

Enhancing Explainability and Scrutability of Recommender Systems

A dissertation submitted towards the degree
Doctor of Engineering (Dr.-Ing.)
of the Faculty of Mathematics and Computer Science
of Saarland University

by

Azin Ghazimatin

Saarbrücken, 2021

Day of Colloquium:	9 December 2021
Dean of the Faculty:	Prof. Dr. Thomas Schuster
Chair of the Committee:	Prof. Dr. Dietrich Klakow
Reviewers:	Prof. Dr. Gerhard Weikum Dr. Rishiraj Saha Roy Prof. Dr. Mounia Lalmas Prof. Dr. Sihem Amer-Yahia
Academic Assistant:	Dr. Panagiotis Mandros

Acknowledgments

I am profoundly grateful to Gerhard Weikum, my advisor, for his invaluable mentorship and tremendous support. Thank you, Gerhard, I have greatly benefited from your wealth of knowledge. This thesis would not have been possible without your excellent guidance.

I am sincerely grateful to Rishiraj Saha Roy, my co-advisor, for his exceptional mentorship and continuous support. Thank you, Rishi for all the invaluable lessons you shared with me throughout my PhD that helped me expand my technical knowledge.

I offer my warmest thanks to Prof. Lalmas and Prof. Amer-Yahia for reviewing my thesis, and for their very insightful feedback. I am also deeply grateful to Prof. Klakow and Dr. Mandros for all the time and effort they spent as the thesis committee members.

A very special thanks to Michelle Carnell for supporting PhD students in every way possible. Thank you Michelle for all your kindness and for having faith in me over these years.

Many thanks to Oana Balalau for being an amazing mentor and colleague. Thank you, Oana for all your support and insightful comments and suggestions.

I am extremely grateful to Petra Schaaf for patiently assisting me in many different ways. Thank you, Petra for all your support.

Heartfelt thanks to my friends, Norine, Noemi, Sebastian, and Clara who have always been a major source of support since the very first day I came to Saarbrücken. Thank you my friends for the countless happy moments you brought for me.

I deeply thank my parents, my sister, Elham, and my brother, Ryan, for their unconditional love and support in all stages of my life.

Last but not least, thank you, Babak, for always being there for me, for bearing with me whenever I was stressed out, and for being the best companion in face of difficulties throughout these years.

Abstract

Our increasing reliance on complex algorithms for recommendations calls for models and methods for explainable, scrutable, and trustworthy AI. While explainability is required for understanding the relationships between model inputs and outputs, a scrutable system allows us to modify its behavior as desired. These properties help bridge the gap between our expectations and the algorithm’s behavior and accordingly boost our trust in AI.

Aiming to cope with information overload, recommender systems play a crucial role in filtering content (such as products, news, songs, and movies) and shaping a personalized experience for their users. Consequently, there has been a growing demand from the information consumers to receive proper explanations for their personalized recommendations. These explanations aim at helping users understand why certain items are recommended to them and how their previous inputs to the system relate to the generation of such recommendations. Besides, in the event of receiving undesirable content, explanations could possibly contain valuable information as to how the system’s behavior can be modified accordingly.

In this thesis, we present our contributions towards explainability and scrutability of recommender systems:

- We introduce a *user-centric* framework, FAIRY, for discovering and ranking *post-hoc explanations for the social feeds* generated by *black-box* platforms. These explanations reveal relationships between users’ profiles and their feed items and are extracted from the local interaction graphs of users. FAIRY employs a learning-to-rank (LTR) method to score candidate explanations based on their relevance and surprisal.
- We propose a method, PRINCE, to facilitate *provider-side explainability* in graph-based recommender systems that use personalized PageRank at their core. PRINCE explanations are comprehensible for users, because they present subsets of the user’s prior actions responsible for the received recommendations. PRINCE operates in a *counterfactual* setup and builds on a polynomial-time algorithm for finding the smallest counterfactual explanations.
- We propose a human-in-the-loop framework, ELIXIR, for *enhancing scrutability* and subsequently the recommendation models by leveraging *user feedback on explanations*. ELIXIR enables recommender systems to collect user feedback on pairs of recommendations and explanations. The feedback is incorporated into the model by imposing a soft constraint for learning user-specific item representations.

We evaluate all proposed models and methods with real user studies and demonstrate their benefits at achieving explainability and scrutability in recommender systems.

Kurzfassung

Unsere zunehmende Abhängigkeit von komplexen Algorithmen für maschinelle Empfehlungen erfordert Modelle und Methoden für erklärbare, nachvollziehbare und vertrauenswürdige KI. Zum Verstehen der Beziehungen zwischen Modellein- und ausgaben muss KI erklärbar sein. Möchten wir das Verhalten des Systems hingegen nach unseren Vorstellungen ändern, muss dessen Entscheidungsprozess nachvollziehbar sein. Erklärbarkeit und Nachvollziehbarkeit von KI helfen uns dabei, die Lücke zwischen dem von uns erwarteten und dem tatsächlichen Verhalten der Algorithmen zu schließen und unser Vertrauen in KI-Systeme entsprechend zu stärken.

Um ein Übermaß an Informationen zu verhindern, spielen Empfehlungsdienste eine entscheidende Rolle um Inhalte (z.B. Produkten, Nachrichten, Musik und Filmen) zu filtern und deren Benutzern eine personalisierte Erfahrung zu bieten. Infolgedessen erheben immer mehr Informationskonsumenten Anspruch auf angemessene Erklärungen für deren personalisierte Empfehlungen. Diese Erklärungen sollen den Benutzern helfen zu verstehen, warum ihnen bestimmte Dinge empfohlen wurden und wie sich ihre früheren Eingaben in das System auf die Generierung solcher Empfehlungen auswirken. Außerdem können Erklärungen für den Fall, dass unerwünschte Inhalte empfohlen werden, wertvolle Informationen darüber enthalten, wie das Verhalten des Systems entsprechend geändert werden kann.

In dieser Dissertation stellen wir unsere Beiträge zu Erklärbarkeit und Nachvollziehbarkeit von Empfehlungsdiensten vor.

- Mit FAIRY stellen wir ein benutzerzentriertes Framework vor, mit dem post-hoc Erklärungen für die von Black-Box-Plattformen generierten sozialen Feeds entdeckt und bewertet werden können. Diese Erklärungen zeigen Beziehungen zwischen Benutzerprofilen und deren Feeds auf und werden aus den lokalen Interaktionsgraphen der Benutzer extrahiert. FAIRY verwendet eine LTR-Methode (Learning-to-Rank), um die Erklärungen anhand ihrer Relevanz und ihres Grads unerwarteter Empfehlungen zu bewerten.
- Mit der PRINCE-Methode erleichtern wir das anbieterseitige Generieren von Erklärungen für PageRank-basierte Empfehlungsdienste. PRINCE-Erklärungen sind für Benutzer verständlich, da sie Teilmengen früherer Nutzerinteraktionen darstellen, die für die erhaltenen Empfehlungen verantwortlich sind. PRINCE-Erklärungen sind somit kausaler Natur und werden von einem Algorithmus mit polynomieller Laufzeit erzeugt, um präzise Erklärungen zu finden.
- Wir präsentieren ein Human-in-the-Loop-Framework, ELIXIR, um die Nachvollziehbarkeit der Empfehlungsmodelle und die Qualität der Empfehlungen zu verbessern. Mit ELIXIR können Empfehlungsdienste Benutzerfeedback zu Empfehlungen und Erklärungen sammeln. Das Feedback wird in das Modell einbezogen, indem benutzerspezifischer Einbettungen von Objekten gelernt werden.

Wir evaluieren alle Modelle und Methoden in Benutzerstudien und demonstrieren ihren

Nutzen hinsichtlich Erklärbarkeit und Nachvollziehbarkeit von Empfehlungsdiensten.

CONTENTS

1	Introduction	1
1.1	Motivation	1
1.1.1	Explainable recommendations	2
1.1.2	Scrutable recommendations	3
1.2	Challenges	4
1.3	Prior work and its limitations	5
1.4	Contributions	6
1.5	Publications	7
1.6	Organization	8
2	Background	9
2.1	Overview of recommender models	9
2.1.1	Models and methods	10
2.1.2	Evaluation criteria	11
2.2	Explaining recommendations	12
2.2.1	Definitions	12
2.2.2	Purposes of explanations	13
2.2.3	Technical properties of explanations	15
2.2.4	Explanation styles	16
2.2.5	Post-hoc explanations	18
2.2.6	Model-aware explanations	19
2.3	Scrutable recommenders	22
3	Post-hoc Explanations for Black-Box Recommendations	25
3.1	Introduction	26
3.2	Discovering explanations	29
3.3	Ranking explanations	31
3.3.1	Learning to rank explanations	32
3.3.2	Learning to rank features	32
3.4	User studies	34
3.5	Evaluation setup	36
3.6	Results and insights	37
3.6.1	Key findings	37
3.6.2	Analysis and discussion	39
3.7	Related work	41
3.8	Conclusion	42

4	Counterfactual Explanations for Recommendations	45
4.1	Introduction	46
4.2	Computational model	48
4.3	The PRINCE algorithm	50
4.4	Correctness proof	51
4.5	Graph experiments	55
4.5.1	Setup	55
4.5.2	Results and insights	57
4.6	User study	59
4.7	Related work	61
4.8	Conclusion	62
5	Using Explanations to Improve Recommender Models	65
5.1	Introduction	66
5.2	The ELIXIR framework	68
5.2.1	Feedback collection	69
5.2.2	Feedback densification	70
5.2.3	Feedback incorporation	71
5.2.4	ELIXIR in recommenders based on PPR	72
5.3	User study for data collection	73
5.3.1	Statement on ethics	74
5.3.2	Setup	74
5.4	Evaluation setup	77
5.4.1	Configurations	77
5.4.2	Metrics	78
5.4.3	Initialization	78
5.5	Results and insights	80
5.5.1	Key findings	80
5.5.2	Analysis	80
5.6	Related work	89
5.7	Conclusion	90
6	Conclusions and Outlook	91
	List of Figures	93
	List of Tables	95
A	FAIRY: User Study Guideline	97
	Bibliography	103

1

INTRODUCTION

Contents

1.1	Motivation	1
1.1.1	Explainable recommendations	2
1.1.2	Scrutable recommendations	3
1.2	Challenges	4
1.3	Prior work and its limitations	5
1.4	Contributions	6
1.5	Publications	7
1.6	Organization	8

1.1 Motivation

Our increasing reliance on complex algorithms for recommendations calls for models and methods for explainable and scrutable AI. While explainability helps us understand the cause of a decision made by an algorithm [Miller, 2019], a scrutable system enables users to correct system’s assumptions when needed [Tintarev and Masthoff, 2007]. These properties bring about trust by bridging the gap between humans and AI.

Aiming to think on our behalf and predict our information need, recommender systems are perceived as advice-givers that can improve our acceptance through explanations [Ricci et al., 2015]. With the emergence of more complex models [Koren et al., 2009] outperforming the simpler and more explainable ones [Sarwar et al., 2001], *Explainable AI* has progressively received more attention from the Recommender Systems (RecSys) community [Zhang and Chen, 2020]. Lack of transparency in recommender systems can have a direct impact on user acceptance, as based on the content personalized for users, they may feel that the system is labeling them inappropriately¹ or misusing their private information². To highlight the gravity of this matter, recently, laws have been passed to establish users’ right to explanations [Goodman and Flaxman, 2017].

¹<https://www.wsj.com/articles/SB1038261936872356908>

²<https://www.wired.co.uk/article/tiktok-filter-bubbles>

Despite the close tie between explainability and scrutability [Balog and Radlinski, 2020], they do not necessarily entail each other. In other words, knowing why the algorithm makes particular choices may not be sufficient for realizing how to modify it. For instance, imagine a user of an online movie streaming service who is frequently recommended with action movies. The system explains its choices by drawing connections between the recommended movies and the action movies the user previously watched on the platform. Now, consider the situation where the user wants to stop receiving such movies as they do not entirely match her interest. Here, the provided explanations do not act as a precise guide as to how she can effectively exert control over her recommendations. Therefore, scrutability in recommender systems requires separate consideration and handling.

Approaches to explainable recommendations differ based on *who consumes the explanations*. For instance, system developers can benefit from detailed statistics to perform error analysis [Wu et al., 2019b]. Such explanations, however, are often beyond users' comprehension and thus hardly useful to them. Furthermore, evidence suggests that too much transparency may hurt users' trust in the system [Ananny and Crawford, 2018]. End users are often interested in receiving *local explanations* which reveal how their *own* inputs affect the system's decision for them [Doshi-Velez et al., 2017]. In this thesis, we assume that explanations are generated *for* the end users, and they are the ones who seek to *scrutinize* the model.

This thesis develops models and methods for enhancing *explainability* and *scrutability* of *recommenders systems* for *end users*.

1.1.1 Explainable recommendations

Recommender systems aim at delivering personalized content such as products, movies, books, and songs to their users. The chosen content is often visualized in a ranked list, where the order reflects the relevance of the items to the user. To compute these relevance scores, recommender systems usually train models based on various inputs collected from their users. User inputs can be explicit (e.g., rating or liking an item) or implicit (e.g., watching a movie or listening to a song). The abundance of implicit signals has facilitated data collection by service providers.

Providing the systems with an enormous amount of data over time, users might not be able to remember all the details of their interactions, and hence experience difficulty in understanding why they receive certain items as their recommendations. This problem particularly worsens when users do not even have access to the complete history of their interaction with the system, a phenomenon referred to as *inverse privacy* [Gurevich and Wing, 2016]. Therefore, it is imperative for the recommender systems to be explainable, i.e., to enable users to understand the relationships between their own input to the system and the recommendations they receive.

To illustrate how a recommendation can be explained, imagine a user who is a member of a social cataloging website like Goodreads³ and receives a book recommendation, titled *Recovery*:

³<https://www.goodreads.com/>

Freedom from Our Addictions. Examples 1.1.1 and 1.1.2 present two possible ways of explaining this recommendation to the user by outlining connections between the given recommendation and her past actions on the platform:

Example 1.1.1. You $\xrightarrow{\text{liked}}$ *Becoming* $\xrightarrow{\text{has genre}}$ *Autobiography* $\xrightarrow{\text{belongs to}}$ *Recovery: Freedom from Our Addictions*

Example 1.1.2. You $\xrightarrow{\text{follow}}$ *Alice* $\xrightarrow{\text{follows}}$ *Addiction Topic* $\xrightarrow{\text{belongs to}}$ *Recovery: Freedom from Our Addictions*

As depicted in Example 1.1.2, showing the full connections between a user and her recommendation may reveal private and possibly sensitive information about other users (*Alice* in this example). This concern could be addressed by showing only the user’s *own* actions in explanations. For instance, to prevent disclosing *Alice*’s interest in the topic of *Addiction*, we can replace the explanation in Example 1.1.2 by “*because you follow Alice*”.

Apart from describing *why* a certain item is relevant to a user, recommender systems are also expected to be able to explain the rankings, i.e., to reason *why* a certain item is *more relevant* than the others. For instance, the following statement explains the cause of receiving the book *Recovery: Freedom from Our Addictions* as the *top-ranked* recommendation:

Example 1.1.3. You are recommended with the book *Recovery: Freedom from Our Addictions* because you *liked* the books *Becoming* and *Dreams from My Father*. *If you did not like these two books, your top-ranked recommendation would be the book **Food and Nutrition**.*

Example 1.1.3 shows that *liking* the books *Becoming* and *Dreams from My Father* is the key reason that the book *Recovery: Freedom from Our Addictions* is more relevant to the user than the book *Food and Nutrition*. The blue text in this example demonstrates the causality between user’s previous action and system’s outcome. Such explanations are referred to as counterfactual; they pinpoint those user actions whose absence would result in a different recommendation for her. Identifying the true reasons behind the recommendations, these explanations pave the way towards scrutability, i.e., they help shed light on how users can control what they see as their recommendations.

1.1.2 Scrutable recommendations

A *scrutable* recommender system allows its users to tell the system when it is wrong and enables users to steer their recommendations accordingly [Tintarev and Masthoff, 2007]. This feature is particularly useful when users experience drifts in their interests or when the system cannot correctly infer their preferences. Evidence suggests that scrutability can improve user’s engagement level and their satisfaction [Hijikata et al., 2012, Knijnenburg et al., 2012a, Parra and Brusilovsky, 2015].

Critique-enabled recommenders have already taken the first step towards scrutability. These systems employ a feedback mechanism called *critiquing* that enable users to express their dissatisfaction with some characteristics of the recommended item [Chen and Pu, 2012]. For instance, imagine a student who relies on an online service like Yelp⁴ to find a nice place to have dinner. The recommended restaurants, however, are not suitable for her as they are mostly expensive and far from her place. In this scenario, she will benefit from system-suggested critiques such as *show me a cheaper or closer restaurant* that enables her to explore other options that suit her interest better. These initial attempts open up new research opportunities for the development of scrutable systems that are capable of learning fine-grained user preferences.

1.2 Challenges

On the path towards explainable and scrutable recommendations, we encounter numerous challenges. In this thesis, we particularly try to overcome the following challenges:

- **Explaining black-box recommendations:** As the generator of recommendations, service providers are in the best position to explain recommendations. Nevertheless, in many online services, recommended content still lacks (satisfying) explanations. A prominent example is social platforms that provide their users with personalized streams of content, also known as *feeds*. For instance, in Quora⁵, a social Q&A platform, user feed constitutes a number of questions and answers posted by other users. These feed items are often only tagged with *Recommended for you* or *Topic you might like*. Such statements, however, hardly yield any insight to the user as to *why* some content is selected for her or *why* she is supposed to like them.

In the absence of system-generated explanations, it is impossible to determine and evaluate the *true* reasons behind the social feeds generated by black-box models. In this situation, however, users would still benefit from *post-hoc explanations* for their feed items. These are merely plausible justifications that might be decoupled from the actual model, and hence are not faithful to the model. Nonetheless, they can still help users realize how their feeds *relate* to their profiles. Considering the abundance and diversity of actions in social platforms (e.g., *following friends or topics, liking content, sharing posts*), it would be a non-trivial task for the user to find these justifications on her own. This demands development of services and tools that provide users with such post-hoc explanations for their social feeds.

- **Counterfactual explanations:** The major criticism on post-hoc rationalizations is that they do not guarantee faithfulness, i.e., they do not necessarily yield an honest account of the underlying recommendation mechanism. Counterfactual explanations, on the other hand, aim at identifying the smallest subset of inputs that have *caused* a particular system outcome, and hence are faithful to the underlying model [Molnar, 2020]. In other words,

⁴<https://www.yelp.com>

⁵<https://www.quora.com/>

such explanations identify a subset of user inputs whose absence would have resulted in a different output. Example 1.1.3 illustrates a counterfactual explanation in the context of recommender systems. Generating such explanations for recommendations can be computationally expensive, as the search space grows exponentially with the user input size. This calls for the development of efficient methods for generating minimal counterfactual explanations for recommendations.

- **Scrutable recommender models:** Counterfactual explanations contain valuable information that can point the user towards *scrutability*, i.e., a course of action to avoid receiving certain recommendations. For instance, based on the explanation in Example 1.1.3, it is natural for the user to infer that if she removed her ratings for the books *Becoming* and *Dreams from My Father*, she would stop receiving books like *Recovery: Freedom from Our Addictions*. Therefore, a trivial solution for controlling recommendations is through undoing or modifying the already performed actions. The manual removal of previous actions, however, can become tedious for the user over time and more importantly it may have an adverse effect on the quality of future recommendations. For instance, by removing the ratings in the previous example, user would stop receiving *all* the interesting recommendations that could come about through their similarity with the books *Becoming* and *Dreams from My Father*. This indicates the need to develop models and methods for enabling users to give lightweight feedback on their recommendations and their associated explanations, and subsequently incorporating the collected user feedback into the model.

1.3 Prior work and its limitations

Despite the growing body of literature for justifying recommendations, very little attention has been paid to explaining social feeds in particular. The initial attempts mostly aimed at raising user awareness of curation algorithms behind the social feeds and analysing the impact of personalization on user's behavior [Eslami et al., 2015, Cotter et al., 2017, Rader et al., 2018]. The existing works for generating post-hoc explanations mainly suffer from two limitations that make them unsuited for explaining social feeds generated by black-box models: (i) they assume access to information such as the complete history of all (or a large group of) users [Peake and Wang, 2018], textual side information about the items [Wang et al., 2018c], latent features of users and items [Yang et al., 2018] or gradient and Hessian matrix of the model [Cheng et al., 2019], which are hardly made available by social platforms, and (ii) they often neglect user's opinion for ranking the explanations, and instead rely on heuristic metrics to score them [Yang et al., 2018].

Decoupled from the underpinning model, post-hoc justifications raise the risk of generating misleading explanations [Lipton, 2018]. To address this shortcoming, several works have attempted to introduce explainable recommendation models. These works bring explainability to the design level and leverage user-generated reviews [Zhang et al., 2014b, Chen et al., 2016,

Wang et al., 2018b, Tao et al., 2019] or structured knowledge [Catherine et al., 2017a, Ai et al., 2018, Huang et al., 2018, Wang et al., 2019a, Xian et al., 2019] to enhance the recommendation performance as well as the explainability of their models. The proposed approaches, however, are not directly applicable to the *existing* opaque recommenders. To address this need, several works have been introduced to generate faithful explanations via techniques such as surrogate models [Nóbrega and Marinho, 2019], subgroup discovery [Lonjarret et al., 2020] or association rule mining [Peake and Wang, 2018]. Most of these works, however, do not validate the *causality* of their explanations. This demands development of methods that guarantee the causality between the explanations and the ranking of the recommendations.

Revealing the true reasons behind the recommendations, causal explanations provide the grounds for actionability. Recently, there has been growing attention on critique-enabled recommender systems whereby users are enabled to critique the explanations of the recommended items [Wu et al., 2019a, Luo et al., 2020a]. For instance, consider a user of a critique-enabled music service who is recommended with the song *In The Zone*. The explanation for this item outlines the features deemed to be relevant to her, such as *Pop*, *Dance*, and *R&B*. Assume that she does not like this recommendation because it is a *Pop* song. In this scenario, the system allows her to *critique* the *Pop* feature and accordingly adjust her future recommendations. In the prior work, critiquing is mostly limited to coarse-grained item features that need to be explicit and extracted apriori. This highlights the need for the development of methods for collection and incorporation of user feedback on explanations that are both lightweight and suitable for learning more fine-grained preferences.

1.4 Contributions

We tackle the challenges outlined in Section 1.2 and address the limitations of the prior work described in the previous section by making the following contributions:

- **Post-hoc explanations for black-box recommendations.** We develop a framework, FAIRY, that generates post-hoc justifications for recommendations generated by black-box models. We showcase application of FAIRY for explaining social feeds, in particular. FAIRY helps users understand the relationships between their actions on the platform and their feed items. For this, we first model the user’s local neighborhood on the platform as an interaction graph. This graph is constructed solely from the information available to the user. In a user’s interaction graph, the set of simple paths connecting the user to her feed item are treated as pertinent explanations. Next, FAIRY scores the discovered explanations with learning-to-rank models built upon users’ judgements on relevance and surprisal of the explanation paths. Longitudinal user studies on two social platforms, *Quora* and *Last.fm*⁶, demonstrate the practical viability and user benefits of this framework in different domains.

⁶<https://www.last.fm/>

The results of this work have been published as a full paper in WSDM 2019 [Ghazimatin et al., 2019].

- **Counterfactual explanations for recommendations.** To address the limitations of the post-hoc justifications, we propose a mechanism, PRINCE, for generating causal and tangible explanations in a class of recommenders which use personalized PageRank at their core. Given a ranked list of recommendations, PRINCE generates a counterfactual explanation with the smallest size for the top-ranked item. In other words, PRINCE explains the most relevant recommendation to the user by identifying the minimum number of her actions whose removal displaces the top-ranked item. PRINCE uses a polynomial-time algorithm to find the minimal counterfactual explanations from an exponential search space, and hence it is efficient. Experiments on two real-world datasets show that PRINCE provides more compact explanations than intuitive baselines and insights from a crowdsourced user-study demonstrate the viability of such action-based explanations. The results of this work have appeared as a full paper in WSDM 2020 [Ghazimatin et al., 2020].
- **Using explanations to improve recommender models.** We develop a human-in-the-loop framework, ELIXIR, that leverages user feedback on explanations to enhance scrutability and subsequently the quality of recommendations for the user. ELIXIR enables recommenders to obtain user feedback on pairs of recommendation and explanation items, where users are asked to give a binary rating on the shared aspects of the items in a pair. To incorporate the collected feedback, ELIXIR proposes a method to learn user-specific latent preference vectors used for updating item-item similarities. The underlying intuition is to increase (decrease) the distance of disliked (liked) items and the like to the user’s profile, such that the quality of future recommendations is improved. Our framework is instantiated using generalized graph recommendation based on personalized PageRank. Insightful experiments with a real user study show significant improvements for movie and book recommendations over item-level feedback. The results of this work have been published as a full paper in The Web Conference 2021 [Ghazimatin et al., 2021].

1.5 Publications

The results of this thesis have appeared in the following conference articles whose lead author is the author of this thesis:

1. Ghazimatin, A., Saha Roy, R., and Weikum, G. (2019). **FAIRY: A framework for understanding relationships between users’ actions and their social feeds.** In *WSDM ’19: The Twelfth ACM International Conference on Web Search and Data Mining, Melbourne, VIC, Australia, February 11-15, 2019*, pages 240–248.
2. Ghazimatin, A., Balalau, O., Saha Roy, R., and Weikum, G. (2020). **PRINCE: provider-side interpretability with counterfactual explanations in recommender systems.** In

WSDM '20: *The Thirteenth ACM International Conference on Web Search and Data Mining, Houston, TX, USA, February 3-7, 2020*, pages 196–204.

3. Ghazimatin, A., Pramanik, S., Roy, R. S., and Weikum, G. (2021). **ELIXIR: learning from user feedback on explanations to improve recommender models**. In *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, pages 3850–3860

In addition, the summary of this thesis has been presented at the Doctoral Consortium at SIGIR 2020 [[Ghazimatin, 2020](#)].

1.6 Organization

The rest of this dissertation proceeds as follows. Chapter 2 discusses the necessary background on explainability and scrutability of recommender systems. Chapter 3 presents FAIRY, our contributions towards explaining social feeds generated by black-box models. Chapter 4 describes PRINCE, a method that enables a class of recommenders to generate counterfactual explanations for their recommendations. Chapter 5 presents ELIXIR, a framework that leverages user feedback on explanations to improve scrutability. Lastly, Chapter 6 concludes the dissertation and presents future research directions.

2

BACKGROUND

Contents

2.1 Overview of recommender models	9
2.1.1 Models and methods	10
2.1.2 Evaluation criteria	11
2.2 Explaining recommendations	12
2.2.1 Definitions	12
2.2.2 Purposes of explanations	13
2.2.3 Technical properties of explanations	15
2.2.4 Explanation styles	16
2.2.5 Post-hoc explanations	18
2.2.6 Model-aware explanations	19
2.3 Scrutable recommenders	22

This chapter presents the necessary background on explainable and scrutable recommendations. In Section 2.1, we briefly discuss the popular models, methods, and evaluation criteria used in recommender systems. Section 2.2, provides an overview of explainable recommendations by describing the aims of explanations, relevant definitions, and classification of methods for explaining recommendations. Section 2.3 covers notions and methods used in scrutable recommenders.

2.1 Overview of recommender models

Recommender systems are crucial to overcoming information overload in the online world. To narrow down user’s choices, recommender systems apply techniques to rank the available options based on user’s preferences and constraints. In this section, we briefly discuss the popular models, methods, and evaluation criteria used in recommender systems.

2.1.1 Models and methods

Content-based recommenders: Content-based recommenders try to find interesting items for a user by matching up the attributes of her profile with the content of the items. Common indicators of content are item descriptions and user-generated reviews for items and users, respectively. In early approaches, content was often represented using simple techniques such as TF-IDF [Ahn et al., 2007]. Modern approaches, however, mostly rely on more advanced techniques such as dimension reduction (e.g., Latent Semantic Analysis) [Musto et al., 2016b], graphical models (e.g., Latent Dirichlet Analysis) [Li et al., 2011], and deep learning [Musto et al., 2018] to encode the content. While content-based recommenders offer transparency owing to their interpretable design, they often fall short on generating diverse recommendations, as they rely on content similarity of items to the ones already liked by the user.

Collaborative filtering (CF): The key idea of collaborative filtering is to automate word-of-mouth recommendations by leveraging the ratings of *all* users. The proposed approaches to CF can be grouped into two classes of *neighborhood-based* and *model-based* methods [Ricci et al., 2015].

In neighborhood-based CF, user-item ratings are directly used to predict unknown ratings. This can be done in two ways: (i) user-based CF, or (ii) item-based CF. In user-based CF, predictions for a target user rely on the ratings of her neighbors (like-minded users) [Miller et al., 2003]. Neighbor users are those who share at least one common rating with the target user. Item-based CF, on the other hand, recommends a user with items similar to those already rated by her [Sarwar et al., 2001]. Here, two items are considered similar, if they have received similar ratings from the crowd.

In contrast to neighborhood-based CF, model-based CF uses ratings to learn a predictive model. The common techniques used in model-based CF include matrix factorization [Koren, 2008, Koren et al., 2009], Bayesian clustering [Breese et al., 1998], support vector machine [Gracar et al., 2006] and neural networks [He et al., 2017].

Graph-based recommenders: Neighborhood-based CF relies on the existence of direct neighbors. As a result, these approaches suffer from limited coverage and sensitivity to sparse data [Ricci et al., 2015]. To address these limitations, graph-based recommenders were introduced [Aggarwal et al., 1999]. In these models, the data is represented in a graph whose nodes are users and items, and edges encode interactions and similarities between the nodes. Such graphs exemplify Heterogeneous Information Networks (HINs) [Sun and Han, 2012]. HINs are graphs that are able to model multiple types of nodes (e.g., users and items) and edges (e.g., user-item interactions and item-item similarities).

To compute the similarity between users and items, these models often use path-based similarity measures such as random-walk similarity. Therefore, nodes that are not direct neighbors can still influence each other. Pixie [Eksombatchai et al., 2018] and RecWalk [Nikolakopoulos and

[Karypis, 2019] are two recent graph-based recommender models that apply biased random walk and personalized PageRank, respectively, to compute recommendation scores for a target user.

Knowledge-based recommenders: Leveraging side information about users and items, knowledge-based recommenders unify the interaction-level and content-level similarities [Guo et al., 2020]. Like graph-based methods, these recommenders represent data in a HIN. Methods for computing the affinity scores between users and items in a HIN are either embedding-based or path-based.

Embedding-based methods often apply multi-task learning to jointly train the recommendation task and learn the low-rank representations of graph entities and relations [Zhang et al., 2016a, Xin et al., 2019]. Some of the techniques for encoding the graph components into low-dimensional vector spaces include TransE [Bordes et al., 2013], TransH [Wang et al., 2014] and TransR [Lin et al., 2015].

In path-based methods, the connectivity patterns of the entities are leveraged to generate recommendations. To capture different similarity semantics, these methods often rely on a set of pre-defined meta-paths (path patterns) [Sun et al., 2011, Yu et al., 2013, Yu et al., 2014]. Recent models such as RippleNet [Wang et al., 2018a] and Graph Neural Network (GNN) [Fan et al., 2019] attempt to unify the embedding-based and path-based methods using preference propagation. More details about knowledge-based recommenders can be found in [Guo et al., 2020].

2.1.2 Evaluation criteria

There are several criteria for evaluating recommender systems. The primary criterion is the accuracy of the model with respect to user preferences which can be measured in multiple ways. For instance, predicted ratings are typically evaluated using (Root) Mean Square Error (RMSE or MSE) or Mean Absolute Error (MAE). In many applications, however, recommender systems predict item usage instead of numerical ratings, i.e., given an item they predict whether the user is going to use it (e.g., liking or buying an item) or not. Similar to the classification tasks, usage prediction is often evaluated using metrics such as precision, mean average precision (MAP), recall, and AUC (Area Under Curve).

To evaluate a ranked list of recommendations, other metrics such as normalized cumulative discounted gain (nDCG), hit ratio (HR), mean reciprocal rank (MRR), and average reciprocal hit rank (ARHR) are used [Ricci et al., 2015]. Assuming the availability of the ground truth ranking, Spearman's ρ or Kendall's τ can be used to measure the correlation between the true and the predicted ranking lists.

Other criteria for evaluating recommender systems include fairness [Beutel et al., 2019], diversity [Kunaver and Pozrl, 2017], serendipity [Manca et al., 2018], and privacy [Beigi et al., 2020]. More evaluation criteria can be found in [Ricci et al., 2015]. Table 2.1 presents formal definitions of common metrics used in recommender systems.

Metric	Formal definition	Description
MSE	$\frac{1}{ \mathcal{T} } \sum_{(u,i) \in \mathcal{T}} (r_{u,i} - \hat{r}_{u,i})^2$	\mathcal{T} : set of user-item interactions $r_{u,i}$: rating of user u for item i $\hat{r}_{u,i}$: predicted rating of user u for item i
P@k	$\frac{\#tp_k}{k}$	$\#tp_k$: no. true positives up to position k
R@k	$\frac{\#tp_k}{\#tp + \#fn}$	$\#tp$: no. true positives $\#fn$: no. false negatives
MAP@k	$\frac{\sum_{i=1}^k AP@i}{k}$	$AP@i = \frac{\sum_{j=1}^i P@j \times rel(j)}{\#tp_i}$ $rel(j)$: 1 if item j is relevant, 0 otherwise.
nDCG@k	$\frac{DCG@k}{IDCG@k}$	$DCG@k = \sum_{i=1}^k \frac{2^{rel(i)} - 1}{\log_2(i+1)}$ $IDCG@k$: Ideal DCG@k
HR@k	$\frac{\sum_{u \in U} I(u,k)}{ U }$	U : set of users $I(u,k)$: 1 if the top- k recommendations for user u contain at least one relevant item, 0 otherwise.
MRR@k	$\frac{1}{ U } \sum_{u \in U} \frac{1}{rank(u,k)}$	$rank(u,k)$: position of the first relevant item for user u among her top- k recommendations.

Table 2.1: Common metrics used for evaluating recommendations [Burges et al., 2005, Ricci et al., 2015].

2.2 Explaining recommendations

The importance of explaining system-generated advice has long been known for recommender systems [Clancey, 1983, Sinha and Swearingen, 2002]. Early recommenders mostly employed neighborhood-based or content-based filtering to personalize items for their users. The resulting recommendations could simply be explained by presenting similar neighbors [Sarwar et al., 2001, Herlocker et al., 2000] or overlapping content tags between recommendation items and users' profiles [Degenmis et al., 2007, Degenmis et al., 2008]. Since the Netflix Prize competition [Bennett and Lanning, 2007], the popularity of Latent factor models (LFM) [Koren, 2008, Koren et al., 2009] surged. The latent features learned by these models, however, are not directly explainable, fueling the progress of research on *Explainable Recommendation* [Zhang and Chen, 2020]. This chapter presents an overview of explainability in recommender systems.

2.2.1 Definitions

Explanation versus interpretation: The term *explanation* is historically defined as the information provided about the causal history of an event [Lewis, 1986]. In the context of machine

learning, the term *interpretation* has a similar definition. Miller [Miller, 2019] defines *interpretability* as the extent to which a human can understand the *cause* of a decision. Kim et al. [Kim et al., 2016] describe an interpretable model as one whose outcome can be consistently predicted by human.

While most researchers use the terms *interpretability* and *explainability* interchangeably, some draw a line between them. For instance, according to Rudin [Rudin, 2019], explanations are post-hoc justifications for black-box models, and unlike interpretations, they may not be faithful to the original model. Therefore, she encourages researchers to stop explaining black-box models (*explainable ML*) and start developing predictive models that are both accurate and understandable by humans (*interpretable ML*). In some online scientific forums, however, explainability refers to the extent to which the internal mechanics of a model can be explained in human terms¹. According to these forums, an interpretation describes a more abstract account of the model whose aim is to only outline the causal relations between the input and output of the model. In this thesis, we use the terms *explanation* and *interpretation* interchangeably.

Justifications: When identifying the true cause of an event is too challenging, humans invent a plausible story consistent with their knowledge to justify the event [Riedl, 2019]. These rationales, however, are not necessarily accurate reflections on actual causes. Inspired by human-like rationales, Vig et al. [Vig et al., 2009] attempt to justify the recommended items by showing relevant item tags that previously received positive sentiment from the user. Another example of such rationales in the context of recommender systems is the statistics such as the rating histogram of a user’s neighbors to justify the relevance of her recommendations [Herlocker et al., 2000].

2.2.2 Purposes of explanations

Research on *explainable recommendations* has received considerable attention from both the Recommender Systems (RecSys) [Zhang and Chen, 2020] and the Human-Computer Interaction (HCI) research communities [Tintarev and Masthoff, 2007]. Tintarev and Masthoff [Tintarev and Masthoff, 2007] highlight seven goals for explaining recommendations which can also be used for evaluating the explanations. These aims are as follows:

Trust: To increase user confidence in the system. Trust in a system can be evaluated explicitly using qualitative surveys [Pu and Chen, 2007] or implicitly by measuring users’ loyalty to the system indicated by their level of engagement [McNee et al., 2003].

Transparency: To help users understand *how the system works* and *how the recommendations are chosen for them*. Sinha and Swearingen [Sinha and Swearingen, 2002] evaluate transparency

¹<https://www.kdnuggets.com/2018/12/machine-learning-explainability-interpretability-ai.html>

by asking users whether they understand *why* the system recommended certain items to them.

Scrutability: To allow users to scrutinize their recommendations and tell the system when it is wrong. Scrutability enables users to modify their profiles and influence their future recommendations accordingly [Czarkowski, 2006]. This feature is particularly useful when users do not like their recommendations [Balog et al., 2019]. To evaluate scrutability, it is common to measure the quality of recommendations after incorporating user feedback [Zhang and Pu, 2006] or to count the number of conversation cycles a system requires to find a suitable item for the user [McCarthy et al., 2005].

Effectiveness: To help users make good decisions. An explanation is effective if it convinces the user to consume an item (e.g., watching a movie) which will be liked by her [Bilgic and Mooney, 2005]. Therefore, the effectiveness of explanations can be evaluated according to users' ratings for the explained items [Herlocker et al., 2000].

Persuasiveness: To convince users to adopt the recommended items. Evidence suggests that providing information about the recommendations affects user's opinions about the items and subsequently increases the likelihood of consuming or buying an item [Herlocker et al., 2000, Cosley et al., 2003, Bilgic and Mooney, 2005].

Efficiency: To help users make decisions faster. Explanations can significantly reduce users' cognitive effort and the amount of time they require for locating a desirable item [Pu and Chen, 2006]. In conversational recommender systems, explanations help elicit user preferences within a smaller number of cycles [Reilly et al., 2004a].

User satisfaction: To increase ease of use or engagement. Tintarev and Masthoff [Tintarev and Masthoff, 2007] distinguish between user satisfaction with the process of recommendation and with the recommended products. Satisfactory explanations inform users about the process of generating recommendations and help improve their overall acceptance of the system. User satisfaction can be evaluated explicitly via questionnaires or implicitly by measuring users' engagement levels [McInerney et al., 2018]. Nunes and Jannach [Nunes and Jannach, 2017] do not consider user satisfaction as a single objective and instead split it into ease of use, enjoyment and usefulness.

Relation between different goals: Most studies on explainable recommendations are concerned with only one of the goals described above [Zhang and Chen, 2020]. These goals, however, are not entirely independent of each other. For instance, an early study by Sinha and Swearingen [Sinha and Swearingen, 2002] suggests that transparency increases user's trust in the system. This connection, however, was rejected by Cramer et al. who showed that transparency and trust are not necessarily related to each other [Cramer et al., 2008]. Another notable relation is between effectiveness and persuasiveness. Persuasive explanations can manipulate users

into consuming items they do not like, and thus might harm effectiveness [Bilgic and Mooney, 2005]. Balog and Radlinski [Balog and Radlinski, 2020] have recently investigated interactions between all the described explanation goals. Their systematic evaluation exhibits that all goals are moderately correlated. Besides, user satisfaction was shown to be the most correlated metric with all others, which along with transparency and scrutability could provide the most complete assessment of the explanation quality.

2.2.3 Technical properties of explanations

Approaches to explainability are often guided by certain technical desiderata for the generated explanations. In what follows, we describe some of the prevalent requirements for explanations.

Counterfactual: A counterfactual explanation of a prediction is defined as *the smallest change to the feature values that causes the prediction to change to an alternative value* [Molnar, 2020]. To give an example, consider a user who is recommended with the book *Harry Potter II*. The explanation *because you read Harry Potter I* is counterfactual, if without watching *Harry Potter I*, the user would not be recommended with *Harry Potter II*. Finding counterfactual explanations for recommendations is particularly challenging for two reasons: (i) the input features are often categorical (e.g., item tags or user actions) which can lead to a combinatorial explosion when searching for minimal changes in the input, and (ii) the set of possible outcomes is huge as it contains all the conceivable item rankings, demanding explicit specification of the alternative, counterfactual outcomes.

Faithfulness: Recently, Jacovi and Goldberg [Jacovi and Goldberg, 2020] have identified three criteria for faithful explanations. The first criterion is the *consistency* of the explanations with the model’s prediction. For instance, the explanation *because you like Italian foods* is not in agreement with the high recommendation score predicted for a Chinese restaurant, and hence is unfaithful to the model. The second criterion demands the *uniqueness* of explanations for the same inputs and outputs [Jain and Wallace, 2019]. This criterion, however, has been brought under scrutiny by some other researchers [Wiegrefe and Pinter, 2019]. Lastly, a faithful explanation should pinpoint the *important features* for model reasoning. The importance/relevance/contribution of features is typically quantified as the amount of change caused in the model output as a result of their removal from the input. For instance, if the removal of the book *Harry Potter I* from the user’s history, substantially reduces the relevance score of *Harry Potter II*, then the explanation *because you read Harry Potter I* for the recommendation item *Harry Potter II* is faithful to the model. This technique is referred to as erasure [Jacovi and Goldberg, 2020], leave-one-out (LOO) [Jain and Wallace, 2019], or ablation [Covert et al., 2020]. A similar notion of faithfulness is credibility which has been used in graph-based recommenders, and is estimated as the strength of the explanation paths between the user and the recommended item [Lin et al., 2014, Yang et al., 2018].

Diversity: Another perspective for evaluating explanations is their diversity. In path-based explanations, heterogeneity of the relation types (edge labels) [Yang et al., 2018] and path patterns [Fu et al., 2020] are indicators of explanations’ diversity. For instance, according to Yang et al. the explanation $You \xrightarrow{\text{liked}} Harry\ Potter\ I \xrightarrow{\text{has author}} J.K.\ Rowling \xrightarrow{\text{authored by}} Harry\ Potter\ II$ is more diverse than $You \xrightarrow{\text{liked}} Harry\ Potter\ I \xrightarrow{\text{liked by}} User1 \xrightarrow{\text{liked}} Harry\ Potter\ II$ as it contains multiple edge types. The benefit of diversifying explanations goes beyond facilitating human comprehension; when brought to the design level, they help diversify and subsequently improve the quality of the recommendations as well [Yu et al., 2009].

Readability: For explanations to be comprehensible, they are expected to be short and readable. Readability can be quantified with the size of the explanation set [Das et al., 2011], the length of the explanation path [Yang et al., 2018], or the number of words and sentences in textual explanations [Costa et al., 2018].

Global versus local explanations: Global explanations help understand the distribution of the target outcome based on the input features [Murdoch et al., 2019, Molnar, 2020]. An example of a global explanation in the context of recommender systems is *because you read Harry Potter I* when the system recommends the book *Harry Potter II*. This explanation reveals the recurring pattern of recommending the next book in a series. Local explanations, on the other hand, zoom in on a single instance and examine why a model predicts a certain outcome for it [Molnar, 2020]. For instance, the explanation *because your friends liked Harry Potter II* is considered local for a given user.

2.2.4 Explanation styles

There are different presentation styles for explaining recommendations. Below, we describe the most common styles [Zhang and Chen, 2020] and present some examples in Table 2.2.

Neighborhood-based explanations: These explanations exploit the similarity of the recommended item to a user’s neighborhood to justify its relevance. These explanations are particularly common in recommenders based on collaborative filtering, and are either user-based [Abdollahi and Nasraoui, 2017, Heckel et al., 2017, Cheng et al., 2019] or item-based [Sarwar et al., 2001, Abdollahi and Nasraoui, 2017, Heckel et al., 2017, Catherine et al., 2017b, Chen et al., 2018b, Peake and Wang, 2018, Cheng et al., 2019]. User-based explanations (e.g., *because users similar to you also liked item A*) are grounded in the ratings given by similar users to the recommended item. Item-based explanations, on the other hand, attribute the relevance of the recommendation item to the user’s similar ratings in the past. A common template for such explanations is *because you liked item B which is similar to item A*. As users might not know other similar users, user-based explanation style is deemed less trustworthy than the item-based style where users are familiar with all the items mentioned in the explanation [Zhang and Chen,

2020].

Feature-based explanations These explanations justify the relevance of the recommended items by relating their content to user's preferences. Content-based recommenders use this strategy for generating recommendations as well as their explanations [Pazzani and Billsus, 2007, Vig et al., 2009]. An example of a feature-based explanation is *because you like movies tagged as "twist ending"* [Balog et al., 2019]. Such explanations require external sources (e.g., review texts [Hou et al., 2019] or social microblogs [Zhao et al., 2014, Zhao et al., 2016]) for extracting crisp user and item features.

Textual explanations These explanations leverage user-generated reviews to justify the relevance of the recommended items in natural language. A common approach to generate these explanations is to exploit the stated user sentiments towards different item features and align them with the aspects present in the recommend item [Zhao et al., 2015, Zhang et al., 2014b]. The resulting explanations are either presented in predefined templates such as *you might be interested in [aspect] on which this product performs well* [Zhang et al., 2014b], produced by crowd-workers [Chang et al., 2016], or automatically generated using sequence-to-sequence models [Costa et al., 2018, Li et al., 2020]. Some examples of textual explanation with free form are *the price and the sound quality is great* for an electronic device and *the bottle is very light and the smell is very strong* for a beauty product [Chen et al., 2021]. According to a comparison made by Chang et al. [Chang et al., 2016], textual explanations lead to a higher level of trust and user satisfaction compared to feature-based explanations.

Visual explanations These explanations use graphical designs to draw connections between the recommended item and user's preferences. These explanations are widely used in graph-based recommenders, where paths [Lao et al., 2011, Yang et al., 2018, Ai et al., 2018, Wang et al., 2019b, Xian et al., 2019, Ma et al., 2019], trees [Tao et al., 2019, Kouki et al., 2019] and subgraphs [Fang et al., 2011, Musto et al., 2016a, Seufert et al., 2016] present the relationships between different entities (in this context, between a user and her recommendation). Another type of visual explanation is saliency maps that highlight the important regions in the image of an item [Chen et al., 2019a]. Histograms have also been long used to summarize information related to the recommended items, user's interests, and her neighborhood [Herlocker et al., 2000, Donkers and Ziegler, 2020]. Word clouds are another type of visual explanations that help users quickly spot the interesting item properties [Wu and Ester, 2015].

Social explanations These explanations establish the relevance of the recommended items through user's social links [Papadimitriou et al., 2012, Sharma and Cosley, 2013]. A common template for these explanations is *because your friends A and B also liked item C*. This style of explanation is particularly effective in social recommender systems [Sánchez et al., 2017].

Explanation style	Example explanation for the recommendation <i>Harry Potter II</i>
Item-based	Because you read <i>Harry Potter I</i> .
User-based	Because similar users to you liked <i>Harry Potter II</i> .
Feature-based	Because you like books tagged as <i>fantasy</i> .
Textual	Because you like books that blend action, mystery and humor.
Path-based	$You \xrightarrow{\text{liked}} \text{Harry Potter I} \xrightarrow{\text{has author}} J.K.R \xrightarrow{\text{authored by}} \text{Harry Potter II}$
Social	Because your friend <i>Alice</i> loved <i>Harry Potter II</i> .

Table 2.2: Examples of different explanation styles.

2.2.5 Post-hoc explanations

Post-hoc explainability aims at justifying a model’s outcomes using methods that are decoupled from the original model [Zhang and Chen, 2020]. In other words, post-hoc explanations are model-agnostic, i.e., they do not assume prior knowledge of the model, and hence are suitable for explaining black-box models. Common techniques for generating post-hoc explanations include surrogate models [Nóbrega and Marinho, 2019, Lee et al., 2020], subgroup/subgraph discovery [Lonjarret et al., 2020], and association rule mining [Peake and Wang, 2018] as described below.

Surrogate models: Inspired by LIME (Local Interpretable Model-agnostic Explanations) [Ribeiro et al., 2016], multiple methods have been developed to explain black-box recommendations using surrogate models. The goal of these methods is to train an interpretable model that can approximate the local behavior of the original model. Common choices for surrogate models include decision trees [Singh and Anand, 2018], logistic regression [Xu et al., 2020], and Bayesian networks [Carmona et al., 2015]. These models, however, often have limited predictive power, curbing their faithfulness to the original model.

To address this limitation, Wang et al. [Wang et al., 2018c] have recently proposed a reinforcement learning framework for generating explanations that are faithful to the model and have a good quality of presentation. For this, they introduce agents for generating explanations and predicting user ratings. The environment rewards the agents if they can correctly predict the output rating and generate explanations that are both readable and consistent with the model’s prediction.

Data mining techniques: Recommendations can be explained by identifying the most contributing user inputs (e.g., the items in user history) to the prediction. To this end, several works have adopted data mining techniques such as *association rule mining*, *subgroup discovery* and *influence functions*. For instance, Peake and Wang [Peake and Wang, 2018] propose a method

for extracting global rules from user profiles. The resulting rules, however, often have limited fidelity, i.e., they may not be able to explain all the recommendations generated by the original model. Besides, the explanations provided in this model are not personalized to a specific user.

Lonjarret et al. [Lonjarret et al., 2020] resolve these issues by applying subgroup discovery to find the most influential user inputs. In this work, a subgroup is defined as a subset of items in a user’s history. The predictive power of each subgroup for a certain recommendation is assessed using a scoring function. Subgroups with the highest scores are then treated as explanations for the respective recommendation. In another work, Ying et al. [Ying et al., 2019] try to explain the predictions of graph neural networks (GNNs) by identifying a subgraph that has the highest mutual information with the given output.

Provided that the gradients and the Hessian matrix of the model are accessible, one can approximate the influence of individual data points on the model’s prediction using influence functions [Cheng et al., 2019]. This method, however, falls short on assessing the influence of a group of data points, reducing its effectiveness for generating counterfactual explanations.

Selected other methods: SHAP (SHapley Additive exPlanations) is another method that employs concepts from game theory to compute contribution scores of input features for a particular prediction [Lundberg and Lee, 2017]. This method, however, has a large computational cost [Molnar, 2020], and hence is not commonly used for explaining recommendations. Other approaches for post-hoc explainability use ad-hoc criteria for discovering and ranking explanations. For instance, Vig et al. [Vig et al., 2009] use item tags (e.g., comedy in movie domain) to justify their relevance to the users. These tags are scored according to their relevance to the item and the user’s profile. Sorted Explanation Paths (SEP) is another method for generating post-hoc explanations for recommendations [Yang et al., 2018]. This method first builds a unified information network to incorporate all available information about users and items. Next, the simple paths connecting the user to her recommendation are extracted and ranked based on three heuristic metrics; credibility, readability, and diversity. The top-ranked paths are selected as explanations for the final interpretation.

2.2.6 Model-aware explanations

Post-hoc explanations may not be faithful to the model and, as a result, can potentially mislead users [Lipton, 2018]. A natural solution to eliminate such a risk is to develop models that are explainable by design. We describe the most common explainable recommendation models (see [Zhang and Chen, 2020] for a broad survey) in the rest of this section.

Content-based models: As described in Section 2.1.1, in content-based models, items are recommended based on their content similarity to the user’s profile. Early approaches represented both the items and the user profile with term vectors, computed their proximity using cosine similarity, and explained recommendations by highlighting the key terms present in

both the item description and user profile [Billsus and Pazzani, 2000, Ahn et al., 2007]. Later approaches adopted ontological similarities and topic modeling to capture word semantics and further improved the ranking models [Magnini and Strapparava, 2001, Middleton et al., 2004, Zhao et al., 2015]. With the recent emphasis on designing interpretable models [Rudin, 2019], content-based models are regaining prominence. For instance, Rana and Bridge [Rana and Bridge, 2018] propose Recommendation by Explanation (r-by-e), where items are sorted according to their content overlap with users' histories. In another work, Balog et al. [Balog et al., 2019] devised a model for learning set-based preferences which uses the probability ranking principle in IR [Robertson, 1977] to rank items according to their probability of being liked by the user. In this model, the extracted set of tags are used as explanations.

Neighborhood-based models: These models comprise a class of collaborative filtering recommenders that rely on the neighbors of the target user [Ricci et al., 2015]. Recommendations generated in these models are simply justified by showing a list of neighbor users [Luo et al., 2008] or neighbor items [Sarwar et al., 2001] along with their similarity weights used at the time of prediction. In a more recent work, Heckel et al. [Heckel et al., 2017] employs co-clustering for generating recommendations and their corresponding neighborhood-based explanations.

Graph-based models: These models enhance explainability through capturing side information in user-item interaction graphs. For instance, He et al. [He et al., 2015] model item aspects as separate nodes and build a tripartite graph that captures user-item-aspect relations. They propose TriRank, a method for scoring the vertices of the graph such that the scores are consistent with user ratings and the nearby vertices receive close values. In this model, recommendations are explained using the top-ranked aspects connecting the target user to her recommended item. In another work, Park et al. [Park et al., 2017] build a unified graph that combines user-item interactions with a social network, and use random walk sampling to pick similar entities for recommendation. The explanations in this model identify the most similar users to the target user and most similar items in user's history to the recommended item.

Knowledge graph-based (KG-based) models: In the context of recommender systems, a knowledge graph is described as a heterogeneous information network (HIN) that models different entity types (e.g., movies, actors, and directors) and the relationships between them (e.g., directed by and stars in) [Guo et al., 2020]. The KG-based models leverage the rich factual information about the items available in knowledge graphs (or knowledge bases) to improve recommendations and their explainability. Catherine et al. [Catherine et al., 2017a] propose a KG-based model that takes as input a set of rules and a database of facts and constructs a proof graph via grounding the rules. Running personalized PageRank on the proof graph, they jointly rank items and entities in the KG where the entities can serve as an explanation for the recommendation.

More recent KG-based models embed the entities and relations in KG into feature vectors and plug them as input to the recommendation model. Huang et al. [Huang et al., 2018] use relation

embeddings to model attribute-level preferences and explain recommendations by referring to the attributes with the highest attention weights. Ai et al. [Ai et al., 2018] adopt a relaxed version of TransE [Bordes et al., 2013] for learning the KG representations and use a soft matching algorithm to generate short explanation paths for the recommendations. Wang et al. [Wang et al., 2018a] learn item-specific user embeddings by propagating user interests iteratively along the KG links and averaging the similarity of the user’s neighborhood to a given item. In this model, the paths with the highest relevance probabilities are used as explanations. In another work [Xian et al., 2019], a policy-guided graph search algorithm is proposed to effectively sample reasoning paths for recommendations.

Factorization models: The lack of intuitive meanings for latent features has motivated researchers to introduce interpretability in latent factor models. One approach is to enforce close proximity of similar entities (users and items) through additional regularizers [Abdollahi and Nasraoui, 2017, Hsieh et al., 2017], and use neighborhood-based explanations to justify the relevance of the recommendations. Another common approach is to exploit user preferences towards explicit item features and combine them with latent factors. Zhang et al. [Zhang et al., 2014b] introduce two additional input matrices, namely the user-feature matrix and the item-feature matrix that were factorized jointly with the rating matrix. They also employ a ranking scheme which aligns users’ preferences with the features present in the items, and thus generate explainable recommendations. Built upon these initial attempts, other approaches have been proposed that utilize techniques such as factorizing user aspect preference (UAP) and item aspect quality (IAQ) matrices [Hou et al., 2019], user-item-feature tensor factorization [Chen et al., 2016, Wang et al., 2018b], aspect topic modeling [Cheng et al., 2018] or aligning latent features to the topics/crisp features extracted from review texts [McAuley and Leskovec, 2013] or knowledge graphs [Anelli et al., 2020].

Deep learning models: The attention mechanism is known to improve interpretability of deep models. Deep recommenders use attention scores to highlight interpretable reasons such as important words/sentences in user’s reviews [Seo et al., 2017, Lu et al., 2018a, Avinesh et al., 2019, Chen et al., 2019c], regions of interests in product images [Chen et al., 2019a], interesting item aspects/features [Gao et al., 2019, Xin et al., 2019, Suzuki et al., 2019, Pan et al., 2020], useful item reviews [Chen et al., 2018a], or relevant items in a user’s history [Chen et al., 2018b]. In another work, Fusco et al. [Fusco et al., 2019] introduce an interpretable neural architecture which adopts layer-wise relevance propagation (LRP) [Bach et al., 2015] to determine the contributing input factors for a prediction. To increase the usefulness of explanations, recent studies have developed interpretable models that generate free-form explanations from review texts using sequence-to-sequence models [Costa et al., 2018, Lu et al., 2018b, Chen et al., 2019d, Chen et al., 2021].

Rule-based models: These models employ techniques such as decision trees for mining rules

for recommendations [Guttag et al., 2000]. Ma et al. [Ma et al., 2019] propose a model that uses different schemes for learning and ranking rules based on existing item-item associations in the training set. The induced rules are then used to generate recommendations that can be explained using the premises of the rules. HyPER (HYbrid Probabilistic Extensible Recommender) [Kouki et al., 2015] is another rule-based recommender that incorporates a wide range of signals such as item-item and user-user similarities using a set of rules. For instance, in the context of music recommendation, these rules can capture the intuition that similar users like similar artists [Kouki et al., 2019]. Wang et al. [Wang et al., 2018d] build a Gradient Boosting Decision Tree (GBDT) to derive a set of predictive rules based on features of users and items. The premises of the resulting decision rules are then used as interpretable input features to an attention-based network where attention weights highlight the importance of features for the prediction.

2.3 Scrutable recommenders

Modeling user preferences is critical to the effectiveness of recommender systems. Incorrect assumptions about users' preferences can lead to wrong or outdated user models. In such scenarios, systems would benefit from scrutability. To recall, scrutable recommenders enable users to inform the system when it is wrong, exert control over the personalization process and accordingly influence their future recommendations [Tintarev and Masthoff, 2007].

To enhance scrutability of recommender systems, it is natural to assume that users must first understand or at least have an intuition of the reasoning behind their received recommendations. Therefore, mechanisms for scrutability are often designed in the context of explanations [Jannach et al., 2016].

Critiquing-based recommender systems are prominent examples of scrutable recommenders, where users are allowed to critique their recommended items [Chen and Pu, 2012]. For instance, if a user wants to purchase a camera, she may ask the system to display similar cameras to the one recommended but with a lower price. Critiquing is a post-filtering functionality that helps users incrementally refine the preference models learned from their interactions with the system as they see more recommendations. Existing critique-based recommenders use either user-initiated, system-suggested, or natural language-based critiquing [Chen and Pu, 2012], which will be explained next. Since user-initiated critiquing is the most common type of critiquing, we dedicate the majority of our discussion to this type.

- **User-initiated critiquing:** This type of critiquing aims at assisting users in creating critiquing criteria on their own. For this, systems enable their users to interact with the attributes of the recommended items or their explanations and modify their influence on future recommendations. The existing methods for user-initiated critiquing mainly differ based on how they collect and incorporate user feedback. Early approaches mostly employed interpretable recommendation models whose inputs or internal parameters could be

directly controlled by the user. Some examples of such direct user control are modifying the weight of recommendation rules [Jannach and Kreutler, 2007], adding terms to (or removing terms from) own profile for better news recommendations [Ahn et al., 2007], decreasing (or increasing) the influence of friends on social recommendations [Bostandjiev et al., 2012, Knijnenburg et al., 2012b], removing tracks from own history to stop receiving similar songs as recommendations [Jin et al., 2018], and deactivating set-based preferences inferred about the user in the movie domain [Balog et al., 2019].

With the recent advances in feature engineering, modern approaches to scrutability are able to offer more fine-grained user control. Wu et al. [Wu et al., 2019a] propose a deep model that jointly predicts item ratings and personalized keyphrases as their explanations. When a user critiques a keyphrase (such as *Hip-hop* for a recommended song), the model generates a new explanation vector by zeroing out the critiqued phrases, updates the learned latent variables accordingly, and subsequently produces new recommendations. Built upon this work, Luo et al. [Luo et al., 2020a] extend the model to enable multi-step critiquing. For this, they formulate a linear programming (LP)-based optimization problem to compute the optimal amount of change in model parameters in response to the user’s critiques.

Another technique to incorporate users’ critiques into the model is through expanding the training set with biased input data to shift the decision boundary in the desired direction. This method is particularly beneficial for exerting control over black-box classifiers whose parameters cannot be directly modified. Lee et al. [Lee et al., 2020] propose a method for creating labeled pseudo-examples according to users’ feedback. When a user gives positive (negative) feedback on an item feature, pseudo-examples are created such that they feature the acted-upon attribute and are assigned positive (negative) labels. The influence of each pseudo-example is determined according to their proximity to the critiqued recommendation.

- **System-suggested critiquing:** This type of critiquing aims at guiding the search process by proposing a set of critique suggestions from which users can select. An example of such a suggestion is “cheaper with lower resolution” for a recommended camera. McCarthy et al. [McCarthy et al., 2004] propose dynamic critiquing, where in each recommendation cycle, association rule mining is applied to discover frequent sets of value differences between the current recommendation and the remaining products. The resulting sets identify multiple ways to filter out the search space (e.g., moving on to cheaper cameras). This approach was later improved by Zhang and Pu [Zhang and Pu, 2006] who also took into account user’s preferences when suggesting critiques.
- **Natural language-based critiquing:** This type of critiquing enables users to enter their critiques in natural language. In conversational recommender systems, this technique helps refine the user’s preferences in the course of a conversation with her. ExpertClerk [Shimazu, 2001], Adaptive Place Advisor [Thompson et al., 2004], and MusicBot [Jin et al., 2019] are examples of conversational recommender systems that exploit natural language-based critiquing for improving their recommendations.

3

POST-HOC EXPLANATIONS FOR BLACK-BOX RECOMMENDATIONS

Contents

3.1	Introduction	26
3.2	Discovering explanations	29
3.3	Ranking explanations	31
3.3.1	Learning to rank explanations	32
3.3.2	Learning to rank features	32
3.4	User studies	34
3.5	Evaluation setup	36
3.6	Results and insights	37
3.6.1	Key findings	37
3.6.2	Analysis and discussion	39
3.7	Related work	41
3.8	Conclusion	42

Users increasingly rely on recommendations generated by social platforms (hereinafter referred to as feeds) for consuming daily information. The items in a feed, such as news, questions, songs, etc., usually result from the complex interplay of a user's social contacts, her interests, and her actions on the platform. The relationship between the user's own behavior and the received feed is often puzzling, and many users would like to have a clear explanation for why certain items were shown to them. Transparency and explainability are critical in social platforms considering the potential concerns such as filter bubbles and privacy risks.

This chapter presents a framework that systematically discovers, ranks, and explains relationships between users' actions and items in their social media feeds. In Section 3.2, we describe how to model the user's local neighborhood on the platform as an interaction graph, a form of a *heterogeneous information network (HIN)* constructed solely from the information that is easily accessible to the concerned user. We posit that paths in this interaction graph connecting the user and her feed items can act as post-hoc explanations for the user. In Section 3.3, we propose

a *learning-to-rank (LTR)* model based on ordinal regression to score the paths based on two criteria; relevance (usefulness) and surprisal. We evaluate our framework using real user studies on two social platforms and demonstrate the practical viability and user benefits of the proposed framework in Sections 3.4-3.6.

3.1 Introduction

Motivation. Web users interact with a huge volume of content every day, be it for news, entertainment, or inside social conversations. To save time and effort, users are progressively depending on curated *feeds* for such content. A feed is a stream of *individualized* content items that a service provider tailors to a user. Well-known kinds of feeds include Facebook and Twitter for social networks, Quora and StackExchange for community question-answering, Spotify and Last.fm for music, Google News and Mashable for news, and so on. Since a feed is a one-stop source for information, it is important that users understand *how items in their feed relate to their profile and activity on the platform*.

On some platforms like Twitter and Tumblr, the feed originates solely from updates in the user’s social neighborhood or from their explicitly stated interest categories, and the connection is almost always obvious to the user. However, as service providers gather an increasing amount of user-specific information in an attempt to better cater to personal preferences, more and more platforms (like Quora, LinkedIn, and Last.fm), are generating complex feeds. Here, a feed results from an intricate combination of one’s interests, friendship network, her actions on the platform, and external trends. Such platforms are our focus in this chapter.

Over time, a user accumulates several thousands of actions that together constitute her profile (posts, upvotes, likes, comments, etc.), making it impossible for the user to remember all these details. Further, the user may not even possess a complete record of her actions on the platform, a common situation that has been referred to as the problem of *inverse privacy* [Gurevich and Wing, 2016].

In such situations, identifying *explanatory relationships* between the users’ online behavior (social network, thematic interests, actions like clicks and votes) and the feed items they receive, is useful for at least three reasons: (i) they can *convince* the user of their relevance, whenever a received item’s connection to the user is non-obvious or surprising; (ii) they can point the user towards future actionability (a course of action to avoid seeing more of certain kinds of items), and (iii) due to the sheer scale and complexity of data and models that service providers deal with, it is not always realistically possible to show end users the real reasons behind the feed items recommended to them; in such cases, these relationships act as justifications that the users could find plausible.

For example, if *Alice* sees a post on making bombs in her feed when she herself is unaware of any explicit connection to such, she might be highly curious as to what she might have done to create such an association. In this context, *Alice* would definitely find it useful if she is now

shown explanations like the following, that could remind her of some relevant actions: (i) her good friend Bob is a close friend of Charlie, who follows Chemistry and the bomb post was tagged as belonging to this category, or, (ii) she recently asked a question about food, that is recorded as a sub-category of Organics in the platform’s taxonomy, and the author of the bomb post has also categorized the post under Organics. In our study involving 20 users each on two platforms, participants reported seeing 2,410 such non-obvious items in their feeds over a period of two months.

Limitations of state-of-the-art. In principle, service providers are in the best position to offer such explanations. But they rarely do so in practice. For example, Quora simply tags items not emanating directly from one’s neighborhood or interest profile with ‘Topic you might like’, and Last.fm often has notes like ‘Similar to Shayne Ward’ for a track recommendation, neither elaborating the user’s relationship to the artist Shayne Ward, nor how the similarity was determined in the first place. Facebook’s explanations have similarly been brought into question [Andreou et al., 2018]. Except [Eslami et al., 2015], there is very little work investigating relationships between user interactions on the platform and items in their feeds (outside of Twitter [Edwards et al., 2014], where the feed is generated exclusively from the network). Relationship discovery [Liang et al., 2016], however, has been explored in other contexts and for different goals, most notably for understanding entity relatedness in knowledge graphs [Lao et al., 2011, Fang et al., 2011, Pirrò, 2015, Seufert et al., 2016, Bianchi et al., 2017], predicting links on social networks [Zhang et al., 2014a], and generating personalized recommendations [Yu et al., 2014, Lao and Cohen, 2010].

Understanding connections between user preferences and online advertisements has been investigated in simulated environments to some extent [Lécuyer et al., 2014, Lécuyer et al., 2015, Parra-Arnau et al., 2017]. We differentiate ourselves from these approaches in the following ways:

- Prior works have aimed to discover the *true provenance* of an item using models of *causality*, and are typically aimed at *reverse-engineering* the platform. This is very different from our goal where we try to unravel connections between a *user’s own actions* and what she sees in her feed, to enable her to better understand the interplay between her and the platform, and for her to have a better handle on the cognitive overload resulting from interactions with the platform;
- A common approach in this setting is to use *what-if* analysis methods [Lécuyer et al., 2014, Lécuyer et al., 2015] like differential correlation, which are intractable in a setting where a user makes thousands of interactions with the platform over an extended period of time;
- Mechanisms underlying advertisement targeting are guided by completely different (financial) incentives in comparison to regular feed items.

Approach. We propose FAIRY, a **F**ramework for **A**ctivity-**I**tem **R**elationship discover**Y**, that

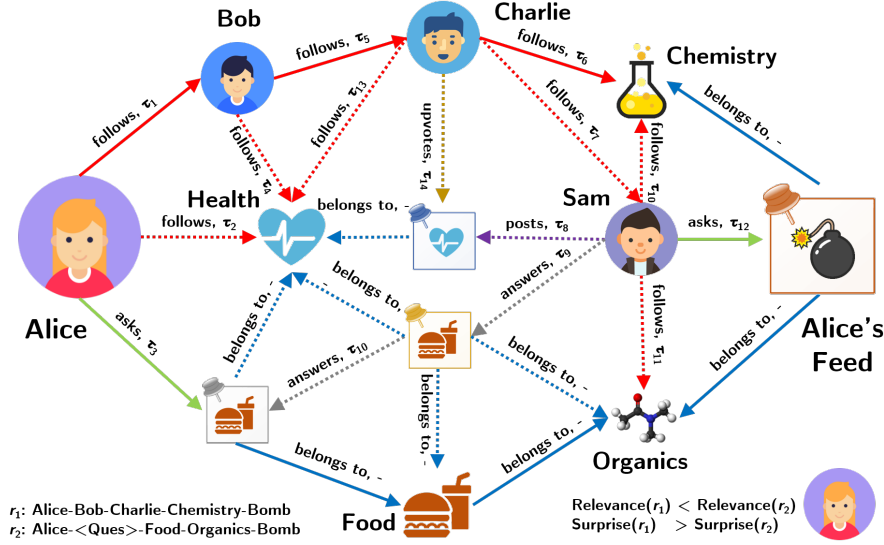


Figure 3.1: Toy interaction graph for Quora user Alice.

(i) addresses the discussed challenges by building user-specific *interaction graphs* exclusively using information visible to the user herself, (ii) learns models for predicting relevance and surprisal, trained on data from real user-studies on two platforms (Quora and Last.fm), and (iii) uses learning-to-rank (LTR) techniques to rank relationships derived from the above interaction graphs.

Since our goal is to pinpoint a set of user actions and their subsequent associations to the recommended feed item, FAIRY starts out by building an interaction graph connecting items in the user’s local neighborhood. An example of such an interaction graph for a Quora user is shown in Fig. 3.1. The interaction graph is modeled as a heterogeneous information network (HIN) [Sun and Han, 2012, Sun et al., 2011, Deng et al., 2011, Liang et al., 2016]. The HIN is a graph with different types of nodes (users $\{u_l\}$, categories $\{c_m\}$, and items $\{i_n\}$) and edges (corresponding to different action types like *follow*, *ask*, and *upvote*, shown in different colors). Nodes and edges in this HIN are weighted (not marked in Fig. 3.1 for simplicity). Edges are directed and *timestamped*, corresponding to the time an action was performed, wherever applicable (τ_j ’s in Fig. 3.1). In the FAIRY framework, each path in this interaction graph that connects the user u (Alice) to a feed item f (post on bombs) corresponds to a potential explanation for that item, provided that each edge e on the path has timestamp $\tau(e) < \tau(f)$, where $\tau(f)$ is the time the feed item was seen by u . Two possible explanation paths (out of many more) are shown via solid edges in Fig. 3.1.

The high number of such explanation paths in the HIN demands a subsequent ranking module. We employ a learning-to-rank (LTR) model based on ordinal regression [Joachims, 2006], that models judgments of relevance and surprisal, collected from the *same* users who received the feed item. Features used in the LTR models correspond to lightweight estimations of user influence, category specificity, item engagement, path pattern frequency, and so on. These are

derived from node and edge weights in the HIN, and are intentionally kept simple, to make them tangible and interpretable to the end user.

We compare FAIRY to three baselines: ESPRESSO [Seufert et al., 2016], REX [Fang et al., 2011], and PRA [Lao and Cohen, 2010], based on different underlying algorithms. These are state-of-the-art methods for computing entity relatedness over knowledge graphs. FAIRY outperforms these baselines on two representative platforms in modeling both relevance and surprisal.

The key contributions presented in this chapter are:

- the first user-centric framework for discovering and ranking explanation paths between users' activities on a social network and items in their feed;
- models for capturing the subtle aspects of user-relevance and surprisal for such explanations, with learning-to-rank techniques over lightweight features;
- extensive experiments including ablation studies, showing systematic improvements over multiple baselines, and identifying vital factors in such models;
- a user study conducted over two months involving 20 users each on Quora and Last.fm, providing useful design guidelines for future research on feed analysis.

3.2 Discovering explanations

The FAIRY framework uses a heterogeneous information network (HIN) [Sun and Han, 2012] to represent a user's presence and activities on a social network S as a graph, and relies on a learning-to-rank (LTR) model to order explanation paths mined from this user-specific interaction graph.

We are given a user $u \in U$, where U is the set of members of S , and her feed on S , F_u^S . Our goal here is to find and rank the set of explanation relations $R_{uf} = \{r_1, \dots, r_k\}$ between u and $f \in F_u^S$, where $k = |R_{uf}|$. R_{uf} is the set of connections between u and f via u 's local neighborhood on S . This local neighborhood is initialized using the set of activities that u performs: $A_u = \{a_1, \dots, a_m\}$, where $m = |A_u|$. Examples of such activities are 'asking a question' on Quora or 'loving an album' on Last.fm. Connections to f via A_u are identified by traversing the vicinity of u in S , and recorded in u 's interaction graph, G_u . Strictly speaking, G_u may not be fully connected. Typically, however, S has an underlying taxonomy \mathbb{T} of topics that all entities in the network must belong to, and overlaying \mathbb{T} on the vicinities of u and f ensures connectivity in G_u . Specifically, a path between any pair of entities may be found by first associating them via the categories they directly belong to, followed by subsequent generalization by traversing higher up in \mathbb{T} , till a connection is established (see [Kasneci et al., 2009] for an application of this strategy in relationship mining). More formally, each G_u is an instance of a HIN, defined as [Liang et al., 2016]:

Definition 3.2.1. The *Interaction Graph* of a user u is a directed and weighted multi-graph $G_u = (N, E, M_N^T, M_E^T, M_N^W, M_E^W, M^\tau)$, where: N is the set of nodes, E is the set of edges, $M_N^T : N \mapsto T_N$ and $M_E^T : E \mapsto T_E$ are functions mapping nodes and edges to their corresponding types (sets T_N and T_E), $M_N^W : N \mapsto \mathbb{R}_{\geq 0}$ and $M_E^W : E \mapsto \mathbb{R}_{\geq 0}$ are node and edge weight mapping functions respectively, and $M^\tau : E \mapsto \mathcal{T}$ maps each edge $e \in E$ to a timestamp $\tau(e)$.

The user u , her actions A_u , and the feed item f are naturally part of G_u ($u, f \in N, A_u \subseteq E$). Fig. 3.1 shows a representative interaction graph where u is the leftmost user *Alice*, and f is the rightmost item marked with a bomb. We now explain each property of this HIN, instantiating them with corresponding features of social networks.

Nodes $\{n \in N\}$ in G_u correspond to entities in the social network. **Node types** are either users or various classes of content (categories, tags, posts, songs, etc.). Via mapping M_N^T , we have $M_N^T(n) = t_N^n$, where $t_N^n \in T_N$ is one of the types above for node n . In Fig. 3.1, $N = \{Alice, \dots, Health, \dots, bomb-post\}$, $M_N^T(Alice) = \text{user}$, and $M_N^T(Health) = \text{category}$.

An **edge** $e \in E$ represents a relationship (interchangeably referred to as *actions* henceforth) between two nodes $n_1, n_2 \in N$. Edges represent the following connections: (i) *user-content*: these capture engagement or actions by the users on the content in S (in Fig. 3.1, user *Alice* $\xrightarrow{\text{follows}}$ category *Health*), (ii) *user-user*: they capture social relationships between users (*Charlie* $\xrightarrow{\text{follows}}$ *Sam*), and (iii) *content-content*: these edges capture relationships between content items (*Food* $\xrightarrow{\text{belongs to}}$ *Organics*). Each e is mapped to an **edge type** $M_E^T(e) = t_E^e$, where $t_E^e \in T_E$, instantiated depending on the platform (e.g., *asks*, *answers*, *follows*, ... for Quora); $t(\text{Charlie} \xrightarrow{\text{follows}} \text{Sam}) = \text{follows}$). For each edge in the HIN, we add an opposite edge *typed* with the *inverse relation* between the same pair of nodes (e.g., we add *Health* $\xrightarrow{\text{follows}^{-1}}$ *Alice*). This enables *bi-directional traversal* of the HIN.

Nodes $n \in N$ and edges $e \in E$ in G_u are associated with non-negative *weights*. The **node weight** $M_N^W(n)$ may reflect node *influence*, *specificity*, or *engagement* depending upon the entity type. The value of $M_N^W(n)$ itself may be derived from measurable features of the platform that are visible to u . For example, if the post on food was upvoted by 30 users, $M_N^W(\text{food-post}) = 30$. An **edge weight** $M_E^W(e)$ counts the number of times the action was performed, e.g., if a Last.fm user *scrobbles* (listens to) a specific song five times, then $M_E^W(e) = 5$.

G_u is a **multi-graph**, implying that there may exist more than one edge between any two nodes, corresponding to multiple actions. For example, Twitter users can both *like* and *re-tweet* a Tweet, and users on Last.fm can both *scrobble* a song, and *love* it.

There exists a unique **edge timestamp** denoted by $\tau(e)$, which is the time when the corresponding action was performed (e.g., $\tau(\text{Bob} \xrightarrow{\text{follows}} \text{Charlie}) = \tau_5$). This is possibly null, if e has existed since the epoch (category memberships are assumed to be such edges in this work). For edges with $w(e) > 1$ (action performed multiple times), we define $\tau(e)$ as the timestamp of the first instance of this action.

The size of the interaction graph G_u is characterized by the *eccentricity* of u , $\varepsilon(u)$. This

is the greatest *graph (geodesic) distance* (length of shortest path) d between u and any other node n' in G_u , i.e., $\varepsilon(u) = \max_{n' \in G_u, n' \neq u} d(u, n')$. In other words, $\varepsilon(u) = 3$ yields the 3-hop neighborhood of u in G_u . In Fig. 3.1, $\varepsilon(\text{Alice}) = 4$.

We are now in a position to discover the set of explanation relations R_{uf} , formally defined below.

Definition 3.2.2. An *Explanation Path* is a path r connecting u and f in G_u such that the timestamp of every edge $\tau(e)$ in r is less than the time when f was seen by u , i.e. $\tau(f) > \max_{e \in r} \tau(e)$.

In Fig. 3.1, for $(\text{Alice}, \text{bomb-post } f)$, if we have $\tau(\text{bomb-post}) = 13$, then $\text{Alice} \xrightarrow{\text{follows}} \text{Health} \xrightarrow{\text{belongs to}^{-1}} \text{health-post} \xrightarrow{\text{posts}^{-1}} \text{Sam} \xrightarrow{\text{asks}} \text{bomb-post}$, is a valid explanation path. However, the following path: $\text{Alice} \xrightarrow{\text{follows}} \text{Bob} \xrightarrow{\text{follows}} \text{Charlie} \xrightarrow{\text{upvotes}} \text{health-post} \xrightarrow{\text{posts}^{-1}} \text{Sam} \xrightarrow{\text{asks}} \text{bomb-post}$, is *invalid* as the timestamp $\tau(\text{Charlie} \xrightarrow{\text{upvotes}} \text{health-post}) = 14 > 13$. Henceforth, *explanations* and *relations* (with or without *paths*) are used interchangeably, and should be understood as equivalent. An explanation path can thus be a combination of user-content, user-user, and content-content edges. For example, the path outlined earlier in this paragraph is a mixture of user-content and content-content edges, but lacks any user-user connection. Thus, given, u , f , and G_u , we extract all explanation paths between u and f from G_u as candidates for further processing.

It can be argued that since the interaction graph could often be dense, explanations could be better presented as *sub-graphs* [Seufert et al., 2016] instead. But we prefer paths in this work due to the following reasons: (i) sub-graphs are difficult to isolate by influence, especially for dense neighborhoods; (ii) paths (simple, without loops), are atomic units of relationships, and subgraphs can, in fact, be reduced to a number of constituent paths; and (iii) subgraphs are harder to interpret for the average user, and may be more difficult to make comparative assessments.

3.3 Ranking explanations

Due to the high activity count aggregated by a user over her time as a member of the platform, and due to the richness of the platform itself (allowing a post to have multiple categories, having a detailed directed acyclic graph (DAG) taxonomy, etc.), the usual number of candidate explanation paths is too high to be processed by a user, if presented all at once. Measurements from our user study show that the number of such paths can vary from a few thousand to even millions (depending on the graph size determined by $\varepsilon(u)$, and length of relation path r). Thus, it is imperative that we rank these paths and present only a top few, to prevent a cognitive overload.

3.3.1 Learning to rank explanations

Over the last decade or so, learning-to-rank (LTR) [Joachims, 2006, Cao et al., 2007] has emerged as the *de facto* framework for supervised ranking in information retrieval and data mining, which motivates its application to FAIRY. LTR has three basic variants: pointwise, pairwise, and listwise, with each type proving beneficial in specific contexts [Radlinski and Joachims, 2005, Bast and Haussmann, 2015].

The common guiding criterion, though, is the nature of gold label judgments that can be collected (and/or inferred). In our case, we want to model relevance and surprisal of the explanation paths. Generally, it is difficult for a user to score a standalone explanation path on scale of 0 – 10, say. At the other extreme, it might be even harder to score a complete list of say, a heuristically chosen sample of ten paths. Collecting *preference judgments* is the most natural thing to do in our setting: it is a conceivable task for an end user (an average social media user, here) to rate a path as being *more relevant* (generally useful as a satisfactory explanation to her), or it being *more surprising* (such as discovering a forgotten/unknown connection in her vicinity), *than another path* connecting her to the same feed item. Collecting such explicit pairwise annotations has been suggested as being cognitively preferable to the user in other contexts like document relevance assessments [Carterette et al., 2008]. In a similar vein, we use a pairwise learning-to-rank model based on ordinal regression, that directly makes use of the users’ preference judgments on pairs of explanations [Joachims, 2006]. We used $SV M^{rank}$ with a linear kernel, as it is very fast to train, and has been shown to be highly effective in ranking result pages for search queries [Schuhmacher et al., 2015].

3.3.2 Learning to rank features

The general principles guiding this work are explainability and transparency. This influences the choice of our features in two ways: (i) while the provided explanation should already be insightful to the user, it is not unreasonable to assume that the user could, in turn, want to know what was responsible for a few chosen paths to be shown as more surprising or relevant than others. This points towards using simple and interpretable features that can give a naive user a handle on what was found to be “important” in this context; and, (ii) the features should be *visible* to the user in question: either public, or easily accessible in a few clicks, or by visiting a user-specific URL. Data that only the service provider has access to (derived measures of user-user similarity), or requires excessive crawling (total number of authors of all posts in a category) are clearly unsuitable in the task at hand. With these guidelines, we define the following sets of features for the FAIRY framework. These are grouped into five sets: (i) user, (ii) category, (iii) item, (iv) path instance, and (v) path pattern. For the first three sets, if there are multiple instances of the same type on a path (two users or three categories), the feature value is averaged over these instances.

3.3.2.1 User features

We consider two factors for users on explanation paths: (i) *user influence*, and (ii) *user activity*. Influence is typically measured using number of followers or the link ratio (ratio of followers to followees, wherever *friendship* is not mutual) on social networks [Srijith et al., 2017]. Higher the link ratio, higher is the perceived influence. For activity, we measure the individual types of activity that the user is allowed on the platform (*scrobbling* tracks, *loving* tracks, and *following* other users on Last.fm). More influential and active users may have a discriminative effect on the user’s judgments for relevance.

3.3.2.2 Category features

(i) *Popularity* or influence can be estimated for categories too, for example, by counting the number of posts in them, or looking at their total numbers of followers or subscribers. Such aggregates are often made visible by providers, and it is not necessary to actually visit and count the items concerned. (ii) We also consider *category specificity* [Ramakrishnan et al., 2005], which is reflected by its depth in the category hierarchy (the higher the level, the more specific it is, root assumed to be level 0) and the number of children (sub-categories) it has in the taxonomy (more children implying less specificity). While popularity may influence user’s judgments on relevance, high category specificity may directly affect the surprisal factor.

3.3.2.3 Item features

(i) *Specificity* for items (songs, posts, etc.) may be analogously computed by counting the number of different categories that the item belongs to. (ii) *Engagement* is an important measure for items, which represents the different actions that have been performed on it (typically the same as the number of users who have interacted with the item). Examples include the number of different listeners of a song on Last.fm, or the number of different answers that a question has on Quora, etc. Engagement may be perceived as analogous to influence or popularity for users and categories in its role in FAIRY.

3.3.2.4 Path instance features

There are some properties of the explanation path as a whole: these can be understood better by further separating them into path instance and path pattern features. Instance features measure aspects of the specific path in question. These include: (i) Aggregate *similarity of feed item f to relation path r* , i.e., $sim(f, r) = \frac{\sum_{n \in r, n \neq u, f} sim(n, f)}{len(r) - 1}$, where n is any *internal node* on the path, and $len(r)$ is the path length (-1 gives the number of such internal nodes). This normalized similarity function $sim(\cdot, \cdot) \in [0, 1]$ is treated as a plug-in and can be instantiated via embedding-based similarities, or computed locally from G_u itself. The item-path similarity

function, in a sense, measures the *coherence* of the extracted relation path to the recommended item. If it is very low, the path could be quite surprising to the user. (ii) Analogous aggregate *similarity of user u with r* : $\text{sim}(u, r) = \frac{\sum_{n \in r, n \neq u, f} \text{sim}(n, u)}{\text{len}(r) - 1}$. This feature models the familiarity of the user with the path as a whole. (iii) *Path length $\text{len}(r)$* : The length of an explanation is easily one of the most tangible factors for a user to make decisions on: while shorter paths may imply obvious connections, longer paths may be more surprising. (iv) *Path recency*: This is a temporal feature, and is defined as the most recent edge on the path with respect to the feed: $\min_{e \in r} \tau(f) - \tau(e)$. The difference is measured in *days*. Highly recent paths will have a low value of this feature; the idea is that relatively newer paths may be more relevant to the user due to freshness, while older paths may have an associated surprise factor. (v) *Edge weights*: This feature averages the number of times each action on the path has been repeated (e.g., number of times a user has listened to a specific song). Note that this feature cannot be used for Quora as none of the actions can be repeated, i.e, the user cannot upvote/follow/ask/answer the same item more than once.

3.3.2.5 Path pattern features

Pattern features drop concrete instantiations $\{n\}$ and $\{e\}$ in r and deal with the underlying sequence of node and edge types $M_N^T(n)$ and $M_E^T(e)$ instead (r_1 in Fig. 3.1 has the path pattern: $\text{user} \xrightarrow{\text{follows}} \text{user} \xrightarrow{\text{follows}} \text{user} \xrightarrow{\text{follows}} \text{category} \xrightarrow{\text{belongs to}^{-1}} \text{post}$). We consider the following features: (i) *Pattern frequency*: This is the average *support* count of a pattern between any u and f (frequent patterns may be less surprising). (ii) *Pattern confidence*: The percentage of (u, f) pairs with at least one observed instance of the pattern. (iii) *Edge type counts*: Users in our study explicitly mentioned the effect of some edge types on their choice of relevance and surprisal. Thus, to zoom into the effect produced by the aggregate measure of pattern frequency, we also considered the counts of each edge (action) type present on the path as features (*#likes*, *#follows*, *#belongs to*, etc).

The general intuition here is that the user has clear mental models of relevance and surprisal; the above features are what she sees in her daily interactions with the platform. These are therefore tangible perceptions that influence her models, and the aim here is to *learn* how these factors combine to mimic her assessments.

3.4 User studies

Platforms. Since user feeds are never public, we needed to design user studies from where we could collect gold judgments on explanation paths extracted and ranked by FAIRY. While there are a plethora of social network platforms providing personalized feeds, we chose Quora and Last.fm as they possess the richness that can truly test the full power of our HIN model. To be specific, these platforms have node types, edge types, non-obvious feed items, properties for

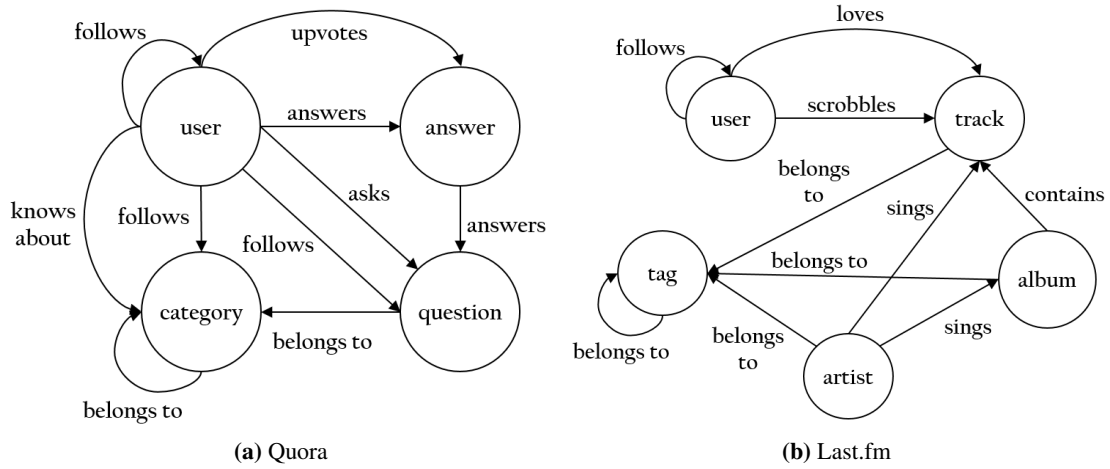


Figure 3.2: Logical schemata for the Quora and Last.fm platforms, showing permissible relationships between node (entity) and edge (action) types.

estimating node and edge weights, have millions of members, and have been subject of previous research (e.g., Quora in [Wang et al., 2013], Last.fm in [Jäschke et al., 2007]). Also, one being a community-question answering site, and the other an online music recommender, they represent two completely different application platforms, and hence are ideal to evaluate FAIRY on. We use a slightly simplified version of these platforms’ schemata, as shown in Fig. 3.2.

Users. We hired 20 users each for Quora and Last.fm, for interacting with the platforms and assessing explanations, in May - July 2018. Users had to set up fresh accounts (using real credentials for accountability) so that all their activity can be recorded. Each user had to spend 20 hours on one platform in total. This total time was divided into one hour sessions per day, with a gap of one day between consecutive sessions. Interactions were planned in this staged manner so as to allow the service provider enough time to build up user profiles, and generate personalized feeds. The hired users were graduate students of mixed background who were familiar with Quora and Last.fm. They were paid \$10 per 1-hour session, for three tasks: (i) interacting with the platform (a minimum of 12 activities per session from permissible actions in Fig. 3.2, no upper bound, “natural” behavior recommended), (ii) identifying non-obvious feed items after going through their complete feed (both platforms have a countable feed at a given point of time), and (iii) providing assessments of relevance and surprisal. Their complete activity and feed on the platforms were recorded. Users were given a random set of ten initial topics to follow on Quora, due to such a requirement by the platform. They were assigned five initial followers from within the study group on Quora and Last.fm. More details about the design of this study can be found in Appendix A.

Judgments. After each session, we updated the interaction graphs of users, selected three non-obvious recommendations per user, and mined explanation paths for these feed items. Path length was restricted to four for Quora and five in Last.fm, which resulted in about $2k - 30k$

Platform	#Activity	$ N $	$ E $	$\#(u, f)$ paths
Quora	217	31,769	532,569	28,527
Last.fm	132	22,815	79,252	1,897

Table 3.1: User study statistics (nos. averaged over users except for the last column).

paths on average over (u, f) pairs. Increasing path length led to an *exponential increase* in the number of explanations. For each session, we chose 1 to 3 random feed items (among the ones reported by the user) and randomly sampled 25 pairs of paths per item to be assessed by the user on two questions: (i) Which explanation path is more relevant (useful) to you for this feed item? (ii) Which explanation path is more surprising to you for this feed item? The users were allowed to give free-text comments on reasons motivating their choices. Several of these intuitions and allusions were encoded into our design considerations.

Statement on ethics. To comply with standard ethical guidelines, all participants were informed about the purpose of the study, and that their data was being collected for research purposes. They signed documents to confirm their awareness of the same. Users signed up with their real credentials; no terms of service of the providers were violated over the course of the research. All users deleted their accounts at the conclusion of the study.

3.5 Evaluation setup

Datasets. For Quora, we used 11,677 pairs of explanation paths evaluated by 20 users as the gold standard. These explanation paths covered 459 distinct (u, f) pairs. Each explanation path appeared in 1.9 pairs on average. For Last.fm, we collected 4,791 evaluated pairs from 20 users. These paths were extracted as potential explanations for 235 distinct (u, f) pairs. Each path occurred in about 1.7 pairs. Details on the interaction graphs are in Table 3.1.

LTR. To run LTR, we divided each dataset into 80% training, 10% development, and 10% test sets. We used SVM^{rank} [Joachims, 2006] with a linear kernel in all our experiments.

Baselines. FAIRY was compared with three baselines for relationship discovery: ESPRESSO [Seufert et al., 2016], REX [Fang et al., 2011] and PRA [Lao and Cohen, 2010]. In ESPRESSO [Seufert et al., 2016], the goal is to find relatedness cores (dense subgraphs) between two sets of query nodes. For this, they first identify a center, i.e., a node with the highest similarity to both the input query sets. Then, they expand the subgraph by adding other key entities, their context entities, and query context entities. In each step, the entities are selected based on random walk-based scores. To apply this algorithm, we consider $\{u\}$ and $\{f\}$ as the input query sets. To compute the score of each path, we first find the most similar node on the path to both u and f as the center. We then expand the set of selected nodes by adding their adjacent nodes on the

Platform	Method	FAIRY	ESPRESSO [Seufert et al., 2016]	REX [Fang et al., 2011]	PRA [Lao and Cohen, 2010]
Quora	Relevance	60.33*	49.93	23.47	30.84
	Surprisal	60.38*	49.93	21.58	51.93
Last.fm	Relevance	56.24*	48.85	37.66	49.06
	Surprisal	54.21*	51.39	36.03	43.29

Table 3.2: Accuracy of FAIRY compared with baselines. The maximum value in a row is marked in bold. An asterisk (*) denotes statistical significance of FAIRY over the strongest baseline, with p -value ≤ 0.05 for a two-tailed paired t -test.

path. At the time of adding each node, we compute their random walk-based similarity to the adjacent (already selected) node on the path. In the end, the score of each path is computed by averaging the scores of its constituent nodes.

REX [Fang et al., 2011] takes a pair of entities and returns a ranked list of its relationship explanations. Like ESPRESSO, the relationships are in the form of subgraphs. REX ranks the extracted relationships based on different classes of measures. These measures can also be used to score paths. We used all aggregate and distributional measures from the original paper. However, for brevity, we only describe the global distributional measure as it performed the best. This measure captures the rarity of relations as a signal of interestingness. For this, we sort explanation paths according to $1 - \text{pattern_confidence}(\cdot)$. To recall, $\text{pattern_confidence}(r)$ for explanation r is the percentage of (u, f) pairs with at least one explanation path with the same pattern as that of r . Accordingly, explanations with less frequent patterns receive higher scores.

For PRA [Lao and Cohen, 2010], we computed scores of paths via pattern-constrained random walks. For instance, a pattern like “user $\xrightarrow{\text{follows}}$ post” only allows the random walker to leave the source node with type “user” to nodes with type “post” via edges of type “follows”. The PRA score of the path r between u and f is the probability that a random walker constrained by the pattern $\text{path_pattern}(r)$ starts at u , traverses path r and visits node f .

Metric. We measured the accuracy of each method (or configuration, as applicable) as the ratio of the correct predictions to all the predictions over pairs of relationship paths.

3.6 Results and insights

3.6.1 Key findings

Comparison of FAIRY with baselines. Table 3.2 shows the comparison of accuracy for the relevance and surprisal models of FAIRY with baselines on both platforms. In all cases, FAIRY significantly outperforms all the baselines (paired t -test with p -value < 0.05). Note that all baselines have the same model for both relevance and surprisal as they try to find either the most ‘*relevant*’ or the most ‘*interesting*’ relationships. All baselines solely rely on the structural properties of the underlying graphs. In ESPRESSO, scores of cores are affected by the degree

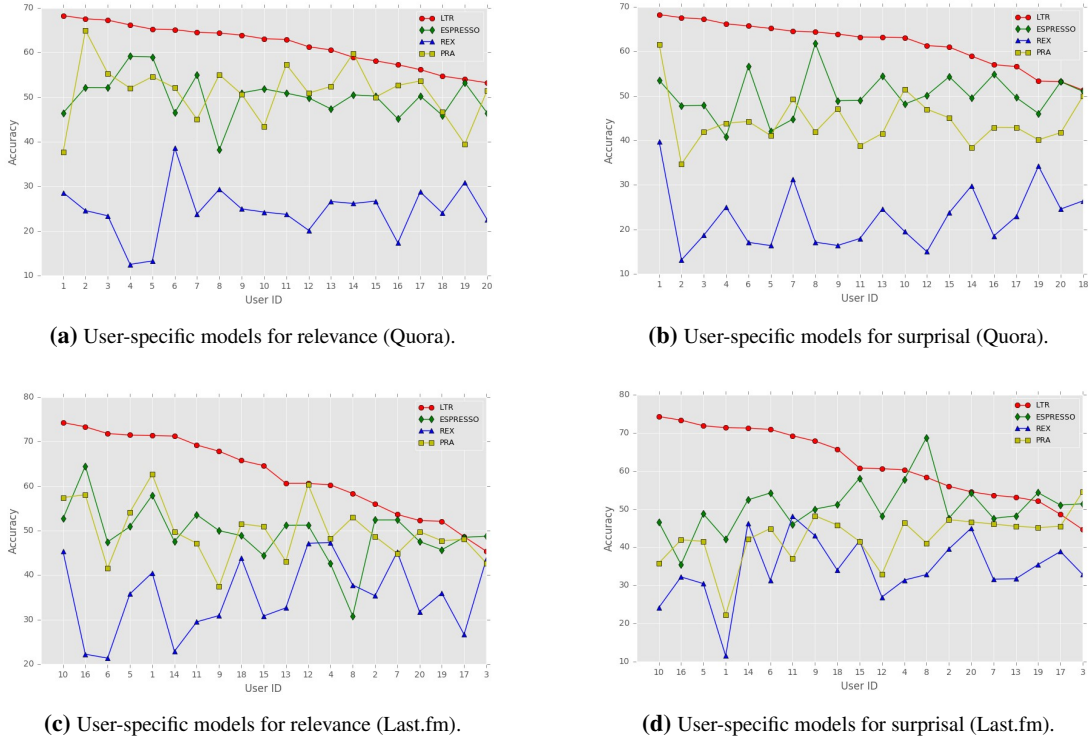


Figure 3.3: Performance of user-specific models for ranking explanations. Red: LTR (FAIRY), green: ESPRESSO [Seufert et al., 2016], blue: REX [Fang et al., 2011], yellow: PRA [Lao and Cohen, 2010].

(no. outgoing edges) of the intermediate nodes. More precisely, on Quora, category nodes have large degrees as they are connected to many other nodes. This affects scores of paths with many intermediate category nodes. A similar problem happens in PRA as it is also based on random walks. Besides, as path scores are computed by multiplying inverse node degrees, PRA is biased toward shorter paths. In REX, we are only able to compare explanation paths with different path patterns. This substantially lowers accuracy, as many explanation paths share the same pattern.

User-specific models. To test subjective preferences for relevance and surprisal, we built and evaluated analogous user-specific LTR models. FAIRY accuracies of user-specific models were observed to be higher than the aggregate global model for most users (Fig. 3.3a-3.3d). There are, however, a few users whose judgments we could not easily predict. User ids' are assigned in descending order of FAIRY accuracy in Fig. 3.3a, and these ids are used as references for comparison across all subsequent figures. We found that FAIRY outperforms baselines in user-specific models as well. Note, again, that baselines do not have separate models for relevance and surprisal.

Features	Quora		Last.fm	
	Relevance	Surprisal	Relevance	Surprisal
All	60.33	60.38	56.24	54.21
No user features	60.21	60.19	56.03	54.15
No category features	60.33	60.23	56.65	54.80
No item features	60.31	60.38	56.24	54.42
No path instance features	51.69	51.73	54.32	53.65
No path pattern features	60.21	60.71	55.78	54.21

Table 3.3: Ablation study results. The highest value in each column is marked in bold.

3.6.2 Analysis and discussion

Ablation study. To analyze the effects of different feature groups on LTR accuracy, we removed one group at a time and retrained the models. The key finding was that the removal of any of the feature groups would hurt the accuracy of the models on at least one platform. For example, while the removal of path instance features does not affect the accuracy of the models on Last.fm, it significantly reduces the accuracy on Quora by around 9% (from $\simeq 60\%$ to $\simeq 51\%$). Therefore, for the sake of consistency, we kept the set of features the same on both platforms. Details of feature group removal are presented in Table 3.3.

To systematically study variations in features, we tested the effects of adding/removing/replacing single features. For instance, to compute the aggregate similarity of the feed item to the explanation path, we plugged in two different similarity functions: embedding-based similarity and graph-based similarity. In the former, we computed the similarity of each entity on the path to the feed item by averaging the pairwise cosine similarity between the embeddings of the categories/tags associated with the feed and the entity. To learn the embedding of each category/tag, we first sampled a set of related sentences. On Quora, we sampled 100 questions at random, posted in the concerned category. For Last.fm tags, we treated each tag as a sentence. Then we learned the embedding of each sentence using the latent variable generative model in Arora et al. [Arora et al., 2017] and represented each category/tag with the average embedding of the sampled sentences.

In the graph-based similarity function, we used the taxonomic distance between categories/tags in the category DAG. Replacing the embedding-based similarity with graph-based one improved the accuracy by 8 percent (from $\simeq 52\%$ to $\simeq 60\%$). This emphasizes the insufficiency of graph structures in capturing the similarity of nodes. We tried several other variations, too. For example, adding counts of *node types* as a new feature, or replacing total edge label counts with edge type counts (user-content, user-user, and content-content) counts, or replacing all user activity features with their maximum values (instead of averages): all of these hurt the accuracy of the Quora relevance model ($\simeq 60\%$) by at least $\simeq 0.05\%$.

Perturbation analysis. To understand the effect of instances of each node type on users' judg-

Sampling	Quora		Last.fm	
	Relevance	Surprisal	Relevance	Surprisal
Random	60.33	60.38	56.24	54.21
Perturb user	56.36	56.66	57.77	58.57
Perturb category	58.51	58.25	55.52	56.84
Perturb item	50.26	50.35	52.85	52.14

Table 3.4: Effect of sampling strategy on FAIRY performance. The highest value in a column is marked in bold.

ments, we changed our sampling strategy so that paths in each pair differ in only one instance. In other words, one path can be obtained by perturbing any one instance of the other path. For example, $Alice \xrightarrow{\text{follows}} Bob \xrightarrow{\text{follows}} Health$ is a user-perturbation of the path $Alice \xrightarrow{\text{follows}} Jack \xrightarrow{\text{follows}} Health$. We generated perturbed paths for each node type and retrained the LTR models. Table 3.4 shows that in most cases, random sampling performed better than such perturbed sampling. The only exception is for user-perturbed paths in Last.fm, indicating that the chosen user features are particularly effective in the music domain. In one-on-one interviews with users at the end of the study, some users expressed their interest in receiving explanation paths with certain friends on it. User-perturbation created pairs of paths where such friends were present in one but not in the other, resulting in clearer preferences and improved modeling. The last row of the table, however, shows that item-perturbations greatly degraded performance. This can be attributed to incomplete knowledge of the users on certain path items in the interaction graphs: replacing such items with yet other unfamiliar items clearly has an arbitrary effect on assessments, which seemed to worsen the model’s accuracy.

Transitivity of judgments. Relevance and surprisal are subtle factors, and it is worthwhile to investigate transitivity in users’ assessments (for both understanding and as a sanity check). So we extracted all triplets (r_i, r_j, r_k) of explanation paths where we had a user’s judgments on all the three possible *pairs* (r_i, r_j) , (r_i, r_k) and (r_j, r_k) built from them. We then computed a $transitivity_score = \frac{\sum_{(r_i, r_j, r_k)} I(r_i, r_j, r_k)}{\sum_{(r_i, r_j, r_k)} 1}$ where $I(r_i, r_j, r_k)$ is the indicator of a transitive triplet. For instance, if $Relevance(r_i) < Relevance(r_j)$ and $Relevance(r_j) < Relevance(r_k)$, then $I(r_i, r_j, r_k) = 1$ if $Relevance(r_i) < Relevance(r_k)$. On a positive note, it turned out that people’s judgments followed transitivity for 80% and 74% of the Quora and Last.fm triplets, respectively.

Surprisal and complexity. To make sure that users were not simply using complexity (say, length) of a path as a proxy for surprisal, we asked about a third of the users to additionally select the “more complex” path. We noticed that for $\simeq 7.5\%$ of pairs, the more surprising path was in fact the simpler (shorter) one, indicating that surprisal is based on more implicit factors.

Anecdotal examples. Table 3.5 presents some examples of correct (in blue) and wrong (in red)

Quora: Relevance	1	Shrey $\xrightarrow{\text{follows}}$ Ali $\xrightarrow{\text{follows}}$ Social Psychology $\xrightarrow{\text{belongs to}^{-1}}$ What are the things you shouldnt say to people who hate themselves Shrey $\xrightarrow{\text{follows}}$ Business $\xrightarrow{\text{belongs to}}$ Advice on working with people $\xrightarrow{\text{belongs to}}$ Social psychology $\xrightarrow{\text{belongs to}^{-1}}$ What are the things ...
	2	Ali $\xrightarrow{\text{follows}}$ Travel $\xrightarrow{\text{follows}^{-1}}$ Stephanie $\xrightarrow{\text{asks}}$ What is the tackiest thing you have ever seen at a wedding? Ali $\xrightarrow{\text{follows}}$ What is something you have tried but will never do again $\xrightarrow{\text{belongs to}}$ Experiences in Life $\xrightarrow{\text{belongs to}^{-1}}$ What is the ...
Quora: Surprisal	3	Amr $\xrightarrow{\text{follows}}$ Cooking $\xrightarrow{\text{follows}^{-1}}$ Ratnesh $\xrightarrow{\text{asks}}$ How can you learn faster $\xrightarrow{\text{answers}^{-1}}$ answer by Ara Amr $\xrightarrow{\text{upvotes}}$ answer by James $\xrightarrow{\text{answers}}$ How can you learn faster $\xrightarrow{\text{answers}^{-1}}$ answer by Ara
	4	Ali $\xrightarrow{\text{upvotes}}$ answer by Gerry $\xrightarrow{\text{answers}}$...something you secretly regret $\xrightarrow{\text{belongs to}}$ life and living $\xrightarrow{\text{belongs to}^{-1}}$ How do I give up on life Ali $\xrightarrow{\text{follows}}$ Health $\xrightarrow{\text{belongs to}^{-1}}$ swimwear $\xrightarrow{\text{belongs to}}$ life and living $\xrightarrow{\text{belongs to}^{-1}}$ How do I give up on life
Last.fm: Relevance	5	Sahar $\xrightarrow{\text{scrobble}}$ Earth $\xrightarrow{\text{sings}^{-1}}$ Sleeping at last $\xrightarrow{\text{belongs to}}$ indie rock $\xrightarrow{\text{belongs to}^{-1}}$ artist : Kodaline Sahar $\xrightarrow{\text{follows}}$ Sana $\xrightarrow{\text{scrobble}}$ Adventure of a lifetime $\xrightarrow{\text{sings}^{-1}}$ Coldplay $\xrightarrow{\text{belongs to}}$ alternative $\xrightarrow{\text{belongs to}^{-1}}$ artist : Kodaline
	6	Bahar $\xrightarrow{\text{follows}}$ Mojtaba $\xrightarrow{\text{scrobble}}$ Baribakh $\xrightarrow{\text{sings}^{-1}}$ Mansour $\xrightarrow{\text{belongs to}}$ Persian $\xrightarrow{\text{belongs to}^{-1}}$ artist : Ebi Bahar $\xrightarrow{\text{loves}}$ Twist in my sobriety $\xrightarrow{\text{belongs to}}$ pop $\xrightarrow{\text{belongs to}^{-1}}$ track : Pickack $\xrightarrow{\text{sings}}$ artist : Ebi
Last.fm: Surprisal	7	Elitsa $\xrightarrow{\text{follows}}$ Engkebert $\xrightarrow{\text{scrobble}}$ The prince $\xrightarrow{\text{belongs to}}$ metal $\xrightarrow{\text{belongs to}^{-1}}$ track : Rigger $\xrightarrow{\text{sings}^{-1}}$ artist : Inflames Elitsa $\xrightarrow{\text{loves}}$ Ohne dich $\xrightarrow{\text{contains}^{-1}}$ Reise, Reise $\xrightarrow{\text{belongs to}}$ metal $\xrightarrow{\text{belongs to}^{-1}}$ track : Acoustic piece $\xrightarrow{\text{sings}^{-1}}$ artist : Inflames
	8	Ali $\xrightarrow{\text{loves}}$ bang bang $\xrightarrow{\text{belongs to}}$ female vocalists $\xrightarrow{\text{belongs to}^{-1}}$ album : Dua Lipa (Deluxe) $\xrightarrow{\text{contains}}$ Dreams $\xrightarrow{\text{sings}^{-1}}$ Dua lipa Ali $\xrightarrow{\text{follows}}$ Anna $\xrightarrow{\text{scrobble}}$ bad girl friend $\xrightarrow{\text{belongs to}}$ pop $\xrightarrow{\text{belongs to}^{-1}}$ track : New rules $\xrightarrow{\text{sings}^{-1}}$ Dua lipa

Table 3.5: Anecdotal cases of path pairs with ids. FAIRY makes correct and wrong predictions for blue and red pairs, respectively.

predictions by FAIRY. In each pair, the first path denotes the preferred one by the user. The diversity of node and edge types in correct pairs shows the ability of FAIRY to learn underlying factors determining relevance or surprisal. The wrongly predicted pairs provide insights on the shortcomings of FAIRY. For example, in the surprisal model, we do not consider *sensitivity* of topics or items. Pair #4 is one such example where explanation paths reveal such sensitive items the users interacted with. Another limitation of FAIRY is that it does not incorporate background information about the users (such as their gender or nationality) which clearly affects their preferences. For example, on Last.fm, some users mentioned that presence of regional tags (such as Persian, Latin or Brazilian) influenced their relevance decisions (pair #6).

3.7 Related work

Social feeds and transparency. Personalizing social feeds has been the focus of many studies, as the amount of information generated by users' networks is overwhelming. To increase user engagement, models have been developed for finding relevant feed items for users by exploiting their past behavior [Freyne et al., 2010, Hong et al., 2012, Soh et al., 2013, Agarwal et al., 2015]. Users, however, are often unaware of the presence of such curation algorithms [Hamilton et al.,

2014, Eslami et al., 2015] as service providers generally do not provide insightful explanations. For example, Cotter et al. [Cotter et al., 2017] demonstrate inadequacy of explanations for feed ranking in Facebook. Therefore, it is imperative to have mechanisms for more transparency in social platforms.

Heterogeneous information networks and meta-paths. Due to the limitations of traditional graphs for capturing complex semantics with different types of entities and relations, heterogeneous information networks (HIN) were introduced to model multiple node and edge types [Sun and Han, 2012]. To analyze such HINs better, a meta-path was defined as the pattern of a path (sequence of node and edge types). Meta-paths have since been used in different applications such as similarity search [Sun et al., 2011, Seyler et al., 2018], relationship discovery [Fang et al., 2011, Kong et al., 2013, Behrens et al., 2018], link prediction [Sun et al., 2012, Zhang et al., 2014a, Dong et al., 2017, Zhang et al., 2018], and generating recommendations [Lee et al., 2013, Liu et al., 2014a, Hu et al., 2018] in HINs.

Relationship discovery in knowledge graphs. Finding interesting relationships among graph concepts is too broad an area to do justice in a short survey: however, mining such connections for knowledge graph entities is a more pertinent sub-problem that has been well-studied. This task is either (semi-)supervised, where users are asked for feedback [Behrens et al., 2018], or unsupervised, where heuristic measures of “interestingness” are applied to detect and rank relationships. However, user utility is probably multi-faceted: while we have explored relevance and surprisal, there are probably more, like coherence or complexity. These measures are normally approximated using topological properties of graphs, such as specificity and rarity of node/edge/path types [Ramakrishnan et al., 2005, Fang et al., 2011] or connectivity/reachability of nodes [Lao and Cohen, 2010, Seufert et al., 2016, Liang et al., 2016]. These scoring strategies, however, implicitly assume a static topology, and may not be useful for dynamic interaction graphs where nodes and edges are added and deleted with each timestep, and it is imperative that relationships should take temporal constraints into account.

3.8 Conclusion

We presented FAIRY, a smart user-centric framework that presents ranked explanations to users for items in their social feeds. Explanations are represented as relationship paths connecting the user’s own actions to the received feed items. FAIRY was trained and evaluated on data from two real user studies on the popular platforms of Quora and Last.fm. It outperformed three baselines on relationship mining on the task of modeling and predicting what users considered relevant and surprising explanations. The success of FAIRY hinges on two key aspects: a powerful heterogeneous information network representation of the user’s local neighborhood that can capture the complexity of current social media platforms, and, (ii) a fast learning-to-rank model that operates with intuitive and interpretable features that are easily accessible to the user.

FAIRY is the first step towards a general goal of improving transparency through the user's lens. Future directions for research would include, among others: (i) better modeling of temporal information in the interaction graph, (ii) further exploiting content features to build better models of entity similarity, and (iii) understanding effects of the user's activities across multiple connected platforms.

4

COUNTERFACTUAL EXPLANATIONS FOR RECOMMENDATIONS

Contents

4.1	Introduction	46
4.2	Computational model	48
4.3	The PRINCE algorithm	50
4.4	Correctness proof	51
4.5	Graph experiments	55
4.5.1	Setup	55
4.5.2	Results and insights	57
4.6	User study	59
4.7	Related work	61
4.8	Conclusion	62

In the previous chapter, we proposed a framework for discovering and ranking post-hoc explanations for social feeds. The resulting explanations, however, are decoupled from the recommendation model, failing to guarantee faithfulness to the model. Besides, the explanation paths may contain sensitive information about other users which could violate their privacy.

In this chapter, we address these limitations by presenting PRINCE: a provider-side mechanism to produce tangible explanations for end users, where an explanation is defined to be *a set of minimal actions performed by the user* that, if removed, changes the recommendation to a different item. We thus posit that PRINCE produces *faithful*, *action-based*, and *concise* explanations, owing to its use of *counterfactual* evidence, a user's *own actions*, and *minimal* sets, respectively.

In Sections 4.3 and 4.4, we present the PRINCE algorithm that finds minimal counterfactual explanations in recommenders based on personalized PageRank (described in Sections 4.2). Section 4.5 discusses experiments on two real-world datasets, showing that PRINCE provides more compact explanations than intuitive baselines. Section 4.6 presents insights from a crowdsourced user study, demonstrating the viability of such action-based explanations.

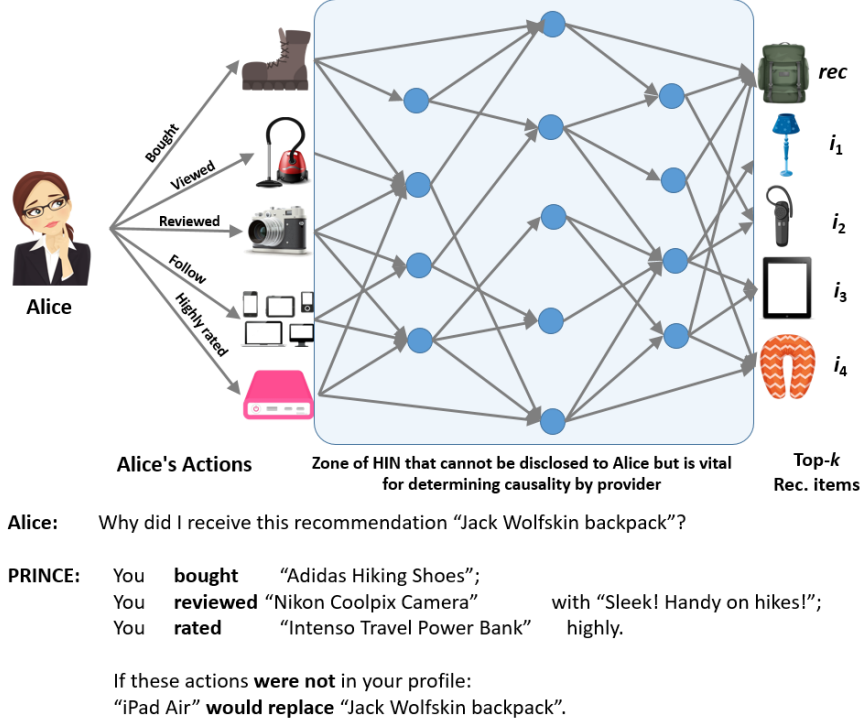


Figure 4.1: PRINCE generates explanations as a minimal set of actions using counterfactual evidence.

4.1 Introduction

Motivation. Explanations for recommenders can take several forms, depending on the generator (*explanations by whom?*) and the consumer (*explanations for whom?*). As generators, only *service providers* can produce true explanations for how systems compute the recommended items [Zhang et al., 2014b, Wang et al., 2018b, Balog et al., 2019]; *third parties* can merely discover relationships and create post-hoc rationalizations for black-box models that may look convincing to users [Peake and Wang, 2018, Wang et al., 2018c, Ghazimatin et al., 2019]. On the consumer side, *end users* can grasp tangible aspects like activities, likes/dislikes/ratings, or demographic factors. Unlike system developers or test engineers, end users would obtain hardly any insight from the transparency of internal system workings. In this chapter, we deal with explanations *by the provider and for the end user*.

Limitations of state-of-the-art. At the core of most recommender systems is some variant of matrix or tensor decomposition (e.g., [Koren et al., 2009]) or spectral graph analysis (e.g., [Jamali and Ester, 2009]), with various forms of regularization and often involving gradient-descent methods for parameter learning. One of the recent and popular paradigms is based on *heterogeneous information networks (HIN)* [Yu et al., 2013, Yu et al., 2014, Shi et al., 2017, Zhang et al., 2019], a powerful model that represents relevant entities and actions as a directed and weighted graph with multiple node and edge types. Prior efforts towards

explanations for HIN-based recommendations have mostly focused on *paths* that connect the user with the recommended item [Shi et al., 2015, Yang et al., 2018, Ai et al., 2018, Wang et al., 2018a, Ghazimatin et al., 2019, Wang et al., 2019b, Xian et al., 2019]. An application of path-based explanations, for an online shop, would be of the form:

User u received item rec because u follows user v , who bought item j , which has the same category as that of rec .

However, such methods come with critical privacy concerns arising from nodes in paths that disclose other users' actions or interests to user u , like the purchase of user v above. Even if user v 's id was anonymized, user u would know whom she is following and could often guess who user v actually is, that bought item j , assuming that u has a relatively small set of followees [Machanavajjhala et al., 2011]. If entire paths containing other users are suppressed instead, then such explanations would no longer be faithful to the true cause. Another family of path-based methods [Peake and Wang, 2018, Wang et al., 2018c, Ghazimatin et al., 2019] presents plausible connections between users and items as justifications. However, this is merely post-hoc rationalization, and not actual causality.

Approach. This chapter presents PRINCE, a method for Provider-side Interpretability with Counterfactual Evidence, that overcomes the outlined limitations. PRINCE is a provider-side solution aimed at detecting the actual cause responsible for the recommendation, in a heterogeneous information network with users, items, reviews, and categories. PRINCE's explanations are grounded in the user's own actions, and thus preclude privacy concerns of path-based models. Fig. 4.1 shows an illustrative example. Here, Alice's actions like bought shoes, reviewed a camera, and rated a power bank are deemed as explanations for her backpack recommendation. One way of identifying a user's actions for an explanation would be to compute scores of actions with regard to the recommended item. However, this would be an unwieldy distribution over potentially hundreds of actions – hardly comprehensible to an end user. Instead, we operate in a *counterfactual* setup [Martens and Provost, 2014]. PRINCE identifies a small (and actually minimal) set of a user's actions such that removing these actions would result in replacing the recommended item with a different item. In Fig. 4.1, the item $rec = \text{"Jack Wolfskin backpack"}$ would be replaced, as the system's top recommendation, by $i_3 = \text{"iPad Air"}$ (the i 's represent candidate replacement items). Note that there may be multiple such minimal sets, but uniqueness is not a concern here.

Another perspective here is that the goal of an explanation is often to show users *what they can do* in order to receive more relevant recommendations. Under this claim, the end user has no *control* on the network beyond her immediate neighborhood, i.e., the network beyond is *not actionable* (shaded zone in Fig. 4.1), motivating PRINCE's choice of grounding explanations in users' own actions.

For true explanations, we need to commit ourselves to a specific family of recommender models. For this, we choose a general framework based on *Personalized PageRank (PPR)*, as

used in the state-of-the-art RecWalk system [Nikolakopoulos and Karypis, 2019], and adapt it to the HIN setup. The heart of PRINCE is a polynomial-time algorithm for exploring the (potentially exponential) search space of subsets of user actions – the candidates for causing the recommendation. The algorithm efficiently computes PPR contributions for groups of actions with regard to an item, by adapting the reverse local push algorithm of [Andersen et al., 2007] to a dynamic graph setting [Zhang et al., 2016b]. In summary, the desiderata for the explanations from PRINCE (in **bold**) connect to the technical approaches adopted (in *italics*) in the following ways. Our explanations are:

- **Faithful**, as they are derived in a *counterfactual* setup;
- **Action-based**, as they are grounded in the user’s own *actions* (analogous to *item-based* explanations, see Table 2.2);
- **Concise**, as they are *minimal* sets changing a recommendation.

Extensive experiments with Amazon and Goodreads datasets show that PRINCE’s minimal explanations, achieving the desired item-replacement effect, cannot be easily obtained by heuristic methods based on contribution scores and shortest paths. A crowdsourced user study on Amazon Mechanical Turk (AMT) provides additional evidence that PRINCE’s explanations are more useful than ones based on paths [Yang et al., 2018].

The key contributions presented in this chapter are:

- PRINCE is the first work that explores counterfactual evidence for discovering causal explanations for recommendations in a heterogeneous information network;
- We present an optimal algorithm that explores the search space of action subsets in polynomial time, for efficient computation of a minimal subset of user actions;
- Experiments with two large datasets and a user study show that PRINCE can effectively aid a service provider in generating user-comprehensible causal explanations for recommended items.

4.2 Computational model

Heterogeneous Information Networks (HIN). A Heterogeneous Information Network (or HIN) is a graph $G = (N, E, M_N^T, M_E^T)$ that consists of a set of nodes N , a set of edges $E \subseteq N \times N$, and mappings M^T from each node and each edge to their types, such that $M_N^T : N \rightarrow T_N$ and $M_E^T : E \rightarrow T_E$ with $|T_N| + |T_E| > 2$ [Shi et al., 2017].

In our problem, a heterogeneous information network (also referred to as a heterogeneous graph) contains at least two node types, users $U \in T_N$ and items $I \in T_N$. For simplicity, we use the notations U and I to refer both to the type of a node and the set of all nodes of that type. A HIN can be *weighted* if there is a weight assigned to each edge, $M_E^W : E \mapsto \mathbb{R}_{\geq 0}$, and is *directed* if E is a set of ordered pairs of nodes. We denote with $N_{out}(n)$ and $N_{in}(n)$ the sets of

out-neighbors and in-neighbors of node n , respectively.

Personalized PageRank (PPR) for recommenders. We use Personalized PageRank (PPR) for recommendation in HINs [Haveliwala, 2003, Nikolakopoulos and Karypis, 2019, Musto et al., 2021]. PPR is the stationary distribution of a random walk in G in which, at a given step, with probability α , a surfer teleports to a set of seed nodes $\{s\}$, and with probability $1 - \alpha$, continues the walk to a randomly chosen outgoing edge from the current node. More precisely, given G , teleportation probability α , a single seed s , the one-hot vector e_s , and the transition matrix W , the Personalized PageRank vector $PPR(s)$ is defined recursively as:

$$PPR(s, \cdot) = \alpha e_s + (1 - \alpha) PPR(s, \cdot) W \quad (4.1)$$

Let $PPR(s, n)$ be the PPR score of node n personalized for s . We define the *PPR recommendation* for user $u \in U$, or the top-1 recommendation, as:

$$rec = \arg \max_{i \in I \setminus N_{out}(u)} PPR(u, i) \quad (4.2)$$

Given a set of u 's actions $A \subset E$ modeled as outgoing edges of u , we use the notation $PPR(u, i|A)$ to define the PPR of an item i personalized for a user u in the graph $G = (N, E \setminus A, M_N^T, M_E^T, M_E^W)$. We refer to this graph as $G \setminus A$. To improve top- n recommendations, Nikolakopoulos and Karypis [Nikolakopoulos and Karypis, 2019] define a random walk in an HIN G as follows:

- With probability α , the surfer teleports to u
- With probability $1 - \alpha$, the surfer continues the walk in the following manner:
 - With probability $1 - \beta$, the random surfer moves to a *node of the same type*, using a similarity-based stochastic transition matrix
 - With probability β , the surfer chooses any outgoing edge at random.

For each node type t in T_V , there is an associated stochastic similarity matrix S_t , which encodes the relationship between the nodes of type t . When nodes of the same type are not comparable, the similarity matrix is the identity matrix, i.e. $S_t = I$. Otherwise, an entry (i, j) in S_t corresponds to the similarity between node i and node j . The stochastic process described by this walk is a nearly uncoupled Markov chain [Nikolakopoulos and Karypis, 2019]. The stationary distribution of the random walk is the PPR with teleportation probability α in a graph G^β whose transition probability matrix is [Nikolakopoulos and Karypis, 2019]:

$$W^\beta = \beta W + (1 - \beta) S \quad (4.3)$$

The matrix W is the transition probability matrix of the original graph G . Matrix $S = \text{Diag}(S_1, S_2, \dots, S_{|T_V|})$ is a diagonal matrix of order $|V|$.

Counterfactual explanations. A user u interacts with items via different types of *actions* A ,

such as clicks, purchases, ratings or reviews, which are captured as *interaction edges* in the graph G . Our goal is to present user u with a set of interaction edges $A^* \subseteq \{(u, n_i) | (u, n_i) \in A\}$ (where n_i is a neighbor of u) responsible for an item recommendation rec ; we refer to this as a *counterfactual explanation*. An explanation is *counterfactual*, if after removing the edges A^* from the graph, the user receives a different top-ranked recommendation rec^* . A counterfactual explanation A^* is *minimal* if there is no smaller set $A' \subseteq A$ such that $|A'| < |A^*|$ and A' is also a counterfactual explanation for rec .

Formal problem statement. Given a HIN $G = (V, E, M_N^T, M_E^T, M_E^W)$ and the top-ranked recommendation $rec \in I$ for user $u \in U$, find a minimum counterfactual explanation for the item rec .

4.3 The PRINCE algorithm

In this section, we develop an algorithm for computing a minimum counterfactual explanation for user u receiving recommended item rec , given the PPR-based recommender framework *RecWalk* [Nikolakopoulos and Karypis, 2019]. A naive optimal algorithm enumerates all subsets of actions $A^* \subseteq A$, and checks whether the removal of each of these subsets replaces rec with a different item as the top recommendation, and finally selects the subset with the minimum size. This approach is exponential in the number of actions of the user.

To devise a more efficient and practically viable algorithm, we express the *PPR* scores as follows [Jeh and Widom, 2003], with $PPR(u, rec)$ denoting the PPR of rec personalized for u (i.e., the random walker can jump back only to u):

$$PPR(u, rec) = (1 - \alpha) \sum_{n_i \in N_{out}(u)} W(u, n_i) PPR(n_i, rec) + \alpha \delta_{u, rec} \quad (4.4)$$

where α denotes the teleportation probability (probability of jumping back to u) and δ is the Kronecker delta function. The only required modification, with regard to *RecWalk* [Nikolakopoulos and Karypis, 2019], is the transformation of the transition probability matrix from W to W^β . For simplicity, we will refer to the adjusted probability matrix as W .

Eq. 4.4 shows that the PPR of rec personalized for user u , $PPR(u, rec)$, is a function of the PPR values of rec personalized for the neighbors of u . Hence, in order to decrease $PPR(u, rec)$, we can remove edges $(u, n_i), n_i \in N_{out}(u)$. To replace the recommendation rec with a different item rec^* , a simple heuristic would remove edges (u, n_i) in decreasing order of their contributions $W(u, n_i) \cdot PPR(n_i, rec)$. However, although this would reduce the PPR of rec , it also affects and possibly reduces the PPR of other items, too, due to the recursive nature of PPR, where all paths matter.

Let A be the set of outgoing edges of a user u and let A^* be a subset of A , such that $A^* \subseteq A$. The main intuition behind our algorithm is that we can express $PPR(u, rec)$ after the removal

of A^* , denoted by $PPR(u, rec|A^*)$, as a function of two components: $PPR(u, u|A^*)$ and the values $PPR(n_i, rec|A)$, where $n_i \in \{n_i | (u, n_i) \in A \setminus A^*\}$ and $n_i \neq u$. The score $PPR(u, u|A^*)$ does not depend on rec , and the score $PPR(n_i, rec|A)$ is independent of A^* .

Based on these considerations, we present Algorithm 1, proving its correctness in Sec. 4.4. Algorithm 1 takes as input a graph G , a user u , a recommendation rec , and a set of items I . In lines 3-9, we iterate through the items I , and find the minimum counterfactual explanation A^* . Here, A^i refers to the actions whose removal swaps the orders of items rec and i . In addition, we ensure that after removing A^* , we return the item with the highest PPR score as the replacement item (lines 10-14). Note that in the next section, we propose an equivalent formulation for the condition $PPR(u, i|A^i) > PPR(u, rec|A^i)$, eliminating the need for recomputing scores in $G \setminus A^*$.

The core of our algorithm is the function `SwapOrder`, which receives as input two items, rec and rec^* , and a user u . In lines 21-25, we sort the interaction edges $(u, n_i) \in A$ in decreasing order of their contributions $W(u, n_i) \cdot (PPR(n_i, rec|A) - PPR(n_i, rec^*|A))$. In lines 26-30, we remove at each step, the outgoing interaction edge with the highest contribution, and update variables sum and A^* correspondingly. The variable sum is strictly positive if in the current graph configuration $(G \setminus A^*)$, $PPR(u, rec) > PPR(u, rec^*)$. This constitutes the main building block of our approach. Fig. 4.2 illustrates the execution of Algorithm 1 on a toy example.

The time complexity of the algorithm is $O(|I| \times |A| \times \log |A|)$, plus the cost of computing PPR for these nodes. The key to avoiding the exponential cost of considering all subsets of A is the insight that *we need only to compute PPR values for alternative items with personalization based on a graph where the set of all user actions A is removed*. In other words, instead of iterating over the $2^{|A|}$ possible subsets of the user's actions, we call the `SwapOrder` function $|I|$ times. This is feasible because the action deletions affect only outgoing edges of the teleportation target u , as elaborated in Sec. 4.4.

The PPR computation could simply re-run a power-iteration algorithm for the entire graph, or compute the principal eigenvector for the underlying matrix. This could be cubic in the graph size (e.g., if we use full-fledged SVD), but it keeps us in the regime of polynomial runtimes. In our experiments, we use the much more efficient reverse local push algorithm [Zhang et al., 2016b] for PPR calculations.

4.4 Correctness proof

We prove two main results:

- (i) $PPR(u, rec|A^*)$ can be computed as a product of two components where one depends on the modified graph with the edge set $E \setminus A$ (i.e., removing all user actions) and the other depends on the choice of A^* but not on the choice of rec .

Algorithm 1: PRINCE

Input: $G = (N, E, M_N^T, M_E^T, M_E^W)$, $I \subset N$, $u \in N$, $rec \in I$
Output: A^* and rec^* for (u, rec)

```

1  $A^* \leftarrow A$ 
2  $rec^* \leftarrow rec$ 
3 foreach  $i \in I$  do
4    $A^i \leftarrow \text{SwapOrder}(G, u, rec, i)$ 
   // Actions  $A^i$  swap orders of  $rec$  and  $i$ 
5   if  $|A^i| < |A^*|$  then
6      $A^* \leftarrow A^i$ 
7      $rec^* \leftarrow i$ 
8   end
9 end
   // Finding the correct replacement item.
10 foreach  $i \in I$  do
11   if  $PPR(u, i|A^*) > PPR(u, rec^*|A^*)$  then
12      $rec^* \leftarrow i$ 
13   end
14 end
15 return  $A^*, rec^*$ 

16 Function  $\text{SwapOrder}(G, u, rec, rec^*)$  :
17    $A \leftarrow \{(u, n_i) | n_i \in N_{out}(u), n_i \neq u\}$ 
18    $A^* \leftarrow \emptyset$ 
19    $H \leftarrow \text{MaxHeap}(\phi)$ 
20    $sum \leftarrow 0$ 
21   foreach  $(u, n_i) \in A$  do
22      $diff \leftarrow W(u, n_i) \cdot (PPR(n_i, rec|A) - PPR(n_i, rec^*|A))$ 
23      $H.insert(n_i, diff)$ 
24      $sum \leftarrow sum + diff$ 
25   end
26   while  $sum \geq 0$  and  $|H| > 0$  do
27      $(n_i, diff) \leftarrow H.delete\_max()$ 
28      $sum \leftarrow sum - diff$ 
29      $A^* \leftarrow A^* \cup (u, n_i)$ 
     //  $(u, n_i)$  contributes the most to  $rec$  and the least to  $rec^*$ .
30   end
31   if  $sum > 0$  then  $A^* \leftarrow A$ 
32   return  $A^*$ 
33 end

```

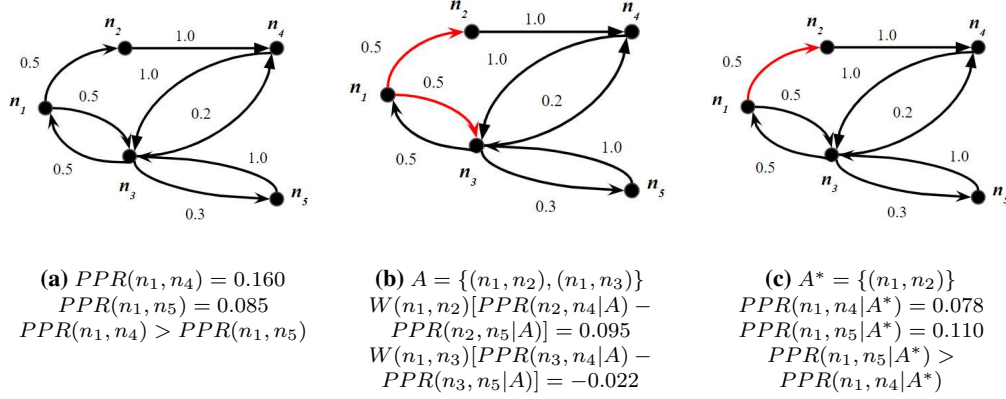


Figure 4.2: Toy Example. (a) A weighted and directed graph where the PPR scores are personalized for node n_1 . Node n_4 has higher PPR than n_5 . (b) Scores in a graph configuration where outgoing edges (n_1, n_2) , and (n_1, n_3) are removed (marked in red). (c) Removing (n_1, n_2) causes n_5 to outrank n_4 .

(ii) To determine if some A^* replaces the top node rec with a different node rec^* which is not an out-neighbor of u , we need to compute only the first of the two components in (i).

Theorem 1. Given a graph $G = (V, E)$, a node u with outgoing edges A such that $(u, u) \notin A$, a set of edges $A^* \subset A$, a node $rec \notin N_{out}(u)$, the PPR of rec personalized for u in the modified graph $G^* = (V, E \setminus A^*)$ can be expressed as follows:

$$PPR(u, rec|A^*) = PPR(u, u|A^*) \cdot f(\{PPR(n_i, rec|A) \mid (u, n_i) \in A \setminus A^*\})$$

where $f(\cdot)$ is an aggregation function.

Proof. Assuming that each node has at least one outgoing edge, the PPR can be expressed as the sum over the probabilities of walks of length l starting at a node u [Andersen et al., 2006]:

$$PPR(u, \cdot) = \alpha \sum_{l=0}^{\infty} (1 - \alpha)^l e_u W^l \quad (4.5)$$

where e_u is the one-hot vector for u . To analyze the effect of deleting A^* , we split the walks from u to rec into two parts, (i) the part representing the sum over probabilities of walks that start at u and pass again by u , which is equivalent to $\alpha^{-1} PPR(u, u|A^*)$ (division by α is required as the walk does not stop at u), and (ii) the part representing the sum over probabilities of walks starting at node u and ending at rec without revisiting u again, denoted by $p_{-u}(u, rec|A^*)$. Combining these constituent parts, PPR can be stated as follows:

$$PPR(u, rec|A^*) = \alpha^{-1} PPR(u, u|A^*) \cdot p_{-u}(u, rec|A^*) \quad (4.6)$$

As stated previously, $p_{-u}(u, rec|A^*)$ represents the sum over the probabilities of the walks from u to rec without revisiting u . We can express these walks using the remaining neighbors

of u after removing A^* :

$$p_{-u}(u, rec|A^*) = (1 - \alpha) \sum_{(u, n_i) \in A \setminus A^*} W(u, n_i) \cdot p_{-u}(n_i, rec|A^*) \quad (4.7)$$

where $p_{-u}(n_i, rec|A^*)$ refers to the walks starting at n_i ($n_i \neq u$) and ending at rec that do not visit u . We replace $p_{-u}(n_i, rec|A^*)$ with its equivalent formulation $PPR(n_i, rec|A)$. $PPR(n_i, rec)$ in graph $G \setminus A$ is computed as the sum over the probabilities of walks that never pass by u . Eq. 4.6 can be rewritten as follows:

$$\begin{aligned} & PPR(u, rec|A^*) \\ &= PPR(u, u|A^*) \cdot \alpha^{-1}(1 - \alpha) \sum_{(u, n_i) \in A \setminus A^*} W(u, n_i|A^*) PPR(n_i, rec|A) \end{aligned} \quad (4.8)$$

This equation directly implies:

$$PPR(u, rec|A^*) = PPR(u, u|A^*) \cdot f(\{PPR(n_i, rec|A) \mid (u, n_i) \in A \setminus A^*\}) \quad (4.9)$$

□

Theorem 2. The minimum counterfactual explanation for (u, rec) can be computed in polynomial time.

Proof. We show that there exists a polynomial-time algorithm for finding the minimum set $A^* \subset A$ such that $PPR(u, rec|A^*) < PPR(u, rec^*|A^*)$, if such a set exists. Using Theorem 1, we show that one can compute if some rec^* can replace the original rec as the top recommendation, solely based on PPR scores from a single graph where the set of all user actions A is removed:

$$\begin{aligned} & PPR(u, rec|A^*) < PPR(u, rec^*|A^*) \\ \Leftrightarrow & \sum_{(u, n_i) \in A \setminus A^*} W(u, n_i|A^*) (PPR(n_i, rec|A) - PPR(n_i, rec^*|A)) < 0 \\ \Leftrightarrow & \sum_{(u, n_i) \in A \setminus A^*} W(u, n_i) (PPR(n_i, rec|A) - PPR(n_i, rec^*|A)) < 0 \end{aligned} \quad (4.10)$$

The last equivalence is derived from:

$$W(u, n_i|A^*) = \frac{W(u, n_i)}{1 - \sum_{(u, n_j) \in A^*} W(u, n_j)} \quad (4.11)$$

For a fixed choice of rec^* , the summands in expression 4.10 do not depend on A^* , and so they are constants for all possible choices of A^* . Therefore, by sorting the summands in descending order, we can greedily expand A^* from a single action to many actions until some rec^* outranks rec . This approach is then guaranteed to arrive at a minimum subset.

□

4.5 Graph experiments

We now describe experiments performed with graph-based recommenders built from real datasets to evaluate PRINCE.

4.5.1 Setup

Datasets. We used two real datasets:

- (i) The Amazon Customer Review dataset (released by Amazon¹), and,
- (ii) The Goodreads review dataset (crawled by the authors of [Wan and McAuley, 2018]²).

Each record in both datasets consists of a user, an item, its categories, a review, and a rating value (on a 1 – 5 scale). In addition, a Goodreads data record has the book author(s) and the book description. We augmented the Goodreads collection with social links (users following users) that we crawled from the Goodreads website.

The high diversity of categories in the Amazon data, ranging from household equipment to food and toys, allows scope to examine the interplay of cross-category information within explanations. The key reason for additionally choosing Goodreads is to include the effect of social connections (absent in the Amazon data). The datasets were converted to graphs with “users”, “items”, “categories”, and “reviews” as *nodes*, and “rated” (user-item), “reviewed” (user-item), “has-review” (item-review), “belongs-to” (item-category) and “follows” (user-user) as *edges*. In Goodreads, there is an additional node type “author” and an edge type “has-author” (item-author). All the edges, except the ones with type “follows”, are bidirectional. Only ratings with value *higher than three* were considered, as low-rated items should not influence further recommendations.

Dataset	#Users	#Items	#Reviews	#Categories	#Actions
Amazon	2k	54k	58k	43	114k
Goodreads	1k	17k	20k	16	45k

Table 4.1: Properties of the Amazon and Goodreads samples.

Sampling. For our experiments, we sampled 500 seed users who had between 10 and 100 actions, from both Amazon and Goodreads datasets. The filters served to prune out both under-active and power users (potentially bots). Interaction graphs were constructed for the sampled users by taking their four-hop neighborhood from the sampled data (Table 4.1). Four is a reasonably small radius to keep the items relevant and personalized to the seed users.

¹s3.amazonaws.com/amazon-reviews-pds/readme.html

²sites.google.com/eng.ucsd.edu/ucsdbookgraph/home

The graphs were augmented with weighted edges to capture node-node similarities. For Amazon, we added review-review edges where weights were computed using the cosine similarity of the review embeddings, generated with Google’s Universal Sentence Encoder [Cer et al., 2018], with a cut-off threshold $\tau = 0.85$ to retain only confident pairs. This resulted in 194 review-review edges. For Goodreads, we added three types of similarity edges: category-category, book-book and review-review, with the same similarity measure (24 category-category, 113 book-book, and 1003 review-review edges). Corresponding thresholds were 0.67, 0.85 and 0.95. We crawled category descriptions from the Goodreads’ website and used book descriptions and review texts from the raw data. For efficient computation of similarity values, we used locality-sensitive hashing (LSH) with random binary projections. Table 4.1 gives some statistics about the sampled datasets.

Initialization. The replacement item for rec is always chosen from the original top- k recommendations generated by the system; we systematically investigate the effect of k on the size of explanations in our experiments (with a default $k = 5$). PRINCE does not need to be restricted to an explicitly specified candidate set, and can actually operate over the full space of items I . In practice, however, replacement items need to be guided by some measure of relevance to the user, or item-item similarity, so as not to produce degenerate or trivial explanations if rec is replaced by some arbitrary item from a pool of thousands. Besides, with the focus of state-of-the-art recommenders on diversification of top- k recommendations, we can make sure that the replacement item is not too close to the original top-ranked recommendations.

We use the standard teleportation probability $\alpha = 0.15$ [Brin and Page, 1998]. The parameter β is set to 0.5. To compute PPR scores, we used the reverse local push method [Zhang et al., 2016b] with $\varepsilon = 1.7e - 08$ for Amazon and $\varepsilon = 2.7e - 08$ for Goodreads. With these settings, PRINCE and the baselines were executed on the constructed HINs to compute an alternative recommendation (i.e., replacement item) rec^* and a counterfactual explanation set A^* .

Baselines. Since PRINCE is an optimal algorithm with correctness guarantees, it always finds minimal sets of actions that replace rec (if they exist). We wanted to investigate, to what extent other, more heuristic, methods approximate the same effects. To this end, we compared PRINCE against two natural baselines:

- (i) *Highest Contributions (HC)*: This is analogous to counterfactual evidence in feature-based classifiers for structured data [Moeyersoms et al., 2016, Chen et al., 2017]. It defines the contribution score of a user action (u, n_i) to the recommendation score $PPR(u, rec)$ as $PPR(n_i, rec)$ (Eq. 4.4), and iteratively deletes edges with highest contributions until the highest-ranked rec changes to a different item.
- (ii) *Shortest Paths (SP)*: SP computes the shortest path from u to rec and deletes the first edge (u, n_i) on this path. This step is repeated on the modified graph, until the top-ranked rec changes to a different item.

Evaluation metric. The metric for assessing the quality of an explanation is its size, that is, the number of actions in A^* for PRINCE, and the number of edges deleted in HC and SP .

4.5.2 Results and insights

We present our main results in Table 4.2 and discuss insights below. These comparisons were performed for different values of the parameter k . Wherever applicable, statistical significance was tested under the 1-tailed paired t -test at $p < 0.05$.

Anecdotal examples of explanations by PRINCE and the baselines are given in Table 4.4. In the Amazon example, we observe that our method produces a topically coherent explanation, with both the recommendation and the explanation items in the same category. The SP and HC methods give larger explanations, but with poorer quality, as the first action in both methods seems unrelated to the recommendation. In the Goodreads example, both HC and SP yield the same replacement item, which is different from that of PRINCE.

k	Amazon			Goodreads		
	PRINCE	HC	SP	PRINCE	HC	SP
3	5.09*	6.87	7.57	2.05*	2.86	5.38
5	3.41*	4.62	5.01	1.66*	2.19	4.37
10	2.66*	3.66	4.15	1.43	1.45	3.28
15	2.13*	3.00	3.68	1.11	1.12	2.90
20	1.80*	2.39	3.28	1.11	1.12	2.90

Table 4.2: Average sizes of counterfactual explanations. The best value per row in a dataset is in **bold**. An asterisk (*) indicates statistical significance of PRINCE over the closest baseline, under the 1-tailed paired t -test at $p < 0.05$.

Parameter	Amazon		Goodreads	
	Pre-computed	Dynamic	Pre-computed	Dynamic
$k = 3$	0.3ms	39.1s	0.3ms	24.1s
$k = 5$	0.6ms	60.4s	0.4ms	34.7s
$k = 10$	1.3ms	121.6s	0.9ms	60.7s
$k = 15$	2.0ms	169.3s	1.5ms	91.6s
$k = 20$	2.6ms	224.4s	2ms	118.8s
$\beta = 0.01$	0.4ms	1.1s	0.3ms	2.9s
$\beta = 0.1$	0.5ms	15.5s	0.3ms	8.9s
$\beta = 0.3$	0.5ms	17.0s	0.4ms	12.5s
$\beta = 0.5$	0.6ms	60.5s	0.4ms	34.7s

Table 4.3: Average runtime of PRINCE, when the scores are pre-computed (**Pre-computed**) and when the scores are dynamically computed using the reverse push algorithm [Zhang et al., 2016b] (**Dynamic**).

Method	Explanation for “Baby stroller” with category “Baby” [Amazon]
PRINCE	Action 1: You rated highly “Badger Basket Storage Cubby” with category “Baby” Replacement Item: “Google Chromecast HDMI Streaming Media Player” with categories “Home Entertainment”
HC	Action 1: You rated highly “Men’s hair paste” with category “Beauty” Action 2: You reviewed “Men’s hair paste” with category “Beauty” with text “Good product. Great price.” Action 3: You rated highly “Badger Basket Storage Cubby” with category “Baby” Action 4: You rated highly “Straw bottle” with category “Baby” Action 5: You rated highly “3 Sprouts Storage Caddy” with category “Baby” Replacement Item: “Bathtub Waste And Overflow Plate” with categories “Home Improvement”
SP	Action 1: You rated highly “Men’s hair paste” with category “Beauty” Action 2: You rated highly “Badger Basket Storage Cubby” with category “Baby” Action 3: You rated highly “Straw bottle” with category “Baby” Action 4: You rated highly “3 Sprouts Storage Caddy” with category “Baby” Replacement Item: “Google Chromecast HDMI Streaming Media Player” with categories “Home Entertainment”
Method	Explanation for “The Multiversity” with categories “Comics, Historical-fiction, Biography, Mystery” [Goodreads]
PRINCE	Action 1: You rated highly “Blackest Night” with categories “Comics, Fantasy, Mystery, Thriller” Action 2: You rated highly “Green Lantern” with categories “Comics, Fantasy, Children” Replacement item: “True Patriot: Heroes of the Great White North” with categories “Comics, Fiction”
HC	Action 1: You follow User ID x Action 2: You rated highly “Blackest Night” with categories “Comics, Fantasy, Mystery, Thriller” Action 3: You rated highly “Green Lantern” with categories “Comics, Fantasy, Children” Replacement item: “The Lovecraft Anthology: Volume 2” with categories “Comics, Crime, Fiction”
SP	Action 1: You follow User ID x Action 2: You rated highly “Fahrenheit 451” with categories “Fantasy, Young-adult, Fiction” Action 3: You rated highly “Darkly Dreaming Dexter (Dexter, #1)” with categories “Mystery, Crime, Fantasy” And 6 more actions Replacement item: “The Lovecraft Anthology: Volume 2” with categories “Comics, Crime, Fiction”

Table 4.4: Anecdotal examples of explanations by PRINCE and the counterfactual baselines.

Approximating PRINCE is difficult. Explanations generated by PRINCE are more concise and hence more user-comprehensible than those by the baselines. This advantage is quite pronounced; for example, in Amazon, all the baselines yield at least one more action in the explanation set on average. Note that this translates into unnecessary effort for users who want to act upon the explanations.

Explanations shrink with increasing k . The size of explanations shrinks as the top- k candidate set for choosing the replacement item is expanded. For example, the explanation size for PRINCE on Amazon drops from 5.09 at $k = 3$ to 1.80 at $k = 20$. This is due to the fact that with a growing candidate set, it becomes easier to find an item that can outrank rec .

PRINCE is efficient. To generate a counterfactual explanation, PRINCE only relies on the scores in the graph configuration $G \setminus A$ (where all the outgoing edges of u are deleted). Pre-computing $PPR(n_i, rec|A)$ (for all $n_i \in \mathcal{N}_{out}(u)$), PRINCE could find the explanation for each $(user, rec)$ pair in about 1 millisecond on average (for $k \leq 20$). Table 4.3 shows runtimes of PRINCE for different parameters. As we can see, the runtime grows linearly with k in both datasets. This is justified by Line 3 in Algorithm 1. Computing $PPR(n_i, rec|A)$ on-the-fly slows down the algorithm. The second and the fourth columns in Table 4.3 present the runtimes of PRINCE when the scores $PPR(n_i, rec|A)$ are computed using the reverse push algorithm for dynamic graphs [Zhang et al., 2016b]. Increasing β makes the computation slower (experimented at $k = 5$). All experiments were performed on an Intel Xeon server with 8 cores@3.2 GHz CPU and 512 GB main memory.

4.6 User study

Quantitative measurement of usefulness. We conducted a user study³ to compare PRINCE to a path-based explanation [Yang et al., 2018] (later referred to as CredPaths). We used the *credibility* measure from [Yang et al., 2018], scoring paths in descending order of the product of their edge weights. We refer to this method as CredPaths. We computed the best path for all 500 user-item pairs (Sec. 4.5.1). This resulted in paths of a maximum length of three edges (four nodes including user and rec). For a fair comparison in terms of cognitive load, we eliminated all data points where PRINCE computed larger counterfactual sets. This resulted in about 200 user-item pairs, from where we sampled exactly 200. As explanations generated by PRINCE and CredPaths have a different format of presentation (a list of actions versus a path), we evaluated each method separately to avoid presentation bias. For the sake of readability, we broke the paths into edges and showed each edge on a new line. Only AMT Master workers⁴ were allowed to provide assessments. These workers are granted with Masters qualification as they have consistently demonstrated a high degree of success in performing a wide range of

³The user study was conducted only on Amazon data for resource constraints.

⁴<https://www.mturk.com/worker/help>

HITs (Human Intelligence Tasks, a unit of job on AMT). Having three AMT Masters for each task, we collected 600 (200×3) annotations for PRINCE and the same number for CredPaths.

A typical data point looks like a row in Table 4.6, that shows representative examples (Goodreads shown only for completeness). We divided the samples into ten HITs with 20 data points in each HIT. For each data point, we showed a recommendation item and its explanation, and asked users about the usefulness of the explanation on a scale of 1 – 3 (“Not useful at all”, “Partially useful”, and “Completely useful”). For this, workers had to imagine that they were a user of an e-commerce platform who received the recommendations as result of doing some actions on the platform. For PRINCE explanations, we provided the following guideline to the workers:

Imagine that you are a customer of an e-commerce platform and have received a certain recommendation. In this task, we provide you with explanations for the above recommendation. These explanations are in terms of actions (such as rated or reviewed certain items) that you did on the platform. Based on this information, you have to rate explanations for 20 items according to their usefulness: (how satisfied are you with these reasons?). Also, please write a short summary for each item justifying your choices (at least one sentence).

We used a similar guideline for explanations of CredPaths, except that the blue lines were replaced by:

Each explanation consists of the best connecting path between you and the recommended item. In other words, the explanations reveal how you are related to the recommendation in terms of other users, items, and categories.

To detect spammers, we planted one honeypot in each of the 10 HITs, that was a completely impertinent explanation. Subsequently, all annotations of detected spammers (workers who rated such irrelevant explanations as “completely useful”) were removed ($\simeq 25\%$ of all annotations).

Method	Mean	Std. Dev.	#Samples
PRINCE	1.91*	0.66	200
CredPaths [Yang et al., 2018]	1.78	0.63	200
PRINCE (Size=1)	1.87	0.66	154
PRINCE (Size=2)	1.88*	0.70	28
PRINCE (Size=3)	2.21*	0.52	18

Table 4.5: Results from the AMT measurement study on usefulness conducted on the Amazon data. An asterisk (*) indicates statistical significance of PRINCE over CredPaths (1-tailed paired t -test at $p < 0.05$).

Table 4.5 shows the results of our user study. It gives average scores and standard deviations, and it indicates statistical significance of pairwise comparisons with an asterisk. PRINCE clearly obtains higher usefulness ratings from the AMT judges, on average. Krippendorff’s alpha [Krippendorff, 2018] for PRINCE and CredPaths were found to be $\simeq 0.5$ and $\simeq 0.3$ respectively, showing moderate to fair inter-annotator agreement. The superiority of PRINCE

Method	Explanation for “Baby stroller” with category “Baby” [Amazon]
PRINCE	Action 1: You rated highly “Badger Basket Storage Cubby” with category “Baby”
CredPaths	You rated highly “Men’s hair paste” with category “Beauty” that was rated by “Some user” who also rated highly “Baby stroller” with category “Baby”
Method	Explanation for “The Multiversity” with categories “Comics, Historical-fiction, Biography, Mystery” [Goodreads]
PRINCE	Action 1: You rated highly “Blackest Night” with categories “Comics, Fantasy, Mystery, Thriller” Action 2: You rated highly “Green Lantern” with categories “Comics, Fantasy, Children”
CredPaths	You follow “Some user” who has rated highly “The Multiversity” with categories “Comics, Historical-fiction, Biography, Mystery”

Table 4.6: Explanations from PRINCE vis-à-vis CredPaths [Yang et al., 2018].

Based on multiple actions explained simply and clearly. [PRINCE]
The recommendation is for a home plumbing item, but the action rated a glue. [PRINCE]
The explanation is complete as it goes into full details of how to use the product, which is in alignment of my review and useful to me. [CredPaths]
It’s weird to be given recommendations based on other people. [CredPaths]

Table 4.7: Turkers’ comments on their score justifications.

also holds for slices of samples where PRINCE generated explanations of size 1, 2 and 3. We also asked Turkers to provide *succinct justifications* for their scores on each data point. Table 4.7 shows some typical comments, where methods for generating explanations are in brackets.

4.7 Related work

Provider-side explainability. Generating explanations has become tightly coupled with building systems that are geared for producing more transparent recommendations (like [Balog et al., 2019]). For broad surveys, see Section 2.2.6 in Chapter 2. Recently, interpretable neural models have become popular, especially for text [Seo et al., 2017, Chen et al., 2018a, Chen et al., 2018b] and images [Chen et al., 2019b], where the attention mechanism over words, reviews, items, or zones in images has been vital for interpretability. The faithfulness of attention scores, however, has been recently brought into question [Jain and Wallace, 2019] as they may fail to identify the *important* features for prediction. Besides, existing works on provider-side interpretability often lack enough evidence on causality of their explanations, and hence have limited scrutability.

In this chapter, we addressed these issues by presenting a method for finding *counterfactual* explanations.

Path-based explanations. Representing users, items, categories and reviews as a knowledge graph or a heterogeneous information network (HIN) has become popular, where explanations take the form of paths between the user and an item. This paradigm comprises a variety of mechanisms: learning path embeddings [Ai et al., 2018, Wang et al., 2018b], propagating user preferences [Wang et al., 2018a], learning and reasoning with explainable rules [Ma et al., 2019, Xian et al., 2019], and ranking user-item connections [Yang et al., 2018, Ghazimatin et al., 2019]. In this chapter, we choose the recent approach in [Yang et al., 2018] as a representative for the family of methods generating path-based explanations to compare PRINCE with. The explanations generated by this method are post-hoc. Other approaches for generating model-agnostic rationalizations include association rule mining [Peake and Wang, 2018], supervised ranking of user-item relationships [Ghazimatin et al., 2019], and reinforcement learning [Wang et al., 2018c].

Personalized PageRank. Random walks over HIN’s have been pursued by a suite of works, including [Desrosiers and Karypis, 2011, Cooper et al., 2014, Christoffel et al., 2015, Eksombatchai et al., 2018, Jiang et al., 2018]. In a nutshell, the Personalized PageRank (PPR) of an item node in the HIN is used as a ranking criterion for recommendations. [Nikolakopoulos and Karypis, 2019] introduced the RecWalk method, proposing a random walk with a nearly uncoupled Markov chain. Our proposed method PRINCE uses this framework. As far as we know, we are the first to study the problem of computing minimum subsets of edge removals (user actions) to change the top-ranked node in a counterfactual setup. Prior research on dynamic graphs, such as [Csáji et al., 2014, Kang et al., 2018a], has addressed related issues, but not this very problem. A separate line of research focuses on the efficient computation of PPR. Approximate algorithms include power iteration [Page et al., 1999], local push [Andersen et al., 2006, Andersen et al., 2007, Zhang et al., 2016b] and Monte Carlo methods [Avrachenkov et al., 2007, Bahmani et al., 2010].

4.8 Conclusion

This chapter explored a new paradigm of action-based explanations in graph recommenders, with the goal of identifying minimum sets of user actions with the counterfactual property that their absence would change the top-ranked recommendation to a different item. In contrast to prior works on (largely path-based) recommender explanations, this approach offers two advantages: (i) explanations are *concise*, *faithful*, and *action-based*, as they are *minimal* sets derived using a *counterfactual* setup over a user’s *own* purchases, ratings and reviews; and (ii) explanations do not expose any information about other users, thus avoiding privacy breaches by design.

The proposed PRINCE method implements these principles using random walks for Personalized PageRank scores as a recommender model. Despite the potentially exponential search space of user-action subsets, we presented an efficient method for generating minimal counterfactual explanations for recommendations.

Extensive experiments on large real-life data from Amazon and Goodreads showed that simpler heuristics fail to find the best explanations, whereas PRINCE can guarantee optimality. Studies with AMT Master workers showed the superiority of PRINCE over baselines in terms of explanation usefulness.

Some future directions for research are: (i) exploring provider-side explainability in other classes of recommenders, (ii) developing methods for generating counterfactual explanations for a slate of recommendations, i.e., explaining the top-k recommendations instead of only the top-ranked item, and (iii) explicitly taking into account the diversity of the replacement items while generating counterfactual explanations.

5

USING EXPLANATIONS TO IMPROVE RECOMMENDER MODELS

Contents

5.1	Introduction	66
5.2	The ELIXIR framework	68
5.2.1	Feedback collection	69
5.2.2	Feedback densification	70
5.2.3	Feedback incorporation	71
5.2.4	ELIXIR in recommenders based on PPR	72
5.3	User study for data collection	73
5.3.1	Statement on ethics	74
5.3.2	Setup	74
5.4	Evaluation setup	77
5.4.1	Configurations	77
5.4.2	Metrics	78
5.4.3	Initialization	78
5.5	Results and insights	80
5.5.1	Key findings	80
5.5.2	Analysis	80
5.6	Related work	89
5.7	Conclusion	90

System-provided explanations for recommendations are an important component towards transparent and scrutable AI. In state-of-the-art research (including our contributions described in Chapters 3 and 4), the primary role of explanations is to improve transparency and subsequently user acceptance. However, it is often not clear how explanations can be leveraged towards scrutability of recommender systems, i.e., how to help users act on the explanations to improve their future recommendations.

In this chapter, we turn the role of explanations around and investigate how they can contribute to enhancing scrutability and improving the quality of the recommendations over time.

We devise a human-in-the-loop framework, called ELIXIR, where user feedback on explanations is leveraged for pairwise learning of user preferences. ELIXIR leverages feedback on pairs of recommendations and explanations to learn user-specific latent preference vectors. In Section 5.2, we formalize ELIXIR and propose a label propagation method with item-similarity-based neighborhoods to overcome sparseness of user feedback. We instantiate our framework in Section 5.2.4 using generalized graph recommendation via Personalized PageRank. Insightful experiments with a real user study described in Sections 5.3 and 5.4 show significant improvements in movie and book recommendations over item-level feedback.

5.1 Introduction

Motivation. Generating explanations for recommendations like movies, music or news, by online service providers, has gained high attention in academic and industrial research [Zhang and Chen, 2020, Balog and Radlinski, 2020, Zhao et al., 2019]. A key goal is to enhance user trust by transparent and scrutable recommendations, so that users understand how the recommended item relates to their prior online behavior (search, clicks, likes, ratings, etc.) Moreover, it is desirable that explanations are *causal* and *actionable*, meaning that i) they refer only to the user’s own action history and not to potentially privacy-sensitive cues about other users (see Chapter 4) and ii) the user can act on the explanation items by giving confirmation or refutation signals that affect future recommendations. Critique-enabled recommendation models [Chen and Pu, 2012, Jin et al., 2019, Luo et al., 2020b, Lee et al., 2020] pursue these goals, but are restricted to user feedback on the recommended items and associated content (e.g., text snippets from item reviews), disregarding the *explanation items*. In this chapter, we extend this regime of actionable user feedback to the explanations themselves, by obtaining additional cues from users in a lightweight manner and incorporating them into a human-in-the-loop framework to improve future recommendations.

Example. Fig. 5.1 shows an illustrative scenario. User u receives a recommendation for the movie *Fight Club* (rec) based on her online history and factors like item-item similarities. This is accompanied by an explanation referring to three items, all previously liked by u and being similar, in some aspects, to rec . We have exp_1 : *Seven Years in Tibet*, exp_2 : *The Prestige*, and exp_3 : *Pulp Fiction*. The system generated these three items for explanation because:

- exp_1 features the actor Brad Pitt who also stars in rec ,
- exp_2 has a surprise ending, similar to rec ,
- exp_3 contains violent content, like rec .

Now suppose that user u loves Brad Pitt and surprise endings but hates disturbing violence (she likes *Pulp Fiction* for other reasons like its star cast and dark comedy, that dominated her opinion, despite the violence). When receiving rec with the above explanation, user u could give different kinds of feedback. The established way is to simply dislike rec , as a

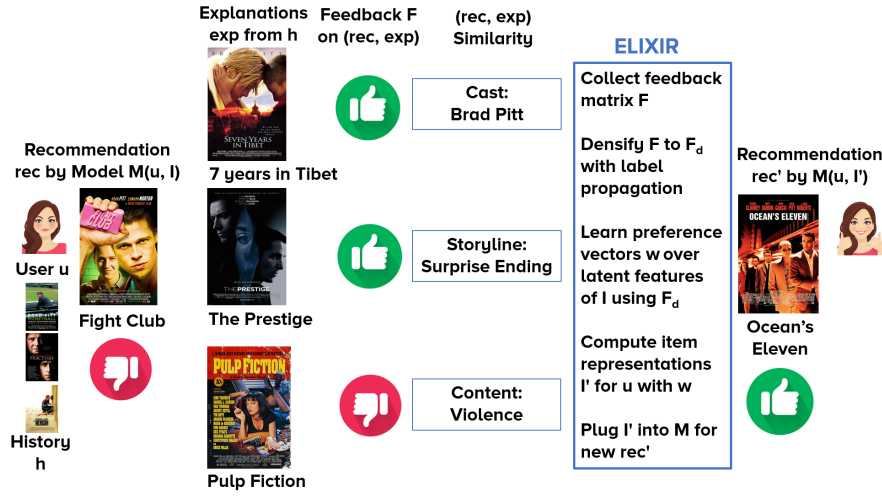


Figure 5.1: Example illustrating the intuitions behind ELIXIR.

signal from which future recommendations can learn. However, this would completely miss the opportunity of learning from how user u views the three explanation items. User u could also dislike the explanation as a whole, but this would give only a coarse signal, too, and would appear conflicting with the fact that she previously liked exp_1 , exp_2 , and exp_3 , confusing the recommender system. The best feedback would be if user u could inform the system that she likes Brad Pitt and surprise endings but dislikes violence, for example, by checking item properties or filling in a form or questionnaire. However, this would be a tedious effort that few users would engage in. Also, the system would have to come up with a very fine-grained feature space of properties, way beyond the usual categories of, say, movie genres.

Problem Statement. The goal in this chapter is to leverage user feedback on explanations. This entails two major problems:

- *Feedback:* How can we elicit user feedback on properties of both recommended and explanation items in a lightweight manner, without burdening the user with too much effort?
- *Actionability:* If we are able to obtain such refined feedback, how can the recommender system learn from it to improve its future outputs for the user?

Approach. This chapter presents ELIXIR (Efficient Learning from Item-based eXplanations In Recommenders), a novel framework, for leveraging explanations to improve future recommendations.

We address the *Feedback* problem by asking users for a binary like/dislike signal about the similarity of an explanation item exp to the recommended item rec . This can be thought of as assessing the quality of the *item pair* $\langle rec, exp \rangle$, but it is important that one of these is an explanation item that was faithfully produced by the recommender system specifically to justify rec . Our experiments compare this choice against asking for assessments on the similarity of the recommendation with the least relevant items in the user's profile, which turns out to be inferior. As we consider only causal explanations that refer to the user's own history of actions, the user

should be reasonably familiar with item exp . This kind of feedback is more refined than simply disliking the entire recommendation. The feedback is optional and can be given for any subset of the possible $\langle rec, exp \rangle$ pairs. Most importantly, the user is not burdened with identifying relevant properties of items, to further explain her feedback to the system. So ELIXIR builds on very lightweight and proactive user feedback.

We address the *Actionability* problem by extending state-of-the-art recommender models with a user-feedback matrix that encodes the like/dislike signals on $\langle rec, exp \rangle$ pairs. Since this matrix is inevitably sparse, ELIXIR densifies this input by means of label propagation on item neighborhoods [Zhu and Ghahramani, 2002]. To avoid the huge cost of computing similarities for all item pairs, we employ locality sensitive hashing (LSH) to find the closest items to every $\langle rec, exp \rangle$ tuple, thereby making ELIXIR efficient and tractable.

The core of our method is the learning of user-specific latent vectors that capture user preferences, by combining the densified feedback matrix and a prior item-item similarity matrix through regularized optimization that models the signals in the feedback matrix as soft constraints. The latent vectors would reflect that user u loves Brad Pitt and sophisticated plots but dislikes violent movies – without referring to these properties, all by means of learning latent representations from lightweight feedback. The per-user vectors are plugged into the actual recommender system to learn user-specific item representations for future recommendations.

We instantiate the ELIXIR framework in a popular family of graph-based recommenders based on personalized PageRank (PPR) (see, e.g., [Nikolakopoulos and Karypis, 2019]), from which explanations can be generated in a faithful and causal manner using PRINCE as described in the previous chapter.

The salient contributions of this chapter are:

- ELIXIR is, to the best of our knowledge, the first framework that leverages user feedback on explanation items, thus making explanations actionable, whereas prior works only tapped into feedback on recommendation items.
- ELIXIR elicits lightweight user feedback to learn user-specific item representations, and incorporates these into the recommender model, instantiated with the PPR methodology.
- We report experiments with data from a longitudinal user study in two domains: (i) movies, and (ii) books. The results demonstrate the viability of ELIXIR and its substantial gains in recommendation quality over item-level feedback.

5.2 The ELIXIR framework

In this section, we describe the components of ELIXIR and present its instantiation for a state-of-the-art recommender, RecWalk [Nikolakopoulos and Karypis, 2019], that is based on personalized PageRank (PPR). Table 5.1 summarizes concepts and notation discussed in this section.

Notation	Concept
u	A single user
v	A single item
\vec{v}	Latent vector for item v
d	Number of latent dimensions
U	Set of all users
I	Set of all items
H_u	Interaction history of user u
$F_u(v_i, v_j)$	Feedback on item pair (v_i, v_j) by user u
v_{ij}	Pseudo-item for item pair (v_i, v_j) in LP
W	Affinity matrix for LP
F_u^d	Densified feedback matrix for user u after LP
m	Number of non-zero elements in F_u^d
$sim(\vec{v}_i, \vec{v}_j)$	Similarity of item pair (v_i, v_j)
\vec{w}_u	Preference vector for user u
$g(\vec{v}, \vec{w}_u)$	Latent representation of item v for user u
γ	Regularization coefficient for learning \vec{w}_u
G	Graph on which the RWR recommender is run
N	Set of nodes in graph
E	Set of edges in graph
A	Matrix of user-item interactions for PPR
S	Item-item similarity matrix for PPR
\vec{e}_u	One-hot vector for user u
α	Restart probability in PPR
β	Probability of walking over interaction edges in PPR
rec	Recommendation item
exp	Explanation item

Table 5.1: Notation for salient concepts in ELIXIR.

5.2.1 Feedback collection

ELIXIR enables recommender systems to combine individual item ratings with feedback on pairs of recommendation and explanation items. The set of item-level signals H_u refers to the set of individual items that appear in the interaction history of user u . Denoting the universe of all items by $I = \{v_1, v_2, \dots, v_{|I|}\}$, we have $H_u \subseteq I$ and usually $|H_u| \ll |I|$.

While most recommenders train a user model solely based on H_u , ELIXIR exploits signals from user feedback on item pairs from recommendations and their explanations. We denote this pair-level feedback by the matrix $F_u \in \{-1, 0, +1\}^{|I| \times |I|}$. The matrix entry $F_u(v_i, v_j)$ represents user u 's feedback on recommendation item v_i and explanation item v_j . To collect such feedback, we ask users whether they like/dislike the similarity between items v_i and v_j . We encode a user's liking, no feedback, and disliking with $+1$, 0 and -1 , respectively.

5.2.2 Feedback densification

As the set of all item-pairs is very large, we expect matrix F_u to be extremely sparse. To mitigate this sparseness, we use the label propagation (LP) algorithm [Zhu and Ghahramani, 2002] on a graph where nodes are *pairs of items*, and edges represent the *similarity between item-pairs*. To define such a graph, we introduce the concept of a pseudo-item v_{ij} for each labeled pair of items (v_i, v_j) (that models an item that is like a mixture of the properties of the two items in the pair) where $F_u(v_i, v_j) \neq 0$, with \otimes denoting the element-wise product:

$$\vec{v}_{ij} = (\vec{v}_i \otimes \vec{v}_j)^{\frac{1}{2}} \quad (5.1)$$

where \vec{v}_i is the feature vector for item v_i . Depending upon the recommender model, item features are either learned by the model [Koren et al., 2009, He et al., 2017, Kang et al., 2018b] or are available from additional sources [Chen et al., 2016, Nikolakopoulos and Karypis, 2019]. More generally, we assume that item features can be cast into a latent representation. Eq. 5.1 defines the pseudo-item \vec{v}_{ij} as the element-wise geometric mean of \vec{v}_i and \vec{v}_j . Compared to the arithmetic mean, the geometric mean is more appropriate for boosting similarities and dampening dissimilarities (higher and lower values in the original vectors, respectively).

The original LP algorithm requires an affinity matrix W which encodes item-item similarities. In our problem, the labels we propagate are feedback points on (v_i, v_j) pairs: so each pseudo-item v_{ij} represents a pair of items and the affinity matrix thus contains pair-pair similarities. This makes W huge ($W \in \mathbb{R}^{|I|^2 \times |I|^2}$) and prohibits full materialization.

Our approach rather is to materialize and populate merely a small subset of W by considering only the k nearest neighbors of each pseudo-item v_{ij} (a labeled feedback point). A naive approach would require the generation of all possible pairs of items in which the nearest neighbors are computed (with complexity $|I|^2$).

To avoid this bottleneck, we compute an approximate kNN set for each pseudo-item v_{ij} using the following technique. We find the kNN set of v_{ij} rather among the items in I , denoted by the itemset kNN_{ij}^I (the superscript I denotes that this is a set of items and not item-pairs or pseudo-items). Instead of searching in $|I| \times |I|$ for the kNN of v_{ij} , we search in $kNN_{ij}^I \times kNN_{ij}^I$. This computation is made efficient using *locality sensitive hashing (LSH)* to deal with the large number of pairings. This way, the search space for label propagation is reduced from $O(|I|^2)$ to $O(|I|)$.

To determine the kNN s of v_{ij} , an item-item similarity measure is required. Different recommenders use different measures for capturing such similarities: cosine similarity [Nikolakopoulos and Karypis, 2019], Euclidean similarity [Hsieh et al., 2017], weighted inner products [Kabbur et al., 2013, Nikolakopoulos et al., 2019, Xin et al., 2019], and angular similarity [Sarwar et al., 2001] are a few common choices. We treat the similarity function as a plug-in module $sim(.,.)$, and instantiate it using cosine similarity when required. Cosine similarity

is emerging as a particularly convenient choice when items (and often users, categories, etc.) are represented as vectors in a shared latent space. Note that we treat items and pseudo-items uniformly and use the same function $\text{sim}(\cdot, \cdot)$ to compute their similarity. The output of this stage is a densified feedback matrix F_u^d .

5.2.3 Feedback incorporation

Optimization problem. We incorporate matrix F_u^d into the recommender by imposing a soft constraint for learning a user-specific mapping function $g(\cdot, \cdot)$ with \vec{w}_u as its parameter vector. The goal is to learn preference vectors \vec{w}_u for each user that can be combined with existing item representations \vec{v} to produce user-specific item representations and then fed into the underlying recommender model. To learn \vec{w}_u , we formulate the following objective function where the signals from the densified feedback matrix are incorporated as a soft constraint:

$$\min_{\vec{w}_u} \frac{1}{m} \sum_{v_i, v_j} F_u^d(v_i, v_j) \cdot (\text{sim}(\vec{v}_i, \vec{v}_j) - \text{sim}(g(\vec{v}_i, \vec{w}_u), g(\vec{v}_j, \vec{w}_u))) + \gamma \|\vec{w}_u\|^2 \quad (5.2)$$

where $m = |\{(v_i, v_j) | F_u^d(v_i, v_j) \neq 0\}|$. Eq. 5.2 includes a mapping or transformation function g (common choices would be vector translation and scaling), to be computed by the optimization solver. This serves to map original item representations and re-arrange their positions in the latent space such that their new similarities reflect the user's feedback on item pairs. The underlying intuition is to decrease or increase pairwise similarities whenever $F_u^d(v_i, v_j) = -1$ or $F_u^d(v_i, v_j) = 1$, respectively. The above objective achieves this two-fold (increasing and decreasing) effect in a unified manner. Additional L2 regularization is used on \vec{w}_u to encourage small magnitude, avoiding drastic changes when it is applied inside $g(\cdot)$.

After learning an (near-) optimal \vec{w}_u , each item vector \vec{v}_i is updated to $g(\vec{v}_i, \vec{w}_u)$; we use these user-specific item vectors to generate new recommendations. We posit that such user-specific item representations helps the recommender model to incorporate the *more liked* and *less disliked* latent aspects of similarity for each user, and helps produce improved recommendations.

An alternative choice of formulating Eq. 5.2 would be to have only the L2 regularization term as the objective and model the signals from F as hard constraints. With this alternative approach, the constraints would become inequality constraints ($F_u^d(v_i, v_j) \cdot (\text{sim}(\vec{v}_i, \vec{v}_j) - \text{sim}(g(\vec{v}_i, \vec{w}_u), g(\vec{v}_j, \vec{w}_u))) < 0$) and would require that the KKT (Karush-Kuhn-Tucker) conditions be satisfied for an optimal solution to exist. In practice, experimenting with the hard constraint formulation resulted in null solutions for most cases; hence our soft-constraint-based method.

Implementation. The optimization in Eq. 5.2 for learning \vec{w}_u is non-convex due to the presence of the cosine function. Stochastic gradient descent (SGD) (available in libraries like PyTorch) is

used for computing near-optimal solutions.

5.2.4 ELIXIR in recommenders based on PPR

Generating recommendations. We incorporate our method into RecWalk [Nikolakopoulos and Karypis, 2019], a state-of-the-art recommender model based on personalized PageRank. The input to this model is a heterogeneous graph (also referred to as a heterogeneous information network, HIN) $G = (N, E, M_N^T, M_E^T, M_E^W)$ with a set of nodes N , a set of edges $E \subseteq N \times N$ and mappings M_N^T and M_E^T from nodes and edges, respectively, to their types, and an edge weight function $M_E^W : E \mapsto \mathbb{R}_{\geq 0}$ (see Section 4.2 for the complete definition). Nodes are either of type *user* or *item*, i.e., $N = U \cup I$. Edges capture user-item interactions, denoted by $A \in \{0, 1\}^{|N| \times |N|}$, and node-node similarities presented by $S \in \mathbb{R}_+^{|N| \times |N|}$. So we have two types of nodes and two types of edges in this graph.

In RecWalk, the recommendation score of item v_i for user u is computed as $PPR(u, v_i)$. To recall, PPR stands for personalized PageRank [Haveliwala, 2003], and in RecWalk is computed as follows [Nikolakopoulos and Karypis, 2019]:

$$PPR(u, \cdot) = \alpha \cdot \vec{e}_u + (1 - \alpha) \cdot PPR(u, \cdot) \cdot [\beta A + (1 - \beta)S] \quad (5.3)$$

where α is the restart probability, \vec{e}_u is the one-hot vector for user u and β is the probability that a walk traverses an interaction edge. According to Eq. 5.3, a walk either visits one of its neighbors with probability $1 - \alpha$ or jumps back to user node u . The neighbors are connected either through interaction or similarity edges. Matrix S encodes similarities between nodes of the same type. Without loss of generality, we assume that a user is similar only to herself, i.e., $S(u_i, u_j) = 1$ if and only if $i = j$. The item-item similarity, however, is defined by the $sim(\cdot, \cdot)$ function, and hence $S(v_i, v_j) = sim(v_i, v_j)$. We simplify the notation and use S to refer only to item-item similarities. Note that RecWalk normalizes matrix S in a certain way to enforce stochasticity. We omit the details for the sake of brevity and refer users to [Nikolakopoulos and Karypis, 2019] for more information. The item v in $|I|$ with the highest $PPR(u, v)$ score is shown as recommendation *rec* to user u .

Generating explanations. Suppose item *rec* is recommended to user u . Item-based explanations $\{exp\}$ in PPR-based recommenders can be generated using the PRINCE algorithm described in detail in Chapter 4. The resulting explanation item sets are minimal and counterfactual: they ensure causality relation using the counterfactual setup that u would not receive *rec* if she did not have items $\{exp\}$ in her history H_u . However, minimality of explanations is not a concern in the current context. Therefore, we take a more straightforward approach to approximate PRINCE by estimating the contribution score of item $v_j \in H_u$ to the recommended item *rec*:

$$contribution(v_j, rec) = PPR(v_j, rec) \quad (v_j \in H_u) \quad (5.4)$$

where $PPR(v_j, rec)$ is the PageRank of node rec personalized for node v_j . We use the top- k items with highest contributions in H_u as the explanation set $\{exp\}$ for item rec .

Incorporating feedback. In RecWalk, item-item similarities are explicitly captured in matrix S , i.e., $S(v_i, v_j) = sim(v_i, v_j)$. Given the items' latent representations (possibly computed by running techniques like NMF or SVD from sparse explicit feature vectors), we define $sim(v_i, v_j)$ as the cosine similarity between v_i and v_j , and hence $S(v_i, v_j) = cos(\vec{v}_i, \vec{v}_j)$. As discussed earlier, to incorporate densified feedback F_u^d , we introduce a user-specific preference vector \vec{w}_u to adjust u 's bias with respect to the latent aspects and update the item representations by adding \vec{w}_u to them. The transformation function g is chosen to be a vector translation, shifting universal item representations onto user-specific ones:

$$g(\vec{v}_i, \vec{w}_u) = \vec{v}_i + \vec{w}_u \quad (5.5)$$

The intuition behind the mapping described in Eq. 5.5 is to highlight (suppress) the effect of liked (disliked) features through addition of positive (negative) bias values. Plugging the definitions for g and sim and the densified matrix F_u^d into the optimization objective (Eq. 5.2), we learn \vec{w}_u as follows:

$$\min_{\vec{w}_u} \frac{1}{m} \sum_{v_i, v_j} F_u^d(v_i, v_j) \cdot (cos(\vec{v}_i, \vec{v}_j) - cos(\vec{v}_i + \vec{w}_u, \vec{v}_j + \vec{w}_u)) + \gamma ||\vec{w}_u||^2 \quad (5.6)$$

Using \vec{w}_u , we build a user-specific similarity matrix S_u defined as:

$$S_u(v_i, v_j) = cos(\vec{v}_i + \vec{w}_u, \vec{v}_j + \vec{w}_u) \quad (5.7)$$

Finally, we update the personalized PageRank recommendation scores accordingly, thereby completing the integration of pairwise feedback into the recommender model:

$$PPR(u, .) = \alpha \cdot \vec{e}_u + (1 - \alpha) \cdot PPR(u, .) \cdot [\beta A + (1 - \beta) S_u] \quad (5.8)$$

5.3 User study for data collection

ELIXIR operates in a unique framework of user judgments on similarities between recommendation and explanation pairs. It hinges on longitudinal observations of the same users providing: i) original profiles, ii) feedback on rec items and iii) feedback on $\langle rec, exp \rangle$ pairs, as well as iv) item-level assessments on the final recommendations. Thus, a study involving real users was imperative to demonstrate the practical viability of our proposal. To this end, we recruited 25 volunteers, who were all Masters' students of the Computer Science Department at the author's institute, with payment per hour comparable to that of master workers on a crowdsourcing platform like Amazon Mechanical Turk.

5.3.1 Statement on ethics

Participants' privacy was fully respected: all personally identifying information concerning participants was kept private during the course of the study, and deleted after its completion. All data was stored locally, with encryption, firewall protection and other measures of this sort. During the course of the study, users had to provide ratings on individual as well as pairs of movies and books. While this is not personally sensitive per se, we recognize that the data reflects users' personal preferences. All participants signed a consent document that they agree to this data being used for research purposes and that it can be released with anonymized identifiers. The user study and the inclusion of results in this work were approved by the Ethics Review Board at the authors' Institute.

The annotation sessions were conducted over online video conferencing, so that participants' browser activity could be monitored. To respect users' privacy, no video recordings were made. A one-hour training session was conducted, where participants were made aware of the goals of the study and their exact tasks, and were guided through examples.

5.3.2 Setup

The user study was conducted in two domains: (i) movies (restricted to Hollywood, because of their popularity and the users' familiarity), and (ii) books. Over the course of six weeks (three weeks for each domain), each user annotated individual as well as pairs of movies and books for a total of 28 hours. The payment was 10 Euros per hour, with the total cost amounting to $25 \times 28 \times 10 = 7,000$ Euros. For each domain, the annotations were collected in three phases that lasted three weeks. Table 5.2 shows some statistics on annotations collected during the study.

Scenario	#Item feedback	#Pair feedback	Sessions	Hours
Phase 1	50	—	1	2
Phase 2	30	300	5	10
Phase 3	72	—	1	2
Total	152	300	7	14
Total (All users)	$\simeq 4000$	7500	175	350

Table 5.2: Annotations per user over stages of the study (spanning a total of 350 person-hours) in each domain.

5.3.2.1 Phase 1: Building users' profiles

It is essential to keep the assessment setup natural: if users were asked to rate arbitrary items and pairs that they are unfamiliar with, the judgments would be unreliable. Thus, as the first step of the study for each domain, we asked users to provide us with 50 movies and books each, that they liked, to build a true history for each user, that would create subsequent recommendations for her. Since movie or book titles can often be ambiguous, users were asked to provide us with MovieLens¹ and Goodreads² URLs in individualized spreadsheets. For each domain, we conducted this phase in a session spanning two hours which provided us with $50 \times 25 = 1,250$ user actions (likes).

5.3.2.2 Phase 2: Collecting feedback on items and pairs

The obtained user profiles were plugged into the PPR-based recommender model RecWalk [Nikolopoulos and Karypis, 2019], where every liked item contributes an interaction edge to the network. The union of all items rated by the 25 users forms our universe of items now, from where we generated the top-30 recommendations for each user. Along with each recommendation, we generated the top-5 explanation items $\{exp\}$ using the approximated version of the PRINCE algorithm (Eq. 5.4).

To investigate the role of faithful explanations in pairwise feedback, we also identified the five items $\{rand\}$ in the user's profile that are the *least similar* to the recommendation item. These serve as a proxy for pairing the recommendation with random items; they are drawn from the user's profile to ensure familiarity. The similarity is computed as the cosine between the item vectors.

The users are now presented with three tasks: (i) rate the generated recommendations (like/dislike); (ii) rate the similarity of each $\langle rec, exp \rangle$ pair (like/dislike); (iii) rate the similarity of each $\langle rec, rand \rangle$ pair (like/dislike). The two kinds of feedback, item-level in (i), and pair-level in (ii)+(iii), have very different semantics, and users were appropriately briefed and guided. Item-level feedback is straightforward, where they comment whether they liked or disliked an item. Rating an item pair, though, needs a bit more reflection on the possible similarities between the two items (two movies or two books), deciding on the most important factor in case of multiple such aspects, and providing the binary preference assessments.

Participants entered their ratings in individualized spreadsheets we prepared for them. Each sheet contained several blocks where each block corresponded to one recommendation item followed by ten different explanation items for it (five $\{exp\}$ and five $\{rand\}$). To avoid any position bias, we randomly shuffled the explanation items in each block.

While the feedback remains lightweight due to the user's potential familiarity with the items,

¹<https://movielens.org/home>

²<https://goodreads.com/>

we provided some help to cue their memory. For instance, we presented each movie with its title and its corresponding MovieLens URL, where the user could see the movie’s summary and key properties. Moreover, MovieLens provides a rich set of tags on actors, directors, genre, storyline and content; for item pairs we displayed the intersection set of top tags for the two movies. Users could nevertheless browse the MovieLens pages or other background sources at their discretion.

Book recommendations were also presented together with some auxiliary information including their descriptions, authors, top genres as listed on their Goodreads pages, and their corresponding URLs. Similar to the movie domain, we facilitated users’ judgments on pairs of books by listing their common properties such as genres and authors.

Note that the assessment of $\langle rec, exp \rangle$ or $\langle rec, rand \rangle$ is decoupled from the fact whether the user likes *rec*, *exp*, or *rand* individually. To make this distinction clear, the users were walked through several reference annotations during the training session. For qualitative analysis, we also asked users to optionally articulate the dimension that was the basis of their similarity feedback. We report on this in the experimental section.

At the end of this stage, each user provided us with 30 item-level ratings (*rec*), and $30 \times 5 \times 2 = 300$ pair-level ratings (five pairs for each of $\langle rec, exp \rangle$ and $\langle rec, rand \rangle$). Therefore, for each domain, we had a total of 750 distinct item-level feedback points and 7,500 distinct pair-level feedback points, for a total of 25 users. This phase required ten hours from each user: to avoid task fatigue, this was spread over five two-hour sessions.

5.3.2.3 Phase 3: Collecting feedback on final recommendations

In the last phase of the longitudinal user study, the collected feedback was incorporated into the ELIXIR framework to produce improved recommendations for every user. *Item-level feedback* was cast into additional interaction edges for the original graph recommender; *pair-level feedback* was incorporated using the procedure described in Sec. 5.2. In addition, we experimented with *combined feedback* incorporating both item-level and pair-level. These are the three top-level configurations in our experimental evaluation.

For incorporating pair-level feedback, there are two possibilities of using either *exp* or *rand*, altogether resulting in five variations of the recommender model. These models were each made to produce 30 recommendations, leading to 180 items to be rated by each user (five pair-level and one item-level strategy, thus a total of $6 \text{ strategies} \times 30 = 180$). However, there were overlaps in the *rec* sets across configurations. At the end, a total of $\simeq 1,800$ ratings were collected from 25 users in each domain ($\simeq 72$ per user). This phase took two hours per user, on average, and was completed in a single session at the end of the third week.

5.4 Evaluation setup

5.4.1 Configurations

We evaluate ELIXIR for different configurations, including the baseline of exploiting solely item-level feedback on *rec* items. But before we can go to the results, we need to explain the basic setup of the experimental framework. Latent vectors necessary to initialize item representations are learnt by running non-negative matrix factorization (NMF) on the sparse matrix of movie-tag memberships from MovieLens and book-genre memberships from Goodreads using the Nimfa Python library³ with the default settings. The number of latent dimensions d is chosen to be 20 which was guided by observations on the reduction of sparsity from the original matrix.

We use the SciPy library⁴ for subsequent label propagation with the cosine kernel. The number k for LP was chosen to be 10, which means that for each pseudo-item, we find the 10 nearest items, and hence $\binom{10}{2} = 45$ pseudo-items. We tried two other values, $k = 5$ and 20, and observed similar results.

For LSH, we used NearPy⁵ with random binary projection as its hash function. LSH assigns each item vector v_i to a bucket where its approximate nearest neighbors lie. While a large number of buckets decreases the probability of neighbors being assigned to the same bucket, a small number reduces the efficiency of k NN queries. Considering the choice of k in LP ($k = 10$), we chose the number of buckets to be 8 (corresponding to 3 random binary projection vectors). With this number of buckets, we reduce the failure rate of LSH to 15%, i.e., for only 15% of the k NN queries with $k = 10$, LSH returns less than 10 neighbors. We use our own implementations of RecWalk and PRINCE for generating recommendations and explanations, respectively.

The following five feedback configurations are compared:

- *Item-level feedback.* This baseline model only absorbs users' binary preferences on individual items *rec*. Such item-level feedback adds interaction edges to the input graph.
- *Pair-level feedback with explanations.* The model captures only the judgments on pairs of $\langle rec, exp \rangle$ items. Such pair-level signals update the similarity matrix used in the recommendation model.
- *Pair-level feedback with random items.* This is similar to the previous configuration, except that the explanation items here are replaced by the least relevant items from the user's history ($\{rand\}$).
- *Item + pair-level feedback with explanations.* The model exploits both individual and pairwise feedback.

³<http://ai.stanford.edu/~marinka/nimfa/>

⁴<https://bit.ly/35lVV10>

⁵<https://pixelogik.github.io/NearPy/>

- *Item + pair-level feedback with random items.* This is similar to the previous configuration except that the explanation items $\{exp\}$ are replaced by $\{rand\}$.

5.4.2 Metrics

We evaluate the quality of recommendations generated after feedback incorporation using three metrics: i) Precision at the top- k ranks ($P@k$), ii) Mean Average Precision at the top- k ranks ($MAP@k$), and, iii) normalized discounted cumulated gain at the top- k ranks ($nDCG@k$, computed with binary non-graded relevance assessments), as defined in Table 2.1. While $P@k$ is a set-based metric considering the top- k items, analogous to a *slate of recommendations*, the latter two are sensitive to the ranks of relevant items in the lists.

All metrics are computed at three rank cut-off values k : 3, 5, and 10. The relatively low values of cut-off ranks are chosen to show the effectiveness of ELIXIR in introducing highly selective items into the top recommendations for individual users.

5.4.3 Initialization

To fairly compare different configurations, we train RecWalk using the same set of parameters. The restart probability α is set to 0.15 as shown effective in prior works [Brin and Page, 1998]. To highlight the effect of similarity edges, we choose $\beta = 0.1$: a lower β indicates a lower likelihood of sampling an interaction edge for the random walker, and a walker thus traverses similarity edges in G with probability 0.9. Using smaller values for β is also suggested in the original model of RecWalk [Nikolakopoulos and Karypis, 2019].

The interaction graph G built for movies had 25 users, 621 movies, $1.3k$ interaction edges, and $11k$ similarity edges. For books, G was larger and denser, with 868 books, $1.3k$ interaction edges, and $41k$ similarity edges. To compute PageRank scores, we use the power-iteration method with a maximum of 500 iterations.

To construct the similarity matrix S for RecWalk, we employ LSH again with a similar configuration as discussed for densification. To avoid too many edges in the graph, we only connect items with large similarities. For this, we use threshold 0.7, i.e., $S(v_i, v_j) = 0$ if $\cos(v_i, v_j) < 0.7$. Matrix A is built from item-level user feedback: we define $A(u_i, v_j) = 1$ if u_i likes item v_j , and zero otherwise. Regularization parameter γ in Eq. 5.6, and the learning rate lr in SGD for finding an optimal \vec{w}_u were tuned on a development set containing 20% of each user's ratings. We tested 10 different values for γ (1, 2, ..., 10) and 3 values for the learning rate (0.001, 0.01, 0.1) and chose the values with best performance on the development set: $\gamma = 3$ and $lr = 0.01$.

Setup [Movies]	P@3	P@5	P@10	MAP@3	MAP@5	MAP@10	nDCG@3	nDCG@5	nDCG@10
Item-level	0.253	0.368	0.484	0.323	0.380	0.448	0.244	0.327	0.422
Pair-level (Exp-1)	0.506*	0.592*	0.580*	0.566*	0.599*	0.625*	0.496*	0.557*	0.565*
Pair-level (Exp-3)	0.506*	0.568*	0.596*	0.630*	0.624*	0.634*	0.504*	0.547*	0.575*
Pair-level (Exp-5)	0.480*	0.512*	0.568*	0.567*	0.579*	0.591*	0.463*	0.490*	0.536*
Pair-level (Rand-5)	0.453*	0.504*	0.560*	0.536*	0.537*	0.596*	0.451*	0.486*	0.532*
Item+Pair-level (Exp-5)	0.533*	0.544*	0.596*	0.563*	0.571*	0.603*	0.500*	0.517*	0.562*
Item+Pair-level (Rand-5)	0.453*	0.488*	0.572*	0.486*	0.520*	0.578*	0.426*	0.458*	0.526*
Setup [Books]	P@3	P@5	P@10	MAP@3	MAP@5	MAP@10	nDCG@3	nDCG@5	nDCG@10
Item-level	0.253	0.336	0.436	0.309	0.343	0.417	0.233	0.296	0.379
Pair-level (Exp-1)	0.506*	0.528*	0.54*	0.603*	0.620*	0.654*	0.493*	0.512*	0.527*
Pair-level (Exp-3)	0.506*	0.536*	0.58*	0.573*	0.596*	0.612*	0.484*	0.511*	0.551*
Pair-level (Exp-5)	0.586*	0.600*	0.656*	0.726*	0.701*	0.692*	0.602*	0.607*	0.644*
Pair-level (Rand-5)	0.480*	0.504*	0.504	0.593*	0.604*	0.588*	0.468*	0.488*	0.494*
Item+Pair-level (Exp-5)	0.493*	0.456*	0.560*	0.543*	0.566*	0.566*	0.482*	0.458*	0.529*
Item+Pair-level (Rand-5)	0.386*	0.416*	0.516*	0.469*	0.509*	0.536*	0.376*	0.399*	0.475*

Table 5.3: Comparison of different modes of feedback incorporation with results from the user study. The best value in every column (for each domain) is marked in **bold**. * denotes statistical significance of all methods over item-level feedback, with p -value ≤ 0.05 using the Wilcoxon signed rank test [Wilcoxon, 1992].

5.5 Results and insights

5.5.1 Key findings

Our key findings are presented in Table 5.3, where different feedback absorption strategies are evaluated over data from the user study. We make the following salient observations:

- **Pair-level feedback improves recommendations.** The most salient observation from the results in both domains is that including pairwise feedback (all table rows except the first one for each domain) on the similarity of item pairs results in substantial improvements over solely considering item-level feedback. This confirms our hypothesis that procuring feedback on pairs provides refined and highly beneficial user signals that cannot be captured through aggregated item-level feedback.
- **Pair-level feedback is more discriminative than item-level.** Next, we compare the effectiveness of pair-level feedback vis-à-vis item-level. To have a fair comparison with respect to the volume of feedback, we introduce a new setup, Pair-level (Exp-1), where the volume of pair-level feedback is similar to that of the Item-level setup. For this, we only consider users' feedback on the most relevant explanation item. This results in the incorporation of 30 pairs with distinct recommendation items, that is compared to item-level feedback on these 30 items. We observe that in both domains, Pair-level (Exp-1) significantly outperforms the Item-level setup. This demonstrates the efficacy of pair-level feedback in capturing users' fine-grained interests. For completeness, we also provide results when we use top-3 and top-5 explanations (Exp-3 and Exp-5, respectively).
- **Using explanations for pair-level feedback is essential.** We set out with the goal of making explanations actionable towards model improvement. This is validated by the observation that item+pair-level for explanations are consistently and substantially better than item+pair-level for random items instead of top-ranked explanation items.

5.5.2 Analysis

Our longitudinal study on collecting pairwise judgments opens up the possibility of gaining insights on several issues on user behavior and feedback:

- Q1. How different are positive and negative feedback by volume?
- Q2. Are users who are more likely to provide negative item-level ratings also biased towards more dislikes on item pairs?
- Q3. What are the most common aspects influencing similarity feedback on item pairs?
- Q4. How does performance improvement correlate with the diversity of the original profile?
- Q5. How do the volumes of positive and negative feedback correlate with performance

improvement?

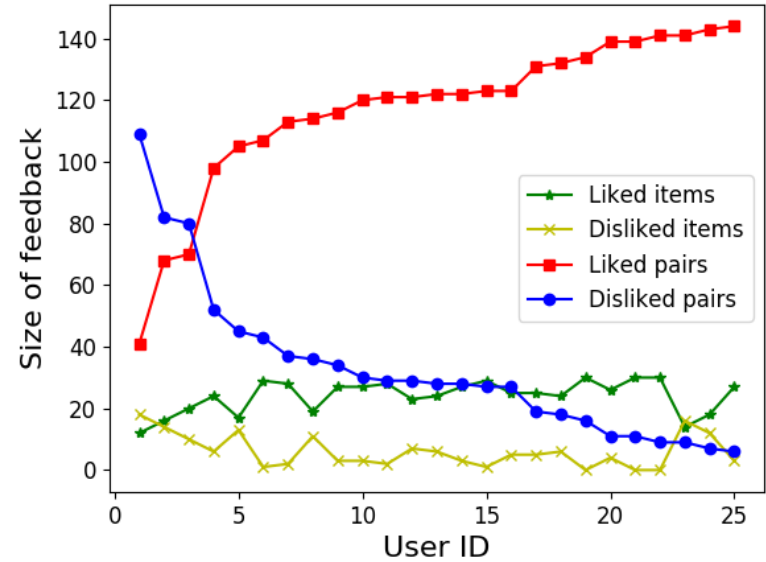
We address these questions in the subsequent analysis. In the end, Fig. 5.7 and 5.8 show four representative examples from the user study.

To see the size distribution of different feedback types (Q1), we plot the number of item-level and pair-level feedback points provided by each user in Fig. 5.2. Users vary in their proportions of positive and negative feedback. Overall, users enter much more positive feedback than negative. Reasonable numbers for all four types of judgments (item-/pair-level \times like/dislike) show that, overall, users are willing to provide the necessary effort towards improving their recommendations. Monitoring feedback assessments over online sessions showed that pairwise feedback requires indeed a lightweight effort (measured by the time taken for completion) and does not impose a substantial cognitive load on users.

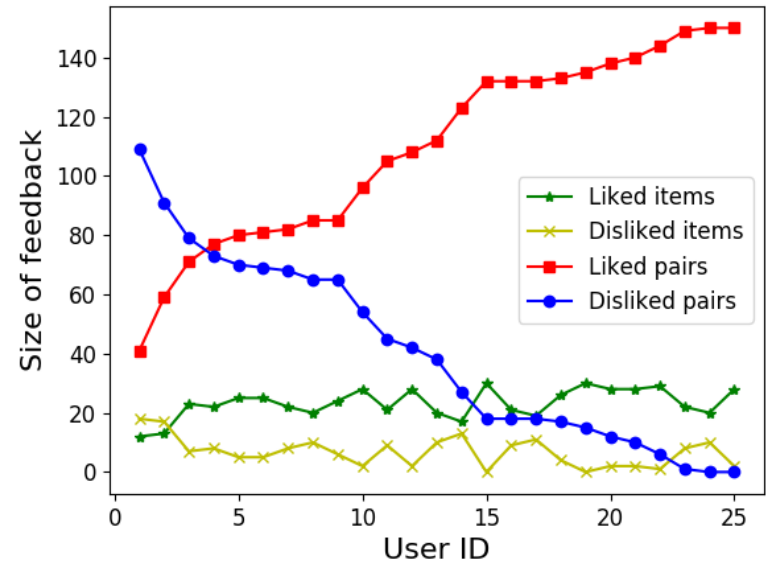
In particular, we find that users who may be biased towards more negative item-level feedback often provide substantial volumes of negative feedback on pairs. The corresponding Pearson correlations in movie and book domains are 0.53 and 0.51, respectively. Therefore, we can conclude that user behavior carries over from items to pairs (Q2). This leads to an extremely crucial insight: negative signals on items cannot be harnessed in graph recommenders. Yet eliciting negative feedback on certain pairs of bad recommendations and good explanation items, can lead to substantial benefit for the recommender system: similar bad recommendations become less likely to be recommended to the user.

Next, we show factors influencing pairwise similarity assessment in Fig. 5.3 (Q3). To compare and contrast, we asked users to mention their reasons for both item- and pair-level feedback. Qualitative analysis reveals that in the movie domain, genres play the biggest role in feedback, followed by content, actor, and then director. We observe that genre and content (the latter includes storylines like plot twists, alien movies, medieval movies, etc.) are much more likely to influence user preferences than the presence of specific actors or directors. This underlines the necessity of latent representation of item properties, as storylines are hard to capture in explicit feature models. Similar trends are observed for books, i.e., genre is the most frequently mentioned, followed by content and author. The interesting observation is that users are systematic in their behavior: in both domains, histograms have the same relative distribution for item- and pair-feedback.

We also investigate whether all users are equally likely to benefit from ELIXIR. Since profile sizes are kept constant to control for other factors, we try to see if performance improvements from item+pair-level feedback is connected to the *diversity* of their original profiles (Q4). To quantify diversity, we measure the entropy of the distribution of tags associated with the 50 items that were used to initialize profiles of the users (higher entropy is higher diversity). Plots for movies and books are shown in Fig. 5.4. Our observation for the movie domain is that ELIXIR helps users with relatively high interest diversity (right half) slightly more (top right) than the users with more particular interests (languishing towards the bottom left corner). The corresponding Pearson correlation is 0.29 which indicates a moderate positive correlation

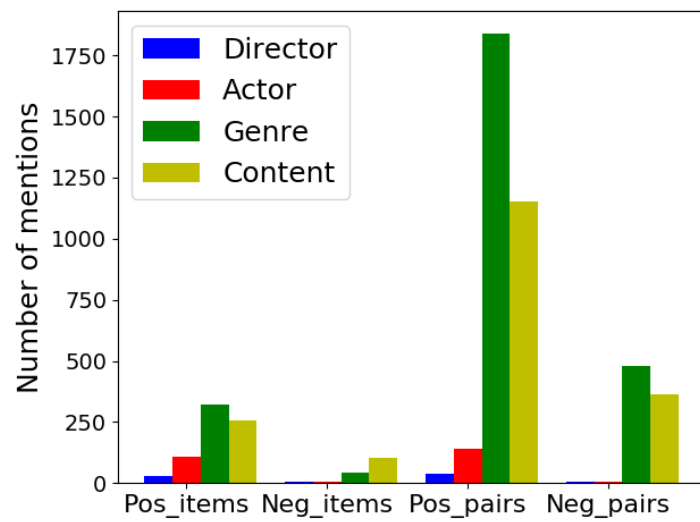


(a) Movies

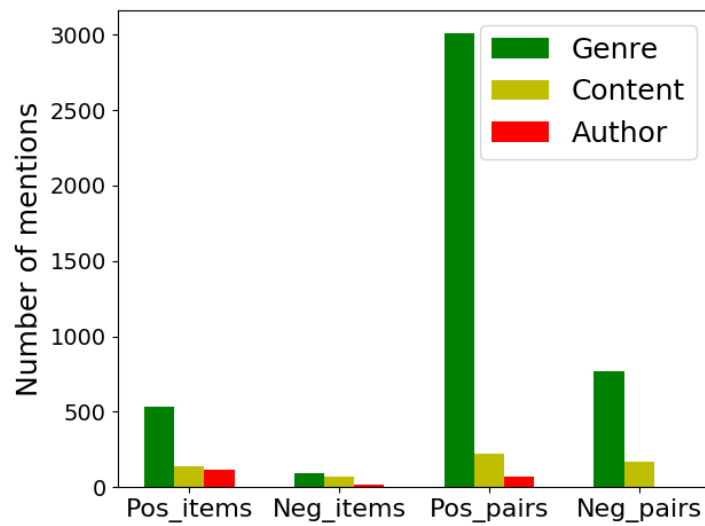


(b) Books

Figure 5.2: Per-user volume of feedback by type.



(a) Movies



(b) Books

Figure 5.3: Key influencers behind feedback assessments.

between profile diversity and improvement level. For books, however, the Pearson correlation is -0.17 implying a small negative correlation between the diversity of profiles and effectiveness of ELIXIR.

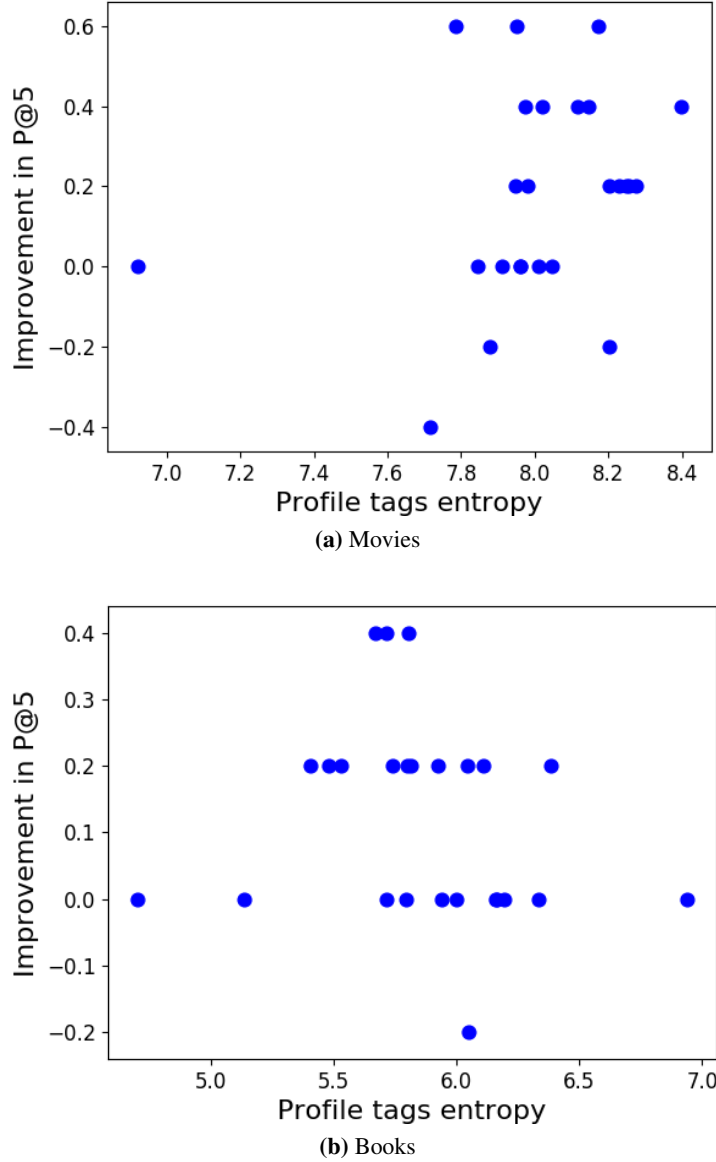


Figure 5.4: Connecting gains via ELIXIR with profile diversity.

Next, we investigate if different volumes of feedback on the four possibilities (item-level like/dislike; pair-level like/dislike) lead to notably differing performance improvements (Q5). We show the effect of feedback size on improvement levels in Fig. 5.5 and 5.6, where the two plots in the top row of each figure correspond to item-level feedback, and the bottom to pair-level. The scales and limits of x -axes within rows (and all y -axes) are kept the same for easy comparison. Here we note that dots along the same row (level) of precision correspond to the same users. The notable observation from these figures is that users who provide more positive

feedback are likely to see higher improvements. This correlation is particularly more pronounced for positive item-level feedback. While the benefit of sending more positive signals as more actionable is understandable, a certain part of the blame may lie on the graph recommender itself, where “negative edges” cannot be included easily: presence or absence of edges is the standard model. This suggests further research to explore ELIXIR with other families of recommenders like matrix or tensor factorization, which can more easily incorporate negative feedback.

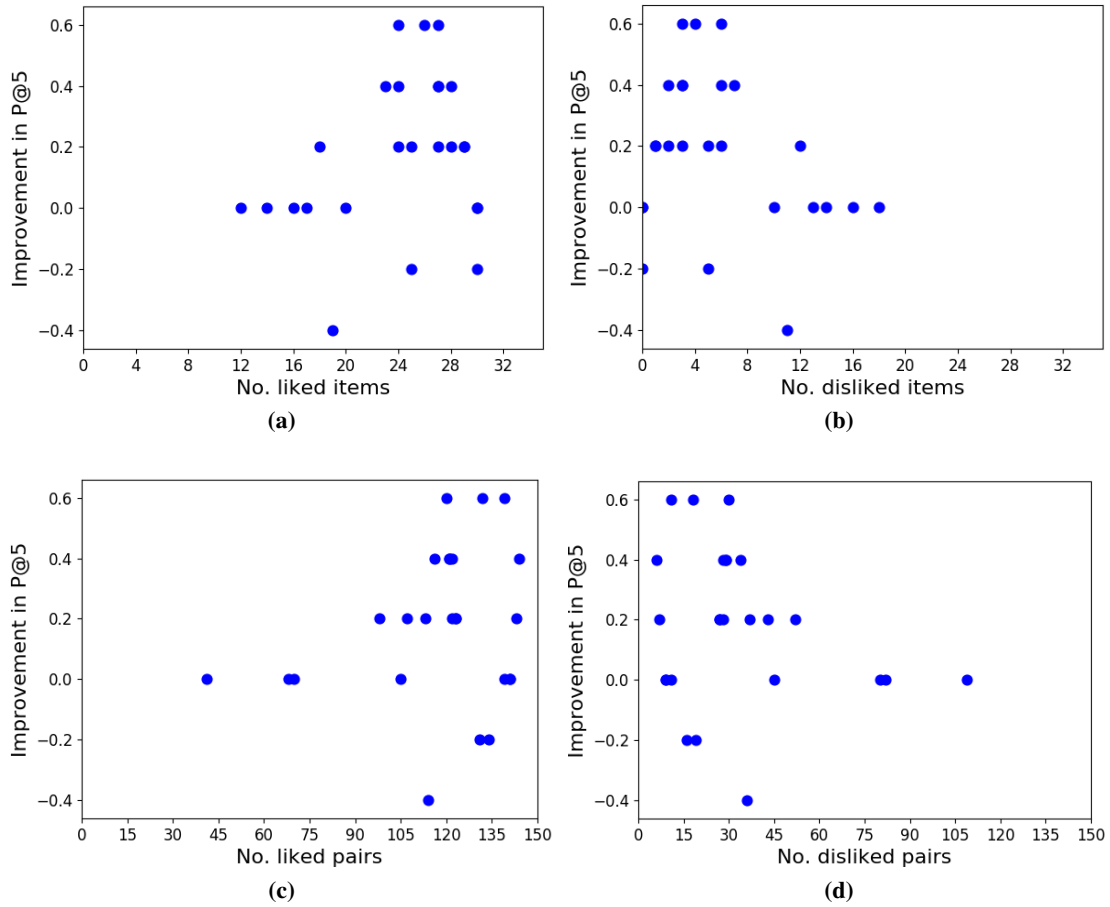


Figure 5.5: P@5-improvement w.r.t. feedback size (Movies).

Finally, we show anecdotal examples from our user study in Fig. 5.7 and 5.8. For movie recommendation (Fig. 5.7), incorporating user feedback on pairwise similarity introduces new items into the top-10 recommendations (*The Chronicles of Narnia*, *The BFG*) for their respective users. These new recommendations possess the similarity aspects liked by the user (*fantasy* for *Narnia*, *based on a book* for *The BFG*), and lack the dimensions that the user has implicitly disliked (*crime* for the first anecdote, *fiction* for the second one). Similarly, we present two instances of improvement in book recommendation in Fig. 5.8, where incorporation of pair-level feedback results in reducing the relevance score of disliked items (*Memoirs of a Geisha*, *DotCom Secrets*) and bringing up more relevant items (*The Iliad*, *Mindset: The New Psychology*).

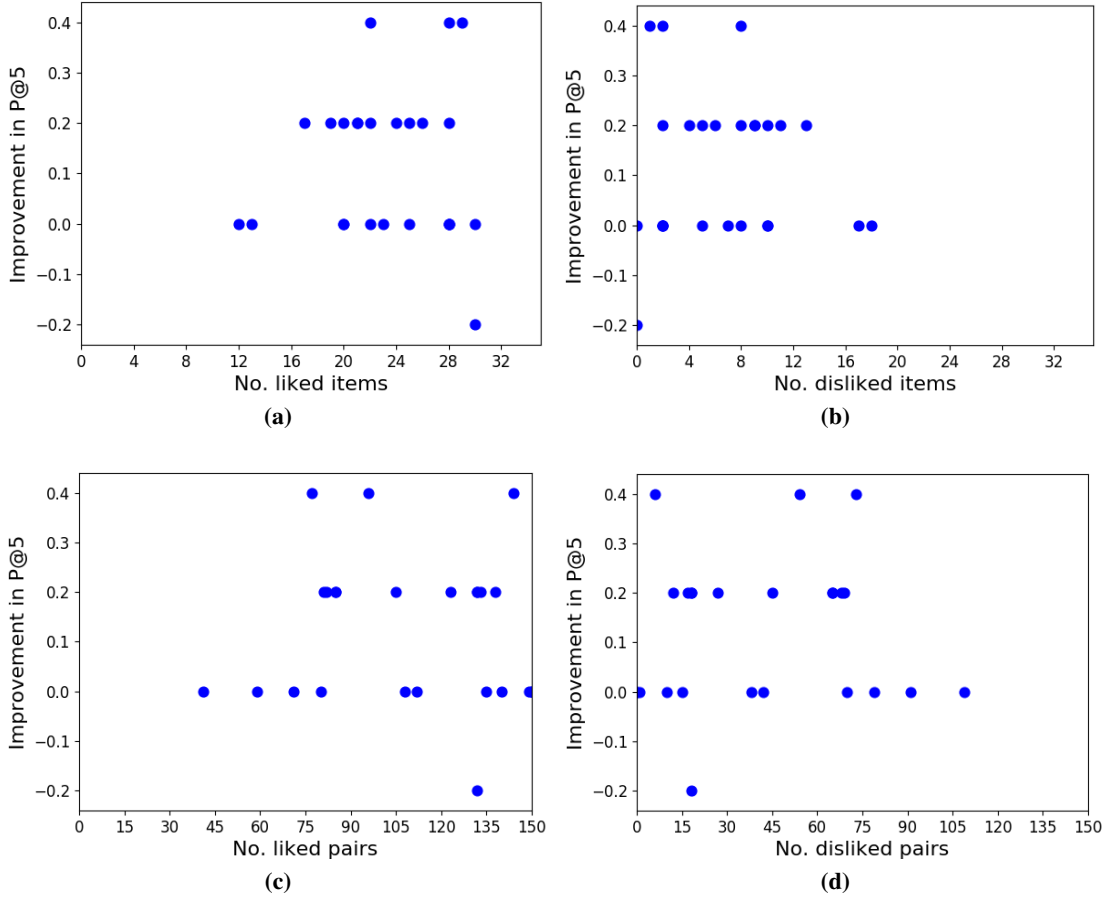


Figure 5.6: P@5-improvement w.r.t. feedback size (Books).

of Success) for the respective users in the ranked list of recommendations.

Limitations imposed by resource constraints. One limitation of our evaluation is the scale of the user study. Evaluating ELIXIR on a larger scale would incur substantially more monetary cost and require design and implementation of a large-scale system suitable for orchestrating and monitoring the longitudinal process of user-system interactions. Resource constraints also impact the possibility of full exploration of the parameter space in this work, such as a thorough search for the best number of latent dimensions d , as that might require the repetition of the whole study. Nevertheless, we evaluated another value $d = 10$ in the movie domain to verify the robustness of ELIXIR. Trends were very similar: P@5 values for item-level, pair-level (top explanations), and item+pair level (top explanations) came out to be 0.520, 0.712, and 0.712, respectively, retaining previously observed statistically significant trends of superiority of configurations involving pair-level feedback over item-level only.




















History h	Recommendations at time T (recs)	Feedback on recs	Explanations exp from h	Feedback on (rec, exp) pairs	(rec, exp) Similarity	Recommendation at time T+1 using only item-level feedback	Recommendation at time T+1 using both item and pair-level feedback
 User u1	 Nightcrawler		 Se7en		Genre: Thriller, Crime	 Joker	 The Chronicles of Narnia: Prince Caspian
	 Fantastic Beasts and Where to Find Them		 The Hobbit: The Battle of the Five Armies		Genre: Fantasy, Adventure		
 User u2	 Divergent		 Doctor Strange		Genre: Science Fiction	 Space Jam	 The BFG
	 Harry Potter and the Deathly Hallows: Part 1		 Charlie and the Chocolate Factory		Content: Based on a book, fantasy		

Figure 5.7: Anecdotal examples in the movie domain.

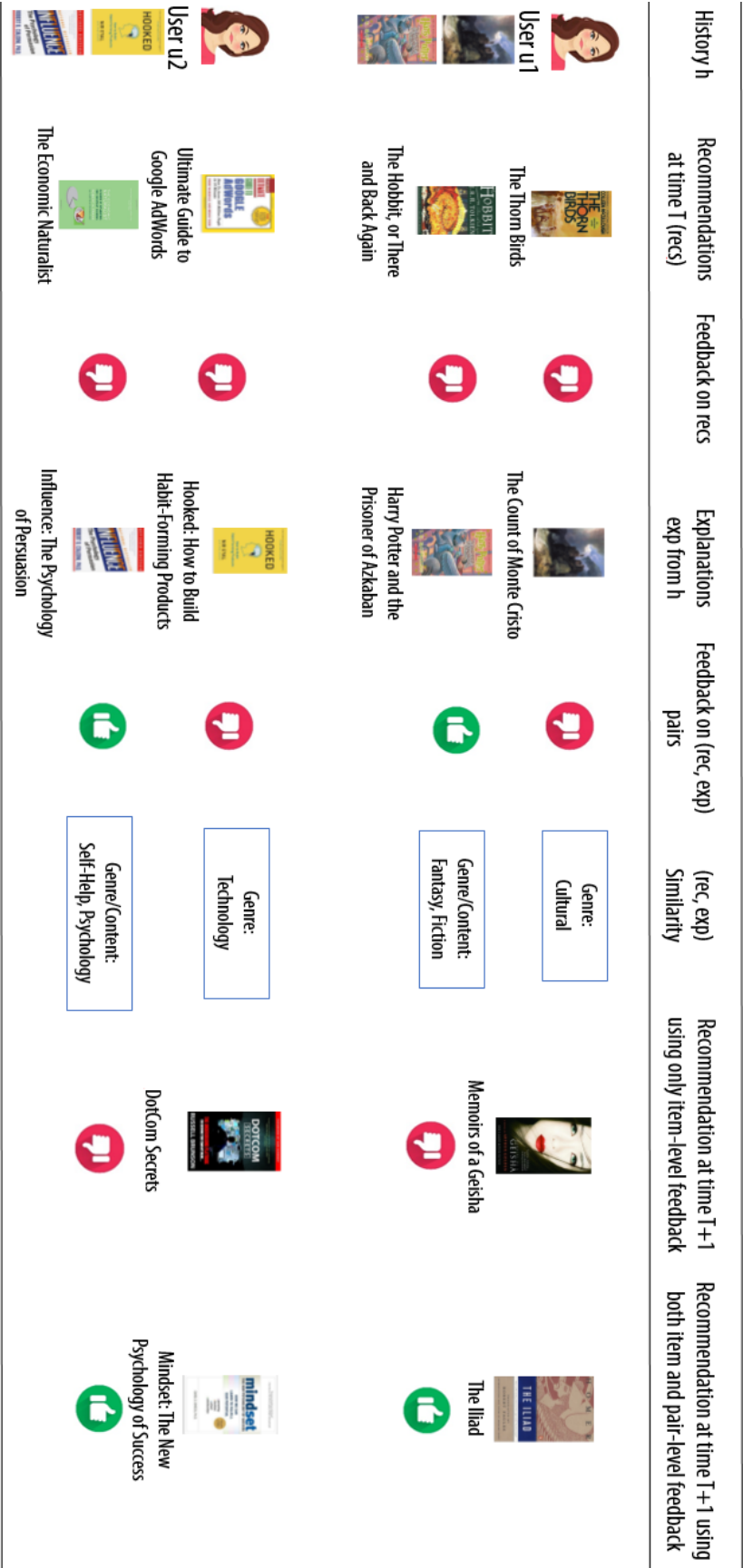


Figure 5.8: Anecdotal examples in the book domain.

5.6 Related work

Critique-based recommenders. In most of the prior works on explainable recommendation, the role of explanations is limited to providing users with insights into the recommendation model. This limits scrutability as users might not have a clue as to how to correct the system’s reasoning. To increase user control over the recommendation process, critique-based recommenders were introduced [Chen and Pu, 2007, Chen and Pu, 2012]. For a broad survey, see Section 2.3.

Critiquing is a method for conversational (a.k.a. sequential and interactive) recommendation that adapts recommended items in response to user preferences on item attributes. Incremental critiquing/tuning [McCarthy et al., 2010, Reilly et al., 2004b, Lee et al., 2020, Chen et al., 2020] was thus proposed to improve recommendation quality over successive recommendation cycles. However, this restricts users to critique/tune based on explicit item properties which are hard to generalize.

Recent works in the form of Deep Language-based Critiquing (DLC) [Wu et al., 2019a, Luo et al., 2020a] address this challenge by accepting arbitrary language-based critiques to improve the recommendations for latent factor-based recommendation models. In [Luo et al., 2020b], Luo et al. improve the complexity of the existing critiquing frameworks by revisiting critiquing from the perspective of Variational Autoencoder (VAE)-based recommendation methods and keyphrase-based interaction.

Existing critique-enabled recommenders mostly focus on negative feedback on concrete features of individual recommendation items. In ELIXIR, we address this limitation by enabling users to give both positive and negative feedback on pairs of recommendation and explanation items.

Set-based preference. Most recommendation approaches rely on signals provided by users on individual items. Another mechanism of eliciting preference is to ask users for feedback on itemsets. Such set-based preference annotations help in faster learning of user interests, especially in cold start situations [Chang et al., 2015]. Moreover, users who are not willing to provide explicit feedback on individual items due to privacy concerns may agree to provide a single rating to a set of items, as it provides a certain level of information abstraction. At the same time, from the given set-based rating, some information regarding item-wise preference can be inferred. In the same vein, in [Sharma et al., 2019], authors gathered users’ preferences on itemsets and developed a collaborative filtering method to predict ratings for individual items in the set.

Apart from understanding user profiles, set-based learning is also useful in works that have focused on recommending lists of items or bundles of items to users such as recommendation of music playlists [Aizenberg et al., 2012], travel packages [Liu et al., 2011], and reading lists [Liu et al., 2014b]. ELIXIR reinforces this viability of set-based feedback.

5.7 Conclusion

In this chapter, we have shown how explanations for recommendations can be made actionable by incorporating user feedback on pairs of items into recommender systems. ELIXIR is a human-in-the-loop system that proactively elicits lightweight user feedback on the similarity of recommendation and explanation pairs. ELIXIR subsequently densifies this feedback using a smart combination of label propagation and locality sensitive hashing, learns user-specific item representations using a soft-constraint-regularized optimization, and seamlessly injects these learned signals into the underlying recommender.

We instantiated this framework with a major family of recommender models based on personalized PageRank, exemplified by the RecWalk method. Our experimental evaluation, based on a longitudinal user study, showed major gains in recommendation quality. This demonstrates the power of the proposed ELIXIR framework to learn more discriminative latent features about user preferences, which are disregarded in traditional item-level ratings.

Future work would naturally focus on extending ELIXIR to other families of recommenders such as matrix/tensor factorization or neural methods, exploring alternative strategies for absorbing pairwise feedback, and investigating the effectiveness of ELIXIR for long-tail users with sparse profiles.

6

CONCLUSIONS AND OUTLOOK

This thesis investigates explainability and scrutability of recommender systems. Findings from our studies demonstrate the benefits of explanations for both end users and service-providers: users gain insight into the personalization process, and service providers enhance their users' experiences by offering more transparency and facilitating user control through feedback on explanations.

In Chapter 3, we presented FAIRY, a framework for explaining black-box recommendations by discovering and ranking relationships between a user and her recommendation item. The practical viability of FAIRY implies its applicability in recommender systems with limited transparency or with complex underlying models where generating faithful explanations is not feasible.

Chapter 4 presented PRINCE, our contribution towards provider-side explainability. PRINCE generates counterfactual explanations that are grounded in the user's own actions. Results from real user studies show that PRINCE explanations are more useful than path-based justifications, corroborating the importance of generating faithful explanations.

In Chapter 5, we presented ELIXIR, a framework for leveraging users' feedback on explanations to improve their recommendations. The results in this chapter show that the role of explanations is not limited to mere insights into the system; they can also be used for preference elicitation and subsequent model improvement. This suggests that there is scope for improving the performance of existing recommenders using explanations.

Our contributions described in this thesis have given rise to many questions in need of further investigation. Below we describe some conceivable extensions and future directions for research on explainable recommendations.

Relationships as explanations. A natural progression of our work on explaining black-box recommendations is to improve the coverage of the discovered relationships between users and their recommendations and to introduce multi-criteria ranking schemes. To discover as many relationships as possible, a browser extension can be used to record all user's heterogeneous inputs to the system. At the time of generating explanations, it would be helpful to allow users to select multiple criteria (e.g., diversity, coherence, simplicity) based on which the discovered relationships can be sorted. Another perceived extension is to devise methods that translate relationships into natural language explanations.

Counterfactual explanations. Research on counterfactual explanations is still in its initial stage. Some future directions related to the generation of such explanations for recommendations include generating counterfactual explanations in neural recommenders, finding the cause for the top-k recommendations as opposed to only the top-ranked item, and studying users' behaviors when they are provided with counterfactual explanations and whether it would be different from when explanations are not counterfactual.

Explanations in action. The results of our studies indicate the benefit of leveraging user feedback on explanation items for improving their recommendations. As explicit feedback is often scarce, it would be beneficial to investigate the possibility of capturing implicit feedback on explanation items. Moreover, the data on pair-level user feedback collected in our user studies could pave the way for designing methods to simulate such feedback.

Offline evaluation of explanations. Evaluating explanations often requires real user studies, and hence is very costly. The community will certainly benefit from introducing benchmarks or developing simulators that can mimic users' responses to explanations.

We hope that this thesis sparks interest in the community towards fulfilling some of these goals and pushing forward the mindsets and infrastructures required for trustworthy AI.

LIST OF FIGURES

3.1	FAIRY: Toy interaction graph for a Quora user.	28
3.2	FAIRY: Logical schemata for the Quora and Last.fm platforms.	35
3.3	FAIRY: Performance of user-specific models for ranking explanations.	38
4.1	PRINCE: Sample explanation.	46
4.2	PRINCE: Toy Example.	53
5.1	ELIXIR: Example illustrating the intuitions.	67
5.2	ELIXIR: Per-user volume of feedback by type.	82
5.3	ELIXIR: Key influencers behind feedback assessments.	83
5.4	ELIXIR: Connecting performance gains with profile diversity.	84
5.5	ELIXIR: P@5-improvement w.r.t. feedback size (Movies).	85
5.6	ELIXIR: P@5-improvement w.r.t. feedback size (Books).	86
5.7	ELIXIR: Anecdotal examples in the movie domain.	87
5.8	ELIXIR: Anecdotal examples in the book domain.	88

LIST OF TABLES

2.1	Common metrics used for evaluating recommendations	12
2.2	Examples of different explanation styles.	18
3.1	FAIRY: User study statistics.	36
3.2	FAIRY: Accuracy compared with baselines.	37
3.3	FAIRY: Ablation study results.	39
3.4	FAIRY: Effect of sampling strategy on performance.	40
3.5	FAIRY: Anecdotal examples.	41
4.1	PRINCE: Properties of the Amazon and Goodreads samples.	55
4.2	PRINCE: Average sizes of counterfactual explanations.	57
4.3	PRINCE: Average runtime.	57
4.4	PRINCE: Anecdotal examples.	58
4.5	PRINCE: Results from the AMT measurement study on usefulness.	60
4.6	PRINCE: Explanations from PRINCE vis-à-vis CredPaths	61
4.7	PRINCE: Turkers' comments on their score justifications.	61
5.1	ELIXIR: Notation for salient concepts.	69
5.2	ELIXIR: Annotations per user over stages of the study.	74
5.3	ELIXIR: Comparison of different modes of feedback incorporation.	79



FAIRY: USER STUDY GUIDELINE

Schedule

As mentioned in the invitation letter, users should spend 20 hours in total. They can work remotely but they should spend 60 minutes on Mondays, Wednesdays, and Fridays. The schedules for the first and second sessions are different from that of other sessions.

First Session

Each user has to create a proxy account in both Quora and Last.fm as described later in this document. Accounts should be created on the same day that the kickoff meeting happens.

Second Session

In this session, users should do the following tasks:

1. Users should start following each other as described next. For this, we assign an ID to each user in the kickoff meeting.

Each user should refer to her corresponding row (and not the column) in Table A.3 and follow all the users listed there on both Quora and Last.fm.

2. Users should interact with both Quora and Last.fm by doing a minimum number of activities described later. Note that users are allowed to do only the activities of types present in Table A.2.
3. After doing the activities, users should visit the activity and content page (in Quora), and the library page (in Last.fm) to make sure they have done the minimum number of activities.
4. **At the end of each session** users should visit a number of pages described later, save them in “Webpage complete” mode and send them to us. Note that, for each platform there is an extra file called “selected_rec.s.txt” that needs to be sent to us.

Other Sessions

Starting from the third session, users are expected to do the following tasks:

- Interacting with both Quora and Last.fm
- Visiting the activity and content page (in Quora) and the library page (in Last.fm) at the end of the session to make sure they have done the minimum number of activities.
- Sending us a number of files described later.
- Evaluating pairs of relations.

End of Study

At the end of the study, users are asked to delete all the accounts created by them. Note that users should delete their accounts only after receiving the instructions from us.

Creating proxy accounts

We need users to create an email account with which they can register on both Quora and Last.fm. Users should enter their real name when signing up. However, there is no restriction for the username. **Note that these accounts must be used solely for the purpose of this study. After creating the email accounts and signing up in the mentioned platforms, please make sure that you sign off from the email account..**

Last.fm

To create an account in Last.fm, users should visit <https://www.last.fm/join>, and sign up with the proxy account they have created. After clicking the “Create my account”, users should check their email and confirm their membership.

Quora

Users should visit [Quora](#) website. Clicking on “Continue with Email”, they can start registering as a member. Note that users should enter their real name as they have to accept the terms of agreement. As users are all in Germany, a window with the title “you can now join Quora in German” will appear. Users should click on “Decide later”. Next, they should select 10 topics to initiate their profile. For this, they should refer to Table A.1 to find the topics they should follow. In the end, users should check their email to confirm their membership.

Activities

Table A.2 shows the actions that users are allowed to do on Quora and Last.fm together with a minimum (or maximum) number of activities expected in either each session or in total. **Users should stick to actions shown in Table A.2. They should not fill up their profile with additional data (such as their place of residence).**

	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10
Music	✗									
Health	✗	✗								
Books	✗	✗	✗							
Food	✗	✗	✗	✗						
Visiting and Travel	✗	✗	✗	✗	✗					
Psychology	✗	✗	✗	✗	✗	✗				
Fashion and Style	✗	✗	✗	✗	✗	✗	✗			
Finance	✗	✗	✗	✗	✗	✗	✗	✗		
Politics	✗	✗	✗	✗	✗	✗	✗	✗	✗	
Education	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
History		✗	✗	✗	✗	✗	✗	✗	✗	✗
Business			✗	✗	✗	✗	✗	✗	✗	✗
Writing				✗	✗	✗	✗	✗	✗	✗
Mathematics					✗	✗	✗	✗	✗	✗
Movies						✗	✗	✗	✗	✗
Cooking							✗	✗	✗	✗
Photography								✗	✗	✗
Sports									✗	✗
Philosophy										✗

Table A.1: Topics to follow when creating a Quora account. The columns are the user IDs and the rows are the topics

To limit the effect of the control study on the general platform outside the group, **users should follow only 3-5 users outside the participants of the user study on both Quora and Last.fm.**

Here we list some important points regarding activities in Quora and Last.fm.

Quora

- In each session, users should ask at least 4 questions, provide answers for at least 4 questions and upvote/follow at least 4 answers/questions per session.
- Throughout the whole study, users should follow only 3-5 external users.
- Users should not ask/write objectionable questions/answers (e.g., the ones that are racist or sexist).

Last.fm

- Users should listen to at least 6 tracks per session. To utilize the time in each session, **users should stop playing a track after a short while.** Normally, after around 2 minutes, the track is added to the user history. To ensure the inclusion of each song, users can visit their history of scrobbles (under the Library tab in their profile).
- Users should avoid simply playing the recommended list to fill up their history.

- Users should not listen to multiple tracks simultaneously.
- Users should not scrobble any song outside of a session.
- Users should love at least 6 tracks per session. Note that users may love a track without playing it.
- During the whole study, users should follow 3-5 external users.

Last.fm	Quora
1. listening to a track (at least 6 per session) 2. following a user (3-5 in total) 3. loving a track (at least 6 per session)	1. asking question (at least 4 per session) 2. answering a question (at least 4 per session) 3. following a question or upvoting an answer (at least 4 per session) 4. following a person (3-5 in all sessions)

Table A.2: Allowed actions in Last.fm and Quora

User network

	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10
U1		✕	✕		✕	✕		✕		
U2	✕		✕			✕	✕		✕	
U3	✕	✕		✕	✕	✕				
U4		✕					✕	✕	✕	✕
U5			✕	✕			✕		✕	✕
U6		✕		✕	✕		✕	✕		
U7			✕		✕	✕		✕		✕
U8				✕	✕	✕			✕	✕
U9	✕	✕		✕	✕					✕
U10				✕		✕	✕	✕	✕	

Table A.3: Network of users. Users should find their followees in their row.

Files

At the end of each session, users should visit specific pages listed below, save the HTML source in “Webpage complete” mode and send them to us.

Note that users should scroll down the pages (to be saved) until the end. Users should put the files from Quora and Last.fm in separate folders named “quora” and “lastfm” respectively.

Other than the HTML pages, users should select **non-obvious recommendations** (whose relationship to their profiles is not clear) and put their urls in a separate file.

The files for each platform are as follows:

Last.fm

- Page of users followed by the user. All users can access this page under the tab “following” in their profile. “Profile pic” → “view profile” → “following tab”
- Page of the followers which can be found under the “followers” tab in their profile.
- Page of recommended artists accessible via link
“<https://www.last.fm/home/artists>”
- Page of recommended albums accessible via link
“<https://www.last.fm/home/albums>”
- Page of recommended tracks accessible via link
“<https://www.last.fm/home/tracks>”
- A text file where each line contains the link to a non-obvious recommended artist, album, or track. Note that users should skim over all the recommendations, but they need not examine each item in detail. In the rare cases where users strongly believe that all received recommendations are obvious, they still need to send us an empty file.

Quora

- Activity page. This page can be found by clicking “Profile pic” → “Profile” → “Activity”
- Page of followed topics. “profile pic” → “Profile” → “Topics”
- Following page. “profile pic” → “Profile” → “Following”
- Content page accessible via link “<https://www.quora.com/content>”.
- Home page. This page lists the recommended items generated by Quora. To reach this page, users should simply click on “Home button”.
- A text file where each line contains the link to a surprising recommended item. Each item can be either a question, an answer or a topic. For example,
“<https://www.quora.com/topic/children>” ,
“<https://www.quora.com/does-god-exists>” ,
“<https://www.quora.com/does-god-exists/answer/azinmatin>”
are example links to a topic, a question and an answer respectively. Note that in Quora, “Link” is another type of recommendation. Users should exclude these items from their choices as they are normally redirected to another websites.

Evaluation of relationships

Starting from the third session, users should give us feedback on the possible relations between their profiles and the recommended items. For this, they receive a file containing pairs of relations. They should decide which one is more surprising and which is more relevant (or useful). Here is an example of a pair of relations together with some meta data:

Recommended item of type topic: Children

Link to recommended item: Link to Children topic Webpage

First explanation: Syndy — follows — Frank — follows — Children

Second explanation: Syndy — follows — Life Lessons — super-category — Children

Which explanation is more surprising to you?:

Which explanation is more relevant/useful to you?:

First explanation's links: Link to Syndy — follow — Link to Frank — follows — Link to Children to Frank

Second explanation's links: Link to Syndy — follow — Link to Life Lessons — super-category — Link to Children

Users should answer the questions in red. For example, if user thinks the first explanation is more surprising than the second, she should write “f” (stands for first) in front of “Which explanation is more surprising to you?”. Here is a valid answer to the red fields:

Which explanation is more surprising to you? f

Which explanation is more useful/relevant to you? s

If users need more information regarding each component on the explanation path, they can visit their corresponding Webpage using the URLs written under “First explanation's links” or “Second explanation's links”.

For each session, a separate sheet containing the relation pairs will be shared with each user. Users should always pay attention to the order of the questions as they are randomly ordered. To avoid task fatigue, users should fill out the sheet for each session before the next one starts.

BIBLIOGRAPHY

- [Abdollahi and Nasraoui, 2017] Abdollahi, B. and Nasraoui, O. (2017). Using explainability for constrained matrix factorization. In *Proceedings of the Eleventh ACM Conference on Recommender Systems, RecSys 2017, Como, Italy, August 27-31, 2017*, pages 79–83.
- [Agarwal et al., 2015] Agarwal, D., Chen, B., He, Q., Hua, Z., Lebanon, G., Ma, Y., Shiv-
aswamy, P., Tseng, H., Yang, J., and Zhang, L. (2015). Personalizing linkedin feed. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015*, pages 1651–1660.
- [Aggarwal et al., 1999] Aggarwal, C. C., Wolf, J. L., Wu, K., and Yu, P. S. (1999). Horting
hatches an egg: A new graph-theoretic approach to collaborative filtering. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, CA, USA, August 15-18, 1999*, pages 201–212.
- [Ahn et al., 2007] Ahn, J., Brusilovsky, P., Grady, J., He, D., and Syn, S. Y. (2007). Open user
profiles for adaptive news systems: help or harm? In *Proceedings of the 16th International Conference on World Wide Web, WWW 2007, Banff, Alberta, Canada, May 8-12, 2007*, pages 11–20. ACM.
- [Ai et al., 2018] Ai, Q., Azizi, V., Chen, X., and Zhang, Y. (2018). Learning heterogeneous
knowledge base embeddings for explainable recommendation. *Algorithms*, 11(9):137.
- [Aizenberg et al., 2012] Aizenberg, N., Koren, Y., and Somekh, O. (2012). Build your own
music recommender by modeling internet radio streams. In *Proceedings of the 21st World Wide Web Conference 2012, WWW 2012, Lyon, France, April 16-20, 2012*, pages 1–10.
- [Ananny and Crawford, 2018] Ananny, M. and Crawford, K. (2018). Seeing without knowing:
Limitations of the transparency ideal and its application to algorithmic accountability. *New Media Soc.*, 20(3):973–989.
- [Andersen et al., 2007] Andersen, R., Borgs, C., Chayes, J. T., Hopcroft, J. E., Mirrokni, V. S.,
and Teng, S. (2007). Local computation of pagerank contributions. In *Algorithms and Models for the Web-Graph, 5th International Workshop, WAW 2007, San Diego, CA, USA, December 11-12, 2007, Proceedings*, volume 4863 of *Lecture Notes in Computer Science*, pages 150–165.
- [Andersen et al., 2006] Andersen, R., Chung, F. R. K., and Lang, K. J. (2006). Local graph
partitioning using pagerank vectors. In *47th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2006), 21-24 October 2006, Berkeley, California, USA, Proceedings*, pages 475–486.
- [Andreou et al., 2018] Andreou, A., Venkatadri, G., Goga, O., Gummadi, K. P., Loiseau, P.,
and Mislove, A. (2018). Investigating ad transparency mechanisms in social media: A case
study of facebook’s explanations. In *25th Annual Network and Distributed System Security*

- Symposium, NDSS 2018, San Diego, California, USA, February 18-21, 2018.*
- [Anelli et al., 2020] Anelli, V. W., Di Noia, T., Di Sciascio, E., Ragone, A., and Trotta, J. (2020). Semantic interpretation of top-n recommendations. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–1.
- [Arora et al., 2017] Arora, S., Liang, Y., and Ma, T. (2017). A simple but tough-to-beat baseline for sentence embeddings. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.
- [Avinesh et al., 2019] Avinesh, P. V. S., Ren, Y., Meyer, C. M., Chan, J., Bao, Z., and Sanderson, M. (2019). J3R: joint multi-task learning of ratings and review summaries for explainable recommendation. In *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2019, Würzburg, Germany, September 16-20, 2019, Proceedings, Part III*, volume 11908 of *Lecture Notes in Computer Science*, pages 339–355.
- [Avrachenkov et al., 2007] Avrachenkov, K., Litvak, N., Nemirovsky, D., and Osipova, N. (2007). Monte carlo methods in pagerank computation: When one iteration is sufficient. *SIAM J. Numer. Anal.*, 45(2):890–904.
- [Bach et al., 2015] Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.-R., and Samek, W. (2015). On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140.
- [Bahmani et al., 2010] Bahmani, B., Chowdhury, A., and Goel, A. (2010). Fast incremental and personalized pagerank. *Proc. VLDB Endow.*, 4(3):173–184.
- [Balog and Radlinski, 2020] Balog, K. and Radlinski, F. (2020). Measuring recommendation explanation quality: The conflicting goals of explanations. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pages 329–338.
- [Balog et al., 2019] Balog, K., Radlinski, F., and Arakelyan, S. (2019). Transparent, scrutable and explainable user models for personalized recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*, pages 265–274.
- [Bast and Haussmann, 2015] Bast, H. and Haussmann, E. (2015). More accurate question answering on freebase. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM 2015, Melbourne, VIC, Australia, October 19 - 23, 2015*, pages 1431–1440.
- [Behrens et al., 2018] Behrens, F., Bischoff, S., Ladenburger, P., Rückin, J., Seidel, L., Stolp, F., Vaichenker, M., Ziegler, A., Mottin, D., Aghaei, F., Müller, E., Preusse, M., Müller, N., and Hunger, M. (2018). Metaexp: Interactive explanation and exploration of large knowledge graphs. In *Companion of the The Web Conference 2018 on The Web Conference 2018, WWW 2018, Lyon, France, April 23-27, 2018*, pages 199–202.
- [Beigi et al., 2020] Beigi, G., Mosallanezhad, A., Guo, R., Alvari, H., Nou, A., and Liu, H. (2020). Privacy-aware recommendation with private-attribute protection using adversarial

- learning. In *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining, Houston, TX, USA, February 3-7, 2020*, pages 34–42.
- [Bennett and Lanning, 2007] Bennett, J. and Lanning, S. (2007). The netflix prize. In *Proceedings of KDD cup and workshop*, volume 2007, page 35.
- [Beutel et al., 2019] Beutel, A., Chen, J., Doshi, T., Qian, H., Wei, L., Wu, Y., Heldt, L., Zhao, Z., Hong, L., Chi, E. H., and Goodrow, C. (2019). Fairness in recommendation ranking through pairwise comparisons. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, pages 2212–2220.
- [Bianchi et al., 2017] Bianchi, F., Palmonari, M., Cremaschi, M., and Fersini, E. (2017). Actively learning to rank semantic associations for personalized contextual exploration of knowledge graphs. In *The Semantic Web - 14th International Conference, ESWC 2017, Portorož, Slovenia, May 28 - June 1, 2017, Proceedings, Part I*, volume 10249 of *Lecture Notes in Computer Science*, pages 120–135.
- [Bilgic and Mooney, 2005] Bilgic, M. and Mooney, R. J. (2005). Explaining recommendations: Satisfaction vs. promotion. In *In Proceedings of Beyond Personalization Workshop, IUI 2005, the Workshop on the Next Stage of Recommender Systems Research (IUI), 2005*, pages 13–18.
- [Billsus and Pazzani, 2000] Billsus, D. and Pazzani, M. J. (2000). User modeling for adaptive news access. *User Model. User Adapt. Interact.*, 10(2-3):147–180.
- [Bordes et al., 2013] Bordes, A., Usunier, N., García-Durán, A., Weston, J., and Yakhnenko, O. (2013). Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 2787–2795.
- [Bostandjiev et al., 2012] Bostandjiev, S., O'Donovan, J., and Höllerer, T. (2012). Tasteweights: a visual interactive hybrid recommender system. In *Sixth ACM Conference on Recommender Systems, RecSys '12, Dublin, Ireland, September 9-13, 2012*, pages 35–42.
- [Breese et al., 1998] Breese, J. S., Heckerman, D., and Kadie, C. M. (1998). Empirical analysis of predictive algorithms for collaborative filtering. In *UAI '98: Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence, University of Wisconsin Business School, Madison, Wisconsin, USA, July 24-26, 1998*, pages 43–52.
- [Brin and Page, 1998] Brin, S. and Page, L. (1998). The anatomy of a large-scale hypertextual web search engine. *Comput. Networks*, 30(1-7):107–117.
- [Burges et al., 2005] Burges, C. J. C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., and Hullender, G. N. (2005). Learning to rank using gradient descent. In *Machine Learning, Proceedings of the Twenty-Second International Conference (ICML 2005), Bonn, Germany, August 7-11, 2005*, volume 119 of *ACM International Conference Proceeding Series*, pages 89–96.

- [Cao et al., 2007] Cao, Z., Qin, T., Liu, T., Tsai, M., and Li, H. (2007). Learning to rank: from pairwise approach to listwise approach. In *Machine Learning, Proceedings of the Twenty-Fourth International Conference (ICML 2007), Corvallis, Oregon, USA, June 20-24, 2007*, volume 227 of *ACM International Conference Proceeding Series*, pages 129–136.
- [Carmona et al., 2015] Carmona, V. I. S., Rocktäschel, T., Riedel, S., and Singh, S. (2015). Towards extracting faithful and descriptive representations of latent variable models. In *2015 AAAI Spring Symposia, Stanford University, Palo Alto, California, USA, March 22-25, 2015*.
- [Carterette et al., 2008] Carterette, B., Bennett, P. N., Chickering, D. M., and Dumais, S. T. (2008). Here or there. In *Advances in Information Retrieval, 30th European Conference on IR Research, ECIR 2008, Glasgow, UK, March 30-April 3, 2008. Proceedings*, volume 4956 of *Lecture Notes in Computer Science*, pages 16–27.
- [Catherine et al., 2017a] Catherine, R., Mazaitis, K., Eskénazi, M., and Cohen, W. W. (2017a). Explainable entity-based recommendations with knowledge graphs. In *Proceedings of the Poster Track of the 11th ACM Conference on Recommender Systems (RecSys 2017), Como, Italy, August 28, 2017*, volume 1905 of *CEUR Workshop Proceedings*.
- [Catherine et al., 2017b] Catherine, R., Mazaitis, K., Eskénazi, M., and Cohen, W. W. (2017b). Explainable entity-based recommendations with knowledge graphs. In *Proceedings of the Poster Track of the 11th ACM Conference on Recommender Systems (RecSys 2017), Como, Italy, August 28, 2017*, volume 1905 of *CEUR Workshop Proceedings*.
- [Cer et al., 2018] Cer, D., Yang, Y., Kong, S., Hua, N., Limtiaco, N., John, R. S., Constant, N., Guajardo-Cespedes, M., Yuan, S., Tar, C., Strophe, B., and Kurzweil, R. (2018). Universal sentence encoder for english. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018: System Demonstrations, Brussels, Belgium, October 31 - November 4, 2018*, pages 169–174.
- [Chang et al., 2015] Chang, S., Harper, F. M., and Terveen, L. G. (2015). Using groups of items to bootstrap new users in recommender systems. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing, CSCW 2015, Vancouver, BC, Canada, March 14 - 18, 2015*, pages 1258–1269.
- [Chang et al., 2016] Chang, S., Harper, F. M., and Terveen, L. G. (2016). Crowd-based personalized natural language explanations for recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, September 15-19, 2016*, pages 175–182.
- [Chen et al., 2018a] Chen, C., Zhang, M., Liu, Y., and Ma, S. (2018a). Neural attentional rating regression with review-level explanations. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, pages 1583–1592.
- [Chen et al., 2017] Chen, D., Fraiberger, S. P., Moakler, R., and Provost, F. J. (2017). Enhancing transparency and control when drawing data-driven inferences about individuals. *Big Data*, 5(3):197–212.

- [Chen et al., 2021] Chen, H., Chen, X., Shi, S., and Zhang, Y. (2021). Generate natural language explanations for recommendation. *CoRR*, abs/2101.03392.
- [Chen and Pu, 2007] Chen, L. and Pu, P. (2007). Hybrid critiquing-based recommender systems. In *Proceedings of the 12th International Conference on Intelligent User Interfaces, IUI 2007, Honolulu, Hawaii, USA, January 28-31, 2007*, pages 22–31.
- [Chen and Pu, 2012] Chen, L. and Pu, P. (2012). Critiquing-based recommenders: survey and emerging trends. *User Modeling and User-Adapted Interaction*, 22(1):125–150.
- [Chen et al., 2019a] Chen, X., Chen, H., Xu, H., Zhang, Y., Cao, Y., Qin, Z., and Zha, H. (2019a). Personalized fashion recommendation with visual explanations based on multimodal attention network: Towards visually explainable recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*, pages 765–774.
- [Chen et al., 2019b] Chen, X., Chen, H., Xu, H., Zhang, Y., Cao, Y., Qin, Z., and Zha, H. (2019b). Personalized fashion recommendation with visual explanations based on multimodal attention network: Towards visually explainable recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*, pages 765–774.
- [Chen et al., 2016] Chen, X., Qin, Z., Zhang, Y., and Xu, T. (2016). Learning to rank features for recommendation over multiple categories. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, SIGIR 2016, Pisa, Italy, July 17-21, 2016*, pages 305–314.
- [Chen et al., 2018b] Chen, X., Xu, H., Zhang, Y., Tang, J., Cao, Y., Qin, Z., and Zha, H. (2018b). Sequential recommendation with user memory networks. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, WSDM 2018, Marina Del Rey, CA, USA, February 5-9, 2018*, pages 108–116.
- [Chen et al., 2019c] Chen, X., Zhang, Y., and Qin, Z. (2019c). Dynamic explainable recommendation based on neural attentive models. 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*, pages 53–60.
- [Chen et al., 2020] Chen, Z., Wang, X., Xie, X., Parsana, M., Soni, A., Ao, X., and Chen, E. (2020). Towards explainable conversational recommendation. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 2994–3000.
- [Chen et al., 2019d] Chen, Z., Wang, X., Xie, X., Wu, T., Bu, G., Wang, Y., and Chen, E. (2019d). Co-attentive multi-task learning for explainable recommendation. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 2137–2143.

- [Cheng et al., 2019] Cheng, W., Shen, Y., Huang, L., and Zhu, Y. (2019). Incorporating interpretability into latent factor models via fast influence analysis. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, pages 885–893.
- [Cheng et al., 2018] Cheng, Z., Ding, Y., Zhu, L., and Kankanhalli, M. S. (2018). Aspect-aware latent factor model: Rating prediction with ratings and reviews. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, pages 639–648.
- [Christoffel et al., 2015] Christoffel, F., Paudel, B., Newell, C., and Bernstein, A. (2015). Blockbusters and wallflowers: Accurate, diverse, and scalable recommendations with random walks. In *Proceedings of the 9th ACM Conference on Recommender Systems, RecSys 2015, Vienna, Austria, September 16-20, 2015*, pages 163–170.
- [Clancey, 1983] Clancey, W. J. (1983). The epistemology of a rule-based expert system - A framework for explanation. *Artif. Intell.*, 20(3):215–251.
- [Cooper et al., 2014] Cooper, C., Lee, S., Radzik, T., and Siantos, Y. (2014). Random walks in recommender systems: exact computation and simulations. In *23rd International World Wide Web Conference, WWW '14, Seoul, Republic of Korea, April 7-11, 2014, Companion Volume*, pages 811–816.
- [Cosley et al., 2003] Cosley, D., Lam, S. K., Albert, I., Konstan, J. A., and Riedl, J. (2003). Is seeing believing?: how recommender system interfaces affect users' opinions. In *Proceedings of the 2003 Conference on Human Factors in Computing Systems, CHI 2003, Ft. Lauderdale, Florida, USA, April 5-10, 2003*, pages 585–592.
- [Costa et al., 2018] Costa, F., Ouyang, S., Dolog, P., and Lawlor, A. (2018). Automatic generation of natural language explanations. In *Proceedings of the 23rd International Conference on Intelligent User Interfaces Companion, Tokyo, Japan, March 07-11, 2018*, pages 57:1–57:2.
- [Cotter et al., 2017] Cotter, K., Cho, J., and Rader, E. J. (2017). Explaining the news feed algorithm: An analysis of the "news feed fyi" blog. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems, Denver, CO, USA, May 06-11, 2017, Extended Abstracts*, pages 1553–1560.
- [Covert et al., 2020] Covert, I., Lundberg, S. M., and Lee, S. (2020). Understanding global feature contributions with additive importance measures. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.
- [Cramer et al., 2008] Cramer, H. S. M., Evers, V., Ramlal, S., van Someren, M., Rutledge, L., Stash, N., Aroyo, L., and Wielinga, B. J. (2008). The effects of transparency on trust in and acceptance of a content-based art recommender. *User Model. User Adapt. Interact.*, 18(5):455–496.
- [Csáji et al., 2014] Csáji, B. C., Jungers, R. M., and Blondel, V. D. (2014). Pagerank optimization by edge selection. *Discret. Appl. Math.*, 169:73–87.

- [Czarkowski, 2006] Czarkowski, M. (2006). *A scrutable adaptive hypertext*. PhD thesis, University of Sydney, Australia.
- [Das et al., 2011] Das, M., Amer-Yahia, S., Das, G., and Yu, C. (2011). MRI: meaningful interpretations of collaborative ratings. *Proc. VLDB Endow.*, 4(11):1063–1074.
- [Degemmis et al., 2007] Degemmis, M., Lops, P., and Semeraro, G. (2007). A content-collaborative recommender that exploits wordnet-based user profiles for neighborhood formation. *User Model. User Adapt. Interact.*, 17(3):217–255.
- [Degemmis et al., 2008] Degemmis, M., Lops, P., Semeraro, G., and Basile, P. (2008). Integrating tags in a semantic content-based recommender. In *Proceedings of the 2008 ACM Conference on Recommender Systems, RecSys 2008, Lausanne, Switzerland, October 23-25, 2008*, pages 163–170.
- [Deng et al., 2011] Deng, H., Han, J., Zhao, B., Yu, Y., and Lin, C. X. (2011). Probabilistic topic models with biased propagation on heterogeneous information networks. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, CA, USA, August 21-24, 2011*, pages 1271–1279.
- [Desrosiers and Karypis, 2011] Desrosiers, C. and Karypis, G. (2011). A comprehensive survey of neighborhood-based recommendation methods. In *Recommender Systems Handbook*, pages 107–144.
- [Dong et al., 2017] Dong, Y., Chawla, N. V., and Swami, A. (2017). metapath2vec: Scalable representation learning for heterogeneous networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, August 13 - 17, 2017*, pages 135–144.
- [Donkers and Ziegler, 2020] Donkers, T. and Ziegler, J. (2020). Leveraging arguments in user reviews for generating and explaining recommendations. *Datenbank-Spektrum*, 20(2):181–187.
- [Doshi-Velez et al., 2017] Doshi-Velez, F., Kortz, M., Budish, R., Bavitz, C., Gershman, S., O’Brien, D., Schieber, S., Waldo, J., Weinberger, D., and Wood, A. (2017). Accountability of AI under the law: The role of explanation. *CoRR*, abs/1711.01134.
- [Edwards et al., 2014] Edwards, C., Edwards, A., Spence, P. R., and Shelton, A. K. (2014). Is that a bot running the social media feed? testing the differences in perceptions of communication quality for a human agent and a bot agent on twitter. *Comput. Hum. Behav.*, 33:372–376.
- [Eksombatchai et al., 2018] Eksombatchai, C., Jindal, P., Liu, J. Z., Liu, Y., Sharma, R., Sugnet, C., Ulrich, M., and Leskovec, J. (2018). Pixie: A system for recommending 3+ billion items to 200+ million users in real-time. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, pages 1775–1784.
- [Eslami et al., 2015] Eslami, M., Rickman, A., Vaccaro, K., Aleyasen, A., Vuong, A., Karahalios, K., Hamilton, K., and Sandvig, C. (2015). "i always assumed that I wasn’t really that close to [her]": Reasoning about invisible algorithms in news feeds. In *Proceedings of the*

- 33rd Annual ACM Conference on Human Factors in Computing Systems, CHI 2015, Seoul, Republic of Korea, April 18-23, 2015*, pages 153–162.
- [Fan et al., 2019] Fan, W., Ma, Y., Li, Q., He, Y., Zhao, Y. E., Tang, J., and Yin, D. (2019). Graph neural networks for social recommendation. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pages 417–426.
- [Fang et al., 2011] Fang, L., Sarma, A. D., Yu, C., and Bohannon, P. (2011). REX: explaining relationships between entity pairs. *Proc. VLDB Endow.*, 5(3):241–252.
- [Freyne et al., 2010] Freyne, J., Berkovsky, S., Daly, E. M., and Geyer, W. (2010). Social networking feeds: recommending items of interest. In *Proceedings of the 2010 ACM Conference on Recommender Systems, RecSys 2010, Barcelona, Spain, September 26-30, 2010*, pages 277–280.
- [Fu et al., 2020] Fu, Z., Xian, Y., Gao, R., Zhao, J., Huang, Q., Ge, Y., Xu, S., Geng, S., Shah, C., Zhang, Y., and de Melo, G. (2020). Fairness-aware explainable recommendation over knowledge graphs. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pages 69–78.
- [Fusco et al., 2019] Fusco, F., Vlachos, M., Vasileiadis, V., Wardatzky, K., and Schneider, J. (2019). Reconet: An interpretable neural architecture for recommender systems. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 2343–2349.
- [Gao et al., 2019] Gao, J., Wang, X., Wang, Y., and Xie, X. (2019). Explainable recommendation through attentive multi-view learning. 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*, pages 3622–3629.
- [Ghazimatin, 2020] Ghazimatin, A. (2020). Explaining recommendations in heterogeneous networks. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, page 2479.
- [Ghazimatin et al., 2020] Ghazimatin, A., Balalau, O., Saha Roy, R., and Weikum, G. (2020). PRINCE: provider-side interpretability with counterfactual explanations in recommender systems. In *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining, Houston, TX, USA, February 3-7, 2020*, pages 196–204.
- [Ghazimatin et al., 2021] Ghazimatin, A., Pramanik, S., Roy, R. S., and Weikum, G. (2021). ELIXIR: learning from user feedback on explanations to improve recommender models. In *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, pages 3850–3860.
- [Ghazimatin et al., 2019] Ghazimatin, A., Saha Roy, R., and Weikum, G. (2019). FAIRY: A

- framework for understanding relationships between users' actions and their social feeds. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, WSDM 2019, Melbourne, VIC, Australia, February 11-15, 2019*, pages 240–248.
- [Goodman and Flaxman, 2017] Goodman, B. and Flaxman, S. R. (2017). European union regulations on algorithmic decision-making and a "right to explanation". *AI Mag.*, 38(3):50–57.
- [Grcar et al., 2006] Grcar, M., Fortuna, B., Mladenic, D., and Grobelnik, M. (2006). knn versus SVM in the collaborative filtering framework. In *Data Science and Classification, Studies in Classification, Data Analysis, and Knowledge Organization*, pages 251–260.
- [Guo et al., 2020] Guo, Q., Zhuang, F., Qin, C., Zhu, H., Xie, X., Xiong, H., and He, Q. (2020). A survey on knowledge graph-based recommender systems. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–1.
- [Gurevich and Wing, 2016] Gurevich, Y. and Wing, J. M. (2016). Inverse privacy. *Commun. ACM*, 59(7):38–42.
- [Gutta et al., 2000] Gutta, S., Kurapati, K., Lee, K. P., Martino, J., Milanski, J., Schaffer, J. D., and Zimmerman, J. (2000). TV content recommender system. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence, July 30 - August 3, 2000, Austin, Texas, USA*, pages 1121–1122.
- [Hamilton et al., 2014] Hamilton, K., Karahalios, K., Sandvig, C., and Eslami, M. (2014). A path to understanding the effects of algorithm awareness. In *CHI Conference on Human Factors in Computing Systems, CHI'14, Toronto, ON, Canada - April 26 - May 01, 2014, Extended Abstracts*, pages 631–642.
- [Haveliwala, 2003] Haveliwala, T. H. (2003). Topic-sensitive pagerank: A context-sensitive ranking algorithm for web search. *IEEE Trans. Knowl. Data Eng.*, 15(4):784–796.
- [He et al., 2015] He, X., Chen, T., Kan, M., and Chen, X. (2015). Trirank: Review-aware explainable recommendation by modeling aspects. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM 2015, Melbourne, VIC, Australia, October 19 - 23, 2015*, pages 1661–1670.
- [He et al., 2017] He, X., Liao, L., Zhang, H., Nie, L., Hu, X., and Chua, T. (2017). Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web, WWW 2017, Perth, Australia, April 3-7, 2017*, pages 173–182.
- [Heckel et al., 2017] Heckel, R., Vlachos, M., Parnell, T. P., and Dünner, C. (2017). Scalable and interpretable product recommendations via overlapping co-clustering. In *33rd IEEE International Conference on Data Engineering, ICDE 2017, San Diego, CA, USA, April 19-22, 2017*, pages 1033–1044.
- [Herlocker et al., 2000] Herlocker, J. L., Konstan, J. A., and Riedl, J. (2000). Explaining collaborative filtering recommendations. In *CSCW 2000, Proceeding on the ACM 2000 Conference on Computer Supported Cooperative Work, Philadelphia, PA, USA, December*

- 2-6, 2000, pages 241–250.
- [Hijikata et al., 2012] Hijikata, Y., Kai, Y., and Nishida, S. (2012). The relation between user intervention and user satisfaction for information recommendation. In *Proceedings of the ACM Symposium on Applied Computing, SAC 2012, Riva, Trento, Italy, March 26-30, 2012*, pages 2002–2007. ACM.
- [Hong et al., 2012] Hong, L., Bekkerman, R., Adler, J., and Davison, B. D. (2012). Learning to rank social update streams. In *The 35th International ACM SIGIR conference on research and development in Information Retrieval, SIGIR '12, Portland, OR, USA, August 12-16, 2012*, pages 651–660.
- [Hou et al., 2019] Hou, Y., Yang, N., Wu, Y., and Yu, P. S. (2019). Explainable recommendation with fusion of aspect information. *World Wide Web*, 22(1):221–240.
- [Hsieh et al., 2017] Hsieh, C., Yang, L., Cui, Y., Lin, T., Belongie, S. J., and Estrin, D. (2017). Collaborative metric learning. In *Proceedings of the 26th International Conference on World Wide Web, WWW 2017, Perth, Australia, April 3-7, 2017*, pages 193–201.
- [Hu et al., 2018] Hu, B., Shi, C., Zhao, W. X., and Yu, P. S. (2018). Leveraging meta-path based context for top- N recommendation with A neural co-attention model. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, pages 1531–1540.
- [Huang et al., 2018] Huang, J., Zhao, W. X., Dou, H., Wen, J., and Chang, E. Y. (2018). Improving sequential recommendation with knowledge-enhanced memory networks. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*, pages 505–514.
- [Jacovi and Goldberg, 2020] Jacovi, A. and Goldberg, Y. (2020). Towards faithfully interpretable NLP systems: How should we define and evaluate faithfulness? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 4198–4205.
- [Jain and Wallace, 2019] Jain, S. and Wallace, B. C. (2019). Attention is not explanation. In *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)*, pages 3543–3556.
- [Jamali and Ester, 2009] Jamali, M. and Ester, M. (2009). *TrustWalker*: a random walk model for combining trust-based and item-based recommendation. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Paris, France, June 28 - July 1, 2009*, pages 397–406.
- [Jannach and Kreutler, 2007] Jannach, D. and Kreutler, G. (2007). Rapid development of knowledge-based conversational recommender applications with advisor suite. *J. Web Eng.*, 6(2):165–192.
- [Jannach et al., 2016] Jannach, D., Naveed, S., and Jugovac, M. (2016). User control in recommender systems: Overview and interaction challenges. In *E-Commerce and Web*

- Technologies - 17th International Conference, EC-Web 2016, Porto, Portugal, September 5-8, 2016, Revised Selected Papers*, volume 278 of *Lecture Notes in Business Information Processing*, pages 21–33.
- [Jäschke et al., 2007] Jäschke, R., Marinho, L. B., Hotho, A., Schmidt-Thieme, L., and Stumme, G. (2007). Tag recommendations in folksonomies. In *Knowledge Discovery in Databases: PKDD 2007, 11th European Conference on Principles and Practice of Knowledge Discovery in Databases, Warsaw, Poland, September 17-21, 2007, Proceedings*, volume 4702 of *Lecture Notes in Computer Science*, pages 506–514.
- [Jeh and Widom, 2003] Jeh, G. and Widom, J. (2003). Scaling personalized web search. In *Proceedings of the Twelfth International World Wide Web Conference, WWW 2003, Budapest, Hungary, May 20-24, 2003*, pages 271–279.
- [Jiang et al., 2018] Jiang, Z., Liu, H., Fu, B., Wu, Z., and Zhang, T. (2018). Recommendation in heterogeneous information networks based on generalized random walk model and bayesian personalized ranking. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, WSDM 2018, Marina Del Rey, CA, USA, February 5-9, 2018*, pages 288–296.
- [Jin et al., 2019] Jin, Y., Cai, W., Chen, L., Htun, N. N., and Verbert, K. (2019). Musicbot: Evaluating critiquing-based music recommenders with conversational interaction. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*, pages 951–960.
- [Jin et al., 2018] Jin, Y., Tintarev, N., and Verbert, K. (2018). Effects of personal characteristics on music recommender systems with different levels of controllability. In *Proceedings of the 12th ACM Conference on Recommender Systems, RecSys 2018, Vancouver, BC, Canada, October 2-7, 2018*, pages 13–21.
- [Joachims, 2006] Joachims, T. (2006). Training linear svms in linear time. In *Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia, PA, USA, August 20-23, 2006*, pages 217–226.
- [Kabbur et al., 2013] Kabbur, S., Ning, X., and Karypis, G. (2013). FISM: factored item similarity models for top-n recommender systems. In *The 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2013, Chicago, IL, USA, August 11-14, 2013*, pages 659–667.
- [Kang et al., 2018a] Kang, J., Wang, M., Cao, N., Xia, Y., Fan, W., and Tong, H. (2018a). AURORA: auditing pagerank on large graphs. In *IEEE International Conference on Big Data, Big Data 2018, Seattle, WA, USA, December 10-13, 2018*, pages 713–722.
- [Kang et al., 2018b] Kang, W., Wan, M., and McAuley, J. J. (2018b). Recommendation through mixtures of heterogeneous item relationships. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22-26, 2018*, pages 1143–1152.
- [Kasneci et al., 2009] Kasneci, G., Ramanath, M., Sozio, M., Suchanek, F. M., and Weikum, G.

- (2009). STAR: steiner-tree approximation in relationship graphs. In *Proceedings of the 25th International Conference on Data Engineering, ICDE 2009, March 29 2009 - April 2 2009, Shanghai, China*, pages 868–879.
- [Kim et al., 2016] Kim, B., Koyejo, O., and Khanna, R. (2016). Examples are not enough, learn to criticize! criticism for interpretability. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 2280–2288.
- [Knijnenburg et al., 2012a] Knijnenburg, B. P., Bostandjiev, S., O'Donovan, J., and Kobsa, A. (2012a). Inspectability and control in social recommenders. In *Sixth ACM Conference on Recommender Systems, RecSys '12, Dublin, Ireland, September 9-13, 2012*, pages 43–50.
- [Knijnenburg et al., 2012b] Knijnenburg, B. P., Bostandjiev, S., O'Donovan, J., and Kobsa, A. (2012b). Inspectability and control in social recommenders. In *Sixth ACM Conference on Recommender Systems, RecSys '12, Dublin, Ireland, September 9-13, 2012*, pages 43–50.
- [Kong et al., 2013] Kong, X., Cao, B., and Yu, P. S. (2013). Multi-label classification by mining label and instance correlations from heterogeneous information networks. In *The 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2013, Chicago, IL, USA, August 11-14, 2013*, pages 614–622.
- [Koren, 2008] Koren, Y. (2008). Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, Nevada, USA, August 24-27, 2008*, pages 426–434.
- [Koren et al., 2009] Koren, Y., Bell, R. M., and Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37.
- [Kouki et al., 2015] Kouki, P., Fakhraei, S., Foulds, J. R., Eirinaki, M., and Getoor, L. (2015). Hyper: A flexible and extensible probabilistic framework for hybrid recommender systems. In *Proceedings of the 9th ACM Conference on Recommender Systems, RecSys 2015, Vienna, Austria, September 16-20, 2015*, pages 99–106.
- [Kouki et al., 2019] Kouki, P., Schaffer, J., Pujara, J., O'Donovan, J., and Getoor, L. (2019). Personalized explanations for hybrid recommender systems. In *Proceedings of the 24th International Conference on Intelligent User Interfaces, IUI 2019, Marina del Ray, CA, USA, March 17-20, 2019*, pages 379–390.
- [Krippendorff, 2018] Krippendorff, K. (2018). *Content analysis: An introduction to its methodology*. Sage.
- [Kunaver and Pozrl, 2017] Kunaver, M. and Pozrl, T. (2017). Diversity in recommender systems - A survey. *Knowl. Based Syst.*, 123:154–162.
- [Lao and Cohen, 2010] Lao, N. and Cohen, W. W. (2010). Relational retrieval using a combination of path-constrained random walks. *Mach. Learn.*, 81(1):53–67.
- [Lao et al., 2011] Lao, N., Mitchell, T. M., and Cohen, W. W. (2011). Random walk inference

- and learning in A large scale knowledge base. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, 27-31 July 2011, John McIntyre Conference Centre, Edinburgh, UK, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 529–539.
- [Lécuyer et al., 2014] Lécuyer, M., Ducoffe, G., Lan, F., Papancea, A., Petsios, T., Spahn, R., Chaintreau, A., and Geambasu, R. (2014). Xray: Enhancing the web’s transparency with differential correlation. In *Proceedings of the 23rd USENIX Security Symposium, San Diego, CA, USA, August 20-22, 2014*, pages 49–64.
- [Lécuyer et al., 2015] Lécuyer, M., Spahn, R., Spiliopolous, Y., Chaintreau, A., Geambasu, R., and Hsu, D. J. (2015). Sunlight: Fine-grained targeting detection at scale with statistical confidence. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, October 12-16, 2015*, pages 554–566.
- [Lee et al., 2020] Lee, B. C. G., Lo, K., Downey, D., and Weld, D. S. (2020). Explanation-based tuning of opaque machine learners with application to paper recommendation. *CoRR*, abs/2003.04315.
- [Lee et al., 2013] Lee, S., Park, S., Kahng, M., and Lee, S. (2013). Pathrank: Ranking nodes on a heterogeneous graph for flexible hybrid recommender systems. *Expert Syst. Appl.*, 40(2):684–697.
- [Lewis, 1986] Lewis, D. (1986). Causal explanation. In *Philosophical Papers*, volume II, pages 214–240. Oxford University Press.
- [Li et al., 2011] Li, L., Wang, D., Li, T., Knox, D., and Padmanabhan, B. (2011). SCENE: a scalable two-stage personalized news recommendation system. In *Proceeding of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2011, Beijing, China, July 25-29, 2011*, pages 125–134.
- [Li et al., 2020] Li, L., Zhang, Y., and Chen, L. (2020). Generate neural template explanations for recommendation. In *CIKM ’20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, pages 755–764.
- [Liang et al., 2016] Liang, J., Ajwani, D., Nicholson, P. K., Sala, A., and Parthasarathy, S. (2016). What links alice and bob?: Matching and ranking semantic patterns in heterogeneous networks. In *Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada, April 11 - 15, 2016*, pages 879–889.
- [Lin et al., 2014] Lin, X., Shang, T., and Liu, J. (2014). An estimation method for relationship strength in weighted social network graphs. *Journal of Computer and Communications*, 2(04):82.
- [Lin et al., 2015] Lin, Y., Liu, Z., Sun, M., Liu, Y., and Zhu, X. (2015). Learning entity and relation embeddings for knowledge graph completion. In Bonet, B. and Koenig, S., editors, *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*, pages 2181–2187. AAAI Press.
- [Lipton, 2018] Lipton, Z. C. (2018). The mythos of model interpretability. *Commun. ACM*,

- 61(10):36–43.
- [Liu et al., 2011] Liu, Q., Ge, Y., Li, Z., Chen, E., and Xiong, H. (2011). Personalized travel package recommendation. In *11th IEEE International Conference on Data Mining, ICDM 2011, Vancouver, BC, Canada, December 11-14, 2011*, pages 407–416.
- [Liu et al., 2014a] Liu, X., Yu, Y., Guo, C., and Sun, Y. (2014a). Meta-path-based ranking with pseudo relevance feedback on heterogeneous graph for citation recommendation. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, CIKM 2014, Shanghai, China, November 3-7, 2014*, pages 121–130.
- [Liu et al., 2014b] Liu, Y., Xie, M., and Lakshmanan, L. V. S. (2014b). Recommending user generated item lists. In *Eighth ACM Conference on Recommender Systems, RecSys '14, Foster City, Silicon Valley, CA, USA - October 06 - 10, 2014*, pages 185–192.
- [Lonjarret et al., 2020] Lonjarret, C., Robardet, C., Plantevit, M., Auburtin, R., and Atzmueller, M. (2020). Why should I trust this item? explaining the recommendations of any model. In *7th IEEE International Conference on Data Science and Advanced Analytics, DSAA 2020, Sydney, Australia, October 6-9, 2020*, pages 526–535.
- [Lu et al., 2018a] Lu, Y., Dong, R., and Smyth, B. (2018a). Coevolutionary recommendation model: Mutual learning between ratings and reviews. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, pages 773–782.
- [Lu et al., 2018b] Lu, Y., Dong, R., and Smyth, B. (2018b). Why I like it: multi-task learning for recommendation and explanation. In *Proceedings of the 12th ACM Conference on Recommender Systems, RecSys 2018, Vancouver, BC, Canada, October 2-7, 2018*, pages 4–12.
- [Lundberg and Lee, 2017] Lundberg, S. M. and Lee, S. (2017). A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 4765–4774.
- [Luo et al., 2008] Luo, H., Niu, C., Shen, R., and Ullrich, C. (2008). A collaborative filtering framework based on both local user similarity and global user similarity. *Mach. Learn.*, 72(3):231–245.
- [Luo et al., 2020a] Luo, K., Sanner, S., Wu, G., Li, H., and Yang, H. (2020a). Latent linear critiquing for conversational recommender systems. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, pages 2535–2541.
- [Luo et al., 2020b] Luo, K., Yang, H., Wu, G., and Sanner, S. (2020b). Deep critiquing for vae-based recommender systems. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pages 1269–1278.
- [Ma et al., 2019] Ma, W., Zhang, M., Cao, Y., Jin, W., Wang, C., Liu, Y., Ma, S., and Ren, X.

- (2019). Jointly learning explainable rules for recommendation with knowledge graph. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pages 1210–1221. ACM.
- [Machanavajjhala et al., 2011] Machanavajjhala, A., Korolova, A., and Sarma, A. D. (2011). Personalized social recommendations - accurate or private? *Proc. VLDB Endow.*, 4(7):440–450.
- [Magnini and Strapparava, 2001] Magnini, B. and Strapparava, C. (2001). Improving user modelling with content-based techniques. In *User Modeling 2001, 8th International Conference, UM 2001, Sonthofen, Germany, July 13-17, 2001, Proceedings*, volume 2109 of *Lecture Notes in Computer Science*, pages 74–83.
- [Manca et al., 2018] Manca, M., Boratto, L., and Carta, S. (2018). Behavioral data mining to produce novel and serendipitous friend recommendations in a social bookmarking system. *Inf. Syst. Frontiers*, 20(4):825–839.
- [Martens and Provost, 2014] Martens, D. and Provost, F. J. (2014). Explaining data-driven document classifications. *MIS Q.*, 38(1):73–99.
- [McAuley and Leskovec, 2013] McAuley, J. J. and Leskovec, J. (2013). Hidden factors and hidden topics: understanding rating dimensions with review text. In *Seventh ACM Conference on Recommender Systems, RecSys '13, Hong Kong, China, October 12-16, 2013*, pages 165–172.
- [McCarthy et al., 2005] McCarthy, K., McGinty, L., Smyth, B., and Reilly, J. (2005). A live-user evaluation of incremental dynamic critiquing. In *Case-Based Reasoning, Research and Development, 6th International Conference, on Case-Based Reasoning, ICCBR 2005, Chicago, IL, USA, August 23-26, 2005, Proceedings*, volume 3620 of *Lecture Notes in Computer Science*, pages 339–352.
- [McCarthy et al., 2004] McCarthy, K., Reilly, J., McGinty, L., and Smyth, B. (2004). On the dynamic generation of compound critiques in conversational recommender systems. In *Adaptive Hypermedia and Adaptive Web-Based Systems, Third International Conference, AH 2004, Eindhoven, The Netherlands, August 23-26, 2004, Proceedings*, volume 3137 of *Lecture Notes in Computer Science*, pages 176–184.
- [McCarthy et al., 2010] McCarthy, K., Salem, Y., and Smyth, B. (2010). Experience-based critiquing: Reusing critiquing experiences to improve conversational recommendation. In *Case-Based Reasoning, Research and Development, 18th International Conference on Case-Based Reasoning, ICCBR 2010, Alessandria, Italy, July 19-22, 2010, Proceedings*, volume 6176 of *Lecture Notes in Computer Science*, pages 480–494.
- [McInerney et al., 2018] McInerney, J., Lacker, B., Hansen, S., Higley, K., Bouchard, H., Gruson, A., and Mehrotra, R. (2018). Explore, exploit, and explain: personalizing explainable recommendations with bandits. In *Proceedings of the 12th ACM Conference on Recommender Systems, RecSys 2018, Vancouver, BC, Canada, October 2-7, 2018*, pages 31–39.
- [McNee et al., 2003] McNee, S. M., Lam, S. K., Konstan, J. A., and Riedl, J. (2003). Interfaces

- for eliciting new user preferences in recommender systems. In *User Modeling 2003, 9th International Conference, UM 2003, Johnstown, PA, USA, June 22-26, 2003, Proceedings*, volume 2702 of *Lecture Notes in Computer Science*, pages 178–187.
- [Middleton et al., 2004] Middleton, S. E., Shadbolt, N., and Roure, D. D. (2004). Ontological user profiling in recommender systems. *ACM Trans. Inf. Syst.*, 22(1):54–88.
- [Miller et al., 2003] Miller, B. N., Ried, J. T., and Konstan, J. A. (2003). GroupLens for usenet: Experiences in applying collaborative filtering to a social information system. In *From Usenet to CoWebs*, Computer Supported Cooperative Work, pages 206–231.
- [Miller, 2019] Miller, T. (2019). Explanation in artificial intelligence: Insights from the social sciences. *Artif. Intell.*, 267:1–38.
- [Moeyersoms et al., 2016] Moeyersoms, J., Dalessandro, B., Provost, F. J., and Martens, D. (2016). Explaining classification models built on high-dimensional sparse data. *CoRR*, abs/1607.06280.
- [Molnar, 2020] Molnar, C. (2020). *Interpretable machine learning*. Lulu. com.
- [Murdoch et al., 2019] Murdoch, W. J., Singh, C., Kumbier, K., Abbasi-Asl, R., and Yu, B. (2019). Definitions, methods, and applications in interpretable machine learning. *Proceedings of the National Academy of Sciences*, 116(44):22071–22080.
- [Musto et al., 2018] Musto, C., Franza, T., Semeraro, G., de Gemmis, M., and Lops, P. (2018). Deep content-based recommender systems exploiting recurrent neural networks and linked open data. In *Adjunct Publication of the 26th Conference on User Modeling, Adaptation and Personalization, UMAP 2018, Singapore, July 08-11, 2018*, pages 239–244.
- [Musto et al., 2021] Musto, C., Lops, P., de Gemmis, M., and Semeraro, G. (2021). Context-aware graph-based recommendations exploiting personalized pagerank. *Knowl. Based Syst.*, 216:106806.
- [Musto et al., 2016a] Musto, C., Narducci, F., Lops, P., de Gemmis, M., and Semeraro, G. (2016a). Explod: A framework for explaining recommendations based on the linked open data cloud. In *Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, September 15-19, 2016*, pages 151–154.
- [Musto et al., 2016b] Musto, C., Semeraro, G., de Gemmis, M., and Lops, P. (2016b). Learning word embeddings from wikipedia for content-based recommender systems. In *Advances in Information Retrieval - 38th European Conference on IR Research, ECIR 2016, Padua, Italy, March 20-23, 2016. Proceedings*, volume 9626 of *Lecture Notes in Computer Science*, pages 729–734.
- [Nikolakopoulos et al., 2019] Nikolakopoulos, A. N., Kalantzis, V., Gallopoulos, E., and Garofalakis, J. D. (2019). Eigenrec: generalizing puresvd for effective and efficient top-n recommendations. *Knowl. Inf. Syst.*, 58(1):59–81.
- [Nikolakopoulos and Karypis, 2019] Nikolakopoulos, A. N. and Karypis, G. (2019). Recwalk: Nearly uncoupled random walks for top-n recommendation. In *Proceedings of the Twelfth*

- ACM International Conference on Web Search and Data Mining, WSDM 2019, Melbourne, VIC, Australia, February 11-15, 2019*, pages 150–158.
- [Nóbrega and Marinho, 2019] Nóbrega, C. and Marinho, L. B. (2019). Towards explaining recommendations through local surrogate models. In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, SAC 2019, Limassol, Cyprus, April 8-12, 2019*, pages 1671–1678.
- [Nunes and Jannach, 2017] Nunes, I. and Jannach, D. (2017). A systematic review and taxonomy of explanations in decision support and recommender systems. *User Model. User Adapt. Interact.*, 27(3-5):393–444.
- [Page et al., 1999] Page, L., Brin, S., Motwani, R., and Winograd, T. (1999). The PageRank citation ranking: Bringing order to the Web. Technical report, Stanford InfoLab.
- [Pan et al., 2020] Pan, D., Li, X., Li, X., and Zhu, D. (2020). Explainable recommendation via interpretable feature mapping and evaluation of explainability. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 2690–2696.
- [Papadimitriou et al., 2012] Papadimitriou, A., Symeonidis, P., and Manolopoulos, Y. (2012). A generalized taxonomy of explanations styles for traditional and social recommender systems. *Data Min. Knowl. Discov.*, 24(3):555–583.
- [Park et al., 2017] Park, H., Jeon, H., Kim, J., Ahn, B., and Kang, U. (2017). Uniwalk: Explainable and accurate recommendation for rating and network data. *CoRR*, abs/1710.07134.
- [Parra and Brusilovsky, 2015] Parra, D. and Brusilovsky, P. (2015). User-controllable personalization: A case study with setfusion. *Int. J. Hum. Comput. Stud.*, 78:43–67.
- [Parra-Arnau et al., 2017] Parra-Arnau, J., Achara, J. P., and Castelluccia, C. (2017). *MyAd-Choices*: Bringing transparency and control to online advertising. *ACM Trans. Web*, 11(1):7:1–7:47.
- [Pazzani and Billsus, 2007] Pazzani, M. J. and Billsus, D. (2007). Content-based recommendation systems. In *The Adaptive Web, Methods and Strategies of Web Personalization*, volume 4321 of *Lecture Notes in Computer Science*, pages 325–341. Springer.
- [Peake and Wang, 2018] Peake, G. and Wang, J. (2018). Explanation mining: Post hoc interpretability of latent factor models for recommendation systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, pages 2060–2069.
- [Pirró, 2015] Pirró, G. (2015). Explaining and suggesting relatedness in knowledge graphs. In *The Semantic Web - ISWC 2015 - 14th International Semantic Web Conference, Bethlehem, PA, USA, October 11-15, 2015, Proceedings, Part I*, volume 9366 of *Lecture Notes in Computer Science*, pages 622–639.
- [Pu and Chen, 2006] Pu, P. and Chen, L. (2006). Trust building with explanation interfaces. In *Proceedings of the 11th International Conference on Intelligent User Interfaces, IUI 2006*,

- Sydney, Australia, January 29 - February 1, 2006, pages 93–100.
- [Pu and Chen, 2007] Pu, P. and Chen, L. (2007). Trust-inspiring explanation interfaces for recommender systems. *Knowl. Based Syst.*, 20(6):542–556.
- [Rader et al., 2018] Rader, E. J., Cotter, K., and Cho, J. (2018). Explanations as mechanisms for supporting algorithmic transparency. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems, CHI 2018, Montreal, QC, Canada, April 21-26, 2018*, page 103.
- [Radlinski and Joachims, 2005] Radlinski, F. and Joachims, T. (2005). Query chains: learning to rank from implicit feedback. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Chicago, Illinois, USA, August 21-24, 2005*, pages 239–248.
- [Ramakrishnan et al., 2005] Ramakrishnan, C., Milnor, W. H., Perry, M., and Sheth, A. P. (2005). Discovering informative connection subgraphs in multi-relational graphs. *SIGKDD Explor.*, 7(2):56–63.
- [Rana and Bridge, 2018] Rana, A. and Bridge, D. (2018). Explanations that are intrinsic to recommendations. In *Proceedings of the 26th Conference on User Modeling, Adaptation and Personalization, UMAP 2018, Singapore, July 08-11, 2018*, pages 187–195.
- [Reilly et al., 2004a] Reilly, J., McCarthy, K., McGinty, L., and Smyth, B. (2004a). Dynamic critiquing. In *Advances in Case-Based Reasoning, 7th European Conference, ECCBR 2004, Madrid, Spain, August 30 - September 2, 2004, Proceedings*, volume 3155 of *Lecture Notes in Computer Science*, pages 763–777.
- [Reilly et al., 2004b] Reilly, J., McCarthy, K., McGinty, L., and Smyth, B. (2004b). Incremental critiquing. In *Research and Development in Intelligent Systems XXI, Proceedings of AI-2004, the Twenty-fourth SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence, Queens' College, Cambridge, UK, 13-15 December 2004*, pages 101–114.
- [Ribeiro et al., 2016] Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). "why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 1135–1144.
- [Ricci et al., 2015] Ricci, F., Rokach, L., and Shapira, B., editors (2015). *Recommender Systems Handbook*. Springer.
- [Riedl, 2019] Riedl, M. O. (2019). Human-centered artificial intelligence and machine learning. *Human Behavior and Emerging Technologies*, 1(1):33–36.
- [Robertson, 1977] Robertson, S. E. (1977). The probability ranking principle in ir. *Journal of documentation*.
- [Rudin, 2019] Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*,

- 1(5):206–215.
- [Sánchez et al., 2017] Sánchez, L. Q., Sauer, C., Recio-García, J. A., and Díaz-Agudo, B. (2017). Make it personal: A social explanation system applied to group recommendations. *Expert Syst. Appl.*, 76:36–48.
- [Sarwar et al., 2001] Sarwar, B. M., Karypis, G., Konstan, J. A., and Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. In *Proceedings of the Tenth International World Wide Web Conference, WWW 10, Hong Kong, China, May 1-5, 2001*, pages 285–295.
- [Schuhmacher et al., 2015] Schuhmacher, M., Dietz, L., and Ponzetto, S. P. (2015). Ranking entities for web queries through text and knowledge. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM 2015, Melbourne, VIC, Australia, October 19 - 23, 2015*, pages 1461–1470.
- [Seo et al., 2017] Seo, S., Huang, J., Yang, H., and Liu, Y. (2017). Interpretable convolutional neural networks with dual local and global attention for review rating prediction. In *Proceedings of the Eleventh ACM Conference on Recommender Systems, RecSys 2017, Como, Italy, August 27-31, 2017*, pages 297–305.
- [Seufert et al., 2016] Seufert, S., Berberich, K., Bedathur, S. J., Kondreddi, S. K., Ernst, P., and Weikum, G. (2016). ESPRESSO: explaining relationships between entity sets. In *Proceedings of the 25th ACM International Conference on Information and Knowledge Management, CIKM 2016, Indianapolis, IN, USA, October 24-28, 2016*, pages 1311–1320.
- [Seyler et al., 2018] Seyler, D., Chandar, P., and Davis, M. (2018). An information retrieval framework for contextual suggestion based on heterogeneous information network embeddings. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*, pages 953–956.
- [Sharma and Cosley, 2013] Sharma, A. and Cosley, D. (2013). Do social explanations work?: studying and modeling the effects of social explanations in recommender systems. In *22nd International World Wide Web Conference, WWW '13, Rio de Janeiro, Brazil, May 13-17, 2013*, pages 1133–1144.
- [Sharma et al., 2019] Sharma, M., Harper, F. M., and Karypis, G. (2019). Learning from sets of items in recommender systems. *ACM Trans. Interact. Intell. Syst.*, 9(4):19:1–19:26.
- [Shi et al., 2017] Shi, C., Li, Y., Zhang, J., Sun, Y., and Yu, P. S. (2017). A survey of heterogeneous information network analysis. *IEEE Trans. Knowl. Data Eng.*, 29(1):17–37.
- [Shi et al., 2015] Shi, C., Zhang, Z., Luo, P., Yu, P. S., Yue, Y., and Wu, B. (2015). Semantic path based personalized recommendation on weighted heterogeneous information networks. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM 2015, Melbourne, VIC, Australia, October 19 - 23, 2015*, pages 453–462.
- [Shimazu, 2001] Shimazu, H. (2001). Expertclerk: Navigating shoppers buying process with the combination of asking and proposing. In Nebel, B., editor, *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence, IJCAI 2001, Seattle, Washington*,

- USA, August 4-10, 2001, pages 1443–1450. Morgan Kaufmann.
- [Singh and Anand, 2018] Singh, J. and Anand, A. (2018). Posthoc interpretability of learning to rank models using secondary training data. *CoRR*, abs/1806.11330.
- [Sinha and Swearingen, 2002] Sinha, R. R. and Swearingen, K. (2002). The role of transparency in recommender systems. In *Extended abstracts of the 2002 Conference on Human Factors in Computing Systems, CHI 2002, Minneapolis, Minnesota, USA, April 20-25, 2002*, pages 830–831.
- [Soh et al., 2013] Soh, P., Lin, Y., and Chen, M. (2013). Recommendation for online social feeds by exploiting user response behavior. In *22nd International World Wide Web Conference, WWW '13, Rio de Janeiro, Brazil, May 13-17, 2013, Companion Volume*, pages 197–198.
- [Srijith et al., 2017] Srijith, P. K., Lukasik, M., Bontcheva, K., and Cohn, T. (2017). Longitudinal modeling of social media with hawkes process based on users and networks. In *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017, Sydney, Australia, July 31 - August 03, 2017*, pages 195–202.
- [Sun and Han, 2012] Sun, Y. and Han, J. (2012). Mining heterogeneous information networks: a structural analysis approach. *SIGKDD Explor.*, 14(2):20–28.
- [Sun et al., 2012] Sun, Y., Han, J., Aggarwal, C. C., and Chawla, N. V. (2012). When will it happen?: relationship prediction in heterogeneous information networks. In *Proceedings of the Fifth International Conference on Web Search and Web Data Mining, WSDM 2012, Seattle, WA, USA, February 8-12, 2012*, pages 663–672.
- [Sun et al., 2011] Sun, Y., Han, J., Yan, X., Yu, P. S., and Wu, T. (2011). Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. *Proc. VLDB Endow.*, 4(11):992–1003.
- [Suzuki et al., 2019] Suzuki, T., Oyama, S., and Kurihara, M. (2019). Explainable recommendation using review text and a knowledge graph. In *2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, December 9-12, 2019*, pages 4638–4643.
- [Tao et al., 2019] Tao, Y., Jia, Y., Wang, N., and Wang, H. (2019). The fact: Taming latent factor models for explainability with factorization trees. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*, pages 295–304.
- [Thompson et al., 2004] Thompson, C. A., Göker, M. H., and Langley, P. (2004). A personalized system for conversational recommendations. *J. Artif. Intell. Res.*, 21:393–428.
- [Tintarev and Masthoff, 2007] Tintarev, N. and Masthoff, J. (2007). A survey of explanations in recommender systems. In *Proceedings of the 23rd International Conference on Data Engineering Workshops, ICDE 2007, 15-20 April 2007, Istanbul, Turkey*, pages 801–810.
- [Vig et al., 2009] Vig, J., Sen, S., and Riedl, J. (2009). Tagsplanations: explaining recommendations using tags. In *Proceedings of the 14th International Conference on Intelligent User*

- Interfaces, IUI 2009, Sanibel Island, Florida, USA, February 8-11, 2009*, pages 47–56.
- [Wan and McAuley, 2018] Wan, M. and McAuley, J. J. (2018). Item recommendation on monotonic behavior chains. In *Proceedings of the 12th ACM Conference on Recommender Systems, RecSys 2018, Vancouver, BC, Canada, October 2-7, 2018*, pages 86–94.
- [Wang et al., 2013] Wang, G., Gill, K., Mohanlal, M., Zheng, H., and Zhao, B. Y. (2013). Wisdom in the social crowd: an analysis of quora. In *22nd International World Wide Web Conference, WWW '13, Rio de Janeiro, Brazil, May 13-17, 2013*, pages 1341–1352.
- [Wang et al., 2018a] Wang, H., Zhang, F., Wang, J., Zhao, M., Li, W., Xie, X., and Guo, M. (2018a). Ripplenet: Propagating user preferences on the knowledge graph for recommender systems. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22-26, 2018*, pages 417–426.
- [Wang et al., 2018b] Wang, N., Wang, H., Jia, Y., and Yin, Y. (2018b). Explainable recommendation via multi-task learning in opinionated text data. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*, pages 165–174.
- [Wang et al., 2018c] Wang, X., Chen, Y., Yang, J., Wu, L., Wu, Z., and Xie, X. (2018c). A reinforcement learning framework for explainable recommendation. In *IEEE International Conference on Data Mining, ICDM 2018, Singapore, November 17-20, 2018*, pages 587–596.
- [Wang et al., 2019a] Wang, X., He, X., Cao, Y., Liu, M., and Chua, T. (2019a). KGAT: knowledge graph attention network for recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, pages 950–958.
- [Wang et al., 2018d] Wang, X., He, X., Feng, F., Nie, L., and Chua, T. (2018d). TEM: tree-enhanced embedding model for explainable recommendation. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, pages 1543–1552. ACM.
- [Wang et al., 2019b] Wang, X., Wang, D., Xu, C., He, X., Cao, Y., and Chua, T. (2019b). Explainable reasoning over knowledge graphs for recommendation. 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*, pages 5329–5336.
- [Wang et al., 2014] Wang, Z., Zhang, J., Feng, J., and Chen, Z. (2014). Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27 -31, 2014, Québec City, Québec, Canada*, pages 1112–1119.
- [Wiegrefe and Pinter, 2019] Wiegrefe, S. and Pinter, Y. (2019). Attention is not not explanation. 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, Hong Kong, China, November 3-7, 2019*, pages 11–20.
- [Wilcoxon, 1992] Wilcoxon, F. (1992). Individual comparisons by ranking methods. In *Breakthroughs in statistics*, pages 196–202.
- [Wu et al., 2019a] Wu, G., Luo, K., Sanner, S., and Soh, H. (2019a). Deep language-based critiquing for recommender systems. In *Proceedings of the 13th ACM Conference on Recommender Systems, RecSys 2019, Copenhagen, Denmark, September 16-20, 2019*, pages 137–145.
- [Wu et al., 2019b] Wu, T., Ribeiro, M. T., Heer, J., and Weld, D. S. (2019b). Errudite: Scalable, reproducible, and testable error analysis. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 747–763.
- [Wu and Ester, 2015] Wu, Y. and Ester, M. (2015). FLAME: A probabilistic model combining aspect based opinion mining and collaborative filtering. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining, WSDM 2015, Shanghai, China, February 2-6, 2015*, pages 199–208.
- [Xian et al., 2019] Xian, Y., Fu, Z., Muthukrishnan, S., de Melo, G., and Zhang, Y. (2019). Reinforcement knowledge graph reasoning for explainable recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*, pages 285–294.
- [Xin et al., 2019] Xin, X., He, X., Zhang, Y., Zhang, Y., and Jose, J. M. (2019). Relational collaborative filtering: Modeling multiple item relations for recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*, pages 125–134.
- [Xu et al., 2020] Xu, S., Li, Y., Liu, S., Fu, Z., and Zhang, Y. (2020). Learning post-hoc causal explanations for recommendation. *CoRR*, abs/2006.16977.
- [Yang et al., 2018] Yang, F., Liu, N., Wang, S., and Hu, X. (2018). Towards interpretation of recommender systems with sorted explanation paths. In *IEEE International Conference on Data Mining, ICDM 2018, Singapore, November 17-20, 2018*, pages 667–676.
- [Ying et al., 2019] Ying, Z., Bourgeois, D., You, J., Zitnik, M., and Leskovec, J. (2019). Gnnexplainer: Generating explanations for graph neural networks. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 9240–9251.
- [Yu et al., 2009] Yu, C., Lakshmanan, L. V. S., and Amer-Yahia, S. (2009). It takes variety to make a world: diversification in recommender systems. In *EDBT 2009, 12th International Conference on Extending Database Technology, Saint Petersburg, Russia, March 24-26, 2009, Proceedings*, volume 360 of *ACM International Conference Proceeding Series*, pages 368–378.
- [Yu et al., 2014] Yu, X., Ren, X., Sun, Y., Gu, Q., Sturt, B., Khandelwal, U., Norick, B., and

- Han, J. (2014). Personalized entity recommendation: a heterogeneous information network approach. In *Seventh ACM International Conference on Web Search and Data Mining, WSDM 2014, New York, NY, USA, February 24-28, 2014*, pages 283–292.
- [Yu et al., 2013] Yu, X., Ren, X., Sun, Y., Sturt, B., Khandelwal, U., Gu, Q., Norick, B., and Han, J. (2013). Recommendation in heterogeneous information networks with implicit user feedback. In *Seventh ACM Conference on Recommender Systems, RecSys '13, Hong Kong, China, October 12-16, 2013*, pages 347–350.
- [Zhang et al., 2018] Zhang, C., Huang, C., Yu, L., Zhang, X., and Chawla, N. V. (2018). Camel: Content-aware and meta-path augmented metric learning for author identification. In Champin, P., Gandon, F., Lalmas, M., and Ipeirotis, P. G., editors, *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, pages 709–718.
- [Zhang et al., 2019] Zhang, C., Swami, A., and Chawla, N. V. (2019). SHNE: representation learning for semantic-associated heterogeneous networks. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, WSDM 2019, Melbourne, VIC, Australia, February 11-15, 2019*, pages 690–698.
- [Zhang et al., 2016a] Zhang, F., Yuan, N. J., Lian, D., Xie, X., and Ma, W. (2016a). Collaborative knowledge base embedding for recommender systems. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 353–362.
- [Zhang et al., 2016b] Zhang, H., Lofgren, P., and Goel, A. (2016b). Approximate personalized pagerank on dynamic graphs. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 1315–1324.
- [Zhang and Pu, 2006] Zhang, J. and Pu, P. (2006). A comparative study of compound critique generation in conversational recommender systems. In *Adaptive Hypermedia and Adaptive Web-Based Systems, 4th International Conference, AH 2006, Dublin, Ireland, June 21-23, 2006, Proceedings*, volume 4018 of *Lecture Notes in Computer Science*, pages 234–243.
- [Zhang et al., 2014a] Zhang, J., Yu, P. S., and Zhou, Z. (2014a). Meta-path based multi-network collective link prediction. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, pages 1286–1295.
- [Zhang and Chen, 2020] Zhang, Y. and Chen, X. (2020). Explainable recommendation: A survey and new perspectives. *Found. Trends Inf. Retr.*, 14(1):1–101.
- [Zhang et al., 2014b] Zhang, Y., Lai, G., Zhang, M., Zhang, Y., Liu, Y., and Ma, S. (2014b). Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *The 37th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '14, Gold Coast, QLD, Australia - July 06 - 11, 2014*, pages 83–92.

- [Zhao et al., 2019] Zhao, G., Fu, H., Song, R., Sakai, T., Chen, Z., Xie, X., and Qian, X. (2019). Personalized reason generation for explainable song recommendation. *ACM Trans. Intell. Syst. Technol.*, 10(4):41:1–41:21.
- [Zhao et al., 2015] Zhao, K., Cong, G., Yuan, Q., and Zhu, K. Q. (2015). SAR: A sentiment-aspect-region model for user preference analysis in geo-tagged reviews. In *31st IEEE International Conference on Data Engineering, ICDE 2015, Seoul, South Korea, April 13-17, 2015*, pages 675–686.
- [Zhao et al., 2014] Zhao, W. X., Guo, Y., He, Y., Jiang, H., Wu, Y., and Li, X. (2014). We know what you want to buy: a demographic-based system for product recommendation on microblogs. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, pages 1935–1944.
- [Zhao et al., 2016] Zhao, W. X., Li, S., He, Y., Wang, L., Wen, J., and Li, X. (2016). Exploring demographic information in social media for product recommendation. *Knowl. Inf. Syst.*, 49(1):61–89.
- [Zhu and Ghahramani, 2002] Zhu, X. and Ghahramani, Z. (2002). Learning from labeled and unlabeled data with label propagation. *CMU Technical Report*.