item and recommend the items with the highest scores.
- **Most popular:** The most popular items in the dataset are suggested to the cold-start users in this approach.
- **Most popular by instrument:** For a new user, we recommend the most popular items among users playing the same instrument as the new user.
- **Most popular by instrument and level:** For a new user, we recommend the most popular items among users playing the same instrument and at the same level as the new user.

Additionally, we also used an approach called Heater [163] as another baseline for recommendation to cold-start users. In this method, the goal is to learn a

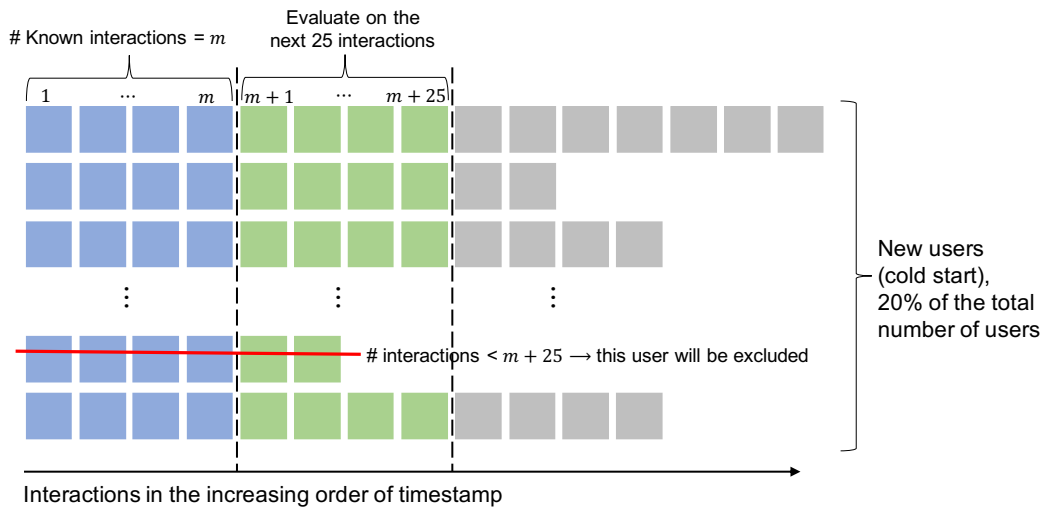


Figure 5.10: Evaluation for the cold start users. For each value of  $m$ , i.e., the number of known interactions for the new users, we compute the embeddings of the new users. Based on these embeddings we evaluate Recall@25 and Precision@25 for the next 25 interactions of each new user. If a user has less than  $m + 25$  interactions, it will be excluded from the experiment.

collaborative filtering representation of new users based on their features while requiring the intermediate representation to be close to a high-quality pre-trained representation. In our case, we use the one-hot encoding of the users' features, i.e., users' level and preferred instrument.

In the first experiment, we evaluate the performance of the above-mentioned baseline approaches. Then, in the subsequent experiments, we will only compare our cold-start recommendation approach with the best baseline. Figure 5.11 shows the performance of the baseline cold-start recommendation approaches in terms of Recall@25 and Precision@25. We can observe that Heater[163] is superior to the other baseline approaches. Therefore, in the next experiment, we only used the Heater approach as a baseline for comparison.

In addition to the baseline approach, we also include the cold-start performance of our neural network-based recommender system (Section 5.4) in our comparison. To address the cold start problem with the neural network-based model, we estimated the embedding vector of a new user using the trained embedding of similar users in the training set. To do so, we averaged the embeddings of users in the training set who had at least one interaction in common with the new user.

Figure 5.12 shows the Recall@25 and Precision@25 as a function of  $m$ , i.e., the number known interactions of new users, for the Heater approach and Our cold-start



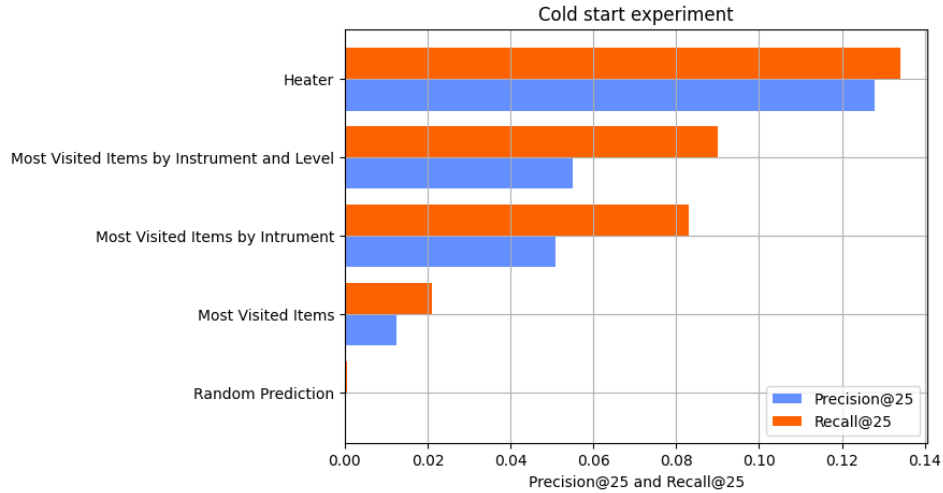


Figure 5.11: Recall@25 and Precision@25 of the 4 baselines on cold start experiments.

recommendations strategies for our neural network-based and graph neural network-based recommender systems. Our cold-start approach based on the GNN model, unlike the other two approaches, takes into consideration the known interactions of the new users. This is reflected by the increase in the cold-start recommendation metrics for the GNN model as we increase the value  $m$ . On the contrary, the cold-start performance of the other two models degrades as the value  $m$  increases. This is mainly due to the fact that these approaches cannot improve the new users' embeddings as the extent of information about the new users increases. Moreover, we can also observe that the cold-start performance of the GNN model slightly degrades for  $m > 50$ . This could be due to our cold-start experiment design (Figure 5.10) and the fact that the set of 25 interactions for which recall and precision are being computed changes at each step.

With these results, we demonstrate that our proposed GNN-based recommender system is well suited for providing relevant recommendations to the users without any, or with little, historical preference information. In addition, with this experiment, we answer the question we mentioned at the beginning of this section, that is, "How many interactions do we need in order to know a new user well enough to make good recommendations"? We observe that having a knowledge of approximately 50 historical interactions of a user provides us with sufficient information in order to generate highly relevant recommendations for them.

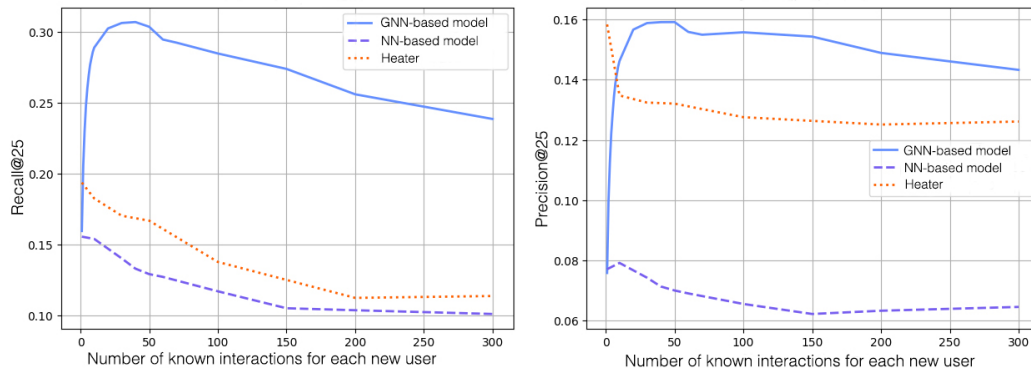


Figure 5.12: Evaluation of the performance for cold start users: Recall@25 (left) and Precision@25 (right) of the Heater approach and our NN-based and GNN-based recommender systems.

### 5.6.5 Explaining the recommendations

In this section, we attempt to leverage the graph characteristics of user interactions together with the GNN recommender system architecture to explain a recommendation made by the model for a specific user. That is, here we attempt to generate local explanations for the recommender system. We explain a specific recommendation by highlighting the most influential items and users that led the model to make this recommendation. For instance, if the model recommends item  $i$  to user  $u$ , the explanation highlights the items among the past interactions of user  $u$  that had the most contribution to this recommendation. Additionally, the explanation highlights the users who previously interacted with the item  $i$  and are the most similar to the user  $u$ . Combining these two sets of information, that is, the most contributing items and users, gives us a clear idea about what has led the model to make this specific recommendation to the user  $u$ .

To generate explanations, we use the GNNExplainer [164] which is a post-hoc interpretability method for explaining the prediction of graph neural networks. The formulation of the GNNExplainer is mainly explained for node classification tasks in the original paper, however, it can be easily extended to link prediction tasks which is the task in recommender systems. GNNExplainer generates an explanation based on a rich sub-graph of the entire graph that the GNN was trained on. Here the key observation is that the computation graph of a node  $v$ , which is defined by the GNN neighborhood-based aggregation mechanism, fully determines all the information that GNN uses to generate a prediction for node  $v$ . The goal of GNNExplainer is to identify a set of pathways within the node  $v$ 's computation graph which are crucial for the classification of node  $v$ . GNNExplainer identifies this set of pathways

by learning a mask matrix through an optimization problem. This mask creates an explanation sub-graph that indicates which edges within the computation graph will be used for prediction. Therefore, the goal of the optimization problem is to maximize the probability of the correct prediction on node  $v$ , given the explanation sub-graph.

To adapt GNNExplainer to the recommendation scenario which is a link prediction task, we need to determine the explanation graph for both of the nodes involved in a prediction, i.e, the user and the item. Additionally, we have to replace the cross entropy loss in the objective of the optimization problem with the BPR loss [156] which is used in recommender systems. In our case, since the GNN with the best recommendation accuracy has only one layer (Table 5.8), the pathways highlighted by GNNExplainer lead us to the most influential items and users toward the recommendation. Figure 5.13 shows an example of the explanation generated by GNNExplainer for a specific recommendation, recommending "Lady Madonna" to the user  $u$ . In this figure, we indicate the importance of the highlighted items and users by the thickness of the arrows. The explanation can identify the items among the past interactions of the user  $u$  which had the most contribution to this recommendation. All of the items highlighted as influential by the explanation are pop/rock songs, similar to the recommended item "Lady Madonna". Interestingly, we can observe that the previous interest of the user  $u$  in The Beatles group, among other factors, has led to this recommendation. Additionally, the explanation also identifies the users who had interacted with the recommended song and had the most contribution to this recommendation. These users have a total of ten interactions in common with the user  $u$ , which shows that their taste is similar to the user  $u$ . Moreover, same as the user  $u$ , these users are all at the intermediate level and play piano. As another example, in Figure 5.14, we show the explanation for the recommendation "La La Land (City of Stars)" to a user. Again here we can observe that the users highlighted as influential by the explanation have a total of 77 interactions in common with the user getting the recommendation, showing that they have very similar interests to this user.

Providing an explanation for a recommendation helps to improve the engagement of a user on the Tomplay platform. For instance in the example depicted in Figure 5.13, the platform can show a message to the user such as "Since you explored *Michell, Let it Be*, etc." and then provide the recommendation "Lady Madonna". Or as another example, in the case of Figure 5.14, the platform can show a message like "Users with similar interests are also practicing *La La Land (City of Stars)*." Providing the explanations in such ways would help to attract the users more to the recommendations and hence improve the users' engagement in the platform.

5 most influential items

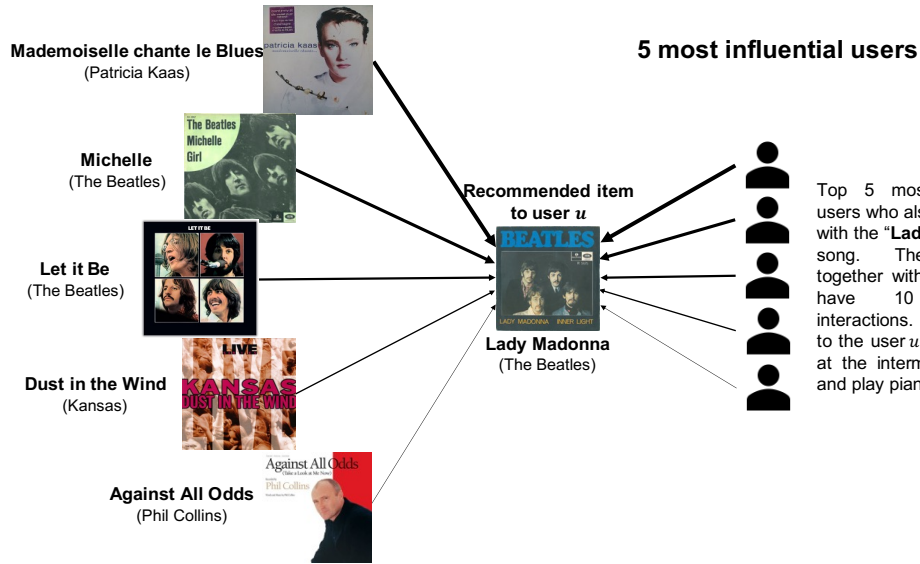


Figure 5.13: Explanation for the recommended track, "You Can Leave Your Hat On", to a specific user. The Explanation shows the top 5 influential items and users towards this recommendation.

5 most influential items

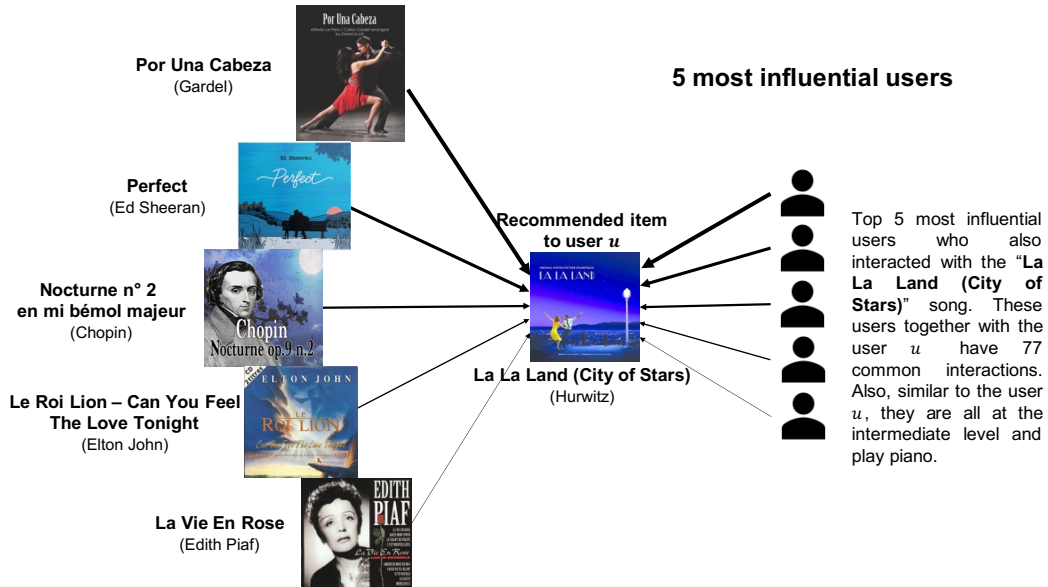


Figure 5.14: Explanation for the recommended track, "La La Land (City of Stars)", to a specific user. The Explanation shows the top 5 influential items and users towards this recommendation.

## 5.7 Discussion

In this work, we introduced a graph neural network-based recommender system with the aim of enhancing the user experience within an educational platform by delivering tailored recommendations. We incorporated several components into the GNN architecture, showcasing their capability to significantly improve recommendation accuracy.

As we look toward the future, a promising avenue for further enhancing such a system lies in transitioning from a homogeneous graph to a heterogeneous graph. This approach allows us to explicitly model the diverse items and user attributes present in the dataset. One exciting prospect is the potential to enhance the recommendation performance by modeling multiple instances of the same songs through specific edge types in this heterogeneous graph. This may unlock new avenues for improving recommendation quality.

Additionally, adopting a heterogeneous graph facilitates an expansion of our explanations beyond merely influential users and items. By incorporating potentially vital user or item features into the explanation of a recommendation, we can provide users with a more comprehensive and insightful rationale behind the suggestions they receive. This dynamic shift from homogeneous to heterogeneous graph modeling paves the way for even more effective and interpretable recommendation systems, further enriching the user experience within educational platforms.

## 5.8 Related Work

**Collaborative Filtering.** Classical recommendation methods such as matrix factorization [165, 157] commonly use separate embedding vectors for users and items. They are often combined using a dot product giving a rankable score for each item. In particular, factorization machines [157] combine factorization models with SVMs tailoring applications of high data sparsity like recommender systems. A key limitation is the limited expressiveness of classical methods preventing them from learning complex relationships between users and items. Deep learning allows to enhance the interaction function as witnessed by a variety of works [159, 158, 166, 161, 162]. The neural factorization machines [158] combine factorization machines and deep learning. Neural collaborative filtering [159] replaces the dot product in classical collaborative filtering with a neural network allowing to learn arbitrary non-linear functions. [161] combined generalized linear models that are simple, effective and interpretable with deep learning techniques to create low-dimensional, dense embeddings to boost generalization. Interpretability has also been a key element in

other works, e.g., [166, 167]. The Deep & Cross network [162] has been proposed in the context of ad click prediction. It utilizes a cross-network to more efficiently learn feature interactions of bounded degree than deep neural networks that learn all interactions implicitly. There are also multiple works on the cold-start problem. For example, recently [168] considered interactively exploring user preferences using a multi-armed bandit approach.

**Cold start issues in recommender systems.** Cold start users and items pose several challenges for modern recommender systems, especially when it comes to new user engagement. There have been several works in the recommender system literature to address the cold start issue. For a complete review, we refer the readers to the systematic review by [169]. Research on cold start strategies can be broadly classified into two categories: data-driven strategies and approach-driven strategies [169]. The primary purpose of data-driven approaches is to effectively use different types of data to alleviate the problems with new users and items. For instance [170] and [171] use social network data to enhance the recommendations for cold-start users. On the other hand, approach-driven strategies focus on defining a new algorithm to address the cold start problems. [172] introduced MSIR, a deep neural network approach that leverages multiplex interaction data to extract latent features. [173] developed a two-stage neural network system, DACR, and NNCF, which utilizes item content and collaborative filtering for cold start item rating predictions. [174] presented SADE, a deep neural network, and timeSVD++, a temporal collaborative filtering model, both addressing cold start scenarios. [175] proposes a two-stage soft-cluster embedding algorithm within a matrix factorization framework that effectively addresses the cold start item problem by leveraging side information about items to create soft item clusters for generating recommendations.

**Graph Neural Networks.** Recent advances in graph neural networks (GNNs) have led to state-of-the-art recommender systems that capture information by exploiting the user-item graph structure of the system. For recent surveys, see [176, 177]. We focus on major developments relevant to our work. In particular, NGCF [178] adapts collaborative filtering using the message-passing architecture of GNNs to propagate the collaborative filtering signal. It uses a model similar to the Graph Convolutional Network and adds a term in the message passing that captures the interaction between users and items. However, it was then observed that removing the weight of NGCF resulted in better efficiency and competitive performance. The observation motivated the design of LightGCN [154], which is a message-passing network that simplifies NGCF by only using neighborhood aggregation. In addition, the effectiveness of recommender systems could be enhanced with extra

knowledge of item properties, leading to the designs of knowledge graph-based graph recommendation models [179, 180, 181]. Furthermore, sequential signals could be leveraged to focus on predicting the next item given users' historical behaviors [182, 183], which overcomes the limitation of traditional recommendation tasks that treat user preferences in a static fashion.

**Sequential Recommender Systems.** Sequential Recommender Systems take into account the time-wise ordering of user-item interactions. Early approaches like [184] relied on matrix factorization and Markov chains. Newer approaches rely on neural architectures such as recurrent neural networks [185, 186], convolutional neural networks [187], and self-attention networks [188, 160]. [160] employed neural networks and, in particular, it considered not only the order of interactions of users with items, but also the time difference between them.

**Educational recommender systems.** Educational recommender systems can be teacher or student facing [189, 190, 191]. [192] recommended MOOC courses using a deep belief network. Given the limited time budget a learner has available, [193] derived algorithms using classical methods such as matrix factorization to maximize the learning outcomes of courses. If a student failed to reach the required scores, the system would recommend auxiliary learning objects. More generally, educational recommender systems propose a learning path [194, 195], that is, a personalized implementation of a curriculum design composed of a set of learning activities that should enable users to achieve particular learning goals. In contrast, we are primarily concerned with matching users' interests.

**Sentence Embeddings.** BERT is a language representation model based on transformers [196] superseding word vectors [197], which are the successors of one-hot encodings of words. Elements of the BERT neural architecture have also been employed for recommender systems, for example, BERT4Rec [198] employs a bidirectional self-attention network. The use of BERT embedding vectors has also been explored recently for recommendations [199] confirming that such vectors contain valuable information on the content of books, music and movies. Reimers et al. [153] claim that vector spaces between languages are not aligned for BERT, that is, sentences with identical content in different languages are mapped to varying locations in the vector space. This motivated sentence-BERT [153] and other multi-lingual models such as LabSE [200] creating sentence embeddings close to each other if the original sentences are semantically similar.

## 5.9 Conclusion

In this study, we examined how to address the issue of choice overload in an educational music learning platform. Our recommendation framework is developed by considering several unique patterns that exist in educational platforms. First, such platforms offer content at varying levels of difficulty to cater to users' learning needs, so we need to determine the user's knowledge level and recommend appropriate content. Second, users interact with fewer items on educational platforms than they do on traditional e-commerce, content streaming, and social networking sites, resulting in sparse interactions. Third, the interactions between users and the multiple instances of items can easily be modeled as a graph.

To address these issues, we incorporate a component that implements a lightweight entity resolution based on text embeddings, which identifies similar or identical content instantiated at various difficulty levels and connects them to users who have interacted with them, thus reducing data sparsity. We propose a deep-learning recommender systems using a graph neural networks architecture and demonstrate its efficacy in performing content filtering for the Tomplay platform. The graph neural networks outperforms a typical deep-learning model in terms of precision and recall. We conduct several experiments to test our predictive model and compare our model against state-of-the-art recommendation systems using data collected from Tomplay's global music learning platform. The results of the experiments provide strong evidence for the effectiveness of our proposed recommendation framework.

Our study contributes to the fields of business and big data analytics [201] and business intelligence [202]. We show that prescriptive analytics based on graph neural recommenders can generate valuable predictions for content to access, something that is beneficial for both the end-users and for the enterprise. The neural and graph-neural recommendation frameworks we developed are significant contributions to the literature on recommendation systems and online learning tools. Our work also highlights the unique access patterns of an educational setting, which has not been previously explored. From a practical standpoint, our recommendation framework can be applied to real-world recommendation problems in learning platforms, as demonstrated in the case of Tomplay, to alleviate information overload. Additionally, we demonstrate the benefits of incorporating an entity resolution component based on text embeddings, which improves the precision and recall of the recommendation system.

Our cold-start experiments provided insights into an interesting question that has been raised in the field of recommender systems: how many historical interactions are necessary to accurately understand a user's preferences? Our findings, based



on the dataset we examined, suggest that 50 historical interactions provide a good understanding of the user’s preference patterns. This result can be compared with existing research in the field of psychological and cognitive sciences, which has explored the accuracy of algorithms in making personality judgments compared to humans [203]. That study found that having access to 60-70 Facebook “Likes” is equivalent to the knowledge captured by a friend of the individual at question.

Our recommendation framework is designed to aid in content discovery on online learning platforms, but its methodology can be applied to a range of problem settings where content is offered in various instantiations and catered to users with different levels of expertise. One potential extension is to combine it with interpretable machine learning, which generates explanations for predictive decisions at varying levels of detail and complexity to serve different stakeholders. For instance, in a medical context, a doctor might receive a more technical explanation than a patient.

# Chapter 6

## Conclusion

Driven by the vital importance of unraveling the intricate inner workings of opaque machine learning (ML) models for enhanced understanding, trust, and fairness, this thesis delved into the realm of Interpretable Machine Learning (IML). Our exploration spanned diverse facets. Initially, we proposed a method to construct a more interpretable low-dimensional embedding space. Embedding methods are a ubiquitous need in modern ML tasks dealing with large-scale high-dimensional data. However, the lack of interpretability in such embeddings hinders their applicability in critical domains. As a pivotal contribution, we inspected the robustness and utility of attribution methods designed to explain deep neural network predictions. Our focus centered on probing these methods' resilience against adversarial attacks aimed at altering explanations. Our findings underscored the vulnerability of common robust explanations and their limitations in achieving desired explanation properties. Notably, we extended our investigation to introduce sparse explanation attacks, shedding light on the potential occurrence of disruptive perturbations in real-life scenarios, which could in turn undermine the reliability of attribution methods. Finally, we showcased the practical utility of IML through a graph neural network-based recommender system that provides insightful explanations for recommendations, exemplifying the tangible benefits of interpretable approaches in real-world applications.

In the following, we review the main contributions of this thesis and discuss some possible future research directions.

## 6.1 Review of main contributions

In chapter 2 we introduced a novel embedding method, MoDE, capable of operating on inexact distance information, characterized by ranges of upper and lower bounds. This novel capability is relevant to compressed data scenarios, which our work primarily addressed. Given its capacity to visualize compressed data, our approach is well-suited for dimensionality reduction and visualization of extensive datasets. Our empirical investigations underscored its competitive embedding quality and classification accuracy compared to prevailing techniques, all while incurring a reduced computational cost. Furthermore, thanks to the objects' order preservation property, in the two-dimensional embedding space resulting from MoDE, the positioning of objects and resulting visualization remains highly interpretable to human observers.

In chapter 3 we assessed the shortcomings and strengths of "smoothed explanations" that are supposed to be robust against adversarial attacks. Our evaluation encompassed two vital aspects: explanation quality and explanation robustness. In terms of quality, a comprehensive examination encompassing sensitivity to model parameters, class discriminability, sparsity, and infidelity was conducted. Notably, our findings indicated that the investigated smoothed explanations underperformed the Gradient method in terms of sensitivity, class discriminability, and sparsity (except for adversarial training). However, smoothed explanations demonstrated improvement in infidelity. Furthermore, we delved into the robustness of explanations under additive and non-additive attacks. The results emphasized the persistence of non-additive attacks as a concern for explanation methods, including the smoothed variants.

In chapter 4 we focused on the sparse attacks to manipulate the explanation of a deep neural network. The main challenges of computing a sparse perturbation for this task are the NP-hardness of  $\ell_0$  minimization, the need for adherence to valid data ranges, and the preservation of the model prediction. Through these challenges, we proposed an innovative algorithm, outperforming the PGD attack with  $\ell_0$  projection. This method not only achieved better results in terms of dissimilarity between original and manipulated explanations but also yielded sparser perturbations. Notably, it showcased the ability to effectively conceal the attribution of  $k$  most important features in the original explanation by perturbing less than  $k$  features in the input data. Our findings emphasized the far-reaching implications of these attacks in domains where reliable explanations are indispensable for ensuring fairness and mitigating bias. Unveiling vulnerabilities within existing explanation methods, our research advocates for the development of more robust explanation techniques.

In chapter 5 we addressed the choice overload in an educational music platform

by tailoring a recommendation system to the unique platform patterns. These patterns include variable content difficulty, sparse user-item interactions, and the graph-like nature of the user-item relationships. Our solution involves a lightweight entity resolution based on text embeddings to reduce data sparsity and utilizing a graph neural network-based recommender system. The proposed model outperforms typical deep-learning models in content filtering, demonstrated through experiments on Tomplay’s platform data. The results underscore the effectiveness of the proposed framework in handling educational platform challenges. Moreover, we demonstrated that by leveraging the user and item sub-graphs we are able to generate insightful explanations for recommendations by highlighting the key users and items that influenced the recommendation.

## 6.2 Future directions

In the context of the increasing adoption of opaque machine learning models in critical applications, the need for transparency and trustworthiness has become imperative. Achieving transparency requires effective explanation methods to illuminate the model’s internal mechanisms. Additionally, selecting suitable explanation methods for specific tasks and ensuring the trustworthiness of these explanations are equally crucial. Our research addressed the quality and robustness of attribution methods for explaining deep neural networks, acknowledging the ongoing challenges.

However, evaluating attribution methods remains complex, particularly due to the absence of clear ground truth for explanations. Presently, evaluations rely on subjective criteria such as smoothness and sparsity, which can bias the process toward human expectations. Such biased criteria might lead to favoring attribution methods that produce results closer to human expectations and might potentially penalize the ones that better explain the underlying model behavior. A promising future direction is the development of objective criteria for evaluation, reducing subjectivity. Furthermore, the evaluation scope predominantly centers on image data, leaving other modalities like tabular, text, and recommendation datasets unexplored.

Regarding robustness, our introduction of attacks to manipulate smoothed explanations highlights the connection between prediction and explanation robustness. Further exploration of this relationship holds potential for future research. Moreover, there is still limited work that analytically explores the explanation robustness from a geometrical standpoint. Analyzing the geometric aspects of deep neural networks could enhance our understanding of explanation robustness and contribute to more

effective attacks. This avenue also presents an interesting research direction.

Considering the emerging applications of Interpretable Machine Learning (IML), the advent of Large Language Models (LLM) like GPT-3 [204] opens doors for generating comprehensive textual explanations. Current approaches rely on large datasets of human-generated explanations to train neural networks to generate explanations [205, 206]. However, human-generated explanations can be biased, and gathering a vast amount of them to make a dataset is very costly. Moreover, using an opaque and complex model to generate explanations is in contrast with the goal of having an explanation for transparency. Leveraging advancements in text and image generation models to produce reliable and adaptable natural language explanations is an exciting direction worth exploring, bridging the gap between complex model-driven explanations and transparency goals. On the other hand, with the advancement of generative AI models, experts in many domains are starting to use such models in their daily tasks. For instance, legal experts are adopting AI to streamline tasks like contract analysis, legal research, and due diligence. IML can provide explanations for AI-generated legal advice, helping lawyers understand how AI systems arrive at their conclusions. This can enhance the trustworthiness of AI-driven legal work.

# Bibliography

- [1] Jing Zheng, Xiao Fu, and Guijun Zhang. Research on exchange rate forecasting based on deep belief network. *Neural Computing and Applications*, 2019.
- [2] Sahar Sohangir, Dingding Wang, Anna Pomeranets, and Taghi M Khoshgoftaar. Big data: Deep learning for financial sentiment analysis. *Journal of Big Data*, 2018.
- [3] Johannes Jurgovsky, Michael Granitzer, Konstantin Ziegler, Sylvie Calabretto, Pierre-Edouard Portier, Liyun He-Guelton, and Olivier Caelen. Sequence classification for credit-card fraud detection. *Expert systems with applications*, 2018.
- [4] Peter C. Esselman, R. Jan Stevenson, Frank Lupi, Catherine M. Riseng, and Michael J. Wiley. Landscape prediction and mapping of game fish biomass, an ecosystem service of michigan rivers. *North American Journal of Fisheries Management*, 2015. doi: <https://doi.org/10.1080/02755947.2014.987887>.
- [5] Renee Obringer and Roshanak Nateghi. Predicting urban reservoir levels using statistical learning techniques. *Scientific Reports*, 2018. doi: 10.1038/s41598-018-23509-w.
- [6] Gregor Stiglic, Primož Kocbek, Nino Fijacko, Marinka Zitnik, Katrien Verbert, and Leona Cilar. Interpretability of machine learning based prediction models in healthcare. *arXiv e-prints*, 2020. doi: 10.48550/arXiv.2002.08596.
- [7] Emmanuel Pintelas, Meletis Liaskos, Ioannis E. Livieris, Sotiris Kotsiantis, and Panagiotis Pintelas. Explainable machine learning framework for image classification problems: Case study on glioma cancer prediction. *Journal of Imaging*, 2020. doi: 10.3390/jimaging6060037.
- [8] Clemens Stachl, Quay Au, Ramona Schoedel, Samuel D. Gosling, Gabriella M. Harari, Daniel Buschek, Sarah Theres Völkel, Tobias Schuwerk, Michelle Oldemeier, Theresa Ullmann, Heinrich Hussmann, Bernd Bischl, and Markus Bühner. Predicting personality from patterns of behavior collected with smartphones. *Proceedings of the National Academy of Sciences*, 2020. doi: 10.1073/pnas.1920484117.
- [9] Xilei Zhao, Xiang Yan, Alan Yu, and Pascal Van Hentenryck. Prediction and behavioral analysis of travel mode choice: A comparison of machine learning and logit models. *Travel Behaviour and Society*, 2020. doi: <https://doi.org/10.1016/j.tbs.2020.02.003>.

- [10] Marcelo O. R. Prates, Pedro H. C. Avelar, and Luis Lamb. Assessing Gender Bias in Machine Translation – A Case Study with Google Translate. *arXiv e-prints*, 2018. doi: 10.48550/arXiv.1809.02208.
- [11] Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad. Intelligent models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2015. Association for Computing Machinery. doi: 10.1145/2783258.2788613.
- [12] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. ”why should I trust you?”: Explaining the predictions of any classifier. In Balaji Krishnapuram, Mohak Shah, Alexander J. Smola, Charu C. Aggarwal, Dou Shen, and Rajeev Rastogi, editors, *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, 2016. doi: 10.1145/2939672.2939778.
- [13] Rebecca Wexler. When a computer program keeps you in jail: How computers are harming criminal justice. *The New York Times*, 2017.
- [14] Brian Edwards. FDA Guidance on Clinical Decision Support: Peering Inside the Black Box of Algorithmic Intelligence. *Chilmark Research*, 2017.
- [15] Ashraf Abdul, Jo Vermeulen, Danding Wang, Brian Y. Lim, and Mohan Kankanhalli. Trends and trajectories for explainable, accountable and intelligible systems: An hci research agenda. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, 2018. Association for Computing Machinery. doi: 10.1145/3173574.3174156.
- [16] Alex Alves Freitas. Comprehensible classification models: a position paper. *SIGKDD Explor.*, 2013. doi: 10.1145/2594473.2594475.
- [17] Johan Huysmans, Karel Dejaeger, Christophe Mues, Jan Vanthienen, and Bart Baesens. An empirical evaluation of the comprehensibility of decision table, tree and rule based predictive models. *Decis. Support Syst.*, 2011. doi: 10.1016/j.dss.2010.12.003.
- [18] David Martens and Bart Baesens. *Building Acceptable Classification Models*, pages 53–74. Springer US, 2010. doi: 10.1007/978-1-4419-1280-0\_3.
- [19] David Lewis. Copyright Page. In *Philosophical Papers Volume II*. Oxford University Press, 1987. doi: 10.1093/0195036468.001.0001.002.003.
- [20] Carl G. Hempel and Paul Oppenheim. Studies in the logic of explanation. *Philosophy of Science*, 1948. doi: 10.1086/286983.
- [21] Zachary C Lipton. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 2018.
- [22] Finale Doshi-Velez and Been Kim. Towards A Rigorous Science of Interpretable Machine Learning. *arXiv e-prints*, 2017. doi: 10.48550/arXiv.1702.08608.

- [23] Tim Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 2019. doi: <https://doi.org/10.1016/j.artint.2018.07.007>.
- [24] James Woodward. *Making Things Happen: A Theory of Causal Explanation*. Oxford University Press, 2004. doi: 10.1093/0195155270.001.0001.
- [25] Wesley C. Salmon. *Four Decades of Scientific Explanation*. University of Minnesota Press, 1989.
- [26] Graziano Mita. *Toward interpretable machine learning, with applications to large-scale industrial systems data. (Contributions à l'apprentissage automatique interprétable : applications aux données de systèmes industriels à grande échelle)*. PhD thesis, Sorbonne University, France, 2021.
- [27] Or Biran and Courtenay V. Cotton. Explanation and justification in machine learning: A survey. 2017.
- [28] Tania Lombrozo. The structure and function of explanations. *Trends in Cognitive Sciences*, 2006. doi: <https://doi.org/10.1016/j.tics.2006.08.004>.
- [29] Fritz Heider and Marianne Simmel. An experimental study of apparent behavior. *The American Journal of Psychology*, 1944.
- [30] Bertram Malle. How the mind explains behavior: Folk explanation, meaning and social interaction. *MIT press*, 2004.
- [31] Been Kim, Oluwasanmi Koyejo, and Rajiv Khanna. Examples are not enough, learn to criticize! criticism for interpretability. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, 2016.
- [32] Aaron Fisher, Cynthia Rudin, and Francesca Dominici. All models are wrong, but many are useful: Learning a variable's importance by studying an entire class of prediction models simultaneously. *J. Mach. Learn. Res.*, 2019.
- [33] Christoph Molnar, Gunnar König, Bernd Bischl, and Giuseppe Casalicchio. Model-agnostic feature importance and effects with dependent features - A conditional subgroup approach. *CoRR*, 2020.
- [34] Amit Dhurandhar, Vijay S. Iyengar, Ronny Luss, and Karthikeyan Shanmugam. TIP: typifying the interpretability of procedures. *CoRR*, 2017.
- [35] Qing Zhou, Fenglu Liao, Chao Mou, and Ping Wang. Measuring interpretability for different types of machine learning models. In Mohadeseh Ganji, Lida Rashidi, Benjamin C. M. Fung, and Can Wang, editors, *Trends and Applications in Knowledge Discovery and Data Mining - PAKDD 2018 Workshops, BDASC, BDM, ML4Cyber, PAISI, DaMEMO, Melbourne, VIC, Australia, June 3, 2018, Revised Selected Papers*, 2018. doi: 10.1007/978-3-030-04503-6\\_29.



- [36] Gregory Plumb, Maruan Al-Shedivat, Ángel Alexander Cabrera, Adam Perer, Eric P. Xing, and Ameet Talwalkar. Regularizing black-box models for improved interpretability. In Hugo Larochelle, Marc’Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.
- [37] Forough Poursabzi-Sangdeh, Daniel G. Goldstein, Jake M. Hofman, Jennifer Wortman Vaughan, and Hanna M. Wallach. Manipulating and measuring model interpretability. In Yoshifumi Kitamura, Aaron Quigley, Katherine Isbister, Takeo Igarashi, Pernille Bjørn, and Steven Mark Drucker, editors, *CHI ’21: CHI Conference on Human Factors in Computing Systems, Virtual Event / Yokohama, Japan, May 8-13, 2021*, 2021. doi: 10.1145/3411764.3445315.
- [38] Sorelle A. Friedler, Chitradheep Dutta Roy, Carlos Scheidegger, and Dylan Slack. Assessing the local interpretability of machine learning models. *CoRR*, 2019.
- [39] Prasad Chalasani, Jiefeng Chen, Amrita Roy Chowdhury, Somesh Jha, and Xi Wu. Concise explanations of neural networks using adversarial training. *CoRR*, 2018.
- [40] Himabindu Lakkaraju, Ece Kamar, Rich Caruana, and Jure Leskovec. Interpretable & explorable approximations of black box models. *CoRR*, 2017.
- [41] Berk Ustun and Cynthia Rudin. Supersparse linear integer models for optimized medical scoring systems. *Mach. Learn.*, 2016. doi: 10.1007/s10994-015-5528-6.
- [42] Hongyu Yang, Cynthia Rudin, and Margo I. Seltzer. Scalable bayesian rule lists. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, 2017.
- [43] Chih-Kuan Yeh, Cheng-Yu Hsieh, Arun Sai Suggala, David I. Inouye, and Pradeep Ravikumar. On the (in)fideliy and sensitivity of explanations. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d’Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, 2019.
- [44] Fabian Offert. “I know it when I see it”. Visualization and Intuitive Interpretability. *arXiv e-prints*, 2017. doi: 10.48550/arXiv.1711.08042.
- [45] Bernease Herman. The Promise and Peril of Human Evaluation for Model Interpretability. *arXiv e-prints*, 2017. doi: 10.48550/arXiv.1711.07414.
- [46] David Gunning and David W. Aha. Darpa’s explainable artificial intelligence (XAI) program. *AI Mag.*, 2019. doi: 10.1609/aimag.v40i2.2850.
- [47] Marco Ancona. *Attribution Methods for Interpreting and Optimizing Deep Neural Networks*. PhD thesis, ETH Zurich, Zürich, Switzerland, 2020.

- [48] Dino Pedreschi, Salvatore Ruggieri, and Franco Turini. Discrimination-aware data mining. In Ying Li, Bing Liu, and Sunita Sarawagi, editors, *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, Nevada, USA, August 24-27, 2008*, 2008. doi: 10.1145/1401890.1401959.
- [49] Sophie Curtis. Google photos labels black people as 'gorillas'. *The Telegraph*, 2015.
- [50] Tolga Bolukbasi, Kai-Wei Chang, James Y. Zou, Venkatesh Saligrama, and Adam Tauman Kalai. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, 2016.
- [51] Marcelo O. R. Prates, Pedro H. C. Avelar, and Luís C. Lamb. Assessing gender bias in machine translation: a case study with google translate. *Neural Comput. Appl.*, 2020. doi: 10.1007/s00521-019-04144-6.
- [52] Christopher Kelly, Alan Karthikesalingam, Mustafa Suleyman, Greg Corrado, and Dominic King. Key challenges for delivering clinical impact with artificial intelligence. *BMC Medicine*, 2019. doi: 10.1186/s12916-019-1426-2.
- [53] Tamra Lysaght, Hannah Lim, Vicki Xafis, and Kee Ngiam. Ai-assisted decision-making in healthcare: The application of an ethics framework for big data in health and research. *Asian Bioethics Review*, 2019. doi: 10.1007/s41649-019-00096-0.
- [54] Sebastian Lapuschkin, Stephan Wäldchen, Alexander Binder, Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. Unmasking clever hans predictors and assessing what machines really learn. *CoRR*, 2019.
- [55] Paul Voigt and Axel Bussche. *The EU General Data Protection Regulation (GDPR): A Practical Guide*. Springer, 2017. doi: 10.1007/978-3-319-57959-7.
- [56] Bryce Goodman and Seth R. Flaxman. European union regulations on algorithmic decision-making and a "right to explanation". *AI Mag.*, 2017. doi: 10.1609/aimag.v38i3.2741.
- [57] Ribana Roscher, Bastian Bohn, Marco F. Duarte, and Jochen Garcke. Explainable machine learning for scientific insights and discoveries. *IEEE Access*, 2020. doi: 10.1109/ACCESS.2020.2976199.
- [58] Richard Tomsett, Dave Braines, Dan Harborne, Alun D. Preece, and Supriyo Chakraborty. Interpretable to whom? A role-based model for analyzing interpretable machine learning systems. *CoRR*, 2018.
- [59] Jessica Dai, Sohini Upadhyay, Ulrich Aivodji, Stephen H. Bach, and Himabindu Lakkaraju. Fairness via explanation quality: Evaluating disparities in the quality of post hoc explanations. *CoRR*, 2022.

- [60] Ann-Kathrin Dombrowski, Maximilian Alber, Christopher J. Anders, Marcel Ackermann, Klaus-Robert Müller, and Pan Kessel. Explanations can be manipulated and geometry is to blame. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, 2019.
- [61] Christopher J. Anders, Plamen Pasliev, Ann-Kathrin Dombrowski, Klaus-Robert Müller, and Pan Kessel. Fairwashing explanations with off-manifold detergent. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, 2020.
- [62] Zifan Wang, Haofan Wang, Shakul Ramkumar, Piotr Mardziel, Matt Fredrikson, and Anupam Datta. Smoothed geometry for robust attribution. In Hugo Larochelle, Marc' Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.
- [63] Scott Lundberg and Su-In Lee. A Unified Approach to Interpreting Model Predictions. *arXiv e-prints*, 2017. doi: 10.48550/arXiv.1705.07874.
- [64] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, 2017.
- [65] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, 2017.
- [66] Nikolaos M. Freris, Michalis Vlachos, and Ahmad Ajalloeian. An interpretable data embedding under uncertain distance information. In *20th IEEE International Conference on Data Mining, ICDM, 2020*.
- [67] Nikolaos M. Freris, Ahmad Ajalloeian, and Michalis Vlachos. Interpretable embedding and visualization of compressed data. *ACM Trans. Knowl. Discov. Data*, 2023. doi: 10.1145/3537901.
- [68] Ahmad Ajalloeian, Seyed-Mohsen Moosavi-Dezfooli, Michalis Vlachos, and Pascal Frossard. On smoothed explanations: Quality and robustness. In *the ACM International Conference on Information & Knowledge Management, 2022*, 2022.
- [69] Ahmad Ajalloeian, Michalis Vlachos, Johannes Schneider, and Alexis Steinmann. A case study in educational recommenders: Recommending music partitures at tomplay. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management, 2022*.
- [70] Trevor F Cox and Michael AA Cox. *Multidimensional Scaling*. Chapman and Hall/CRC, 2000.

- [71] Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 2000.
- [72] Sam T Roweis and Lawrence K Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 2000.
- [73] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 2008.
- [74] Ella Bingham and Heikki Mannila. Random projection in dimensionality reduction: applications to image and text data. In *Proceedings of the 7th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2001.
- [75] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Comput.*, 2003.
- [76] Leland McInnes and John Healy. UMAP: Uniform Manifold Approximation and Projection for dimension reduction. *arXiv preprint:1802.03426*, 2018.
- [77] Michail Vlachos, Nikolaos Freris, and Anastasios Kyrillidis. Compressive mining: fast and optimal data mining in the compressed domain. *The International Journal on Very Large Data Bases (VLDB Journal)*, 2015.
- [78] Emmanuel Candes, Justin Romberg, and Terence Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, 2006.
- [79] Johan Paratte, Nathanaël Perraudin, and Pierre Vandergheynst. Compressive embedding and visualization using graphs. *arXiv preprint:1702.05815*, 2017.
- [80] Jian Tang, Jingzhou Liu, Ming Zhang, and Qiaozhu Mei. Visualizing large-scale and high-dimensional data. In *Proceedings of the 25th International Conference on World Wide Web (WWW)*, 2016.
- [81] José Camacho. Visualizing big data with compressed score plots: Approach and research challenges. *Chemometrics and Intelligent Laboratory Systems*, 2014.
- [82] Shlomo Zilberstein. Using anytime algorithms in intelligent systems. *AI magazine*, 1996.
- [83] John P. Cunningham and Zoubin Ghahramani. Linear dimensionality reduction: Survey, insights, and generalizations. *Journal of Machine Learning Research*, 2015.
- [84] Karl Pearson. LIII. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 1901.
- [85] Nikolaos Passalis and Anastasios Tefas. Dimensionality reduction using similarity-induced embeddings. *IEEE Trans. Neural Networks Learn. Syst.*, 2018.

- [86] Hujun Yin. Nonlinear dimensionality reduction and data visualization: A review. *Int. J. Autom. Comput.*, 2007.
- [87] Jiarui Ding, Anne Condon, and Sohrab P. Shah. Interpretable dimensionality reduction of single cell transcriptome data with deep generative models. *Nature Communications*, 2018.
- [88] Bernhard Schölkopf, Alexander J. Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Comput.*, 1998.
- [89] Elnaz Barshan, Ali Ghodsi, Zohreh Azimifar, and Mansoor Zolghadri Jahromi. Supervised principal component analysis: Visualization, classification and regression on subspaces and submanifolds. *Pattern Recognition*, 2011.
- [90] Mahdokht Masaeli, Glenn Fung, and Jennifer G. Dy. From transformation-based dimensionality reduction to feature selection. In *International Conference on Machine Learning (ICML)*, 2010.
- [91] Yale Chang, Junxiang Chen, Michael H. Cho, Peter J. Castaldi, Edwin K. Silverman, and Jennifer G. Dy. Clustering with domain-specific usefulness scores. In *Proceedings of the SIAM International Conference on Data Mining*, 2017.
- [92] Kelum Gajamannage, Randy Paffenroth, and Erik M. Bollt. A nonlinear dimensionality reduction framework using smooth geodesics. *Pattern Recognition*, 2019.
- [93] Shlomo Hoory, Nathan Linial, and Avi Wigderson. Expander graphs and their applications. *Bulletin of the American Mathematical Society*, 2006.
- [94] Arnold D Well and Jerome L Myers. *Research design & statistical analysis*. Psychology Press, 2003.
- [95] Laurens van der Maaten. Learning a parametric embedding by preserving local structure. In *International Conference on Artificial Intelligence and Statistics*, 2009.
- [96] F. Pedregosa et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 2011.
- [97] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017. URL <http://archive.ics.uci.edu/ml>.
- [98] David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen, and Klaus-Robert Müller. How to explain individual classification decisions. *J. Mach. Learn. Res.*, 2010.
- [99] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLOS ONE*, 2015. doi: 10.1371/journal.pone.0130140.

- [100] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, 2017. doi: 10.1109/ICCV.2017.74.
- [101] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin A. Riedmiller. Striving for simplicity: The all convolutional net. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings*, 2015.
- [102] Amirata Ghorbani, Abubakar Abid, and James Y. Zou. Interpretation of neural networks is fragile. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, 2019. doi: 10.1609/aaai.v33i01.33013681.
- [103] Juyeon Heo, Sunghwan Joo, and Taesup Moon. Fooling neural network interpretations via adversarial model manipulation. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d’Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, 2019.
- [104] Tom J. Viering, Ziqi Wang, Marco Loog, and Elmar Eisemann. How to manipulate cnns to make them lie: the gradcam case. *CoRR*, 2019.
- [105] Pieter-Jan Kindermans, Sara Hooker, Julius Adebayo, Maximilian Alber, Kristof T. Schütt, Sven Dähne, Dumitru Erhan, and Been Kim. The (un)reliability of saliency methods. In Wojciech Samek, Grégoire Montavon, Andrea Vedaldi, Lars Kai Hansen, and Klaus-Robert Müller, editors, *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, volume 11700 of *Lecture Notes in Computer Science*, pages 267–280. Springer, 2019. doi: 10.1007/978-3-030-28954-6\_14.
- [106] Adam Ivankay, Ivan Girardi, Chiara Marchiori, and Pascal Frossard. FAR: A general framework for attributional robustness. *CoRR*, 2020.
- [107] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Jonathan Uesato, and Pascal Frossard. Robustness via curvature regularization, and vice versa. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, 2019. doi: 10.1109/CVPR.2019.00929.
- [108] Chongli Qin, James Martens, Sven Gowal, Dilip Krishnan, Krishnamurthy Dvijotham, Alhussein Fawzi, Soham De, Robert Stanforth, and Pushmeet Kohli. Adversarial robustness through local linearization. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d’Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, 2019.

- [109] Chaowei Xiao, Jun-Yan Zhu, Bo Li, Warren He, Mingyan Liu, and Dawn Song. Spatially transformed adversarial examples. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018.
- [110] Cassidy Laidlaw and Soheil Feizi. Functional adversarial attacks. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d’Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, 2019.
- [111] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018.
- [112] Himabindu Lakkaraju, Nino Arsov, and Osbert Bastani. Robust and stable black box explanations. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, 2020.
- [113] Jiefeng Chen, Xi Wu, Vaibhav Rastogi, Yingyu Liang, and Somesh Jha. Robust attribution regularization. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d’Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, 2019.
- [114] Ann-Kathrin Dombrowski, Christopher J. Anders, Klaus-Robert Müller, and Pan Kessel. Towards Robust Explanations for Deep Neural Networks. *arXiv e-prints*, 2020.
- [115] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda B. Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise. *CoRR*, 2017.
- [116] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.
- [117] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: A simple and accurate method to fool deep neural networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, 2016. doi: 10.1109/CVPR.2016.282.
- [118] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In Yoshua Bengio and Yann LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

- [119] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 2015. doi: 10.1007/s11263-015-0816-y.
- [120] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian J. Goodfellow, Moritz Hardt, and Been Kim. Sanity checks for saliency maps. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, 2018.
- [121] Jianming Zhang, Zhe L. Lin, Jonathan Brandt, Xiaohui Shen, and Stan Sclaroff. Top-down neural attention by excitation backprop. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part IV*, 2016. doi: 10.1007/978-3-319-46493-0\\_33.
- [122] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft COCO: Common Objects in Context. *arXiv e-prints*, 2014.
- [123] Christoph Molnar. *Interpretable Machine Learning*. 2019.
- [124] Prasad Chalasani, Jiefeng Chen, Amrita Roy Chowdhury, Xi Wu, and Somesh Jha. Concise explanations of neural networks using adversarial training. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, 2020.
- [125] Niall P. Hurley and Scott T. Rickard. Comparing measures of sparsity. *IEEE Trans. Inf. Theory*, 2009. doi: 10.1109/TIT.2009.2027527.
- [126] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, 2018. doi: 10.1109/CVPR.2018.00068.
- [127] Logan Engstrom, Andrew Ilyas, Hadi Salman, Shibani Santurkar, and Dimitris Tsipras. Robustness (python library), 2019. URL <https://github.com/MadryLab/robustness>.
- [128] Alex Krizhevsky. Learning multiple layers of features from tiny images. *University of Toronto*, 2012.
- [129] David Alvarez-Melis and Tommi S. Jaakkola. On the robustness of interpretability methods. *CoRR*, 2018.
- [130] Christian Etmann, Sebastian Lunz, Peter Maass, and Carola Schönlieb. On the connection between adversarial robustness and saliency map interpretability. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, 2019.



- [131] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.
- [132] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 2012.
- [133] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 2017.
- [134] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, 2019. doi: 10.18653/v1/n19-1423.
- [135] Yury Gorishniy, Ivan Rubachev, Valentin Khurlov, and Artem Babenko. Revisiting deep learning models for tabular data. In Marc’Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, 2021.
- [136] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, 2017.
- [137] Ulrich Aivodji, Hiromi Arai, Olivier Fortineau, Sébastien Gambs, Satoshi Hara, and Alain Tapp. Fairwashing: the risk of rationalization. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, 2019.
- [138] Himabindu Lakkaraju and Osbert Bastani. ”how do I fool you?”: Manipulating user trust via misleading black box explanations. In Annette N. Markham, Julia Powles, Toby Walsh, and Anne L. Washington, editors, *AIES ’20: AAAI/ACM Conference on AI, Ethics, and Society, New York, NY, USA, February 7-8, 2020*, 2020. doi: 10.1145/3375627.3375833.
- [139] Dylan Slack, Sophie Hilgard, Emily Jia, Sameer Singh, and Himabindu Lakkaraju. Fooling LIME and SHAP: adversarial attacks on post hoc explanation methods. In Annette N. Markham, Julia Powles, Toby Walsh, and Anne L. Washington, editors, *AIES ’20: AAAI/ACM Conference on AI, Ethics, and Society, New York, NY, USA, February 7-8, 2020*, 2020. doi: 10.1145/3375627.3375830.
- [140] Kevin Eykholt, Ivan Evtimov, Earlene Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust physical-world attacks on deep learning visual classification. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, 2018. doi: 10.1109/CVPR.2018.00175.

- [141] Francesco Croce and Matthias Hein. Sparse and imperceivable adversarial attacks. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, 2019. doi: 10.1109/ICCV.2019.00482.
- [142] Apostolos Modas, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. Sparsefool: A few pixels make a big difference. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019*, 2019.
- [143] Xiaoyi Dong, Dongdong Chen, Jianmin Bao, Chuan Qin, Lu Yuan, Weiming Zhang, Nenghai Yu, and Dong Chen. Greedyfool: Distortion-aware sparse adversarial attack. In *Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.
- [144] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.
- [145] Ulrich Aïvodji, Hiromi Arai, Olivier Fortineau, Sébastien Gambs, Satoshi Hara, and Alain Tapp. Fairwashing: the risk of rationalization. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, 2019.
- [146] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017. URL <http://archive.ics.uci.edu/ml>.
- [147] Alexander Chernev, Ulf Böckenholt, and Joseph Goodman. Choice overload: A conceptual review and meta-analysis. *Journal of Consumer Psychology*, 2015.
- [148] Francesco Ricci, Lior Rokach, and Bracha Shapira. Recommender systems: introduction and challenges. *Recommender systems handbook*, 2015.
- [149] Pearl Pu, Li Chen, and Rong Hu. Evaluating recommender systems from the user’s perspective: survey of the state of the art. *User Modeling and User-Adapted Interaction*, 2012.
- [150] Dokyun Lee and Kartik Hosanagar. How do recommender systems affect sales diversity? a cross-category investigation via randomized field experiment. *Information Systems Research*, 2019.
- [151] Xitong Li, Joern Grahl, and Oliver Hinz. How do recommender systems lead to consumer purchases? a causal mediation analysis of a field experiment. *Information Systems Research*, 2022.
- [152] Daniel Fleder and Kartik Hosanagar. Blockbuster culture’s next rise or fall: The impact of recommender systems on sales diversity. *Management science*, 2009.
- [153] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP*, 2019.

- [154] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, 2020.
- [155] Yehuda Koren, Robert M. Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 2009.
- [156] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, 2009. AUAI Press.
- [157] Steffen Rendle. Factorization machines. In *ICDM 2010, The 10th IEEE International Conference on Data Mining*, 2010.
- [158] Xiangnan He and Tat-Seng Chua. Neural factorization machines for sparse predictive analytics. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2017.
- [159] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*, 2017.
- [160] Jiacheng Li, Yujie Wang, and Julian J. McAuley. Time interval aware self-attention for sequential recommendation. In *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining*, 2020.
- [161] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. Wide & deep learning for recommender systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, 2016.
- [162] Ruoxi Wang, Bin Fu, Gang Fu, and Mingliang Wang. Deep & cross network for ad click predictions. In *Proceedings of the ADKDD'17*, 2017.
- [163] Ziwei Zhu, Shahin Sefati, Parsa Saadatpanah, and James Caverlee. Recommendation for new users and new items via randomized training and mixture-of-experts transformation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2020.
- [164] Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. Gnnexplainer: Generating explanations for graph neural networks. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, 2019.

- [165] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 2009.
- [166] Francesco Fusco, Michalis Vlachos, Vasileios Vasileiadis, Kathrin Wardatzky, and Johannes Schneider. Reconet: An interpretable neural architecture for recommender systems. In *IJCAI*, 2019.
- [167] Hongyu Lu, Weizhi Ma, Yifan Wang, Min Zhang, Xiang Wang, Yiqun Liu, Tat-Seng Chua, and Shaoping Ma. User Perception of Recommendation Explanation: Are Your Explanations What Users Need? *ACM Transactions on Information Systems*, 2023.
- [168] Shijun Li, Wenqiang Lei, Qingyun Wu, Xiangnan He, Peng Jiang, and Tat-Seng Chua. Seamlessly unifying attributes and items: Conversational recommendation for cold-start users. *ACM Transactions on Information Systems (TOIS)*, 2021.
- [169] Deepak Kumar Panda and Sanjog Ray. Approaches and algorithms to mitigate cold start problems in recommender systems: a systematic literature review. *J. Intell. Inf. Syst.*, 2022. doi: 10.1007/s10844-022-00698-5.
- [170] Da-Cheng Nie, Zi-Ke Zhang, Qiang Dong, Chongjing Sun, and Yan Fu. Information filtering via biased random walk on coupled social network. *TheScientificWorldJournal*, 2014. doi: 10.1155/2014/829137.
- [171] Sajad Ahmadian, Mohsen Afsharchi, and Majid Meghdadi. An effective social recommendation method based on user reputation model and rating profile enhancement. *J. Inf. Sci.*, 2019. doi: 10.1177/0165551518808191.
- [172] Yutao Ma, Xiao Geng, and Jian Wang. A deep neural network with multiplex interactions for cold-start service recommendation. *IEEE Transactions on Engineering Management*, 2021. doi: 10.1109/TEM.2019.2961376.
- [173] Chieh-Yuan Tsai, Yi-Fan Chiu, and Yu-Jen Chen. A two-stage neural network-based cold start item recommender. *Applied Sciences*, 2021. doi: 10.3390/app11094243.
- [174] Jian Wei, Jianhua He, Kai Chen, Yi Zhou, and Zuoyin Tang. Collaborative filtering and deep learning based recommendation system for cold start items. *Expert Systems with Applications*, 2017. doi: <https://doi.org/10.1016/j.eswa.2016.09.040>.
- [175] Shameem Ahamed Puthiya Parambath and Sanjay Chawla. Simple and effective neural-free soft-cluster embeddings for item cold-start recommendations. *Data Min. Knowl. Discov.*, 2020. doi: 10.1007/s10618-020-00708-6.
- [176] Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. Graph neural networks in recommender systems: a survey. *ACM Computing Surveys*, 2022.
- [177] Chen Gao, Yu Zheng, Nian Li, Yinfeng Li, Yingrong Qin, Jinghua Piao, Yuhan Quan, Jianxin Chang, Depeng Jin, Xiangnan He, et al. A survey of graph neural networks for recommender systems: Challenges, methods, and directions. *ACM Transactions on Recommender Systems*, 2023.

- [178] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. Neural graph collaborative filtering. In *Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval*, 2019.
- [179] Ninghao Liu, Yong Ge, Li Li, Xia Hu, Rui Chen, and Soo-Hyun Choi. Explainable recommender systems via resolving learning representations. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 2020.
- [180] Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. Kgat: Knowledge graph attention network for recommendation. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2019.
- [181] Hongwei Wang, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. Ripplenet: Propagating user preferences on the knowledge graph for recommender systems. In *Proceedings of the 27th ACM international conference on information and knowledge management*, 2018.
- [182] Jianxin Chang, Chen Gao, Yu Zheng, Yiqun Hui, Yanan Niu, Yang Song, Depeng Jin, and Yong Li. Sequential recommendation with graph neural networks. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2021.
- [183] Qiaoyu Tan, Jianwei Zhang, Jiangchao Yao, Ninghao Liu, Jingren Zhou, Hongxia Yang, and Xia Hu. Sparse-interest network for sequential recommendation. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, 2021.
- [184] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th international conference on World wide web*, 2010.
- [185] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939*, 2015.
- [186] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. Neural attentive session-based recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, 2017.
- [187] Jiayi Tang and Ke Wang. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the eleventh ACM international conference on web search and data mining*, 2018.
- [188] Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *IEEE International Conference on Data Mining (ICDM)*, 2018.
- [189] Maria-Iuliana Dascalu, Constanta-Nicoleta Bodea, Monica Nastasia Mihailescu, Elena Alice Tanase, and Patricia Ordoñez de Pablos. Educational recommender systems and their application in lifelong learning. *Behaviour & information technology*, 2016.

- [190] Robert Bodily and Katrien Verbert. Review of research on student-facing learning analytics dashboards and educational recommender systems. *IEEE Transactions on Learning Technologies*, 2017.
- [191] George Karypis. Improving higher education: Learning analytics & recommender systems research. In Paolo Cremonesi, Francesco Ricci, Shlomo Berkovsky, and Alexander Tuzhilin, editors, *Proc. of RecSys*, 2017. doi: 10.1145/3109859.3109870.
- [192] Hao Zhang, Tao Huang, Zhihan Lv, Sanya Liu, and Heng Yang. Moocrc: A highly accurate resource recommendation model for use in mooc environments. *Mobile Networks and Applications*, 2019.
- [193] Amir Hossein Nabizadeh, Daniel Goncalves, Sandra Gama, Joaquim Jorge, and Hamed N Rafsanjani. Adaptive learning path recommender approach using auxiliary learning objects. *Computers & Education*, 2020.
- [194] Amir Hossein Nabizadeh, José Paulo Leal, Hamed N Rafsanjani, and Rajiv Ratn Shah. Learning path personalization and recommendation methods: A survey of the state-of-the-art. *Expert Systems with Applications*, 2020.
- [195] Weijie Jiang and Zachary A. Pardos. Time slice imputation for personalized goal-based recommendation in higher education. In *Proc. of RecSys*, 2019. doi: 10.1145/3298689.3347030.
- [196] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019.
- [197] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 2013.
- [198] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM international conference on information and knowledge management*, 2019.
- [199] Gustavo Penha and Claudia Hauff. What does bert know about books, movies and music? probing bert for conversational recommendation. In *Fourteenth ACM Conference on Recommender Systems*, 2020.
- [200] Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. Language-agnostic BERT sentence embedding. *arXiv preprint arXiv:2007.01852*, 2020.
- [201] Zhiqiang Zheng. Introduction to big data analytics and the special issue on big data methods and applications. *Journal of Management Analytics*, 2015.

- [202] Ahmed Abbasi, Suprateek Sarker, and Roger HL Chiang. Big data research in information systems: Toward an inclusive research agenda. *Journal of the association for information systems*, 2016.
- [203] Wu Youyou, Michal Kosinski, and David Stillwell. Computer-based personality judgments are more accurate than those made by humans. *Proceedings of the National Academy of Sciences*, 2015.
- [204] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language Models are Few-Shot Learners. *arXiv e-prints*, 2020. doi: 10.48550/arXiv.2005.14165.
- [205] Dong Huk Park, Lisa Anne Hendricks, Zeynep Akata, Anna Rohrbach, Bernt Schiele, Trevor Darrell, and Marcus Rohrbach. Multimodal explanations: Justifying decisions and pointing to the evidence. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, 2018. doi: 10.1109/CVPR.2018.00915.
- [206] Ramakrishna Vedantam, Samy Bengio, Kevin Murphy, Devi Parikh, and Gal Chechik. Context-aware captions from context-agnostic supervision. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, 2017. doi: 10.1109/CVPR.2017.120.