Improving the explainability of user-facing recommender systems

Ville Kuosmanen

University of St Andrews

20th April 2020

Supervisor: Dr David Harris-Birtill

Abstract

Recommender systems are increasingly being deployed into production use in various systems, such as online stores, social networks and video-sharing platforms. Modern recommender systems tend to use black-box, machine learning -based, techniques such as matrix factorization or deep learning which can significantly increase the accuracy of predictions while making the system's behaviour hard to understand or explain.

This dissertation shows how the explainability of black-box recommender systems can be improved by providing a generated explanation to recommendations. It includes a survey of current state-of-the-art recommender systems and the attempts at making them explainable, as well as the implementation of a recommender system and two post-hoc explanation generators. The report then describes an evaluation of the explanation generators through a user study, where 41 participants were asked to rate the persuasiveness and trustworthiness of film recommendations coupled with explanations of different classes. The user study found a statistically significant difference in both persuasiveness (p=0.008) and trust (p=0.001) between explanation types in favour of the association rules explainer, while quantitative evaluation showed the explainer suffered from a low model fidelity, potentially adding a selection bias to the results. The report also proposes several exciting topics of further study, such as how the mined association rules can be used to analyse the behaviour of the recommender system as a whole.

Declaration

Table of Contents

Overview

Recommender systems help users discover new items, and they have seen increased use in various applications including video streaming [2][20], e-commerce [2][4][10][17] and social media [18]. One of the main goals of recommender systems is to help users filter the large number of items available in most online platforms to just a handful [2], thus making it easier for users to find the items they like. Recommender systems usually aim to customise the recommendations to each user, as personalised recommendations tend to be more accurate than general ones [4]. Modern systems use machine learning to deliver highly accurate recommendations to each user, but tend to sacrifice interpretability: it’s often hard to understand or explain why a particular item was recommended to a user [1].

This dissertation aims to help deliver more understandable recommendations to users by improving the explainability of recommender systems. This is done through *post-hoc explainability*, which means creating a separate component (explainer) that generates an explanation to a recommendation given by a black-box (i.e. uninterpretable) recommender system [1]. The key advantage of post-hoc explainability over other approaches is that it’s model-agnostic: the same explainer can generate explanations for all kinds of models and requires no domain-specific additional data to work [1].

This dissertation includes the implementation of a recommender system based on matrix factorisation [23] and two post-hoc explainers based on mined association rules and influences, respectively. Particular attention is paid to the problem of evaluating explainability in recommender systems. The context survey introduces seven criteria that systems can be evaluated against[21]: out of them, the project focuses on trust and persuasiveness. The metric *model fidelity* for evaluating post-hoc explanation generators is also introduced, which describes the share of recommendations that can be made explainable [1].

The explanations were evaluated through a user study. 41 participants were asked to rate films they’ve seen before using a custom-built React.js [29] web application, after which they were presented with film recommendations with various types of explanations. The users were then asked to rate how interested they’d be in watching the film and how much they trust the recommendation. These questions were set to measure the persuasiveness and trustworthiness of the explanations. As a result, a statistically significant difference in both persuasiveness (p=0.008) and trust (p=0.001) was measured between explanation types, and post-hoc tests showed the differences to be in favour of the association rules explainer. However, this explainer suffered from a drastically low model fidelity, limiting its usefulness in real systems and potentially distorting the experiment results by introducing a selection bias.

In the end, this dissertation shows how post-hoc explanations can be added to a realistic recommender system and how they can potentially improve the trust and persuasiveness of the system. The dissertation also highlights the importance of model fidelity as a metric for post-hoc explainers and the issues a low model fidelity can cause. Looking towards the future, the dissertation proposes a method to optimise model fidelity for association rules explainers by automatically selecting the best support and confidence bounds, and proposes a novel application of association rules for analysing the behaviour of the recommender system as a whole. This is explored by mapping the recommender system as a graph, which could then be studied objectively to discover new knowledge, for example whether the system suffers from the *extreme content problem* which the dissertation defines.

Context survey

This survey first presents a short overview of recommender systems in general, and a motivation for why explainability should be considered when designing them. The survey then describes several approaches in making recommender systems explainable, and how explainable recommender systems have been evaluated by their authors. It also explores the uses of explainable recommender systems in real-world systems, paying attention to criticism unexplainable recommendations have received.

Recommender systems

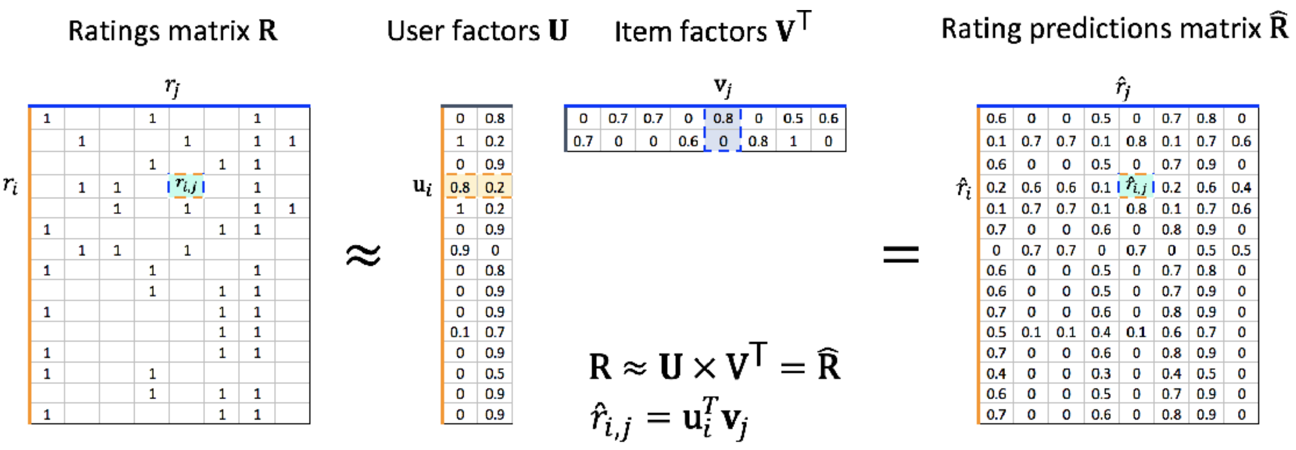
Recommender systems help users discover new items in a system by providing suggestions of items they might find useful. [2] In this context, "item" refers to what the system is recommending to users, and could be anything from movies and toothbrushes to profiles of other users. Recommender systems are often most valuable in systems with a large number of items to choose from, as having too many choices can overwhelm users. [2][3] A good recommender system can therefore help limit the number of items to choose from, from many to just a handful. While the items recommended by the recommender system can be the same for all users, the systems often aim to personalise recommendations to each user, as different users will be interested in different kinds of items. For example, Amazon.com has shown that targeting recommendations to each user vastly outperforms untargeted advertising, therefore boosting the company's sales and revenue. [4]

Recommender systems can be built using several classes of techniques, but this project focuses on collaborative filtering (CF), which is one of the most commonly used techniques today. Most importantly, recommender systems using deep learning [35] will not be covered in this report. CF methods produce recommendations based on the users' previous ratings with the theory that if users A and B liked similar items in the past, it's likely that A will also like other items rated highly by B [7][2]. Before CF, the most prevalent class of recommender system was content-based systems [22]. Content-based systems consider which items are most similar to the items previously liked by a user based on content, for example categories and genres of items. For example, in a movie streaming site a user who has liked horror films in the past would be likely to be recommended other movies belonging to the horror category. CF-techniques have a few key advantages over content-based approaches. Firstly, CF works well even if the items can't be easily categorised or labelled by their content, and it allows the system to recommend very different kinds of items, which may be a desirable property in a system. [6] CF has been considered the most popular technique for building a recommender system in the past few years [1].

CF systems don't directly predict which items to recommend to a given user. Instead, they predict the rating a user would give to each item they haven't yet rated, and then choose the top-n highest ratings as the recommended items. As such, the problem of generating recommendations can be expressed as a matrix completion problem, a well-known problem in mathematics [34]. Given a sparse matrix of ratings by users U on items I, can the unseen values of ratings be predicted accurately? This problem, and one approach in solving it, is illustrated in Figure 1. In order to predict the ratings, the system needs to find a relation between two different classes of entities, users and items. This problem can be overcome by two main approaches: neighbourhood methods which focus on modelling relations inside an entity class (i.e. items with items, or users with users), and latent factor models which automatically learn a set of common factors by which users and items can be compared against each other. [7]

Neighbourhood methods are based on finding the *k* nearest neighbours of entities of the same class, and using them to predict the rating. In this way, only relationships between the same category of entities are modelled. Item-oriented models predict rating *r* by a user *u* for an item *i* by finding the *k* nearest neighbours of *i* (i.e. items most similar to *i*), and taking a weighted average of the ratings *u* has given to them. In user-oriented models *r* is predicted by taking the *k* nearest neighbours of *u* (i.e. users most similar to *u*) and once again taking a weighted average of the ratings these users have given to *i*. Neighbourhood models are relatively simple and have seen widespread adoption in the past, but as they fail to take full advantage of the data their performance tends to be suboptimal. [7][23]

Latent factor models (LFM) use machine learning in solving the problem of relating users and items. LFMs use the same set of factors to describe users and items, allowing their similarity to be compared easily. This means that the items most similar to a user can be found directly, unlike in neighbourhood models where only the most similar users can be found. The factors are automatically learned from the set of numeric ratings (called the **training set** of the model), and they describe the different dimensions the items and users can be described with. In the case of movies, one could envision the system learning dimensions such as scariness, movies one watches during Christmas, or films with strong female characters. Formally, this means LFMs map users and items on a joint latent factor space of some dimensionality *n*, where each factor represents a dimension. [7][23] Figure 1 shows how the original ratings matrix is transformed into the matrices of user and item factors.



Ratings matrix **R**
User factors **U**
Item factors $\mathbf{V}^T$
Rating predictions matrix $\hat{\mathbf{R}}$
r_j
r_i
$r_{i,j}$
$\mathbf{u}_i$
$\mathbf{v}_j$
$\hat{r}_j$
$\hat{r}_i$
$\hat{r}_{i,j}$
$\approx$
$=$
$\mathbf{R} \approx \mathbf{U} \times \mathbf{V}^T = \hat{\mathbf{R}}$
$\hat{r}_{i,j} = \mathbf{u}_i^T \mathbf{v}_j$

Figure 1: *The transformation of a sparse ratings matrix R into factors matrices for users U and items V^T. These matrices can be used to generate a complete predicted ratings matrix* $\hat{R}$*. Reproduced from Figure 1, Peake, G., & Wang, J. (2018, July). 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 (pp. 2060-2069). ACM.

Within the factor space, users and items are associated with a vector. For items, the vector’s values in each dimension rate how well the item is described by that factor, while for users the values rate that user’s interest in that factor. An interaction between a user and an item can then be described by the dot product of their vectors. [23] Since the factors are learned from numeric user ratings without any reference to the items themselves, they don’t directly map to any interpretable factors or categories – for this reason LFMs are said to be black-box models [1].

LFMs can be generated by various matrix factorisation models. Perhaps the best-known of them is singular value decomposition, or SVD. SVD was developed as a better-performing alternative to neighbourhood approaches during the Netflix Prize competition. [23] An SVD model is trained using the training data on user ratings of items, and outputs a model consisting of learned factors for each user and item, as well as user and item *biases*. When predicting a rating, the biases of the user and item in question are added on the dot product of the corresponding factors – this helps the system deal with users who consistently give higher or lower ratings, and items that get particularly high or low ratings. SVD models can be trained either by using stochastic gradient descent, or a gradient descent-based algorithm named alternating least squares (ALS). [7]

SVD was further improved by considering the implicit feedback that the user gives by choosing to rate certain items. Models of this class are called SVD++, and they tend to perform better than standard SVD at the cost of added complexity. [7] Since then, other more complex LFMs have also been developed [28], but they are out of scope of this review. As is usual for machine learning models, great caution should be taken to avoid overfitting the model to the ratings. As such the hyper-parameters of the model should be optimised using cross-validation [7], and the final evaluation of the model’s accuracy should be reported on unseen test data.

Explainability

While machine learning techniques have greatly increased the abilities of many artificial intelligence systems, they have also become so complex that humans can’t understand why and how they make decisions. If the model’s behaviour can’t be understood well by humans, it is said to be uninterpretable or a *black box*. Uninterpretable models are a problem for many reasons, perhaps most importantly due to lack of trust. If the system can’t tell its user why it made a particular decision, why should the user trust that decision? Besides trust, Table 1 shows the seven explanatory criteria previously identified for explainability [21].

Criterion	Definition
Transparency	Explain how the system works
Scrutability	Allow users to tell the system it is wrong
Trust	Increase users’ confidence in the system
Effectiveness	Help users make good decisions
Persuasiveness	Convince users to try or buy
Effectiveness	Help users make decisions faster
Satisfaction	Increase the ease of use or enjoyment

Table 1: *Seven criteria for evaluating the explainability of recommendation systems. Adapted from Table 15.1, Tintarev, N., & Masthoff, J. (2011). Designing and evaluating explanations for recommender systems. In Recommender systems handbook (pp. 479-510). Springer, Boston, MA.*

Justifying the decisions made by the model is especially important in areas like the medical industry, where incorrect decisions can have disastrous results [8]. While recommender systems are often used in less serious applications such as online stores and social networks, making the recommendations explainable is still important since it’s been shown that users prefer recommendations which they perceive as transparent [9].

Explainable recommendations approaches

Making recommender systems explainable is an area of active research, and so several approaches have been tried. This report divides the approaches to two main classes, embedded and post-hoc explanations as proposed by Wang et al. [13], paying particular attention to post-hoc techniques as they are the focus of this project. A more extensive survey on explainable recommendations has been conducted by Zhang and Chen, which will offer further perspective into the topic [11].

Embedded explanations

In embedded methods the explanation generation is integrated with the recommendations model itself [13]. This can be achieved either by using an interpretable model, or adding some notion of explainability into the training inputs. When the model itself is interpretable (i.e. white-box), explaining its behaviour becomes significantly easier. An example of such a model would be an item-oriented neighbourhood model, where the predicted rating for a given user on an item *i* is based entirely on the ratings said user has given to items *i* is most similar to. In such a simple model, the explanation can be generated easily by listing the items *i* is most similar to, and the ratings the current user gave to them. [6] But since interpretable models tend to be simpler than black-box models, their recommendations tend to be less accurate as well. This is known as the accuracy-interpretability trade-off and it has been observed in many applications of machine learning. [1]

MF as a technique can be adapted by adding a notion of explainability into the model’s training inputs. One approach is called Constrained Matrix Factorization (CMF) where constraints of some kind are imposed in training the MF model, resulting in a model that is no longer a black box [1].

For example, Abdollahi and Nasraoui applied CMF when training MF models by adding an “explainability score” as an additional input alongside with the set of ratings users gave to items. The explainability score is calculated mathematically, and is higher if the current user has rated many similar items, or if similar users have rated the current item (i.e. “Similar to items you liked...” or “Users similar to you liked...”). The resulting recommender system gave higher rankings for items deemed explainable than what they would have had in a regular MF-based recommender system. [14]

Embedded approaches also allows researches to add external data beyond the matrix of user ratings of items as inputs to the MF model. This is not necessary, as demonstrated by CMF [14], but adding external data about the users or items can not only help with explanations, but also with the accuracy of the models themselves. Zhang, et al. proposed Explicit Matrix Factorization (EMF), where the factors in a MF model are not automatically learned but rather collected from textual user reviews. The system constructs matrices for how much users cared about each item feature, and how well each item was described by the feature. An MF model could then be built using these matrices. The system is easily interpretable and explained by simply checking which factors items and users score the highest. [10]

Collaborative Topic Poisson Factorization (CTPF) was developed as a model for recommending articles to users that considers the textual content of the article alongside the ratings given by users to articles. As in EMF, in CTPF the factors for items, which in this case are the articles, directly correspond to topics articles are about. [15] When the system recommends articles to a user who has expressed interested in, let’s say, biology (a topic articles are about), they will receive articles that are about biology (inferred by analysing article content), as well as articles that are popular amongst people who have read biology articles (inferred by collaborative filtering). These two approaches show how data other than numeric ratings can be used in training recommender systems, and how advances in related subjects such as natural language processing play an important role in improving the accuracy and explainability of recommender systems.

Embedded explanations tend to naturally have good explainability because the explanations and recommendations models are not decoupled. In terms of accuracy, it is unclear if embedding explanations to the model generally increases or decreases it’s accuracy, as both positive and negative results have been reported [1][13][14]. In addition, unlike post-hoc methods, embedded explanations are not model-agnostic and require a particular type of model (such as MF) or domain (such as academic research papers) to function, therefore making them less flexible. External data isn’t always available either, making some approaches infeasible for certain datasets [1].

Post-hoc explanations

Another way to make the black box’s recommendations explainable is to build a white-box model that generates explanations to a decision made by the black-box recommendations model [11]. This approach is called post-hoc explainability because the recommendations are made explainable after they’ve been produced by the original model, which is left untouched.

Peake and Wang used the post-hoc approach by extracting logical association rules from the recommendation model’s inputs and outputs. The rules extracted were of form {X => Y}, where X is an item the user has experienced preference for, and Y is the recommendation. The association rule in question can be expressed in words as “Because you liked X, we recommend Y”. The association rules themselves can be used to generate recommendations by selecting the set of rules {X, Y} for a user U where X has been liked by the user while Y has not, rating the rules by some metric and selecting the top *n* as “explainable recommendations”. The association rules are combined with the black box model by marking recommendations that are produced by both models as “explainable recommendations”. [1] Here, the researchers limited the size of the association rules to just two items (X and Y in the above example) – experimenting with rules consisting of more items is an important consideration for further study. While a MF model was used in the research, the recommendations model is model-agnostic, meaning that it can generate explanations for any model. [1]

Other post-hoc methods have also been tried. An approach called Fast Influence Analysis (FIA) used influence functions to calculate which inputs to the recommendations model (i.e. the user’s reviews) had the greatest (positive or negative) influence on the prediction. This was done by calculating the difference in predictions when the input was excluded from the training set of the model. Because retraining the model every time would be computationally expensive, an influence function was used to estimate this influence. [12] An advantage of FIA to association rules is that it was able to give an explanation to every recommendation, while association rules could only explain certain proportion of recommendations (measured by a metric called “Model Fidelity”) [1][12].

Machine learning techniques have also been employed in generating post-hoc explanations. Wang et al. designed a framework based on reinforcement learning that not only attempts to explain the recommendations of any black-box system but also attempts to optimise the generated textual explanations according to some knowledge of what good explanations should look like, such as sentence length. [13][11] While Machine Learning can empower the explanation generation, it risks kicking the explainability problem down the road – if the explanation generator is a black box itself, do you then need another system to explain the behaviour of the explanation generator?

Overall, the post-hoc approach is the most flexible way to generate explanations to recommender systems because it allows the original recommendations model to be a black box and will therefore work with most models. [11] However, post-hoc models may not be able to explain every recommendation [1], or they may be uninterpretable black boxes themselves [13].

Evaluating explainable recommendations

Like everything in science, new recommender systems need to be evaluated according to some metric to make sure they are useful to users. There are two main concerns in evaluating explainable recommendations systems: evaluating the accuracy of the recommendations themselves, and evaluating the quality of the explanations. [11]

In terms of accuracy, the goal for an explainable recommender system is to perform as well as a standard, non-explainable recommendations model. The accuracy of the system can be evaluated using standard methods, such as root mean square error (RMSE) of predicted numeric ratings, or precision and recall when predicting top-n recommendations. [11]

While evaluating accuracy is relatively easy, it is much harder to evaluate the quality of the explanations themselves. This is because quality in this case is somewhat subjective – users *A* and *B* might prefer different kinds of explanations. Both quantitative and qualitative techniques have been developed to rate the quality of explanations. One way to quantitatively evaluate explainability is to calculate the proportion of recommendations made by the system that are deemed explainable. This metric, called **model fidelity**, was first proposed by Peake and Wang for evaluating their association rules -based explanations. [1] Similar approach was taken by Abdollahi and Nasraoui, who proposed explainability precision and -recall for measuring precision and recall for explainable recommendations [14]. Just as precision measures the proportion of relevant items in recommendations, explainable precision describes the proportion of explainable items amongst those recommended. Explainable precision is therefore a similar metric to model fidelity.

These metrics are useful for models where only a proportion of all recommendations are explainable. They are not relevant for models that can generate some type of explanation to all recommended items, as they would achieve perfect fidelity by definition. They also don't say anything about the quality of the explanations, meaning that, with model fidelity for example, a model may have high fidelity but produce explanations that are effectively useless for humans.

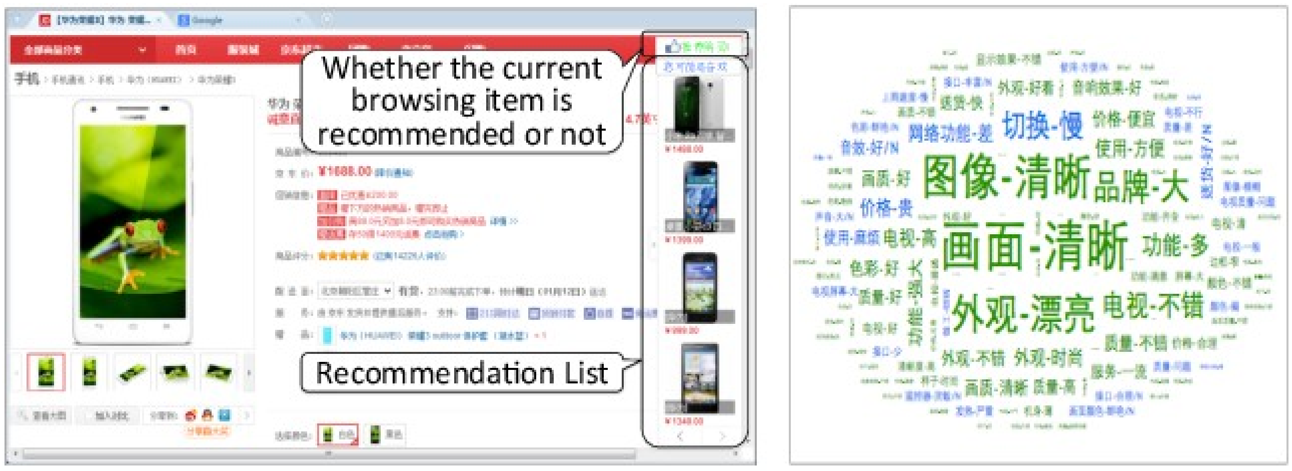
Explanations can also be evaluated using user studies. This is a good approach for measuring explainability because the motivation for improving explainability is to help users understand or trust the recommendations. [21] In their research on using reinforcement learning in explaining recommender systems, Wang et al. used human subjects to evaluate their system alongside specialised quantitative methods. Using a database of Yelp reviews on restaurants, their system generated textual explanations for why the restaurant is recommended to the user. In each task, the users were asked to choose the explanations that adequately explained why the restaurant was recommended to them. The new system was tested alongside two other systems, one that selected sentences from the reviews randomly and a "state-of-the-art explainable recommendations model", where the new system was found to have performed the best. [13] The study specifically measured the "usefulness" of the explanations, which roughly corresponds to the effectiveness-metric in the seven explainability metrics listed earlier.

If the researches have access to real-world systems, experiments can be run using their users in much larger scale and more realistic environment than what is possible in the laboratory. Zhang et al. tested their Explicit Factor Model in the Chinese online shopping website JingDong, which at the time of research had more than 100 million users. The researchers found that in an A/B test, the Click-Through Rate (CTR) was significantly higher for users who received their feature-level explanation than for those who received a generic "People also viewed..." explanation or no explanation at all. [10] CTR is a measure of the persuasiveness of the explanation, which is especially valuable as improving it can have a significant impact on a company's revenue in industrial applications. While experiments like this are easy to run for companies with an established user base, it is important to consider the potential risks to users' privacy as they may not have explicitly signed up to participate in the experiment, and to make sure gathered data is anonymous or properly anonymised.

Visualising explanations

After the explanation has been generated, it needs to be presented to the user. This is an important step: no matter how insightful an explanation, it won't help if the user can't understand it. While the presentation of explanations falls into the fields of human-computer interaction and user interface design which are out of scope for this review, it's important to explore some of the presentations used in previous applications.

The user study in the e-commerce platform JingDong used word clouds of feature-opinion pairs as its explanation (Figure 2) [10], while an example by Amazon.com mentioned in the section below uses a list of items that influenced the recommendation (Figure 3) [17]. Both represent explanations deployed in real systems with real users. It's also worth noting that both explanations require some user action to be shown: the JingDong explanation is displayed when the user hovers over a recommendation, while Amazon's is hidden deep in the platform and under a special pop-up window. This suggests that in real systems, explanations should be shown as "details on-demand" items instead of being the main focus of the view.



(a) In browser recommendation (b) Word cloud

Figure 2: *The interface design of an explanation deployed in JingDong in an A/B test. The top-4 recommendations are shown to user as they browse items in the e-commerce platform a), and the word cloud b) is shown as an explanation when the user hovers over a recommendation. Reproduced from Zhang, Y., Lai, G., Zhang, M., Zhang, Y., Liu, Y., & Ma, S. (2014, July). Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval (pp. 83-92). ACM.*

In addition, Tintarev and Masthoff have suggested several methods and interaction techniques for presenting explanations, as well as the recommendations themselves. Perhaps most interestingly, Table 2 shows their example of how to present influence-style explanations for a recommender system. [21]

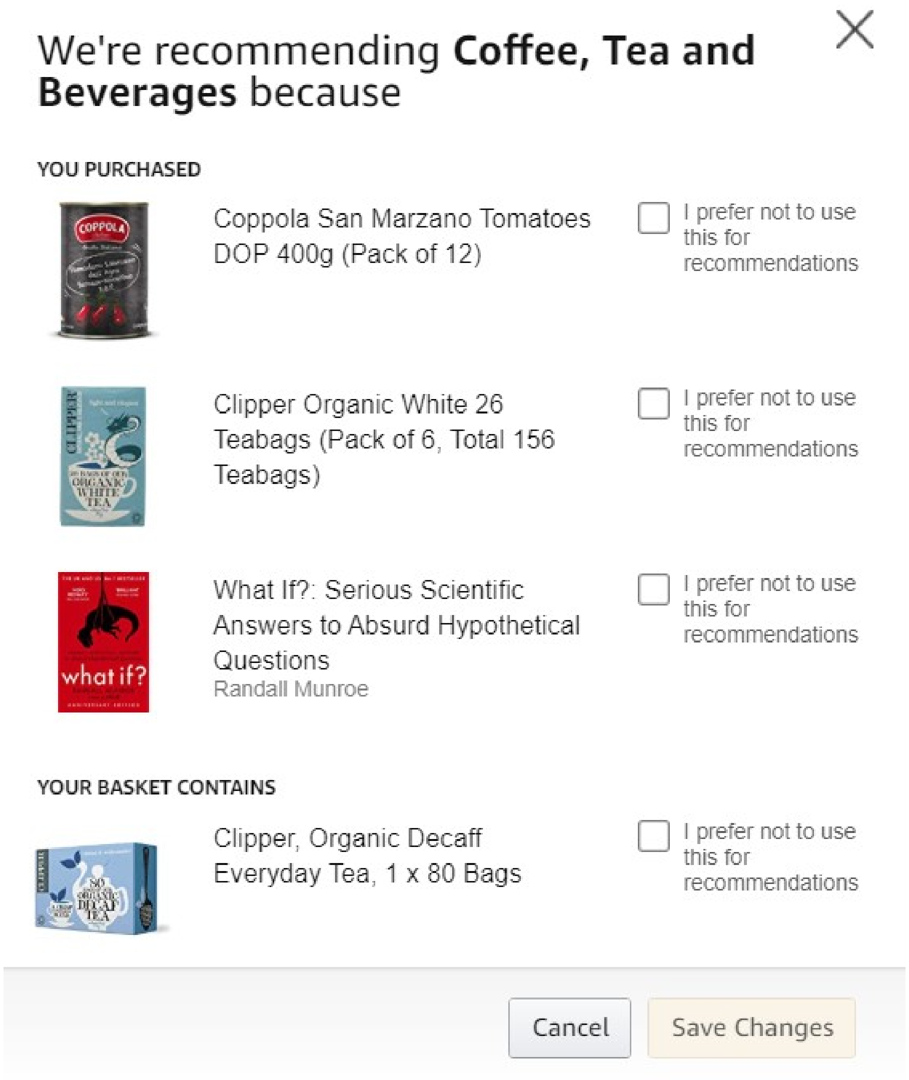
Book	Your rating (out of 5)	Influence (out of 100)
Of Mice and Men	4	54
1984	4	50
Till We Have Faces: A Myth Retold	5	50
Crime and Punishment	4	46
The Gambler	5	11

Table 2: *An example on how to present an influence-style explanation (i.e. which of the previously rated books influenced the recommendation the most) to a user. Adapted from Tintarev, N., & Masthoff, J. (2011). Designing and evaluating explanations for recommender systems. In Recommender systems handbook (pp. 479-510). Springer, Boston, MA.*

Uses of explainable recommendations in industry

Major web platforms have pioneered the use of recommender systems in their products, but so far few have focused on improving the explainability of the systems. There's little academic research published on the applications of explainable recommendations in industry, so a variety of less reliable sources are used in this section. In addition, due to the quick pace of innovation in today's technology companies, the examples used here will likely be out of date quickly.

Amazon has been working on their sophisticated recommendations models since at least the early 2000s [4], and they have used a variety of explanations in their systems. Firstly, some recommendations are labelled with a sentence describing how it was found, such as "Customers who bought this item also bought...". The latter explanation type has been copied by various other web platforms such as the investing app Robinhood. [16] Amazon has also introduced the ability for users to influence how much a purchase should influence their recommendations, or if it should be filtered out altogether. This can be useful for filtering out items that were purchased as gifts, for instance. Users can also see which of their previous purchases or ratings influenced a recommendation, as shown in Figure 3 [17]. This explanation style is similar to the association rules [1] and influence analysis [12] described in previous sections and implemented in this project.



We're recommending **Coffee, Tea and Beverages** because

YOU PURCHASED

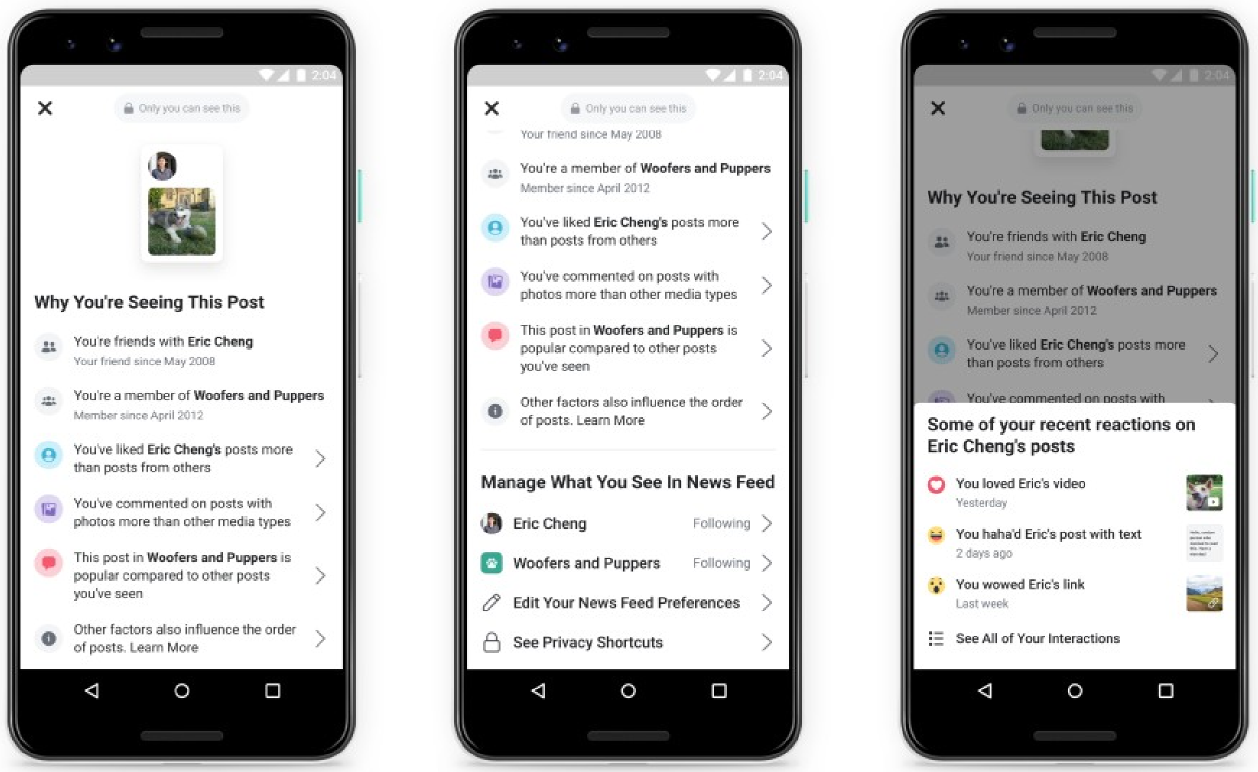
Coppola San Marzano Tomatoes DOP 400g (Pack of 12)	☐ I prefer not to use this for recommendations
Clipper Organic White 26 Teabags (Pack of 6, Total 156 Teabags)	☐ I prefer not to use this for recommendations
What If?: Serious Scientific Answers to Absurd Hypothetical Questions Randall Munroe	☐ I prefer not to use this for recommendations

YOUR BASKET CONTAINS

Clipper, Organic Decaff Everyday Tea, 1 x 80 Bags	☐ I prefer not to use this for recommendations

Figure 3: *An explanation by Amazon.com explaining why the category "Coffee, Tea and Beverages" is recommended to the author, containing items the author has purchased in the past. The widget also allows excluding an item for consideration (for example, if it was a gift or the user didn't like it), thus improving the scrutability of the system.*

Facebook uses artificial intelligence in deciding what content to show to its users. The decision on which posts or advertisements to show to a given user can be thought of as a recommendation problem. Facebook launched its first explanations in 2014, when they released their "Why am I seeing this ad?" feature. In 2019, the system was improved and extended to cover posts. With the new feature users can see why they're seeing a post as well as indications on why the posts are ordered in the way they were. Users are also able to see detailed information on why an advertisement was shown to them, including a timeline on how the advert creator has interacted with their personal data. Screenshots of the feature are shown in Figure 4. By releasing the feature Facebook wanted to increase transparency in its products, as well as help users control their own news feed, i.e. improve the scrutability of the product. [18] In addition, fears of Facebook being used by political actors to show highly targeted, often false or inaccurate, political advertisements increases the need for transparency in social networks [19], and explainable recommender systems can play a role in it.


Only you can see this

Why You're Seeing This Post

You're friends with **Eric Cheng**
Your friend since May 2008

You're a member of **Woofers and Puppers**
Member since April 2012

You've liked **Eric Cheng's** posts more than posts from others

You've commented on posts with photos more than other media types

This post in **Woofers and Puppers** is popular compared to other posts you've seen

Other factors also influence the order of posts. Learn More

Manage What You See In News Feed

Eric Cheng — Following

Woofers and Puppers — Following

Edit Your News Feed Preferences

See Privacy Shortcuts

Some of your recent reactions on Eric Cheng's posts

You loved Eric's video
Yesterday

You haha'd Eric's post with text
2 days ago

You wowed Eric's link
Last week

See All of Your Interactions

***Figure 4:** Facebook's "Why are you seeing this post?" explanation, telling the user what past actions caused the post to be shown to them. Reproduced from Sethuraman, R. (2019, March 31). Why Am I Seeing This? We Have an Answer for You. Retrieved October 2, 2019, from https://newsroom.fb.com/news/2019/03/why-am-i-seeing-this/.*

Recently some recommender systems have faced criticism for directing users to a "rabbit hole" of increasingly radical content. An example of such system is YouTube's recommendation system, which recommends videos and channels for a user based on their watch history. When studying right-wing channels, researchers found some evidence that YouTube's recommendations algorithm radicalised users by recommending increasingly radical channels to them [20]. It's worth noting that this doesn't mean there's anything *wrong* with the recommender system – in fact, it seems to work exactly as it was designed to, recommending videos accurately to users who would be interested in watching them. This problem can be called the **radical content problem**, and little academic research has been conducted on solving it. An interesting research question for the future would be seeing if any of the explainability approaches could be used in discouraging the recommendation systems from recommending increasingly radical content, or discovering if the problem exists in the first place. Using association rules to visualise and analyse the recommender system as a network to identify the existence of the radical content problem in the system is briefly considered at the end of this report.

Conclusion

Research on explainable recommendations (and explainable AI in general) has advanced rapidly over the past few years. This has likely been motivated by the increased deployment of recommender systems in real-world systems as well as the development of more accurate but less interpretable models such as matrix factorization. The task of developing perfectly explainable recommendation systems is still far from complete. Due to the negative media attention big social media companies like Facebook and YouTube have received for their recommendation systems, it is likely that we will see significant progress in real-word adoption of some of the above-mentioned explainability approaches in the next few years, suggesting a bright future for explainable AI research.

Goals

The central goal of the project was to improve the explainability of user-facing recommender systems. To achieve that, a number of following primary goals were set. The goals were:

1. Implement a state-of-the-art recommender system.
2. Implement two ways of generating post-hoc explanations to recommendations
3. Define the metrics on how to best evaluate the explanations, and conduct experiments to evaluate the performance of the explanations under the metrics

As shown in the context survey, “state-of-the-art” was determined to mean a latent factor model -based recommender system. It was decided that the explanations should be evaluated using a user study as well as offline experiments, and as such setting and running a user study on the explanations was chosen as the main method for achieving the third goal.

The scope of primary goals was uncertain at the start of the project. Most importantly, it was not clear whether a user study would be required for evaluating the explanations. For this reason, two secondary and tertiary goals were also defined. The secondary goals were improving the association rules -based explanation generation algorithm and allowing users to influence the system’s recommendations by telling it they are not interested in a recommended movie (i.e. improving the scrutability of the recommender system). The tertiary goals were finding ways to objectively measure and test the best ways to visualise explanations to the users, and exploring the ways explainability can help data scientists understand the recommender system better.

Design

The primary goals of the project directly correspond to the steps taken in the project. This section describes the design and implementation of the recommender system, both types of explanation generators and the web application for running the user study. The user study itself is described in the evaluation-stage. Figure 5 shows the order of user and server actions in the user study, which illustrates how the various components of the system fit together.

The source code used in the research has been made available under the permissive MIT License [33]. It can be found in two separate repositories: repository SHProject includes the Python code used in investigation and running the back-end server in the user study, and repository WebAppSHProject features the React.js front-end application used in the user study. The applications are hosted in GitHub under the URLs https://github.com/villekuosmanen/SHProject and https://github.com/villekuosmanen/WebAppSHProject. The code can be used to repeat the experiments presented in this study, and with minor modifications the implemented explanation generators should be usable in any recommender system.

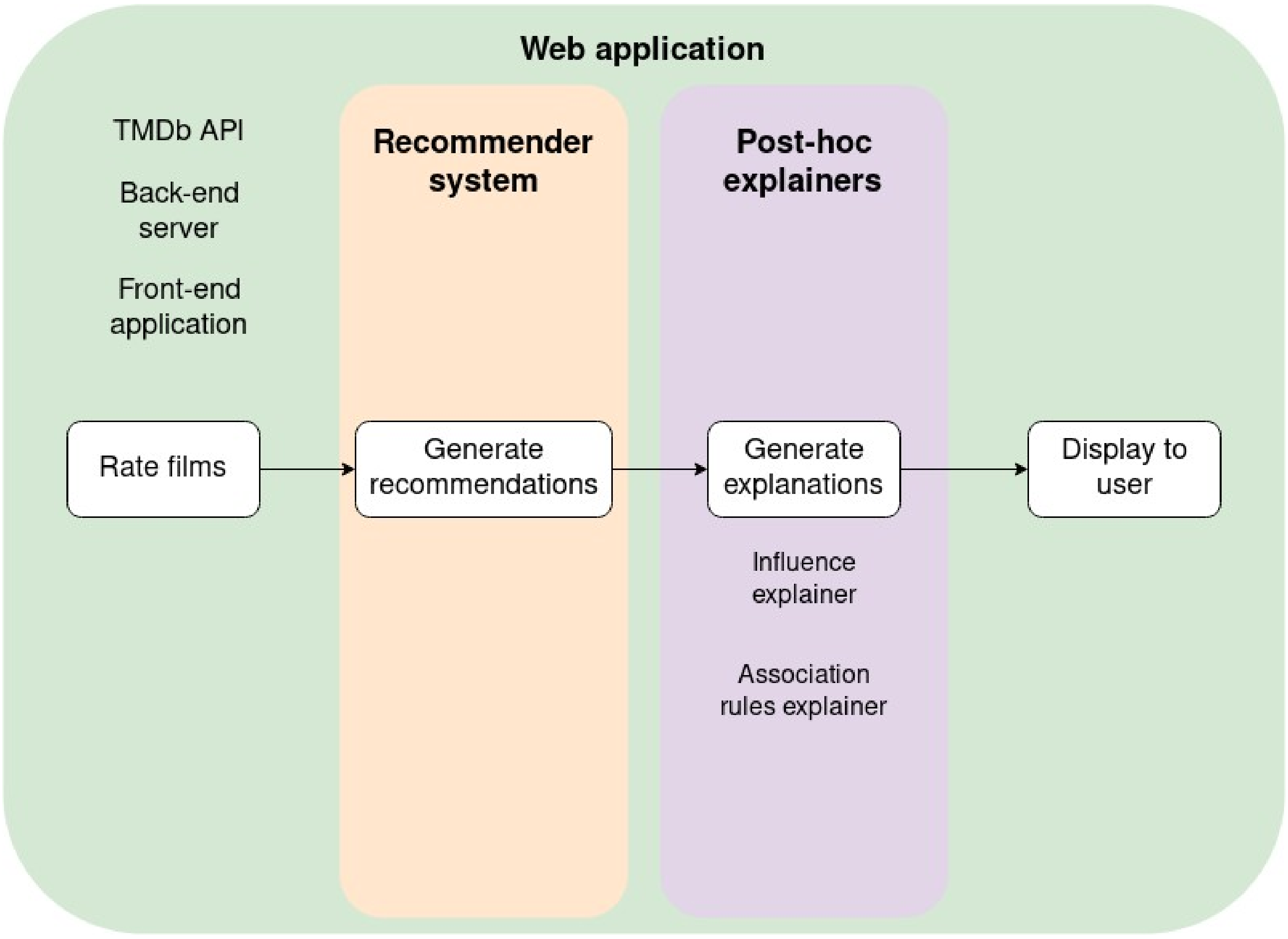


Figure 5: *A flowchart describing the order of actions in the user study. Using the web application, the user first rates films. The recommender system then generates recommendations for them, the explainers generate explanations for the recommendations, which are then shown to the user.*

Recommender System

The recommender system was designed to be based on collaborative filtering (CF) techniques. In CF , the recommendations are based on reviews from other users similar to you [7]. CF was chosen as the technique of choice due to its high accuracy, domain-independence (CF techniques don't need domain knowledge about the items, unlike content-based ones) and abundance of high-quality datasets. The dataset used for the problem was the publicly available MovieLens dataset consisting of film ratings by real users [24]. This dataset is widely used in recommender systems research, and since CF is not domain-specific, the models and algorithms will usually generalise to other fields besides film ratings. When developing the recommender system, the development dataset containing 100'000 ratings was used to lower training time; the 20 million rating benchmark set was used for evaluation and use study.

The recommender system was implemented in Python using the Surprise library [25], which provides a range of CF algorithms for implementing recommender systems. Out of these, the implementation of SVD was selected. SVD is a matrix factorisation algorithm used to generate latent factor models, which can then be used to predict ratings that a user *u* would give to an item *i*

[7]. SVD was selected over other, generally better-performing models such as SVD++ due to its relative simplicity and status as the most well-known matrix factorisation algorithm. Because the goal of the project is to study explainability, the small deficiency in performance doesn't matter.

Before training the latent factor model, the data was split to train and test data using a random 75%-25% train-test split of ratings. This may not the most ideal way to split data: this way some users and items won't have any ratings in one of the sets, making their predictions non-personalised (i.e. prediction is the bias of the user or film). There's surprisingly little literature on the best practices of splitting datasets for matrix factorisation models; as such, the before-mentioned simple split of data was used.

After training, the recommender system can be used to predict ratings for the items and users in the training set. This is enough for static evaluation: however, the user study that was conducted using the system requires dynamic recommendations. Generating new personalised recommendations for a newly-added user with a few ratings is not possible: the model first needs to learn the user's latent factors. This can be done by fully retraining the model – however, this method is infeasible in live systems as training the model for millions of ratings is computationally expensive and would need to be done at every new rating. Because the existing model is "almost correct" and would work as a good starting point for adding the user, the method can be optimised in many ways depending on the underlying implementation of the SVD algorithm. For example, some systems based on gradient descent could be initialised with the original model's weights (here, latent factors), which would allow the algorithm to convergence faster. However, Surprise's SVD uses a fixed number of epochs in running its gradient descent, and doesn't stop until all epochs are done.

As such, a new operation for the SVD model was devised, which adds a new user to the model. This operation only trains the new user's latent factors, and leaves the factors for items and other users unchanged. This radically speeds up training. Of course, this also means that the new user's ratings won't influence the factors of items and improve the model for other users. This is an acceptable trade-off though, as the number of new ratings (~10) is so much smaller than the number of old ratings (~20'000'000). In a real system, the model would likely be re-trained regularly (e.g. once a day) to integrate the ratings into the model. The difference between retraining and using the optimisation is evaluated objectively in the Evaluation-section.

Hyper-parameter optimisation

Before deploying the system, its hyper-parameters were optimised. The Surprise SVD algorithm contained two important hyperparameters to evaluate: number of factors in the latent factor model (**n_factors**) and number of epochs in training (**n_epochs**). These parameters were optimised by heuristically sampling the space of reasonable values for the two variables, and running cross-validation on the selected values using the training set. The results of this procedure are plotted in Figures 6 and 7 for n_factors and n_epochs, respectively.

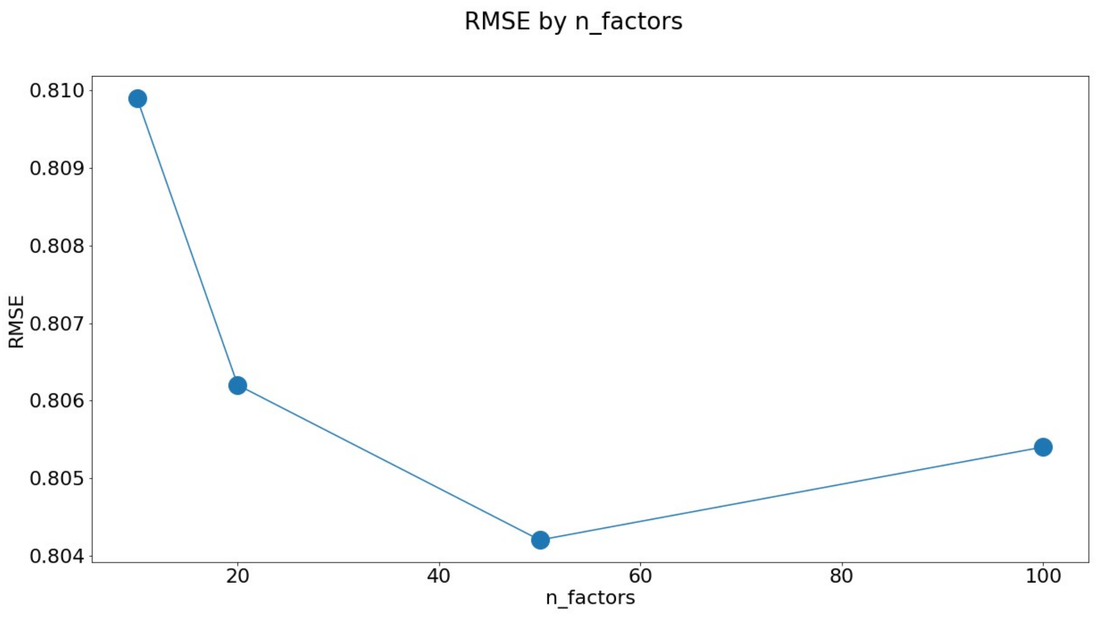


Figure 6*: The measured RMSE in cross-validation (n_folds=4) by various values of the n_factors hyper-parameter for the recommender system.*

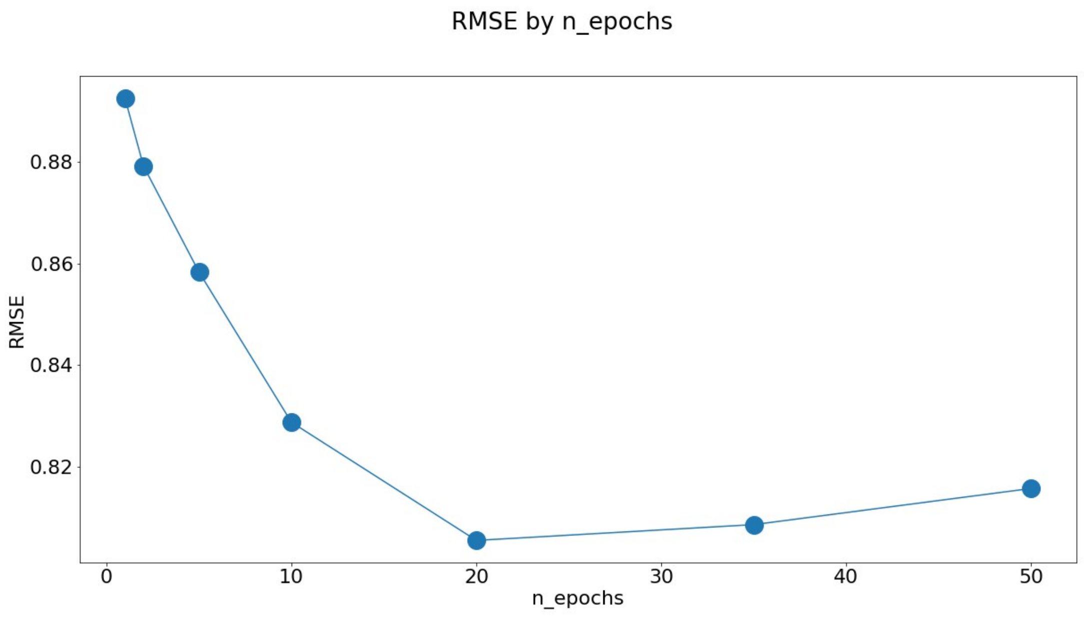


Figure 7*: The measured RMSE in cross-validation (n_folds=4) by various values of the n_epochs hyper-parameter for the recommender system.*

Based on the above results, the chosen values for n_factors and n_epochs were 50 and 20, respectively. The results can't be considered conclusive because the tests were carried separately; it's likely that the two parameters influence each other at least in some ways. In addition, more parameter values could have been tested for both parameters to improve the confidence of the best value. Despite these flaws, the tests give a good estimate at what the optimal values for the parameters roughly are, and that suboptimal hyper-parameter values aren't causing a significant negative effect on the overall performance of the model.

Ethics

Because the used dataset contains opinions of real people, an ethical approval was required to use it in research. Since the data is anonymised and public, no major concerns arose and ethical approval was granted with the code **CS14599**, which is provided as Appendix 2.

Explanations

Latent factor models perform well, but are inherently uninterpretable because the factors are learned automatically from data with no concern given to the attributes of the items and users. To make these recommendations explainable, two post-hoc methods were tested: a method based on mining association rules from the set of top-n recommendations, and a method based on calculating how much the users' previous ratings influence the prediction.

Association Rules

The method for association rules follows the approach developed by Peake and Wang [1]. As a post-hoc explanation, the method works after the recommender system has already been trained.

Firstly, the top-*n* recommendations for each user are outputted. Unlike in the original paper, the items previously rated by the user were not filtered out (though they were filtered out in the evaluation stage). This way the top-*D* items selected for training the association rules were simply the top-n predicted ratings. The value of *n* and *D* selected was (*n, D)*=30, in accordance with the source paper.

The set of top-*D* items for all users were then used to train the set association rules. These rules describe IF => THEN relationships in the data, and can be mined from a set of transactions using a known algorithm. The set of transactions used was the set of top-*D* items for all users, and the apriori algorithm [26] was used to generate the association rules. An implementation of the apriori algorithm from the mlxtend Python library [27] was used. The set of association rules was the filtered according to a few rules. Firstly, the maximum number of items in antecedents (the IF-side of the rule) was three, and maximum number of items in consequents (the THEN-side of the rule) was one. This meant that rules of type ((A => D), (A & B => D) and (A & B & C => D)) were allowed. The minimum support of the rules was set as 0.05, while the minimum confidence was set as 0.3. Here, support refers to the share of all transactions that contain all items in the rule, and confidence refers to the share of transactions containing the antecedents that also contain the consequent.

Once the association rules were filtered, they were sorted according to support. The association rules can then be used to generate explainable recommendations in two ways. Firstly, the rules can be used directly as a recommender model by filtering the rules to those whose antecedents appear in the user's list of previously rated films. The consequents in the remaining rules work as explainable recommendations with the explanation "*Because you watched X [antecedents], we recommend Y [consequent]*". However, this method doesn't properly personalise the recommendations to each user since the recommendations are sorted by global support instead of the predicted rating of the recommender system. The second way is to generate recommendations as normal using the recommender system, and try to make them explainable afterwards by using the association rules.

This is called post-hoc explainability, and in association rules works if a rule is found where the consequent is the predicted item, and the antecedents have all been previously rated by the user. Since not all recommendations can be made explainable, this method introduces the concept of model fidelity. Model fidelity describes the share of recommendations that can be made explainable amongst all of them, and is explored further in the Context study and Evaluation.

Influences

The second type of post-hoc explanation considered in this project was showing how much each previously rated film influenced the rating. Influence analysis has been previously applied to recommender systems in the Fast Influence Analysis system [12]. The influence of a rated film is the difference between the predictions with and without that rating in the recommender system's training set. Formally, for a user with a set of ratings $R = \{r_1, r_2, r_3 \dots r_n\}$ and a recommender system S outputting a predicted rating for an item I and a set of user ratings R, the influence of r_1 to a predicted rating for any previously unrated film is:

$$\mathit{Influence} = S(I, R) - S(I, R \setminus r_1) \tag{1}$$

The simplest way to calculate the influence of a previously rated item on a recommendation is to re-train the model without that rating in the training set, and compare the resulting predictions. However, this idea is absurd due to the high computational cost in fully re-training the model, especially since an influence score would need to be calculated for every previously rated item in the training set. FIA solved the problem by applying statistical influence functions to the calculation [12]; in this project, an optimised method to approximate the influence by allowing fast re-training of a single user was used, which is described in the Recommender System -section.

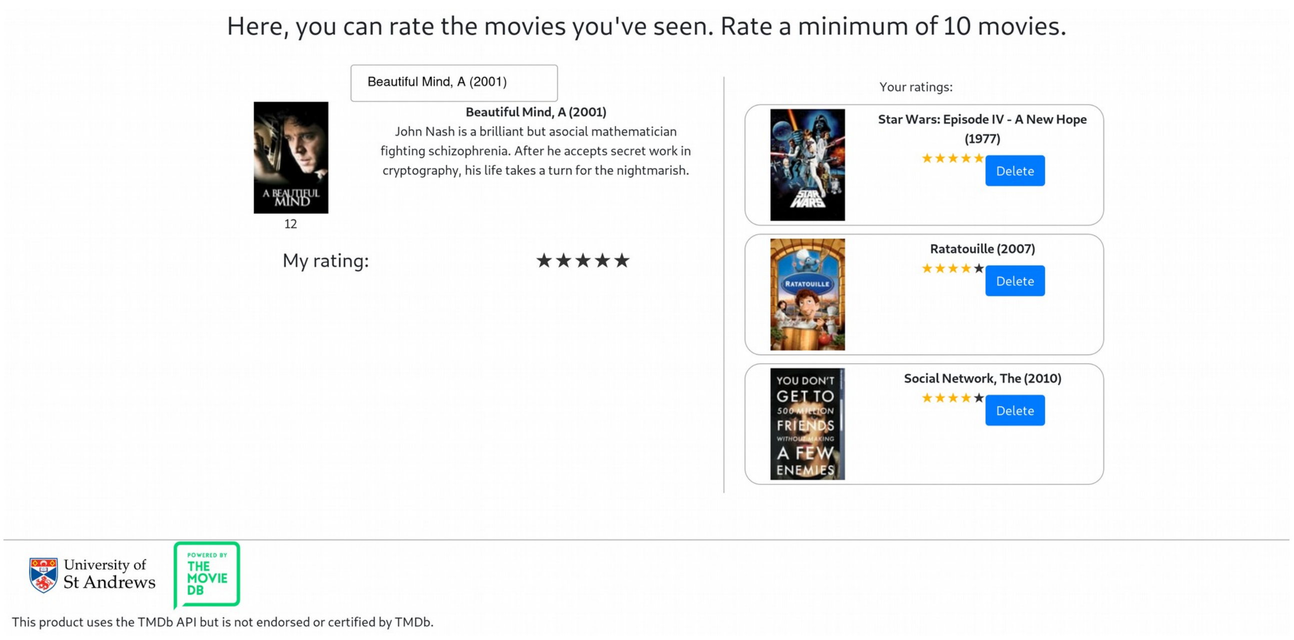
The biggest benefit of the influence method is that explanations can be calculated for any recommendation, which means that model fidelity of the explainer is always 100%. When showing the explanation to users, it may make sense to restrict the influences shown to just the most affecting ones – in the user study, the top three positive and negative influences were shown.

Web application

To study whether real users find the explainable recommendations useful, a user study was conducted. The study was conducted through the internet, which required the implementation of a web platform that would allow users to rate films, generate and present the explainable recommendations to them on-demand, and persist the generated research data and email address. The web application was implemented in two components: a front-end application built in React.js, and a back-end RESTful API that's responsible for actually generating the recommendations and explanations.

Front-end

The front-end was built as a single-page React.js [29] application using the tool create-react-app [30]. The front-end fetches data from the API and renders it to the user. All rendering is done client-side. Screenshots of the web application are shown in Figure 8 and 9, and further highlights of the explanations are shown in figures 13, 14 and 15 in later sections.


Here, you can rate the movies you've seen. Rate a minimum of 10 movies.

Beautiful Mind, A (2001)

Beautiful Mind, A (2001)
John Nash is a brilliant but asocial mathematician fighting schizophrenia. After he accepts secret work in cryptography, his life takes a turn for the nightmarish.

12

My rating: ★★★★★

Your ratings:

Star Wars: Episode IV - A New Hope (1977) ★★★★★ Delete

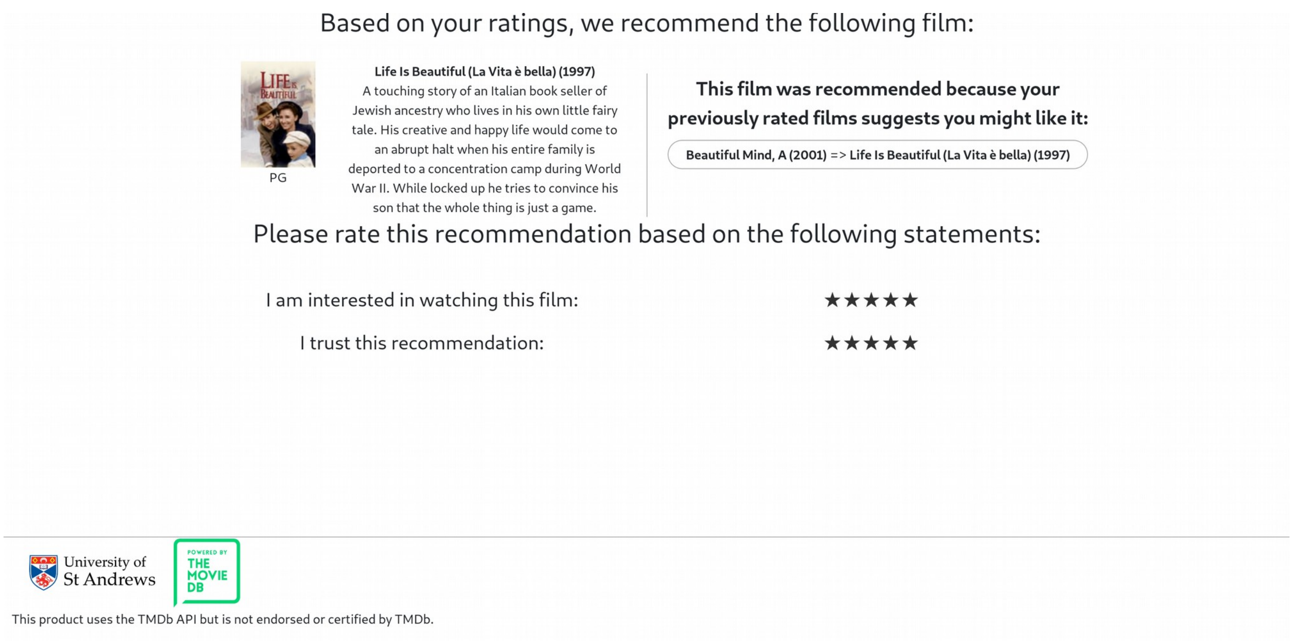
Ratatouille (2007) ★★★★★ Delete

Social Network, The (2010) ★★★★★ Delete

University of St Andrews | POWERED BY THE MOVIE DB

This product uses the TMDb API but is not endorsed or certified by TMDb.

Figure 8: *A screenshot of the rate movies screen of the web application, with several films already rated (right side of the screen).*


Based on your ratings, we recommend the following film:

Life Is Beautiful (La Vita è bella) (1997)
A touching story of an Italian book seller of Jewish ancestry who lives in his own little fairy tale. His creative and happy life would come to an abrupt halt when his entire family is deported to a concentration camp during World War II. While locked up he tries to convince his son that the whole thing is just a game.

PG

This film was recommended because your previously rated films suggests you might like it:

Beautiful Mind, A (2001) => Life Is Beautiful (La Vita è bella) (1997)

Please rate this recommendation based on the following statements:

I am interested in watching this film: ★★★★★

I trust this recommendation: ★★★★★

University of St Andrews | POWERED BY THE MOVIE DB

This product uses the TMDb API but is not endorsed or certified by TMDb.

Figure 9: *A screenshot of the recommendations screen of the web application. Here, the user is shown details of the recommended film, and explanation (association rules) and a widget to rate the recommendation.*

API

The API was built using the Flask web framework [31], and contains five endpoints:

- /movies (GET)
- /movies/details/<int:movie_id> (GET)
- /movies/movie-ratings (POST)

- /recommendations/responses (POST)
- /emails (POST)

All of these endpoints are stateless, although they do have an implied ordering: the emails and responses endpoints aren’t supposed to be used until the end of the study.

The movies-endpoint is used to fetch the titles and ids of available films in the database. This allows the client to offer an auto-complete search bar to the user, at the expense of transferring a relatively large amount of data. The endpoint for movie details is a proxy for searching its data from The Movie Database (TMDb) and returning the result to the user. This request is proxyed through the back-end for three reasons: TMDb uses different film identifiers than the rest of the application (which uses MovieLens ids) so the back-end also performs id translation, direct requests to the TMDb API from the client could be blocked by Cross-Origin Resource Sharing (CORS) policy [32], and embedding the TMDb API key in client code would compromise the key, which is considered secret.

The POST-request to movies/movie-ratings posts the user’s rated movies to the back-end. This triggers a long computation in the back-end, as it adds a new user to a copy of the pre-trained recommender system, generates recommendations to the user, and finally generates explanations to them. The explainable recommendations are then returned to the client. POST-requests to responses and emails are persisted in the private part of the file system of the server, in different files.

Deployment

The web application was deployed to the host server provided by the School of Computer Science in the University of St Andrews. The built version of the front-end was deployed in the public folder of the nginx server, while requests to the API were proxy-ed to the WSGI server running the API process on the host server. As a result the web application was accessible to end users in the address https://vik.host.cs.st-andrews.ac.uk/, and research data was saved in the encrypted, access-controlled school servers.

To make sure the application is working as expected, information and errors from the web application were logged to log files. These logs did not compromise the privacy of users, since there’s no way of knowing which requests were made by which users (as the users had no individual id’s).

Evaluation

Explanations exist for humans, not machines. As such, the best way to evaluate explanations is to use live users. While explainability offers many benefits to end users, the two central benefits are increase in trust and user interest. Previous studies have found that adding explanations to recommendations made users like the system more, as well as feel more confident about its recommendations [9] as well as increase their Click-Through Rate (CTR) in comparison to recommendations with no explanations [10]. Because of this, a user study was conducted test if the type of explanation offered affects the trustworthiness and persuasiveness of the recommendation.

In addition, the recommender system and explanation generators were evaluated offline (i.e. without feedback from human users) using a variety of metrics.

User study

The study was conducted to test the effect of explanation type to the measured trustworthiness and persuasiveness of the recommendation. The explanation types studied were

1. baseline condition with no personalised explanation (baseline)
2. explanation based on association rule mining (association rules), and
3. explanation based on calculated influence of rated films (influence).

To see recommendations, users would first rate a number of films they've seen before, on the scale of 1-5. Afterwards, the system would attempt to show 5 recommendations for each explanation type to them. For each recommendation, the user would rate how interested they are in viewing the film (persuasiveness) and how much they trust the recommendation (trustworthiness).

Recommendations for association rules explanation were selected in a different way than for the other conditions due to the low model fidelity of the explanation generator (evaluated in the Model Fidelity section below). The top-1000 films recommended were checked for explainability, and the top-5 of them were selected, if available. In the uncommon case where no explainable recommendations were found, no association rules explanations were shown for the user. For the other two conditions, the top-10 recommendations (excluding ones used for association rules) were randomly split between the conditions. Before presenting the recommendations to the user, their order was shuffled.

To allow users to make more informed decisions on whether they'd like to see the recommended film or not, the film's poster and a short description was included with it. This data was fetched through The Movie DB API [36].

Data collection

The study was conducted online in a purpose-built web application. The application is described in the "Web Application" subsection of Design, and it was hosted in the researcher's host server in the University of St Andrews domain. Test subjects were recruited by circulating the study advert in social media and university newsletters. The study took around 10-15 minutes to complete, and at the end participants were given an opportunity to participate in a raffle to win a £50 Amazon voucher. Overall, 41 participants took part in the study.

Ethics

An ethical approval for the study was requested before any research activities were conducted. To protect the participants, data collection for the main part of the study was done anonymously. In addition, users' rated movies and recommendations concerned were not recorded, since these ratings could, in theory, leak the users' identity. The email addresses collected to raffle off the prize were stored separately from the main data.

In addition, it was deemed possible for certain participants to be affected by strong adult content or themes in some recommended movies. This was mitigated by allowing users to opt out of seeing movies rated 18 or R18 by the British Board of Film Classification in the application. With these concerns resolved, ethical approval was granted to the study with the code **CS14723**, which is provided as Appendix 3.

Results

The results of the study were analysed as a one-way repeated measures ANOVA to compare the effect of explanation type on recommendation interest and trust. The value used for interest and trust in each condition was the mean of all ratings (in that condition, by that user). This was done to simplify the data analysis. In addition, ratings for recommendations which lacked a description and poster in The Movie DB API were excluded from the analysis, as were users who had not given a score for one of the three categories. This led to the sample size reducing from 41 to 32 participants. The types of explanations tested, referred to as "conditions", were "baseline", "association rule" and "influence" (terms defined in Experiment Design). As repeated measures ANOVA was selected as the statistical significance test, Mauchly's test was used to confirm that the data fulfills the sphericity assumption.

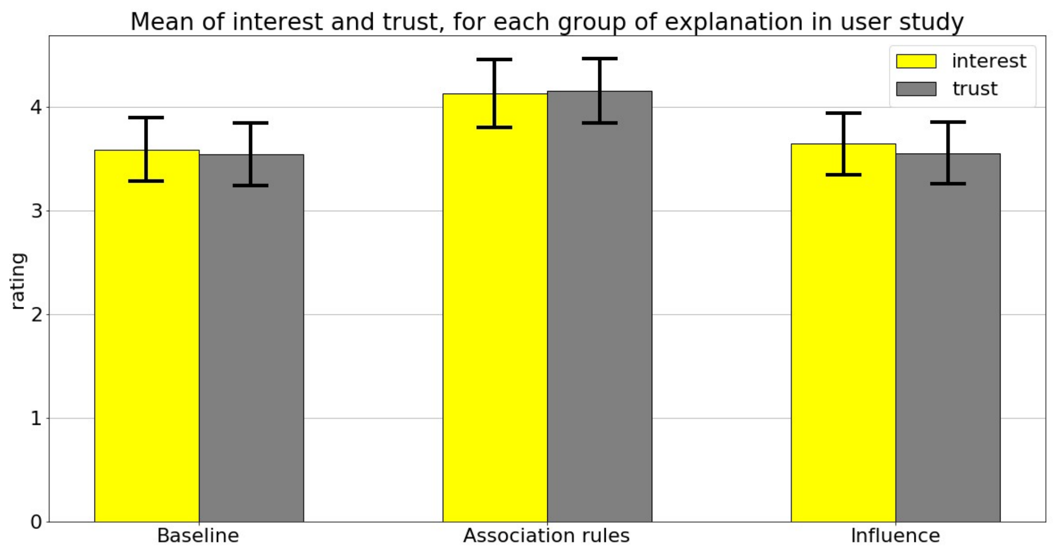


Figure 10: *A bar chart of mean of user-rated interest (yellow) and trust (grey) measured in the user study, for each type of explanation. Error bars (i.e. 95% confidence intervals) are also shown*

For both interest and trust, Mauchly's test showed that the data fulfills the sphericity assumption, χ2 = 2.746, 2.052, p = 0.253, 0.358. There was a significant effect of explanation type to interest, F(2, 62) = 5.166, p = 0.008, and to trust, F(2, 62) = 7.254, p = 0.001.

Bonferroni corrected paired samples t-tests were used to make post hoc comparisons between conditions, for both interest and trust and between all conditions. For interest, there was a statistically significant difference between the ratings for association rules (M= 4.13, SD=0.91) and baseline (M= 3.59, SD=0.85) conditions, p=0.031, and the difference between the ratings for association rules (M= 4.13, SD=0.91) and influence (M= 3.64, SD=.82) conditions were close to statistical significance, p=0.061. For trust, there was a statistically significant difference between the ratings for association rules (M= 4.15, SD=0.85) and baseline (M= 3.54, SD=0.83) conditions, p=0.006, and association rules (M= 4.15, SD=0.85) and influence (M= 3.55, SD=.83) conditions, p=0.003. The means of measured interest and trust for each explanation type are plotted in Figure 10 along with their 95% confidence intervals.

Recommender system

The recommender system was evaluated by predicting ratings for all user-item pairs in the test set, and calculating the error metrics in the prediction. The metrics used were root mean square error (RMSE) and mean absolute error (MAE) which are commonly used for evaluating recommender systems [7]. The measured errors on the tet set with hyperparameters n_factors=50, n_epochs=20 was: **rmse=0.7897**, **mae=0.6010**. Most benchmark results for the MovieLens datasets come from the 10M ratings dataset, where a good benchmark result for SVD measured a RMSE of 0.7720 using a 90:10 percent split in training and testing data [28]. While the 10M and 20M datasets, as well as the used splits aren't directly comparable, the fact that the literature value on a smaller dataset outperforms the result in this report suggests that in terms of performance, the model is suboptimal. Since the recommender system's performance is not the main focus of this project, it's not a major issue.

Model Fidelity

As defined by Peake and Wang, model fidelity describes the share of recommendations that can be explained by the explanation model, or formally:

$$Model\,Fidelity = \frac{explainable\ items \cup recommended\ items}{recommended\ items} \tag{2}$$

Where *explainable items* and *recommended items* are subsets of all items in the system. [1]

Some post-hoc explainers, such as the influence explainer shown here, have a model fidelity of 100%, because they can by design explain every recommendation. For such explainers, model fidelity is not a sensible measure. For the association rules explainer, model fidelity is a key measure since not every recommendation can always be explained.

Model fidelity was measured by listing the top-*n* recommendations for all users in the dataset. The previously-generated association rules were then used to generate explanations for the recommendations. Association rules were generated using the threshold support and confidence values of 0.05 and 0.3, respectively. The number of explainable recommendations was then divided with the number of all recommendations to get the result.

To run the calculation, a sample of 1/10th of users was selected. The users were selected by picking every 10th user by their id value. In addition, recommendations of items the users had previously rated (in their training set) were filtered out. In this way, model fidelity was measured to be 0.0650, or around 6.50%. This is almost an order of magnitude lower than the results reported previously [1], though the numbers aren't directly comparable due to different thresholds and datasets used. The significance of this result is discussed further in the Discussion-section.

Evaluating influence calculation

To evaluate the accuracy of the influence explanation quantitatively, two experiments were set up. The first experiment studied how much the measured influence of a previously rated film varies when the user is added to the model several times. Ideally, the variance should be low: the influence of a film on a prediction should remain consistent. The second experiment compared how the optimised method of training the model for a user compares to full retraining of the model, in measured precision and recall of predictions. In order for the optimisation to be valid, there should not be a major difference between the methods. Finally, the achieved speed-up by using the optimisation versus fully re-training the model was studied.

Experiment 1

In the first experiment, 100 simulated users were created, with 10 random ratings for random films each. Each user was then predicted their top-6 recommendations. For each recommendation, the influence of all the films rated by the user was calculated 20 times, and the mean and standard deviation of the calculated influences of a particular film was recorded. In the end, the mean of means all influences was measured to be 0.166, and the mean of standard deviations of the repeated mean calculation was 0.128. The distributions of measured means and standard deviations are plotted in Figures 11 and 12.

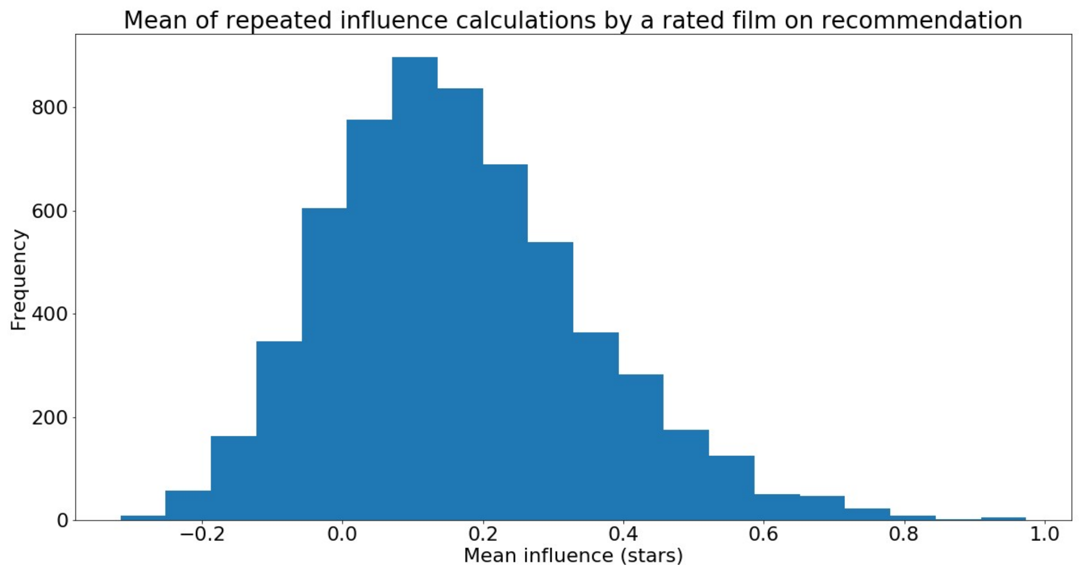


Figure 11*: A previously rated film’s influence on a recommendation was calculated repeatedly (*n*=20). The density of the means of these influences is plotted here.*

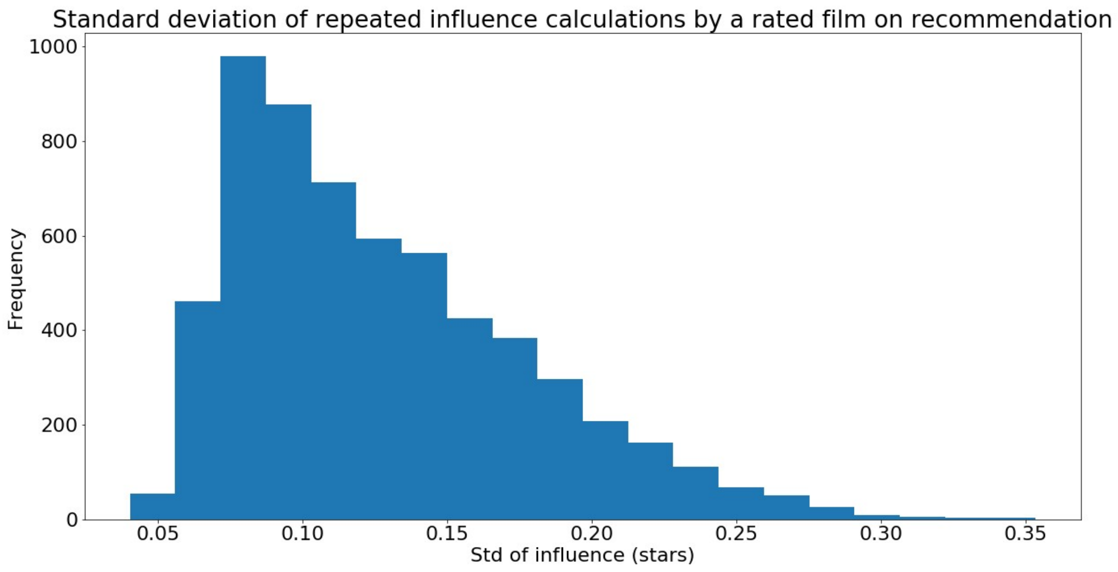


Figure 12*: A previously rated film’s influence on a recommendation was calculated repeatedly (*n*=20). The density of the standard deviations of these influences is plotted here.*

The measured standard deviation shows that the influence calculation has a relatively high variance. If the average rated film’s calculated influence on the prediction is 0.166 stars, recalculating the influence will have lower and upper 95% confidence bounds (+-2SD) of 0.09 – 0.422 rating points. These results shows that the calculated influences are more like “rough guesses” than exact values, but that this uncertainty is more due to the random effects in training this recommender system than due to issues with the optimised retraining method.

Experiment 2

Second experiment studied the precision of recommended films between the optimised training and a full retraining of the model. Due to the high computation cost of retraining, the development dataset containing 100’000 ratings was used for this experiment. For 50 times, a random user was constructed with 20 random ratings. Then, three recommendation algorithms were trained: One fully trained with the training set and the user added through the optimisation (the effect algorithm), and two trained with the user added directly to the training set (the baseline and control algorithms). 20 film recommendations were then generated for each algorithm. The precision of the control and effect algorithms was calculated as the number of recommendations that were also recommended to the baseline algorithm, divided by total number of recommendations.

In the end, the mean and standard deviation of precisions for effect and control algorithms was calculated. The results were:

Effect: mean=28.1%, std=7.8%

Control: mean=28.7%, std=8.2%

As the means are so close to each other, there doesn’t seem to be a meaningful difference between the algorithms, and as such the optimisation for adding a user to the algorithm seems valid. However, the precision of both groups is surprisingly low. This indicates that the recommender system used has a relatively high effect of randomness in training it.

Experiment 3

In experiment 3, the speed-up achieved through the optimisation was compared to fully retraining the model. To do this, a timing of the model was first measured for ordinary training. Then, an operation that adds and trains a new user on the model was timed. Both timings were taken three times. The results were:

- Full training: 10min 46s, 11min 13s, 10min 40s (mean = 10min 53s)
- Optimised: 0.4s, 0.01s, 0.01s (mean = 0.14s)

Based on the results, the optimised method is thousands of times faster (~4566 times, in fact!). It’s also more scalable – the time takes should not increase significantly as the number of ratings increases.

Discussion

This section first discusses whether the goals set at the beginning we achieved, the validity and implications of the user study, and the challenges in directly adding the implemented explanation generators to real systems. The report then defines the *extreme content problem*, and the possibility of using association rules to map the recommender system and critically evaluate whether the system suffers from the problem.

Goals

The primary goals of the project were:

1. Implement a state-of-the-art recommender system.
2. Implement two ways of generating post-hoc explanations to recommendations
3. Define the metrics on how to best evaluate the explanations, and conduct experiments to evaluate the performance of the explanations under the metrics

All primary goals have been achieved. While the goals didn't define the usefulness of the explanations as an explicit goal, it's implicitly assumed to be a metric of success. As such, the Discussion-section below contains a thorough discussion on how the explanations could be added to a real system, and whether adding them would add value to its users. The recommender system implemented here is state-of-the-art in the algorithms used in implementing it – in order to use it in a production-grade system, the model would likely require some improvements to its accuracy as mentioned in the Evaluation-section.

In order to focus on the thorough evaluation of the implemented explanation generators, the secondary and tertiary goals were explored in lesser extent. While this project allowed the use of association rules with more than one antecedent, no theoretical analysis was conducted as to why the explanation generation here is better than in the past. In addition, scrutability was not considered as a metric of explainability in this research. Of the tertiary goals, the goal of using explanations in data analysis was explored through mapping the association rules as a network. This allows visualising the recommender system, and potentially analysing it quantitatively using network analysis tools. This is described later in the Discussion-section, and is considered to have major potential for further study. The need for a more objective approach in visualising the explanations to users was recognised in the project, but it was considered to be out of scope. The goals are further summarised in Table 3.

Goal	Priority	Status
Implement a state-of-the-art recommender system	Primary	Yes
Implement two post-hoc explanation generators	Primary	Yes
Define and evaluate metrics for explanations	Primary	Yes
Improve method for generating association rules explanations	Secondary	No
Use explainability to improve scrutability of the recommender system	Secondary	No
Use explanations in analysing recommender system behaviour	Tertiary	Partial
Define an objective approach in evaluating explanation visualisations.	Tertiary	No

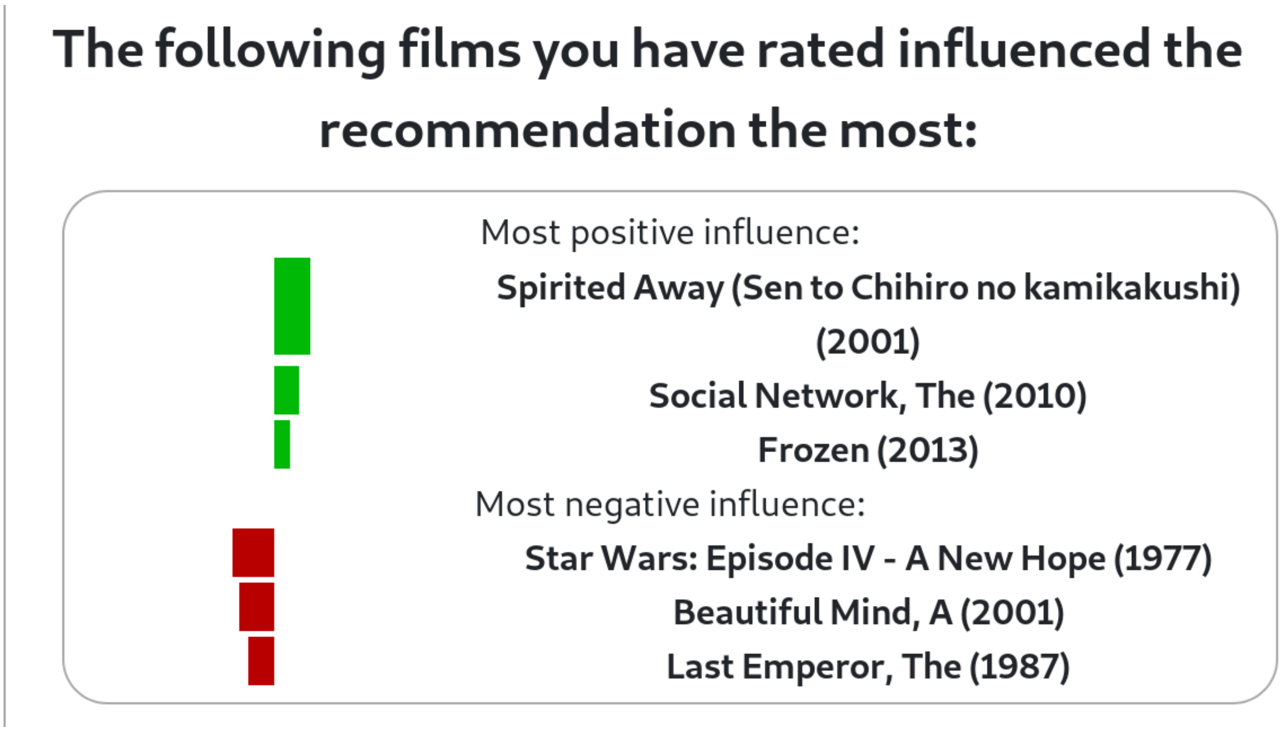
Table 3: *Summary of goals, and whether they were achieved or not. Note that the secondary and tertiary goals were set as alternatives in case the evaluation-step would not require an extensive user study. With the user study, completing them was out of scope.*

User Study

The trustworthiness of a recommender system has been previously linked to two properties: the transparency of the recommendation, and the accuracy of the recommendation algorithm. The interface design of web pages has also be measured to influence user trust.[21] Since users are thought to prefer products they trust, the user study aimed to measure whether adding explanations to recommendations, and as such increasing their transparency, increases the trustworthiness of the system. Persuasiveness was used to simulate how explanation type might affect CTR in real systems, and is an interesting metric since it measures benefit to the system itself (by increased usage), not the user. In the past, persuasiveness in film recommender systems has been measured by simply asking the user how likely they would be to see a film, which was the strategy used here [21].

The user study observed a statistically significant difference in the persuasiveness and trustworthiness of different explanations. The post-hoc tests further showed that the difference was in favour of the association rules explanation. However, there are various possible explanations for the measured effect.

Most obviously, the effect can be because users found the explanations useful. Such effect of adding explainability to recommender systems has been observed before [9][10]. However, there’s no clear reason why the effect would be between association rules and influence, and why no effect between the influence and baseline conditions was observed. One possibility for this is that the users preferred the simpler association rules to the more complex influence explanation, and perhaps didn’t fully understand what the influence explanation meant. Association rules also give a stronger justification for the recommendation: in it the antecedents directly caused the recommendation, while in influence they only influenced it. It’s possible that users preferred this clarity of the association rules explanation. The presentation of the two explanations is shown in Figures 13 and 14, while the presentation of the baseline explainer is shown in Figure 15.



The following films you have rated influenced the recommendation the most:

Most positive influence:

Spirited Away (Sen to Chihiro no kamikakushi) (2001)

Social Network, The (2010)

Frozen (2013)

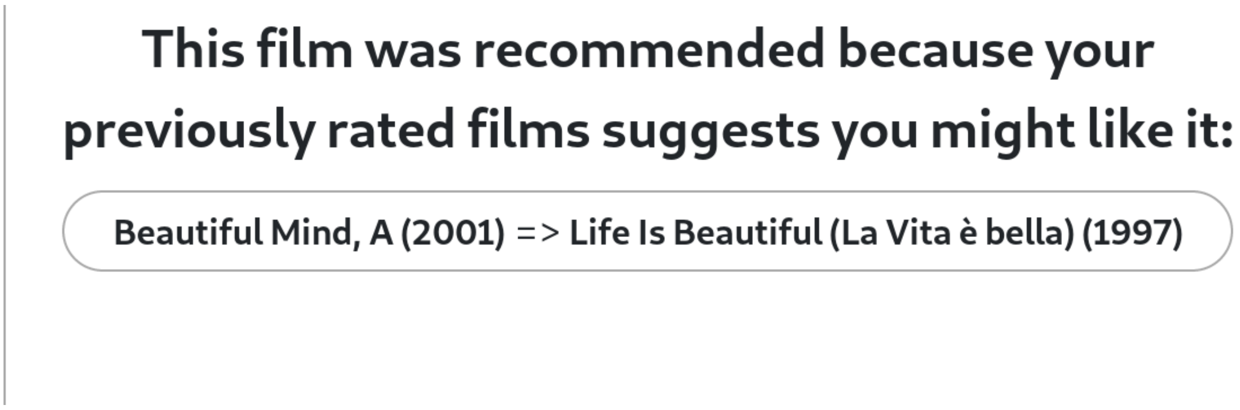
Most negative influence:

Star Wars: Episode IV - A New Hope (1977)

Beautiful Mind, A (2001)

Last Emperor, The (1987)

Figure 13: *How the influence explanation is presented to the user in the web interface. The hue of the bar shows whether the film's influence is positive or negative, and its width shows the strength of the effect.*



This film was recommended because your previously rated films suggests you might like it:

Beautiful Mind, A (2001) => Life Is Beautiful (La Vita è bella) (1997)

Figure 14: *How the association rule explanation is presented to the user in the web interface. The antecedent(s) is shown left-hand side of the rule, and the consequent is shown on the right.*

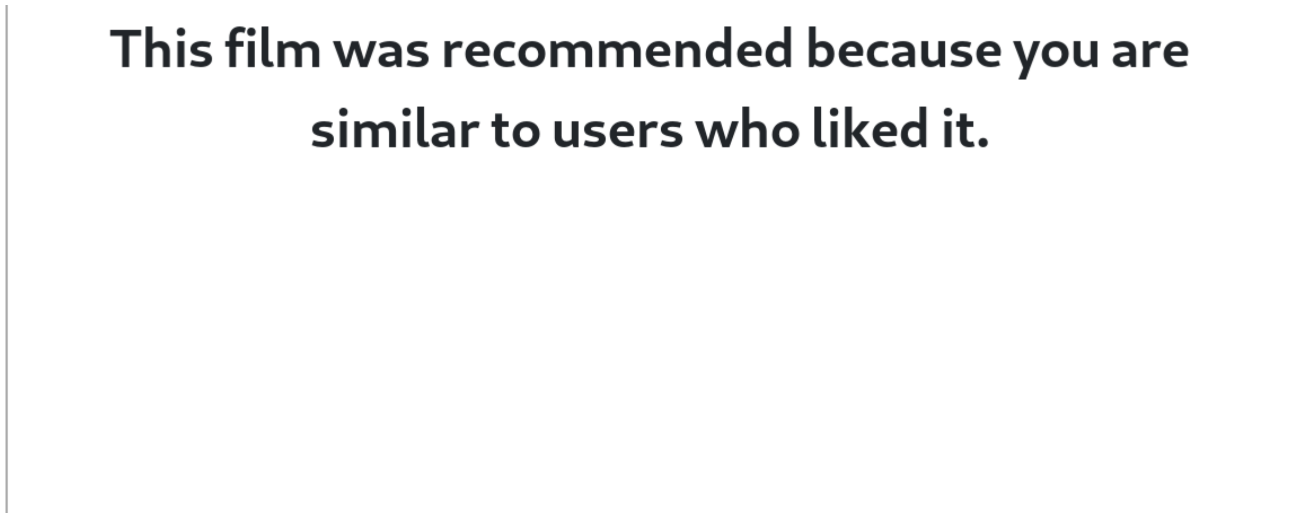

This film was recommended because you are similar to users who liked it.

Figure 15: *How the baseline (i.e. “no explanation”) explanation is presented to the user in the web interface. The explanation gives no personalised insight into why the film was recommended.*

The effect can also arise from the different selection strategy of recommendations between association rules and other conditions. Because the explainable recommendations were generated from global association rules, they tend to only include popular content, and don’t adjust to the user’s individual tastes as well as recommendations in general. However, it is surprising that this would result in an increased effect – in general, users ought to prefer recommendations that are better-targeted to them. It is possible that the “fake” nature of the research platform affects this – the popular but less targeted content may represent films the users knew beforehand and can easily tell that they want to watch them, but would not watch in a real system. Popular films are also more likely to have a poster and description, but this was controlled by filtering out cases without such details from the analysis. Because the films with a poster and description received higher ratings for persuasiveness and trust, it can be concluded that interface design heavily affects these properties as well (a finding that has been described before [21]).

As concluded in the Evaluation-section for influence calculation, the calculated influences suffer from a relatively high variance. This can result in influence values that differ dramatically from the “real” underlying influence of rated films. For example, if an user who had rated a *Star Wars* film highly saw a recommendation of a *Star Wars* film, they’d expect the previous ratings from the franchise to have highly influenced the recommendation. If this influence was understated, the user might lose trust in the recommender system. As such, the inexact nature of the influence calculation may have affected its ratings amongst users.

To conclude, while the study found statistical significance between the explanation methods it’s too early to state that association rules is the better way to generate explanations for recommender systems. In the future, models with a higher model fidelity should be used so association rules explanations can be better compared with other explanation generation methods. Local or cluster methods for mining association rules described in previous work could be used to improve model fidelity [1], though it’s unclear how they would work for real-time systems. Secondly, explanation generation and visualisation should be decoupled as much as possible in further studies. In this study, it’s not clear if the effect is caused by the explanation generation algorithm, or the way the explanations are visualised to the user. Finally, data should be gathered from a larger and more diverse group of participants, preferably in a real system with real users to make sure the results are more representative and generalise better.

In the end, the difference between persuasiveness and trustworthiness should also be highlighted: the measured effect in expressed trust was higher, but only slightly. If persuasiveness is a valid measure for click-through rate (CTR) of real systems, this predicts that adding explanations could have a significant effect on revenue generated by recommender systems. It's also possible that users didn't really consider trustworthiness and persuasiveness as separate metrics, but rather saw them as a single measure of recommendation quality.

Adding explanations to real systems

Association Rules

The association rules explanation was liked by users, and would therefore make a good candidate as an explanation to be added to real systems. As opposed to Peake and Wang (2018), the association rules here allowed more than one antecedent [1]. This was hypothesised to allow for richer association rules to be mined from the dataset, though it's unclear whether this effect was achieved or not. This project also only used global rules – if association rule explanations are to be added to a real system, the developer needs to decide whether the rules used should be global, or the local or cluster-based rules explored by Peake and Wang (2018) [1].

The key advantage for global rules is that the process for mining the rules has to be done once only. The apriori algorithm [26] used in mining association rules is computationally expensive, especially for large datasets, which might cause significant issue in scaling up local- or cluster based approaches. This is most evident in live systems such as the web application used in the user study: you can't expect users to wait several hours for their local association rules calculation to finish! Peake and Wang identified scalable association rules mining as a key area of future study [1], a finding this study affirms.

The biggest problem with using this type of explanation in real systems is model fidelity. As shown by the evaluation step, the measured model fidelity was significantly lower than in the previous study [1]. This can be due to several reasons: firstly, the MovieLens 20M dataset has more items (roughly 27,000) than the Channel 4 dataset (681 in the training set of previous study [1]). Because the association rules are mined from co-located items in a transaction (i.e. a single user's reviews), having fewer items is favourable as any two items are be more likely to be co-located in a transaction. Datasets with large numbers of items aren't doomed to fail though, they just need to set their threshold of support lower, as well as likely set the threshold of confidence higher to avoid including noise. Using local or cluster-based rules could also help, since they were observed to have a higher model fidelity [1].

If the association rules explainer is treated as a machine learning model itself, then the thresholds of support and confidence (as well as other parameters for filtering the set of association rules) can be seen as its hyper-parameters. While the thresholds were set by hand to the researcher's best guess, they could also be determined through hyper-parameter optimisation. This was out of scope for this project, but could be an interesting way to make the association rules step more scientific and allow the method to be more easily applied to more diverse datasets. The cost function for optimising the hyper-parameters would need to be chosen carefully though, perhaps by aiming for the highest possible model fidelity while punishing small thresholds for confidence and support, or by using a version of explainable precision and recall [14].

Influences

While the user study found no statistically significant effect in user-rated trust and persuasiveness for adding the influence-based explanations, they still warrant further study. Most importantly, the method of presenting the explanation to the users can be optimised further. Even if the influence explanations don't improve the trustworthiness and persuasiveness of the model, they do improve transparency. They could also be used for implementing special user interface components: for example, an UI element could show top recommendations amongst those that are most influenced by a specific item. This could be useful for implementing a more personalised version of the classic *"This item is similar to the following items..."* recommender system, which are often item- based neighbourhood models, or content-based models.

The biggest benefit of the influence explanation is its perfect model fidelity, which means that an explanation can always be generated. If an explanation should always be added to a recommendation, it's possible that the two explanation types could be used together: association rules explanations as the primary method, and influences as the fallback. On the other hand, showing that a recommendation can't be explained can be good for transparency, and in some cases for trust as well [21]: by adding a fallback option to association rules, this transparency is lost.

Visualising association rules

Recommender systems based on latent factor models are highly uninterpretable. So far post-hoc explanation methods such as association rules have been used in explaining individual recommendations. However, since the set of association rules encapsulates the recommender system's behaviour as a whole, it could be used to discover knowledge about the recommender system as a whole.

To do that, the recommender system was mapped as a network using the generated association rules. Since the rules are of form (X, Z => Y), they can be considered a directed graph where X, Z and Y are nodes of the graph. This graph can then be visualised using standard graph visualisation techniques. Figure 16 shows a generated graph using the association rules mined from the MovieLens 20m dataset. Due to its large size, a clearer image is provided as Appendix 1. Note that rules of type (X, Y) => Z are shown in two edges, X => Z and Y => Z.

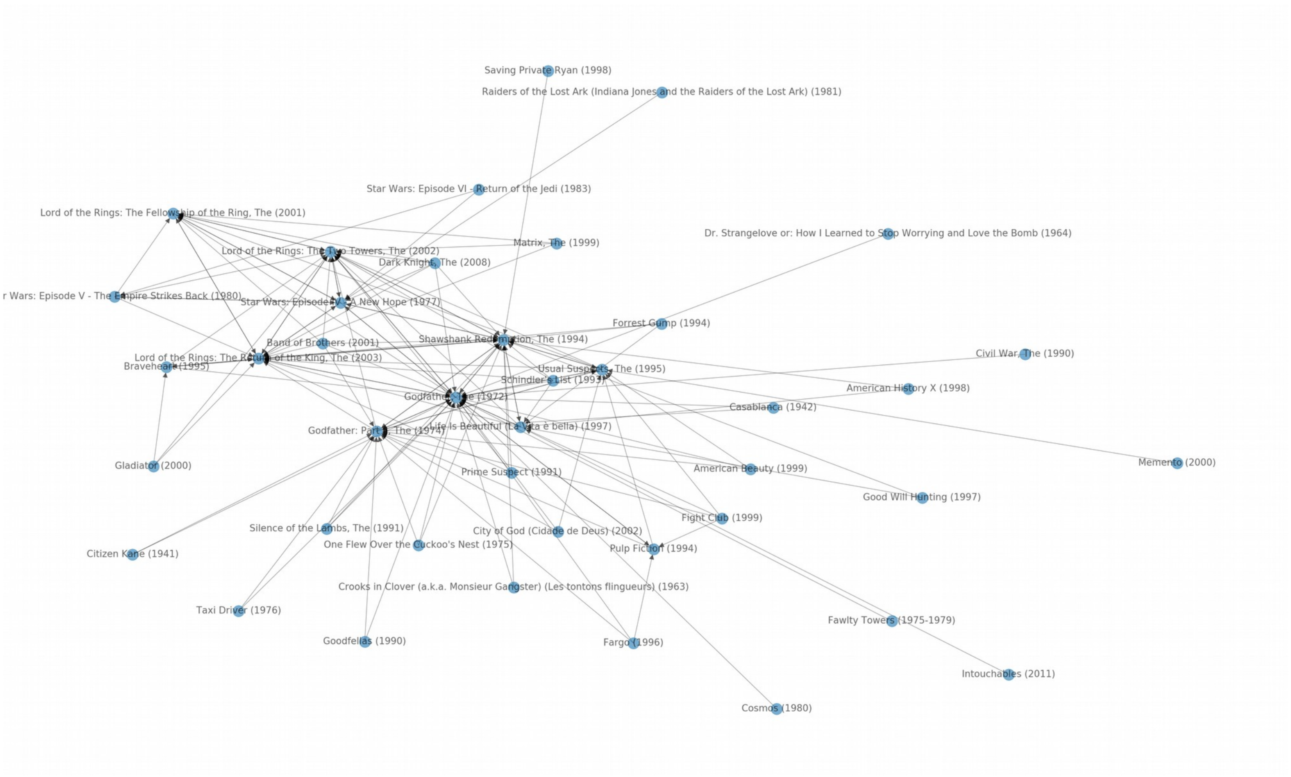

Saving Private Ryan (1998)
Raiders of the Lost Ark (Indiana Jones and the Raiders of the Lost Ark) (1981)
Star Wars: Episode VI - Return of the Jedi (1983)
Lord of the Rings: The Fellowship of the Ring, The (2001)
Dr. Strangelove or: How I Learned to Stop Worrying and Love the Bomb (1964)
Matrix, The (1999)
Lord of the Rings: The Two Towers, The (2002)
Dark Knight, The (2008)
r Wars: Episode V - The Empire Strikes Back (1980)
Star Wars: Episode IV - A New Hope (1977)
Forrest Gump (1994)
Band of Brothers (2001)
Shawshank Redemption, The (1994)
Civil War, The (1990)
Lord of the Rings: The Return of the King, The (2003)
Braveheart (1995)
Usual Suspects, The (1995)
Schindler's List (1993)
American History X (1998)
Godfather, The (1972)
Casablanca (1942)
Godfather: Part II, The (1974)
Life Is Beautiful (La Vita è bella) (1997)
Memento (2000)
Gladiator (2000)
Prime Suspect (1991)
American Beauty (1999)
Good Will Hunting (1997)
Fight Club (1999)
Silence of the Lambs, The (1991)
City of God (Cidade de Deus) (2002)
Citizen Kane (1941)
One Flew Over the Cuckoo's Nest (1975)
Pulp Fiction (1994)
Crooks in Clover (a.k.a. Monsieur Gangster) (Les tontons flingueurs) (1963)
Taxi Driver (1976)
Fawlty Towers (1975-1979)
Goodfellas (1990)
Fargo (1996)
Intouchables (2011)
Cosmos (1980)

Figure 16: *The association rules mined from a recommender system visualised as a directed graph. Each node represents a film, while each edge (arrow) represents a causal relationship in an association rule, i.e. A → B.*

The visualised graph can potentially be used to answer many questions about the underlying recommender system, but the most interesting one is whether the system suffers from the **extreme content problem**, referring to a recommender system that recommends more and more extreme, niche items to users who have rated popular, mild items. This problem is of particular interest to open video platforms like YouTube, which has been accused of promoting more extreme content to viewers of popular, milder “gateway” videos [20]. To analyse the network quantitatively, a proper metric for “extremeness” should be defined. Every item could then be assigned an “extremeness” score, and parts of the network where extremeness increases one moves closer to the centre of a cluster could be identified automatically.

In this case, visual inspection of the network seems to place films such as *The Shawshank Redemption, The Lord of the Rings: The Two Towers and The Godfather* to the centre of the network, which seem to be no more extremist than their antecedents. As such, the network suggests that the recommender system described here does not suffer from the extreme content problem.

Conclusion

This dissertation has shown two alternative approaches in adding explanations to a recommender system, and shown through a study conducted on real users how it can improve the trustworthiness and persuasiveness of the recommendations. This effect was found to be statistically significant (p=0.008, p=0.001) between the explanation types. An investigation on model fidelity highlighted the importance of selecting the right hyper-parameters for the association rule explanation generator, an issue which is considered the biggest roadblock in deploying the explanations to real systems.

The dissertation also proposed an approximation of adding a new user into a latent factor model without a full re-training, and showed that the method doesn’t lower the precision of the resulting top-n recommendations. The approximate training method was found to be thousands of times faster than a full re-train, making rapid addition, update and removal of users feasible in live systems. Finally, the potential use of association rules in analysing the recommender system as a whole was explored through visualising the rules (and by proxy, the recommender system as a whole) as a directed graph.

Limitations

The main limitations of the project concern the user study. Out of the 41 participants, only 32 were generated recommendations with all types of explanations, thus reducing the number of data points available. Furthermore, the users were likely biased to be more European, young and tech-savvy than the average person in the world (though due to the anonymity of the participants, this can’t be confirmed conclusively). In addition, the “fake” nature of the research platform means that the findings, especially those concerning persuasiveness, may not translate to real systems. It’s also possible that the findings are somewhat domain-specific and won’t translate to benefits in domains outside of movie ratings.

In addition, the abysmal model fidelity in the association rules explainer added a selection bias towards popular content when selecting recommendations. This could mean that the observed effect is caused by the selection of recommendations rather than the explanation. Such an effect would be somewhat surprising as the recommender system is supposed to select the best recommendations.

Further study

Based on the results of this report, several areas of further study can be identified. In a future user study, an association rules explainer with a significantly higher model fidelity should be compared against the “baseline” of no explanation in a user study to show how much of the observed effect was due to the explanation, and how much was due to the association rules explainer’s selection bias. It would also be interesting to see whether users would prefer explanations generated from local or cluster-based association rules over global rules. Local and cluster-based rules were reported to have higher model fidelity than global rules [1] and conceptually they focus on describing the parts of the recommender system most relevant to the user. As post-hoc explanations are model-agnostic, the explanations could also be tested with different types of recommender system models, and with different data domains to confirm the assumed robustness of post-hoc explanation methods. Further user studies would be preferred to be conducted in real systems with as many diverse users as possible to make sure the effect translates to the real world.

As mentioned before, an analytical method for setting the optimal bounds for the confidence and support bounds for mining association rules would also be an exciting future area of study. The study would need to find a method of measuring the exact benefits and costs of lowering the bounds: lower bounds result in a greater model fidelity, but also decrease the quality of the average explanation. The use of association rules in data analysis is highly interesting as well. This report identified the extreme content problem as a potential benefactor for mapping the recommender system through association rules, but it is believed that the method could be used in other problems as well. The influence explainer could be of interest in modelling temporal effects in recommender systems – the influence of the most recently rated items can be seen as a derivative of the ratings in the recommender system (i.e. the direction of how the predicted ratings are changing right now) and could potentially be of interest in improving the prediction of future ratings in the system.

References

[1]: Peake, G., & Wang, J. (2018, July). 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 (pp. 2060-2069). ACM.

[2]: Ricci, F., Rokach, L., & Shapira, B. (2011). Introduction to recommender systems handbook. In *Recommender systems handbook* (pp. 1-35). Springer, Boston, MA.

[3]: Schwartz, B. (2004, January). The paradox of choice: Why more is less. New York: Ecco.

[4]: Linden, G., Smith, B., & York, J. (2003). Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing*, (1), 76-80.

[5]: Melville, P., Mooney, R. J., & Nagarajan, R. (2002). Content-boosted collaborative filtering for improved recommendations. *Aaai/iaai*, *23*, 187-192.

[6]: Desrosiers, C., & Karypis, G. (2011). A comprehensive survey of neighborhood-based recommendation methods. In *Recommender systems handbook* (pp. 107-144). Springer, Boston, MA.

[7]: Koren, Y., & Bell, R. (2015). Advances in collaborative filtering. In *Recommender systems handbook* (pp. 77-118). Springer, Boston, MA.

[8]: Gunning, D. (2017). Explainable artificial intelligence (xai). *Defense Advanced Research Projects Agency (DARPA), nd Web*, *2*.

[9]: Sinha, R., Swearingen, K.: The role of transparency in recommender systems. In: Conference on Human Factors in Computing Systems, pp. 830–831 (2002)

[10]: Zhang, Y., Lai, G., Zhang, M., Zhang, Y., Liu, Y., & Ma, S. (2014, July). Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval (pp. 83-92). ACM.

[11]: Zhang, Y., & Chen, X. (2018). Explainable recommendation: A survey and new perspectives. *arXiv preprint arXiv:1804.11192*.

[12]: Cheng, W., Shen, Y., Huang, L., & Zhu, Y. (2019, July). Incorporating Interpretability into Latent Factor Models via Fast Influence Analysis. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (pp. 885-893). ACM.

[13]: Wang, X., Chen, Y., Yang, J., Wu, L., Wu, Z., & Xie, X. (2018, November). A Reinforcement Learning Framework for Explainable Recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)* (pp. 587-596). IEEE.

[14]: Abdollahi, B., & Nasraoui, O. (2017, August). Using explainability for constrained matrix factorization. In *Proceedings of the Eleventh ACM Conference on Recommender Systems* (pp. 79-83). ACM.

[15]: Gopalan, P. K., Charlin, L., & Blei, D. (2014). Content-based recommendations with Poisson factorization. In *Advances in Neural Information Processing Systems* (pp. 3176-3184).

[16]: Makkar, G. (2019, February 16). Robinhood and the overkill of "customers also bought". Retrieved October 2, 2019, from https://uxdesign.cc/robinhood-and-the-overkill-of-customers-also-bought-234266c525f.

[17]: Amazon.com. (2010). Improve Your Recommendations. Retrieved October 2, 2019, from https://www.amazon.com/gp/help/customer/display.html/ref=hp_16465201_FAQ_recommendations?nodeId=13316081.

[18]: Sethuraman, R. (2019, March 31). Why Am I Seeing This? We Have an Answer for You. Retrieved October 2, 2019, from https://newsroom.fb.com/news/2019/03/why-am-i-seeing-this/.

[19]: Moore, M. (2019, August 4). Political ads are all over Facebook. But voters are in the dark about where they come from. Retrieved October 2, 2019, from https://www.theguardian.com/commentisfree/2019/aug/04/facebook-ads-new-polticial-norm-voters-in-dark-where-they-come-from.

[20]: Ribeiro, M. H., Ottoni, R., West, R., Almeida, V. A., & Meira, W. (2019). Auditing Radicalization Pathways on YouTube. *arXiv preprint arXiv:1908.08313*.

[21]: Tintarev, N., & Masthoff, J. (2011). Designing and evaluating explanations for recommender systems. In *Recommender systems handbook* (pp. 479-510). Springer, Boston, MA.

[22]: Schafer, J. B., Frankowski, D., Herlocker, J., & Sen, S. (2007). Collaborative filtering recommender systems. In *The adaptive web* (pp. 291-324). Springer, Berlin, Heidelberg.

[23]: Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, (8), 30-37.

[24]: Harper, F. M., & Konstan, J. A. (2015). The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, *5*(4), 1-19.

[25]: Hug, N. (2017). Surprise, a Python library for recommender systems. *URL: http://surpriselib.com*.

[26]: Agrawal, R., & Srikant, R. (1994, September). Fast algorithms for mining association rules. In *Proc. 20th int. conf. very large data bases, VLDB* (Vol. 1215, pp. 487-499).

[27]: Raschka, S. (2018). MLxtend: Providing machine learning and data science utilities and extensions to Python's scientific computing stack. Journal of open source software, 3(24), 638.

[28]: Rendle, S., Zhang, L., & Koren, Y. (2019). On the difficulty of evaluating baselines: A study on recommender systems. arXiv preprint arXiv:1905.01395.

[29]: React – A JavaScript library for building user interfaces. (n.d.). Retrieved March 26, 2020, from https://reactjs.org/

[30]: Facebook. (2020, March 24). facebook/create-react-app. Retrieved March 26, 2020, from https://github.com/facebook/create-react-app

[31]: Flask. (n.d.). Retrieved March 26, 2020, from https://palletsprojects.com/p/flask/

[32]: Cross-Origin Resource Sharing (CORS). (n.d.). Retrieved March 26, 2020, from https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS

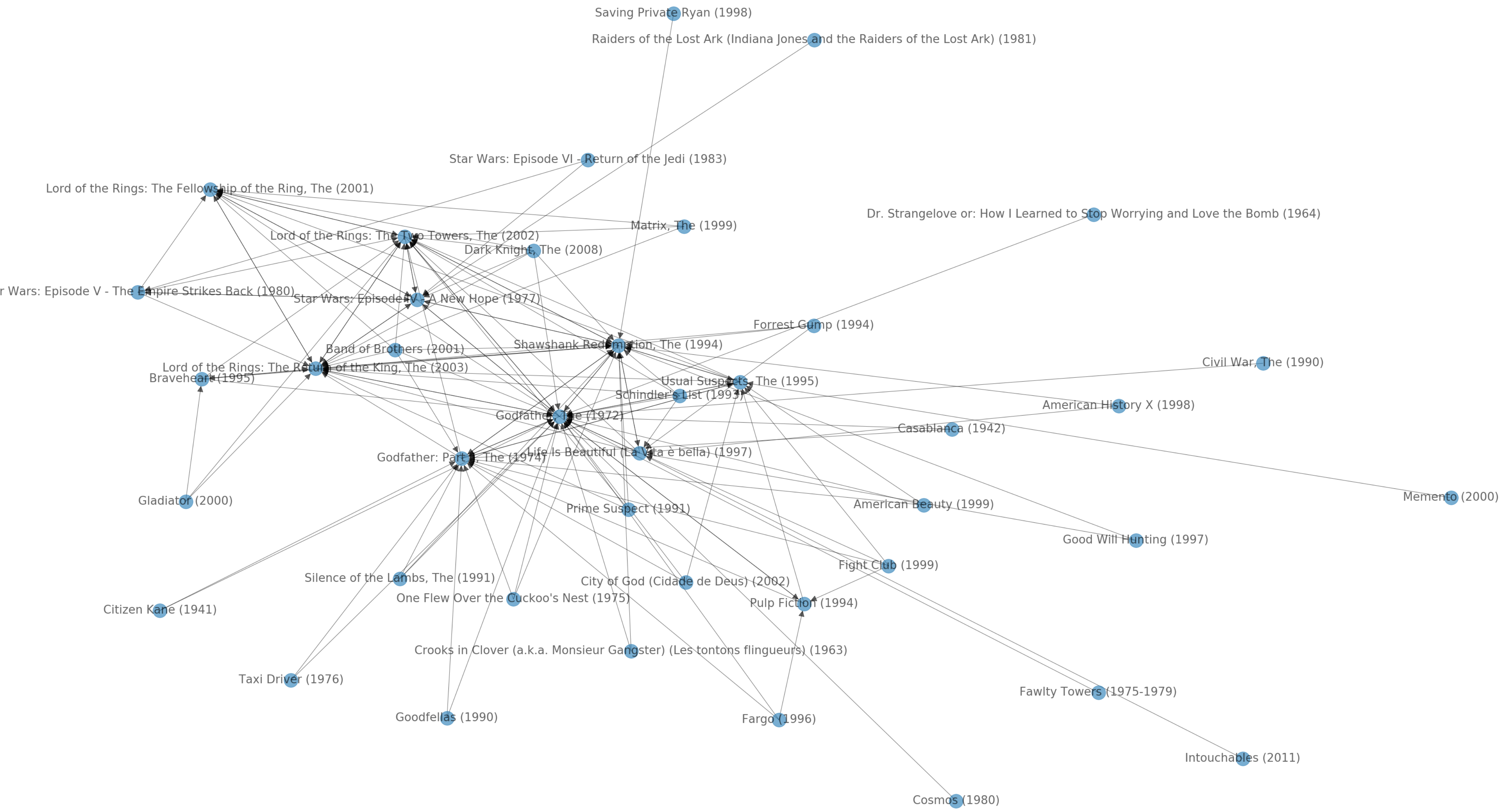
[33]: Open Source Initiative. (2006). The MIT license. 2015b.[Online]. Available: https://opensource. org/licenses/MIT.[Accessed 27 March 2017].

[34]: Candès, E. J., & Recht, B. (2009). Exact matrix completion via convex optimization. *Foundations of Computational mathematics*, *9*(6), 717.

[35]: Wang, H., Wang, N., & Yeung, D. Y. (2015, August). Collaborative deep learning for recommender systems. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 1235-1244).

[36]: The Movie DB. (n.d.). The Movie DB API. Retrieved April 16, 2020, from https://www.themoviedb.org/documentation/api?language=en-US

Appendix list


Saving Private Ryan (1998)
Raiders of the Lost Ark (Indiana Jones and the Raiders of the Lost Ark) (1981)
Star Wars: Episode VI - Return of the Jedi (1983)
Lord of the Rings: The Fellowship of the Ring, The (2001)
Dr. Strangelove or: How I Learned to Stop Worrying and Love the Bomb (1964)
Matrix, The (1999)
Lord of the Rings: The Two Towers, The (2002)
Dark Knight, The (2008)
r Wars: Episode V - The Empire Strikes Back (1980)
Star Wars: Episode IV - A New Hope (1977)
Forrest Gump (1994)
Shawshank Redemption, The (1994)
Band of Brothers (2001)
Civil War, The (1990)
Lord of the Rings: The Return of the King, The (2003)
Braveheart (1995)
Usual Suspects, The (1995)
Schindler's List (1993)
American History X (1998)
Godfather, The (1972)
Casablanca (1942)
Life Is Beautiful (La Vita è bella) (1997)
Godfather: Part II, The (1974)
Memento (2000)
Gladiator (2000)
Prime Suspect (1991)
American Beauty (1999)
Good Will Hunting (1997)
Fight Club (1999)
Silence of the Lambs, The (1991)
City of God (Cidade de Deus) (2002)
One Flew Over the Cuckoo's Nest (1975)
Pulp Fiction (1994)
Citizen Kane (1941)
Crooks in Clover (a.k.a. Monsieur Gangster) (Les tontons flingueurs) (1963)
Taxi Driver (1976)
Fawlty Towers (1975-1979)
Goodfellas (1990)
Fargo (1996)
Intouchables (2011)
Cosmos (1980)

University of St Andrews | FOUNDED 1413

University Teaching and Research Ethics Committee

16 April 2020

Dear Ville,

Thank you for submitting your ethical application, which was considered by the School of Computer Science Ethics Committee on Wednesday 16th October, where the following documents were reviewed:

1. Ethical Application Form

The School of Computer Science Ethics Committee has been delegated to act on behalf of the University Teaching and Research Ethics Committee (UTREC) and has granted this application ethical approval. The particulars relating to the approved project are as follows -

Approval Code:	CS14599	**Approved on:**	06.11.19	**Approval Expiry:**	06.11.2024
Project Title:	Improving Explainability in User-facing Recommendations System				
Researcher(s):	Ville Kuosmanen				
Supervisor(s):	David Harris-Birtill				

Approval is awarded for five years. Projects which have not commenced within two years of approval must be re-submitted for review by your School Ethics Committee. If you are unable to complete your research within the five year approval period, you are required to write to your School Ethics Committee Convener to request a discretionary extension of no greater than 6 months or to re-apply if directed to do so, and you should inform your School Ethics Committee when your project reaches completion.

If you make any changes to the project outlined in your approved ethical application form, you should inform your supervisor and seek advice on the ethical implications of those changes from the School Ethics Convener who may advise you to complete and submit an ethical amendment form for review.

Any adverse incident which occurs during the course of conducting your research must be reported immediately to the School Ethics Committee who will advise you on the appropriate action to be taken.

Approval is given on the understanding that you conduct your research as outlined in your application and in compliance with UTREC Guidelines and Policies (http://www.st-andrews.ac.uk/utrec/guidelinespolicies/). You are also advised to ensure that you procure and handle your research data within the provisions of the Data Provision Act 1998 and in accordance with any conditions of funding incumbent upon you.

Yours sincerely

Wendy Boyter

School Ethics Committee Administrator

University of St Andrews | FOUNDED 1413

University Teaching and Research Ethics Committee

16 April 2020

Dear Ville,

Thank you for submitting your ethical application, which was considered by the School of Computer Science Ethics Committee on Wednesday 11th December, where the following documents were reviewed:

1. Ethical Application Form
2. Participant Information Sheet
3. Participant Consent Form
4. Participant Debrief Form
5. Advertisement

The School of Computer Science Ethics Committee has been delegated to act on behalf of the University Teaching and Research Ethics Committee (UTREC) and has granted this application ethical approval. The particulars relating to the approved project are as follows -

Approval Code:	CS14723	**Approved on:**	05.02.20	**Approval Expiry:**	05.02.2025
Project Title:	Improving Explainability in User-facing Recommendation Systems				
Researcher(s):	Ville Kuosmanen				
Supervisor(s):	David Harris-Birtill				

Approval is awarded for five years. Projects which have not commenced within two years of approval must be re-submitted for review by your School Ethics Committee. If you are unable to complete your research within the five year approval period, you are required to write to your School Ethics Committee Convener to request a discretionary extension of no greater than 6 months or to re-apply if directed to do so, and you should inform your School Ethics Committee when your project reaches completion.

If you make any changes to the project outlined in your approved ethical application form, you should inform your supervisor and seek advice on the ethical implications of those changes from the School Ethics Convener who may advise you to complete and submit an ethical amendment form for review.

Any adverse incident which occurs during the course of conducting your research must be reported immediately to the School Ethics Committee who will advise you on the appropriate action to be taken.

Approval is given on the understanding that you conduct your research as outlined in your application and in compliance with UTREC Guidelines and Policies (http://www.st-andrews.ac.uk/utrec/guidelinespolicies/). You are also advised to ensure that you procure and handle your research data within the provisions of the Data Provision Act 1998 and in accordance with any conditions of funding incumbent upon you.

Yours sincerely

Wendy Boyter

School Ethics Committee Administrator