Transparent, Scrutable and Explainable User Models for Personalized Recommendation

Krisztian Balog
Google
London, UK
krisztianb@google.com

Filip Radlinski
Google
London, UK
filiprad@google.com

Shushan Arakelyan*
USC Information Sciences Institute
Marina Del Rey, CA, USA
shushan@isi.edu

ABSTRACT

Most recommender systems base their recommendations on implicit or explicit item-level feedback provided by users. These item ratings are combined into a complex user model, which then predicts the suitability of other items. While effective, such methods have limited scrutability and transparency. For instance, if a user's interests change, then many item ratings would usually need to be modified to significantly shift the user's recommendations. Similarly, explaining how the system characterizes the user is impossible, short of presenting the entire list of known item ratings. In this paper, we present a new set-based recommendation technique that permits the user model to be explicitly presented to users in natural language, empowering users to understand recommendations made and improve the recommendations dynamically. While performing comparably to traditional collaborative filtering techniques in a standard static setting, our approach allows users to efficiently improve recommendations. Further, it makes it easier for the model to be validated and adjusted, building user trust and understanding.

CCS CONCEPTS

• **Information systems** → **Recommender systems**.

KEYWORDS

Recommendations, explainability, transparency, scrutability

ACM Reference Format:
Krisztian Balog, Filip Radlinski, and Shushan Arakelyan. 2019. Transparent, Scrutable and Explainable User Models for Personalized Recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '19), July 21–25, 2019, Paris, France.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3331184.3331211

1 INTRODUCTION

The importance of explainable AI has been recognized in recent years [33]. Recommender systems represent an important branch of AI research and the *explainability* of recommendations has attracted considerable attention [49, 54]. Generally, explanations "seek to show how a recommended item relates to a user's preferences" [51]. Explanations can serve a multiplicity of aims, including *transparency* (helping users to understand how the system works), *justification* (explaining individual recommendations), and *scrutability* (allowing users to tell the system if it is wrong) [49].

- You like movies that are tagged as 'action', especially those that are tagged as 'violent', such as Aliens.
- You like movies that are tagged as 'twist ending', such as A Pure Formality.
- You don't like movies that are tagged as 'adventure', unless they are tagged as 'thriller', such as Twister.
- You like movies that are tagged as 'cheesy', such as Who Framed Roger Rabbit?
- You like movies that are tagged as 'australia', such as Crocodile Dundee II.

Figure 1: Example summary of a user's preferences.

There is an important distinction between transparency and justifications [51]. The former should give an honest account of how recommendations are selected, while the latter merely gives a plausible description that might be decoupled from the recommendation algorithm. Often, the underlying algorithm is too complex to be described in a human-interpretable manner (e.g., ensemble and deep learning models) or may involve technology that the system provider wishes to protect. Contemporary recommender systems therefore often opt for providing justifications [54], rather than offering genuine transparency.

Scrutability is also lacking in most recommender systems. There is usually little recourse to tell a system if it incorrectly inferred preferences, as the only instruments at the user's disposal are usually removing items from the history and modifying ratings made in the past. It is especially cumbersome to exclude an entire set of items (such as a specific genre in movie recommendations) when, for example, a user's interests shift.

Thus transparency and scrutability are closely tied together, yet one does not imply the other. So far there have only been preliminary attempts at making explanations both transparent and scrutable [19]. With this paper, we aim to fill that gap. Our main research question is the following: How much recommendation accuracy would one need to sacrifice by making a recommender system both transparent and scrutable? We address this question by developing a recommendation approach that satisfies the following desiderata: (1) reveal to users how their preferences are generally understood (explainability); (2) faithfully represent and expose the reasoning behind the recommendation mechanism (transparency); (3) provide users with a direct and meaningful way to revise their model (scrutability).

A fundamental difference between this work and prior research is that we take explainability to the level of user preferences, as opposed to that of item recommendations. That is, instead of explaining the user why a given *item* was recommended, we present an approach to provide a textual description that summarizes the system's understanding of the *user's preferences*. We allow the user to scrutinize this summary and thereby directly modify his or her user model, as illustrated in Figure 1.

*Work was performed while the author was at Google.

SIGIR '19, July 21–25, 2019, Paris, France

ACM ISBN 978-1-4503-6172-9/19/07.
https://doi.org/10.1145/3331184.3331211

Our proposed approach hinges on the notion of *set-based preferences*. People often reason about categories and groups of items when reasoning about possible recommendations to make [3, 14]. Since attaching labels to clusters of items is an inherently difficult problem (e.g., [7]), we instead reverse the process and use tags to define sets. Thus, following related research [17, 42, 51], a basic assumption made in this paper is that user preferences as well as items can be characterized by a set of tags or keywords. These tags may be provided by users (social tagging) or extracted automatically. Given explicit ratings of specific items, which is the most common way of eliciting preferences today, we infer set-based preferences by aggregating over items that are associated with a given tag. We also present a novel pairwise tag-interaction approach that leads to much more semantically rich tag-based preferences being modeled.

This set-based user preference model enables us to generate item recommendations in a transparent manner. For explaining preferences, we opt for sentence-level textual explanations as this provides scrutability, by letting users provide feedback on individual sentences. Any change to the user's preferences has an immediate impact, thereby endowing users with more direct control over the recommendations they receive.

In summary, we make the following contributions: (1) we present an efficient method for inferring set-based user preferences from ratings given to individual items based on the tags associated with those items; (2) we develop a simple, effective, and computationally efficient recommendation model that operates on item tags and set-based user preferences; (3) we propose a simple algorithm for generating natural language explanations of user preferences; (4) we show the value of such a transparent and scrutable model.

2 RELATED WORK

Explainable recommendations refers to personalized recommendation algorithms that "not only provide the user with recommendations, but also make the user aware why such items are recommended" [54]. In addition to improving user acceptance of recommendations (persuasiveness), explanations can serve a multiplicity of aims, such as inspiring the user's confidence in the system (trust), helping users make good decisions (effectiveness) as well as make decisions faster (efficiency) and increasing the ease of use of a system (satisfaction) [49]. Various forms of explanations have been explored in prior work, including sentences [21, 55], tag/keyword clouds [17, 30, 53], as well as different kinds of visualizations [10, 22, 31, 47]. Gedikli et al. [17] evaluate different explanation types and propose a set of guidelines for designing and selecting suitable explanations for recommender systems.

Generation Approaches. Another way to classify explainable recommendation research is by the model used for generating explanations [54]. They can broadly be categorized into content-based, collaborative filtering, and hybrid approaches. *Content-based* methods attempt to match items to users based on features/attributes, such as genre or director in the movies domain. These approaches naturally lend themselves to intuitive explanations by listing content features/attributes that made an item appear in the recommendations [13]. Instead of relying on content information, *collaborative filtering* techniques leverage the "wisdom of the crowds" and make recommendations based on patterns of ratings or usage [24]. Two families of methods may be distinguished: neighborhood-based and model-based. *Neighborhood-based* methods estimate ratings by those made by like-minded users (user-based) or known ratings made by the user on similar items (item-based). Of the two, item-based methods often have better scalability and improved accuracy [40], and are also more amenable to explaining the reasons behind the recommendations, as "users are familiar with items previously preferred by them, but do not know those allegedly like-minded users" [24]. *Model-based* methods capture salient characteristics of users and items by parameters learned from training data. Perhaps the most prominent are latent factor models based on matrix factorization [25], where user-item interactions are modeled as inner products in a lower dimensional space. Although collaborative filtering methods have achieved significant improvements over content-based methods in terms of accuracy, they are less intuitive to explain [54]. Specifically, the challenge is that the meaning of each latent dimension is unknown. Zhang et al. [55] propose to alleviate this problem by aligning each latent dimension with a particular explicit feature extracted from textual user reviews. Note that this work, along with many others [9, 11, 27, 30, 43, 52], generates explanations with the help of reviews written for items, which may not always be available. Others consider neighborhood-style explanations for matrix factorization ("users who are the most similar to you also liked the recommended item") and incorporate an "explainability regularizer" into the objective function [1]. Neural models, which have recently attracted attention for explainable recommendations, also fall under the category of model-based approaches [27, 43]. These leverage attention weights over words in user reviews to indicate which parts are relevant for the recommendation that was made. The recommendation model, however, is still a black box and "the explainability of the deep model itself also needs further exploration" [54].

It is important to note that "the underlying algorithm of a recommender engine will to a certain degree influence the types of explanations that can be generated" [49]. Therefore, explainability is a main design decision for us. Our model would be classified as content-based, as user preferences are characterized as a set of model parameters which are inferred from item ratings and then used to predict how much the user would like an unseen item.

Transparency provides insights into how the recommendation process works, and is closely related to explainability. Indeed, one of the aims that explanations can serve is to provide transparency [49]. A crucial difference is that transparency "should give an honest account of how the recommendations are selected and how the system works" [49], while justification merely provides a plausible reason that may be decoupled from the recommendation algorithm [51]. Explanations can also help to make a system *scrutable*, that is, allow users to correct the system's reasoning or modify preferences in the user model [36]. A preliminary attempt at transparent and scrutable explanations is presented in [19]. There, the authors consider a *similar item search* scenario and provide explanations in the form of overlapping and difference tag clouds between a seed item and a recommended item. Users can then steer the recommendations by manipulating the tag clouds.

The utilization of tags for explainable recommendations has been particularly well studied in the movie domain [17, 51]. Vig et al. [51] use tags for explaining movie recommendations and evaluate different interface designs that show how user preferences relate

Table 1: Notation. (All ratings are w.r.t. a given target user.)

$\mathbf{C}$	Set of candidate statements	i	Item ($i \in \mathbf{I}$)
$\mathbf{I}$	Set of all items	$\hat{r}_t$	Inferred rating of t
$\mathbf{I}^*$	Items rated by the user	r_μ	Neutral rating threshold
$\mathbf{I}_t$	Items labeled with t	r_i	Rating of item i
$\mathbf{I}_t^*$	Rated items labeled with t	t	Tag ($t \in \mathbf{T}$)
$\mathbf{R}_t$	Set of all ratings from items from $\mathbf{I}_t^*$		
$\mathbf{S}_k$	Top-k user preference statements selected		
$\mathbf{T}$	Set of all tags		
$\mathbf{T}^{+/-}$	Set of user tags liked/disliked by the user		
$\mathbf{W}$	User model (weighted set of tags)		

to items. One key observation was that displaying *tag preference* (the user's sentiment towards a tag) is more important than *tag relevance* (the weight of a tag for a given movie). They explain this as follows: "Users may prefer seeing tag preference because they are skeptical that a recommender system can accurately infer their preferences." Gedikli et al. [17] find content-based tag cloud explanations effective and particularly well accepted by users. We also use tags to represent user preferences, but unlike existing work, we strive for natural language explanations instead of word clouds.

Set-Based Preferences. Many papers consider the case where preferences over sets are given, and used to infer preferences for individual items [3, 4, 7, 14, 44]. We investigate the reverse process, starting from item preferences and infering sets. Of the above, [7] is most relevant. Users are presented with sets of items along with some tags (e.g. "based on a comic, dark hero, superhero"), and asked to indicate which sets match their interests. The authors show that users are able to complete the preference elicitation process more rapidly than with traditional item ratings. Other studies also suggest that it may be easier for users to express preferences implicitly in sets rather than item by item [3, 14]. Sharma et al. [44] study how users' ratings on sets of items relate to their ratings on the constituent items. They find that for most users the rating on a set can be accurately approximated by the average rating of the items in that set. There is, however, a considerable user population that tends to over- or under-estimate set-level ratings, especially for sets that contain items with diverse ratings.

3 MODELING USER PREFERENCES

This section describes our user model, which is based on set-based preferences. It is a core enabling component for providing transparent item recommendations and for generating scrutable textual explanations of user preferences.

For the ease of presentation, all our examples are from the domain of our evaluation dataset, movies, as this has been a fertile area of recommendation systems research both from an algorithmic and from an explainability perspective [6, 8, 17, 22, 48, 51]. However, there is nothing domain specific in our approach.

3.1 A Case for Set-based Preferences

Related work has shown that set-based preference elicitation can lead to an accurate model of user preferences and, consequently, to high-quality item recommendations [7]. However, generating descriptions for sets is notoriously hard. Therefore, instead of first clustering items and then naming clusters (as per [7]), we capture sets in terms of social tags, and generate explanations based on those sets. While we will be using tags for capturing sets, we note that sets could also be defined in alternative ways, e.g., using item attributes (movie genres, directors, etc.). We further acknowledge that there are known issues regarding the quality [41] and redundancy [18] of social tags. Some have suggested that certain types of tags are more suitable than others for generating explanations [51]. The quality of item-tag assignments can thus have a significant impact on recommendation accuracy as well as on explanation quality. However, we regard this as a data quality issue, and beyond the scope of this paper; it does not affect our modeling in any way.

3.2 Inferring Set-level Preferences

Let us assume that $\mathbf{I}$ is the set of all known items where each item i represents a single movie, TV series, TV show, or similar. Let us also assume that we have a set of items $\mathbf{I}^*$ that are rated by a given target user u and $r_i \in [-1, 1]$ is the (normalized) rating of each item $i \in \mathbf{I}^*$. Rating values lower than 0 correspond to disliking the item, while values higher than 0 show liking. We also have a number of tags t, each of which represents a semantic set like "comedy film" or "movies directed by Christopher Nolan." We refer to Table 1 for the notation used throughout the paper. Note that we will be referring to ratings and preferences of a single user, therefore the subscript u is generally omitted for the ease of notation.

We infer set-level preferences from ratings given to individual items, taking the mean rating of items labeled with a given tag to be the tag's average appeal to the user:

$$\hat{r}_t = \frac{1}{|\mathbf{I}_t^*|} \sum_{i \in \mathbf{I}_t^*} r_i \,, \tag{1}$$

where $\mathbf{I}_t^*$ is the set of items rated by the user and tagged with t. Note that we take the relationship between items and tags (referred to as *tag relevance* in [51]) to be binary (as opposed to a value on a continuous scale). It means that we are assuming that all tags associated with an item describe that item equally well. The binary approach is motivated both by its conceptual simplicity and by the fact that weighted item-tag assignments are unavailable in the public dataset we are working with. A tag-weighted variant of Eq. (1) would be a straightforward extension in future work.

3.3 Modeling Pairwise Set Interactions

Prior work modeling user interests with tags has focused on tags that succinctly characterize a single set of items, e.g., "science-fiction movies" or "movies starring Tom Cruise." However, reasoning about high-level preferences through single tags produces preferences that are rarely rich enough to capture realistic interests particularly well. For instance, consider the above example of science fiction movies. A wide variety of science fiction movies exist, and a user's preferences are rarely black and white for such a large class of items. It may well be that the user generally dislikes science fiction movies, while some may still appeal to her. For this reason, one of our key contributions is representing user preferences by *interactions* between pairs of tags.

To explain this concept further, consider Figure 2. Here, the user does not like science fiction movies in general, but does like science fiction movies that are about space exploration. Our approach allows describing this situation as "You don't like science fiction movies unless they are about space exploration." Note, that in this

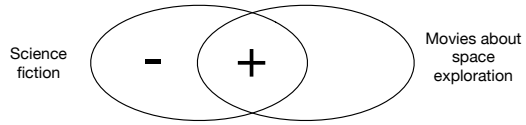

Figure 2: Here, the user does not like science fiction movies in general, but does like those about space exploration.

example it is not necessarily true that the user would also like documentaries or horror movies featuring space exploration. Note also that multi-set interactions could be modeled in a similar way. However, our goal is to produce scrutable models, and we hypothesize that pairwise interactions capture sufficient richness while still providing a unit that can be naturally scrutinized by a user. For instance, it would be natural to ask someone "What sort of science fiction movies do you like?", but not usually natural to ask "What sort of science fiction movies about space exploration do you like?".

We formalize and consider different interactions between sets, which are presented in Figure 3. Each pairwise interaction can also be translated into a simple sentence that captures the meaning of the interaction. While language is naturally ambiguous, we provide a specific meaning to the base interactions allowed.

3.4 User Model

We define the user model as a weighted set $\mathbf{W}$ of tags. For a given tag t, w_t denotes the preference of the target user for that tag, such that the absolute value expresses the strength of the preference, while the sign indicates the direction of preference (i.e., like/dislike).

The preferences for the individual values can then naturally be derived from the inferred tag ratings:

$$w_t = \hat{r}_t - r_\mu \,, \tag{2}$$

where r_μ corresponds to the neutral rating. This may be personalized by using a user-specific value, however, we use the objective neutral rating, i.e., a value of 0, for all users.[1] Given a tagged item corpus, pairwise tag interactions are encoded by introducing any required pairwise tags, as pseudo tags, allowing a weight to be inferred where pairwise interactions exist. In particular, for a pair of tags t_a, t_b encoding "You [don't] like t_a *connective* t_b," and noting that whenever both tags apply then t_a must also apply, we model:

$$w_{t_a,t_b} = \hat{r}_{t_a,t_b} - w_{t_a}. \tag{3}$$

Recall that our approach is designed with scrutability in mind. That is, the user can explicitly state a positive or negative preference for a given tag. In practical terms this simply means overwriting the estimated w_t for that tag with a specific value, such as 1 or -1, or excluding preference on a tag by removing it from $\mathbf{W}$.

4 IDENTIFYING USER PREFERENCES

This section introduces our approach to selecting which tags and pairwise tags, referred to as *statements* hereinafter, should be included in a user model. The algorithm works as follows. A set of candidate statements $\mathbf{C}$ is generated. These are ranked based on utility to the user model, and the set $\mathbf{S}$ of selected statements is included in the user model. Before detailing each of these steps below, we define desirable properties of the set of selected statements.

[1] It is known that some users tend to either underrate or overrate items [44], therefore, r_μ may also be set to the mean user rating. However, in our experiments, this achieved inferior performance to using an objective neutral rating.

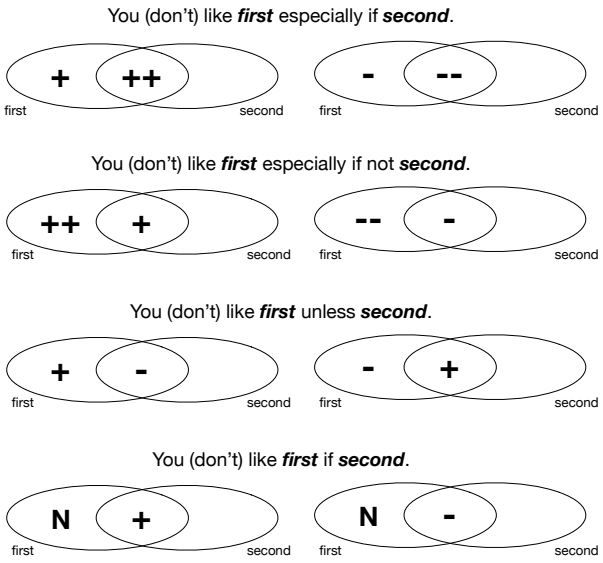

Figure 3: Pairwise set interactions allowed in the user model. *First* is the tag of the left set and *second* is that of the right set. +, - and N indicate positive, negative and neutral average user appeal, with double symbols indicating stronger signals.

4.1 Desirable Properties

We define the following desirable properties for the generated summary: (1) we aim to provide a *correct* description, i.e., one that correctly describes the user and does not contradict ratings provided by the user; (2) we aim to provide a *complete* description, i.e., covering as many preferences from the set of rated items $\mathbf{I}^*$ as possible, and ideally not leaving any significant portion of the user's interests uncovered; (3) we aim for *precision* to avoid generating generic or abstract descriptions, such as "you like TV shows."

4.2 Generating Candidate Statements

A candidate statement $s \in \mathbf{C}$ models either the user's preference about a single tag t or about interaction between a pair of tags (t_a, t_b). We start with the simpler case of a single tag.

Single Tag Statements. Candidate statements are generated for any tag which satisfies a set of conditions parameterized by Θ:

- The tag applies to at least Θ_k items rated by the user. I.e., it is generalizable.
- $\mathbf{R}_t$, the set of ratings for t has a mean that is statistically significantly different from zero (using the Wilcoxon Signed Rank test), with p-value below Θ_p. I.e., it indicates a significant preference.

The utility of statement s_t for a single tag t in the user model is the weight of the tag in the user model, corrected for coverage and significance:

$$U(s_t) = cov(s_t) \cdot sig(s_t) \cdot |w_t| \,, \tag{4}$$

where $cov(s_t)$ represents how many of the item-level user observations are covered by this statement, and $sig(s_t)$ discounts statements over tags that are only weakly statistically significant. Specifically, the coverage is defined as the lesser of the fraction of items rated by the user that do, or do not, contain the tag:

$$cov(s_t) = \min\left(\frac{|\mathbf{I}^*_t|}{|\mathbf{I}^*|}, \frac{|\mathbf{I}^*| - |\mathbf{I}^*_t|}{|\mathbf{I}^*|}\right). \tag{5}$$

The utility of statements for which the statistical strength is less than two standard errors is discounted by defining:

$$sig(s_t) = \min\left(2, |w_t| \Big/ \frac{\sigma_t}{\sqrt{|\mathbf{I}_t^*|}}\right) \tag{6}$$

where σ_t is the estimated variance of the ratings provided in $\mathbf{R}_t$.

Pairwise Tag Statements. For generating pairwise interactions, these requirements apply to the first tag, as well as to the second tag in the context of the first:

$$U(s_{t_a,t_b}) = U(t_a) + cov(s_{t_a,t_b}) \cdot sig(s_{t_a,t_b}) \cdot |w_{t_a,t_b}|, \tag{7}$$

Specifically, for a pairwise interaction we redefine the coverage as:

$$cov(s_{t_a,t_b}) = \min\left(\frac{|\mathbf{I}_{t_a}^* \cap \mathbf{I}_{t_b}^*|}{|\mathbf{I}_{t_a}^*|}, \frac{|\mathbf{I}_{t_a}^*| - |\mathbf{I}_{t_a}^* \cap \mathbf{I}_{t_b}^*|}{|\mathbf{I}_{t_a}^*|}\right). \tag{8}$$

Significance and weight are computed as before, but over $\mathbf{I}_{t_a}^* \cap \mathbf{I}_{t_b}^*$.

4.3 Selecting Statements

While statements can be generated in this way for all tags attached to items in the corpus, as well as for all pairs of tags that co-occur on a sufficient number of items, this would yield an inscrutable user model. Specifically, we started with the observation that it is often impractical for a user to review or update all items they have provided feedback about to validate a model. In this section we show how we select down from all the possible tags and pairwise tags to a smaller set that can be presented to a user.

Recall that $\mathbf{C}$ is the set of all single tag and pairwise candidate statements for a given user. Inspired by the MMR algorithm [5], we perform a greedy selection over $\mathbf{C}$ to obtain a subset $\mathbf{S}_k$ that captures a user's top-k preferences. In each round, we select a statement to add to $\mathbf{S}_k$ by picking the one with highest incremental utility over those statements already selected. Thus, we select:

$$s^* = \arg\max_{s \in \mathbf{C}} U(s|\mathbf{S}_k) \tag{9}$$

$$\text{where} \quad U(s|\mathbf{S}_k) = cov(s|\mathbf{S}_k) \cdot sig(s) \cdot |w_s|, \tag{10}$$

$$cov(s|\mathbf{S}_k) = \min\left(\frac{|\mathbf{I}_s^*/\mathbf{I}_{\mathbf{S}_k}|}{|\mathbf{I}^*|}, \frac{|\mathbf{I}^*| - |\mathbf{I}_s^*/\mathbf{I}_{\mathbf{S}_k}|}{|\mathbf{I}^*|}\right), \tag{11}$$

defining $\mathbf{I}_{\mathbf{S}_k}$ as the set of items in $\mathbf{I}^*$ that influence one of the existing statements in $\mathbf{S}_k$. Below, in Section 7.2, we will evaluate how the value of k impacts recommendation quality.

4.4 Generating Textual Representations

Using the templates presented in Figure 3, the entire model can be presented to the user in natural language. We also note that the intensity of the user's preferences can be accentuated using terms such as "like," "love," "hate," "don't like," etc. In our experiments, we only included two grades, namely "like" and "don't like."

Finally, when generating sentences to present to the user, we additionally select a representative example from the specific items the user has rated for each statement. Our preliminary experiments showed that this disambiguates the meaning of the tag(s) to the user, and grounds it in specific movies that the user knows, thus improving the user's understanding. An example of a single-tag statement would thus be "You don't like science fiction movies, such as The Day After Tomorrow."

5 GENERATING ITEM RECOMMENDATIONS

This section introduces our set-based model for generating item recommendations. It may be classified as a content-based approach, which scores individual items by matching the tags assigned to them against the tag preferences of the user. It is by design a transparent and explainable process that can directly incorporate user feedback, without complex interactions.

5.1 Set-based Model

Let L be a binary random variable that indicates whether item i is liked (ℓ^+) or disliked (ℓ^-). Following the probability ranking principle in IR [39], items should be ranked according to the probability $P(L = \ell^+|u, i)$, which is equivalent to ranking items based on the odds ratio:

$$O(L = \ell^+|u, i) = \frac{P(L = \ell^+|u, i)}{P(L = \ell^-|u, i)}. \tag{12}$$

Applying Bayes' rule and then decomposing the joint probability $P(u, i|L)$, we get:

$$O(L = \ell^+|u, i) \overset{\text{rank}}{=} \frac{P(u, i|L = \ell^+)}{P(u, i|L = \ell^-)} = \frac{P(u|i, L = \ell^+)}{P(u|i, L = \ell^-)} \times \frac{P(L = \ell^+|i)}{P(L = \ell^-|i)}. \tag{13}$$

There are two main components in this model. The *user likelihood*, $P(u|i, L)$ is, intuitively, the probability that user u likes/dislikes item i. The term $P(L|i)$ can be interpreted as the prior probability of item i being liked/disliked (by any user). It is worth pointing out the similarity between this model and *negative query generation* for document retrieval [28]. Here, the user is the query that is being generated by the document (item). However, the estimation of the model's components, which we describe below, is very different.

5.2 User Likelihood

We start by discussing the estimation of $P(u, i|L = \ell^+)$ and $P(u, i|L = \ell^-)$. To reduce the number of equations, we shall use the symbol $\circ$ to denote either + or −. Assuming that tags for an item are sampled identically and independently, the user likelihood is estimated as:

$$P(u|i, L = \ell^\circ) = \prod_{t \in \mathbf{T}_u^\circ} P(t|\theta_i)^{w_{t,u}^\circ}, \tag{14}$$

where $P(t|\theta_i)$ is the probability of tag t given the model of item i, $\mathbf{T}_u^\circ$ is the set of tags liked/disliked the user, and $w_{t,u}^\circ$ are the corresponding (positive/negative) user-tag weights. We detail the estimation of these components below.

Modeling Items. Notice that Eq. (14) resembles the query likelihood formula in ad hoc document retrieval. There is, however, an important difference. The probability of a given tag in the item's model, $P(t|\theta_i)$, should only depend on the presence/absence of that tag. It should not be influenced by what other tags are associated with the item. (Otherwise, the number of tags that are associated with an item would have an undesired influence on the ranking.) Therefore, we model each item as a single sample from a multiple-Bernoulli distribution, where each binary trial corresponds to the event that some tag is associated with the item or not.

Let each item i be represented by a vector $\vec{w}_i \in \{0, 1\}^{|\mathbf{T}|}$, where $w_{t,i} = 1$ iff tag t is assigned to i. From this single sample, we wish to estimate a smoothed tag model θ_i for the item. We assume a prior over the model, specifically a multiple-Beta distribution, which is

the conjugate prior for the multiple-Bernoulli distribution. The probability of a tag given the item's model can then be written as:

$$P(t|\theta_i) = \frac{w_{t,i} + \alpha_t - 1}{\alpha_t + \beta_t - 1} , \tag{15}$$

where α_t and β_t are parameters of the model. Analogously to how smoothing is applied in language modeling for ad hoc document retrieval [32], we set:

$$\alpha_t = \mu \frac{|\mathbf{I}_t|}{|\mathbf{I}|} + 1 \qquad \beta_t = \frac{|\mathbf{I}|}{|\mathbf{I}_t|} + \mu(1 - \frac{|\mathbf{I}_t|}{|\mathbf{I}|}) - 1 , \tag{16}$$

where $\mathbf{I}$ is the set of all items, $\mathbf{I}_t$ is the set of items tagged with t, and μ is the smoothing parameter. Plugging Eq. (16) back into Eq. (15) yields:

$$P(t|\theta_i) = \frac{w_{t,i} + \mu\,|\mathbf{I}_t|/|\mathbf{I}|}{\mu + |\mathbf{I}|/|\mathbf{I}_t|} . \tag{17}$$

Modeling Users. The other ingredient for the estimation of the user likelihood in Eq. (14) is the positive and negative user-tag weights, $w^+_{t,u}$ and $w^-_{t,u}$. Intuitively, these correspond to positive and negative weights in the user model $\mathbf{W}$ (cf. Sect. 3.4). There is, however, the requirement of explainability that we also wish to satisfy. Instead of considering all tags for which preference could be inferred, we restrict ourselves to those tags that were selected to describe the user's preferences, i.e., part of the selected statements, $\mathbf{S}_k$. Recall that statements are generated both for single tags and for pairwise tag interactions. Pairwise tags, nevertheless, are introduced as pseudo tags with appropriate relative weights (cf. Eq. (3)), which means that those can also be treated as simple tags here.

Given the set of top-k user preference statements, we divide tags involved into those liked ($\mathbf{T}^+$) and disliked ($\mathbf{T}^-$) by the user:

$$\mathbf{T}^+ = \{t : t \in \mathbf{S}_k, w_t > 0\} \qquad \mathbf{T}^- = \{t : t \in \mathbf{S}_k, w_t < 0\} , \tag{18}$$

where w_t is the user's preference for tag t according the user model (cf. Sect. 3.4). By definition, $\mathbf{T}^+$ and $\mathbf{T}^-$ are mutually exclusive. Then, the positive and negative user-tag weights are defined as:

$$w^+_{t,u} = \begin{cases} w_t & \text{if } t \in \mathbf{T}^+ \\ 0 & \text{otherwise} \end{cases} \qquad w^-_{t,u} = \begin{cases} -w_t & \text{if } t \in \mathbf{T}^- \\ 0 & \text{otherwise} \end{cases} . \tag{19}$$

We refer to the above formulation as the *fully transparent* user model. Alternatively, one may use all tags from all candidate statements $\mathbf{C}$ to rank items (i.e., replacing $\mathbf{S}_k$ with $\mathbf{C}$ in Eq. (18)), while verbalizing only the top-k statements to the user. We call this a *partially transparent* model.

5.3 Item Priors

So far, our model is purely content-based. However, it is also possible to leverage the wisdom of the crowds, that is, the rating of other users on a given item. This is achieved by setting:

$$\frac{P(L = \ell^+|i)}{P(L = \ell^-|i)} \propto \frac{n^+_i + 1}{|\mathbf{U}| + 1 - n^-_i} , \tag{20}$$

where n^+_i and n^-_i are the number of users who liked and disliked the item,[2] respectively, and $|\mathbf{U}|$ is the total number of users.

[2]Here, we use the neutral objective rating (i.e., 0.5) to decide if ratings given to this item in the training dataset correspond to likes or dislikes.

5.4 Final Model

By taking the logarithm of Eq. (13), we obtain:

$$\begin{aligned} \log O(L = \ell^+|u,i) \quad \stackrel{\text{rank}}{=} \quad & \log P(L = \ell^+|i) - \log P(L = \ell^-|i) \\ & + \log P(u|i, L = \ell^+) - \log P(u|i, L = \ell^-) , \end{aligned} \tag{21}$$

where the first two terms are item priors that are independent of the user and the last two terms express the probability of the user liking/disliking the given item. Substituting Eqs. (14) and (20), we obtain the following final ranking function:

$$\begin{aligned} \log O(L = \ell^+|u,i) \quad \stackrel{\text{rank}}{=} \quad & \log(n^+_i + 1) - \log(|\mathbf{U}| + 1 - n^-_i) \\ & + \sum_{t \in \mathbf{T}^+} w^+_{t,u} \log P(t|\theta_i) \\ & - \sum_{t \in \mathbf{T}^-} w^-_{t,u} \log P(t|\theta_i) . \end{aligned} \tag{22}$$

When item priors are used, they may be seen as a collaborative filtering element, making the overall approach a hybrid method. In the absence of item priors, the model is a purely content-based one.

6 EXPERIMENTAL SETUP

To assess our model in terms of effectiveness and scrutability, we conduct both a traditional benchmark-style evaluation as well as a user study. We next provide a high level overview of the methodology and describe the evaluation dataset. Following this, we present the experiment design in detail and describe baselines.

6.1 Measurement Approach

Two broad categories of evaluation methodologies are commonly distinguished in the literature for measuring recommendation quality [46]: (1) *rating prediction* estimates the rating a user would assign to an item in the collection (often measured in terms of root mean square error) and (2) *item recommendation* (a.k.a. *top-N recommendation*), where a small set of items assessed for suitability as user recommendations (typically measured using rank-based metrics, such as precision or NDCG). While rating prediction has traditionally been more popular, it can only be measured with respect to observed ratings in the dataset. In practice, these are biased towards items users like or know [29, 45]. Many real-world scenarios, instead, are concerned with suggesting a few specific items from *all* items in the catalog, and therefore should instead be modeled as ranking problems [2, 12, 46]. Accordingly, for a given user u, the output of the recommendation is a ranked list of the top-N highest scoring items for that user.

6.2 Benchmark Dataset

We employ the MovieLens-20M (ML-20M) dataset, which has been extensively used in recommendation research [20]. It describes users' movie preferences, expressed as 5-star ratings. The dataset also contains social tags that users have assigned to individual movies. These are typically single words or short phrases whose meaning and purpose is determined by the user. Tag quality is known to be mixed in this collection [17, 51]. Therefore, we filter tags on multiple criteria. Following [51], we limit the vocabulary of tags to those that have been applied by at least five different users and to at least two different items. Further, we remove tags deemed inappropriate (e.g., adult content) or of low utility (e.g., "don't remember"). For a given item, we only keep tags that have

Table 2: Characteristics of the MovieLens-20M dataset.

	#Ratings	#Users	#Items	#Tags
Original data	20 000 263	138 493	27 278	38 641
Filtered data used	17 710 309	133 638	5 839	5 543

been assigned by at least two users. Finally, we filter out movies that have fewer than two tags assigned to them. Table 2 provides descriptive statistics on the original and filtered datasets.

We partition the dataset into train and test sets by randomly selecting 1000 users and sampling 20% of their ratings as test data. All other users and the remaining ratings of our test users constitute the training split. For each user, we generate a ranked list of 100 items. We report standard IR measures: mean average precision (MAP), mean reciprocal rank (MRR), and normalized discounted cumulative gain (NDCG). For measures that operate on binary relevance (MAP and MRR), following [26], items rated 4 stars or more are considered relevant. When relevance is graded (NDCG), the gain value is the star-rating minus 2, i.e., a 3-star item receives gain 1 and a 5-star item receives gain 3.

A major challenge in rank-based evaluation is that only some of the items users like are generally known. One evaluation strategy (referred to as the *TestRatings* methodology in [2]) is to only consider items that have been rated by the user in the test set. This, however, "does not test the recommender's ability to identify interesting items from a large pool" [15] and thus amounts to solving an easier problem, often considerably overestimating the true performance of a system. Instead, we consider all items rated by any user in the training set a potential candidate (except those already rated by the target user)—known as the *TrainingItems* methodology in [2]. Items that have not been rated by the user are all assumed to be non-relevant. As there are many such items, some of which are possibly relevant, this leads to underestimated performance [2].

6.3 User Study

We also design a user study to evaluate the transparency and scrutability of our model. Taking place on a crowdsourcing platform (restricted to US-based users), it was performed in two rounds, as illustrated in Figure 4. First, a preference elicitation round is conducted, where users are asked to provide ratings on a number of movies. This data serves as input to the set-based model and to the subsequent summary generation. The second round is further subdivided into two steps. In step 2A, users are presented with a summary of their preferences, or more precisely, a number of sentences, which they are asked to inspect (scrutinize). In step 2B, users are given a set of movie recommendations to rate. We provide specific details regarding the data collection process below.

We took a stratified sample of 500 movies from the filtered MovieLens-20M dataset as our item collection. This sample is made up of (i) the top 150 movies by the number of ratings received and (ii) a random movie for each year between 1980 and 2014, and for each of the top 10 most popular genres,[3] that is not already in the top-150 set. For the random movies, we required each to be rated by at least 20 users (to avoid movies too far down in the long tail). We randomly split the set of movies into a preference elicitation set (400 movies) and a candidate set (100 movies).

[3]These are: action, adventure, documentary, comedy, crime, drama, horror, romance, sci-fi, and thriller.

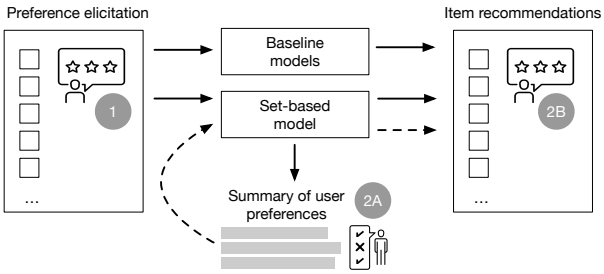

Figure 4: Experimental design for evaluating transparency and scrutability.

Table 3: Illustration of scrutability options for a summary preference statement in our user study.

(1)	You like movies that are tagged as 'action', especially those that are tagged as 'sword fight', such as The Princess Bride.
(2)	You like movies that are tagged as 'action', especially those that are tagged as 'sword fight'.
(3)	You like movies that are tagged as 'action', such as The Princess Bride.
(4)	You like movies that are tagged as 'action'.
(5)	You like The Princess Bride.
(6)	None of the above.

In round one, users were asked to rate at least 20 and at most 80 movies, using a three-point rating scale, corresponding to 1-star (dislike), 3-star (neutral), and 5-star (like) ratings on a 5-star scale. To prevent workers from providing random ratings, we also requested them to specify, in a text input field, what they liked/disliked about the given movie. To incentivize workers, they were promised an invitation to a generously-compensated follow-up task (i.e., round two), subject to the quality of their responses. We performed a manual inspection of the text inputs provided, and filtered out users with low-quality responses. In this way, we collected two rounds of responses from 122 users.

Round two was divided into two steps. First, in 2A, users were asked to inspect and provide feedback on the top five summary sentences generated for them using Eq. (9). To make user feedback actionable, we start with the complete statement, then remove segments progressively. Participants are asked to inspect the statements in order, and select the first that accurately describes their preferences. We refer to Table 3 for a specific example. For each option, we interpret the choice by updating the user's model if the option is selected. For instance, if the user were to select the third option in Table 3, we would update their user model to set the weight of the pairwise tag (action,sword fight) to zero.

Finally, in 2B, we presented users with personalized movie recommendations to rate (selected from the candidate set). Note that we perform 2A and 2B in a single round to reduce user dropout. We thus need to anticipate possible changes to the user summaries. At the same time, we need to keep the number of items to rate reasonable, to avoid attention fatigue. Therefore, users are presented with 20 movies, pooled from the following sources: two baseline methods (ItemKNN and BPRSLIM), the initial set-based model (i.e., no scrutinization), and the set-based model with different user model variants, based on the anticipated changes. Specifically, for each summary statement and for each scrutability option, we create a corresponding updated model, then generate recommendations using that user model. To form the pool of items to rate, we take the top-5 recommendations of the two baselines and of the initial

Table 4: Benchmark item recommendation results. %UJ refers to the fraction of unjudged items in the top 10.

Method	MRR	MAP	NDCG	%UJ
MostPopular	0.272	0.071	0.214	85%
Item-kNN	0.360	0.129	0.341	76%
BPR-MF	0.387	0.137	0.334	80%
WR-MF	0.439	0.155	0.349	78%
SoftMarginRanking-MF	0.275	0.082	0.240	86%
WeightedBPR-MF	0.265	0.073	0.207	89%
BPR-SLIM	0.441	0.158	0.359	77%
Tags-cosine	0.319	0.073	0.195	86%
Tags-cosine + priors	0.351	0.094	0.245	83%
Set-based	0.308	0.081	0.228	85%

set-based model, and supplement with items by the sum of reciprocal ranks at which they appear in other user models. Similarly to the benchmark evaluation, when using binary relevance only the liked items are considered relevant. When relevance is graded, neutral (3-star) items have gain of 1, and liked items have gain of 3.

6.4 Baselines

We compare against the following collaborative filtering methods, as implemented in the MyMediaLite recommender library [16]:[4]

- **MostPopular**: A simple non-personalized baseline that recommends the most popular items (i.e., those with the most ratings).
- **Item-kNN**: Item-based k-Nearest Neighbors [40], a classical collaborative filtering algorithm that is usually a strong baseline.
- **WR-MF**: Weighted Regularized Matrix Factorization [23, 35] is a regularized version of singular value decomposition (SVD). It can be seen as a pointwise regression approach that learns latent factors by minimizing the square-loss.
- **BPR-MF**: Bayesian Personalized Ranking [38] is a matrix factorization model directly optimized for ranking. It is "built on the assumption that a user prefers an item with a positive feedback to an item without an observed feedback" [37].
- **BPR-SLIM**: Sparse Linear Methods (SLIM) [34] learn a sparse coefficient matrix for items solely from the user rating profiles by solving a regularized optimization problem. We employ a variant that is optimized for the BPR-Opt criterion [38], using Stochastic Gradient Ascent.

All methods use the default configuration settings in MyMediaLite. MostPopular and Item-kNN operate on the original 5-star ratings. For matrix factorization and sparse linear methods, ratings are binarized (corresponding to the problem of learning from binary, positive-only feedback [50]). Specifically, ratings scored by 4 stars and above are taken as positive feedback [26].

Since our set-based model operates on tags, we also consider a content-based approach, which calculates the similarity between a user's profile vector and an item's tag vector.

- **Tags-cosine**: Items are scored according to the cosine similarity between the item's tag vector $\vec{w}_i$ and the user's preference vector $\vec{w}_u$. For a given tag t, the user's preference for t is given by:

$$w_{t,u} = \frac{1}{|\mathbf{I}^*|} \sum_{i \in \mathbf{I}_t^*} (r_i - r_\mu) \cdot w_{t,i} \,, \tag{23}$$

where r_μ is the neutral rating. We also consider another variant with the popularity prior (cf. Eq. (20)) incorporated by way of multiplication.

[4] http://www.mymedialite.net (version 3.11).

Table 5: User study item recommendation results.

Method	MRR	MAP	NDCG@5	NDCG@10
MostPopular	0.882	0.515	0.721	0.628
Item-kNN	0.479	0.245	0.452	0.364
BPR-SLIM	0.709	0.378	0.624	0.511
Set-based model				
Full transp., no priors	0.710	0.393	0.543	0.499
Full transp., priors	0.835	0.529	0.748	0.643
Partial transp., no priors	0.748	$0.516^{\ddagger}$	$0.663^{\ddagger}$	$0.648^{\ddagger}$
Partial transp., priors	0.866	$0.554^{\ddagger}$	$0.782^{\ddagger}$	$0.670^{\ddagger}$

7 RESULTS

We now present a detailed analysis of the performance of our approach both in terms of recommendation quality, and the extent to which it satisfies the desirable properties listed in Sect. 4.1.

7.1 Effectiveness

Our first research question concerns the effectiveness of item recommendations produced by our method. This is evaluated on the benchmark (*Benchmark*) and user study (*UserStudy*) datasets.

We start by considering the standard *Benchmark* dataset and compare against all the baselines; see Table 4. We note that popularity is a relatively weak baseline, and the simple tags-cosine approach is reasonably competitive, particularly when movie scores are weighted with a popularity prior. Our set-based approach (also including the prior, and all candidate statements in $\mathbf{C}$) does not perform best, but also does not perform dramatically more poorly than many reasonable baselines.

However, we particularly note that in this dataset, the vast majority of items that contribute to the performance of a model are not rated by users as shown in the %UJ column. This introduces the potential for bias, and the results for all methods must thus be treated as lower bound estimates for algorithm performance.

For the *UserStudy*, we consider three representative baselines: MostPopular, Item-kNN, and BPR-SLIM. Note that these are trained on the ML-20M collection. We use these to assess four specific variants of our set-based model.

First, we observe that the performance numbers in Table 5 are generally much higher, as users have rated a much larger fraction of items. In particular, almost all items (%UJ < 3%) in the top 5 positions for all the algorithms were judged by the user study participants. No more than 39% of items in the top 10 positions were unjudged (and %UJ < 25% for any variant of the set-based model). This makes these results much more representative than those in Table 4.

As noted earlier, a scrutable model is one that has a limited number of statements, which could be presented to a user (with a possibility to revise/correct them). We consider an algorithm transparent if it only uses such a model, i.e., defined by $\mathbf{S}_k$, consisting of no more than $k = 5$ statements. We term *partial transparency* use of the model $\mathbf{C}$, which, while larger, still selects down for tags and pairs of tags with significant coverage of the user's judged items.

On the *UserStudy* dataset, the set-based model performs best on three of the metrics, while the most popular model performs best

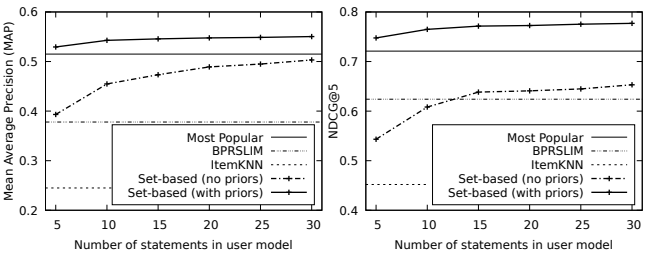

Figure 5: Recommendation performance as a function of model size. Note the strong effect of popularity.

on MRR. As such, we also see the large role that the popularity prior plays, and the relatively small cost of full transparency. We will consider this effect in more detail next.

7.2 Transparency

Our second research question asks about the effects of providing full transparency as opposed to partial transparency. According to Table 5, the relative difference between full and partial transparency is at most 31% without priors and less than 5% when using priors, for any metric. The differences, however, are statistically significant for all metrics except MRR (indicated by ‡, using a two-tailed paired t-test with a p-value threshold of 0.001). Figure 5 further studies the effect of transparency in detail, by varying the length of the user summary. We see that as the number of statements S_k in the user model is increased, recommendation quality improves. While there is a substantial improvement as the model size increases from 5 to 15 statements, beyond this the performance increases more slowly. We also clearly see the effect of the popularity prior.

7.3 Scrutability

Finally, our third research question considers scrutability: Given user feedback on the generated summaries, what impact does it have on the recommendations? Are the inferred statements correctly capturing the participant's interests?

Table 6 shows the distribution of user responses on the generated (top-5) statements. First, we note that, surprisingly, around 19% of two-tag statements and 27% of single-tag statements had the participant disagree with the example belonging to the correct statement. As the example was always rated by the participant, and had the relevant tags, there are two possibilities. Either the participant may disagree with the tag(s) being appropriate for the example. Alternatively, the participant's opinion about the example may have changed between the first and second stages of the user study. As the stages were conducted within a day of each other, we expect that this illustrates the aforementioned data quality concern with tags, where perhaps a significant fraction of tags do not universally represent the movies to which they are attached. Exploring the effect of tag quality is an important direction for understanding where the performance of tag-based algorithms is suboptimal.

Second, we note that only 34% and 57% of participants agree with the entire statements (ignoring the example). This suggests that there is a lot of potential for users to improve their inferred user model in tag-based approaches.

Table 7 shows the effect of scrutability on recommendation quality. Specifically, *Initial* shows the performance of the set-based algorithm using the inital model (matching the *full transparency* results from Table 5). If statements from the fully transparent model

Table 6: Distribution of responses on the generated user preference statements.

Statements for tag interactions	Count	Ratio
(1) [first] [interaction] [second] [example]	39	27%
(2) [first] [interaction] [second]	10	7%
(3) [first] [example]	33	23%
(4) [first]	14	10%
(5) [example]	44	31%
(6) None of the above	3	2%
Total	143	100%
Statements for single tags		
(1) [first] [example]	154	37%
(2) [first]	82	20%
(3) [example]	150	36%
(4) None of the above	31	7%
Total	417	100%

Table 7: Scrutability results for the set-based model. We report on the fully transparent variant (k=5).

Method	MRR	MAP	NDCG@5	NDCG@10
Initial, no priors	0.710	0.393	0.543	0.499
Initial, w/ priors	0.835	0.529	0.748	0.643
Corrected, no priors	0.678	0.381	0.530	0.484
Corrected, w/ priors	0.855	0.532	0.751	0.645

with which the participants disagree are removed from the user model, and are replaced with the next candidate statements from **C** to yield a new full transparent recommendation model consisting of 5 statements, we obtain the performance listed as *Corrected*. We see that in the set-based model with priors, this improves overall recommendation performance. We therefore conclude that scrutability is in fact being achieved. At the same time, we note that without the popularity prior, the corrected models perform slightly worse than the initial ones. We attribute this to the fact that user feedback is not utilized to its full possible extent. In particular, we are only removing tag preferences when the user did not agree with a statement. It would also be possible to update the weights of tags that belonged to statements with which users agreed. It should be noted that the observed differences between *Initial* and *Corrected* are not statistically significant, they should thus be regarded as indicative.

8 CONCLUSIONS

The task we set out in this paper is develop recommendation approaches that are more transparent and scrutable. We presented a novel set-based recommendation model, which we showed to return recommendations that are comparable quality to state-of-the-art recommendation algorithms despite being transparent and explainable. In a user study with very high coverage of rated movies, we demonstrated how the *user model* can be explicitly scrutinized by users, leading to improved recommendations. We also showed how the size of the model that must be scrutinized can be traded off against recommendation quality.

This study suggests a number of avenues for future work. Beyond questions of tag quality, and non-binary tag weighting for set-based models, we also leave open the question of how best to benefit from inferred preferences that users have scrutinized and agree with.

REFERENCES

[1] Behnoush Abdollahi and Olfa Nasraoui. 2016. Explainable Matrix Factorization for Collaborative Filtering. In *Proc. of WWW Companion*. 5–6.

[2] Alejandro Bellogin, Pablo Castells, and Ivan Cantador. 2011. Precision-oriented Evaluation of Recommender Systems: An Algorithmic Comparison. In *Proc. of RecSys'11*. 333–336.

[3] R. I. Brafman, C. Domshlak, S. E. Shimony, and Y. Silver. 2006. Preferences over Sets. In *Proc. of AAAI'06*. 1101–1106.

[4] Gerhard Brewka, Mirosław Truszczyński, and Stefan Woltran. 2010. Representing Preferences Among Sets. In *Proc. of AAAI'10*. 273–278.

[5] Jaime Carbonell and Jade Goldstein. 1998. The Use of MMR, Diversity-based Reranking for Reordering Documents and Producing Summaries. In *Proc. of SIGIR'98*. 335–336.

[6] Rose Catherine, Kathryn Mazaitis, Maxine Eskénazi, and William W. Cohen. 2017. Explainable Entity-based Recommendations with Knowledge Graphs. In *Proc. of the Poster Track of RecSys'17*.

[7] Shuo Chang, F. Maxwell Harper, and Loren Terveen. 2015. Using Groups of Items for Preference Elicitation in Recommender Systems. In *Proc. of CSCW'15*. 1258–1269.

[8] Shuo Chang, F. Maxwell Harper, and Loren Gilbert Terveen. 2016. Crowd-Based Personalized Natural Language Explanations for Recommendations. In *Proc. of RecSys'16*. 175–182.

[9] Xu Chen, Zheng Qin, Yongfeng Zhang, and Tao Xu. 2016. Learning to Rank Features for Recommendation over Multiple Categories. In *Proc. of SIGIR'16*. 305–314.

[10] Xu Chen, Yongfeng Zhang, Hongteng Xu, Yixin Cao, Zheng Qin, and Hongyuan Zha. 2018. Visually Explainable Recommendation. *CoRR* abs/1801.10288 (2018). arXiv:1801.10288

[11] Zhiyong Cheng, Ying Ding, Lei Zhu, and Mohan Kankanhalli. 2018. Aspect-Aware Latent Factor Model: Rating Prediction with Ratings and Reviews. In *Proc. of WWW'18*. 639–648.

[12] Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. 2010. Performance of Recommender Algorithms on Top-n Recommendation Tasks. In *Proc. of RecSys'10*. 39–46.

[13] Marco de Gemmis, Pasquale Lops, Cataldo Musto, Fedelucio Narducci, and Giovanni Semeraro. 2015. Semantics-Aware Content-Based Recommender Systems. In *Recommender Systems Handbook* (2nd ed.), Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor (Eds.). Springer US, Chapter 4, 119–159.

[14] Marie desJardins, Eric Eaton, and Kiri L. Wagstaff. 2006. Learning User Preferences for Sets of Objects. In *Proc. of ICML'06*. 273–280.

[15] Michael D. Ekstrand and Vaibhav Mahant. 2017. Sturgeon and the Cool Kids: Problems with Random Decoys for Top-N Recommender Evaluation. In *Proc. of FLAIRS'17*. 639–644.

[16] Zeno Gantner, Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2011. MyMediaLite: A Free Recommender System Library. In *Proc. of RecSys'11*. 305–308.

[17] Fatih Gedikli, Dietmar Jannach, and Mouzhi Ge. 2014. How Should I Explain? A Comparison of Different Explanation Types for Recommender Systems. *Int. J. Hum.-Comput. Stud.* 72, 4 (April 2014), 367–382.

[18] Scott A. Golder and Bernardo A. Huberman. 2006. Usage Patterns of Collaborative Tagging Systems. *J. Inf. Sci.* 32, 2 (April 2006), 198–208.

[19] Stephen J. Green, Paul Lamere, Jeffrey Alexander, François Maillet, Susanna Kirk, Jessica Holt, Jackie Bourque, and Xiao-Wen Mak. 2009. Generating Transparent, Steerable Recommendations from Textual Descriptions of Items. In *Proc. of RecSys'09*. 281–284.

[20] F. Maxwell Harper and Joseph A. Konstan. 2015. The MovieLens Datasets: History and Context. *ACM Trans. Interact. Intell. Syst.* 5, 4, Article 19 (Dec. 2015), 19:1–19:19 pages.

[21] Xiangnan He, Tao Chen, Min-Yen Kan, and Xiao Chen. 2015. TriRank: Review-aware Explainable Recommendation by Modeling Aspects. In *Proc. of CIKM'15*. 1661–1670.

[22] Jonathan L. Herlocker, Joseph A. Konstan, and John Riedl. 2000. Explaining Collaborative Filtering Recommendations. In *Proc. of CSCW'00*. 241–250.

[23] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In *Proc. of ICDM'08*. 263–272.

[24] Yehuda Koren and Robert Bell. 2015. Advances in Collaborative Filtering. In *Recommender Systems Handbook* (2nd ed.), Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor (Eds.). Springer US, Chapter 3, 77–118.

[25] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer* 42, 8 (Aug. 2009), 30–37.

[26] Nathan N. Liu, Xiangrui Meng, Chao Liu, and Qiang Yang. 2011. Wisdom of the Better Few: Cold Start Recommendation via Representative Based Rating Elicitation. In *Proc. of RecSys'11*. 37–44.

[27] Yichao Lu, Ruihai Dong, and Barry Smyth. 2018. Coevolutionary Recommendation Model: Mutual Learning Between Ratings and Reviews. In *Proc. of WWW'18*. 773–782.

[28] Yuanhua Lv and ChengXiang Zhai. 2012. Query Likelihood with Negative Query Generation. In *Proc. of CIKM'12*. 1799–1803.

[29] Benjamin M. Marlin and Richard S. Zemel. 2009. Collaborative Prediction and Ranking with Non-random Missing Data. In *Proc. of RecSys'09*. 5–12.

[30] Julian McAuley and Jure Leskovec. 2013. Hidden Factors and Hidden Topics: Understanding Rating Dimensions with Review Text. In *Proc. of RecSys'13*. 165–172.

[31] David McSherry. 2005. Explanation in Recommender Systems. *Artif. Intell. Rev.* 24, 2 (Oct. 2005), 179–197.

[32] Donald Metzler, Victor Lavrenko, and W. Bruce Croft. 2004. Formal Multiple-bernoulli Models for Language Modeling. In *Proc. of SIGIR'04*. 540–541.

[33] Don Monroe. 2018. AI, Explain Yourself. *Commun. ACM* 61, 11 (Oct. 2018), 11–13.

[34] Xia Ning and George Karypis. 2011. SLIM: Sparse Linear Methods for Top-N Recommender Systems. In *Proc. of ICDM'11*. 497–506.

[35] Rong Pan, Yunhong Zhou, Bin Cao, Nathan N. Liu, Rajan Lukose, Martin Scholz, and Qiang Yang. 2008. One-Class Collaborative Filtering. In *Proc. of ICDM'08*. 502–511.

[36] Pearl Pu, Li Chen, and Rong Hu. 2012. Evaluating Recommender Systems from the User's Perspective: Survey of the State of the Art. *User Modeling and User-Adapted Interaction* 22, 4-5 (Oct. 2012), 317–355.

[37] Shuang Qiu, Jian Cheng, Ting Yuan, Cong Leng, and Hanqing Lu. 2014. Item Group Based Pairwise Preference Learning for Personalized Ranking. In *Proc. of SIGIR'14*. 1219–1222.

[38] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *Proc. of UAI'09*. 452–461.

[39] S. E. Robertson. 1977. The Probability Ranking Principle in IR. *Journal of Documentation* 33, 4 (1977), 294–304.

[40] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based Collaborative Filtering Recommendation Algorithms. In *Proc. of WWW'01*. 285–295.

[41] Shilad Sen, F. Maxwell Harper, Adam LaPitz, and John Riedl. 2007. The Quest for Quality Tags. In *Proc. of GROUP'07*. 361–370.

[42] Shilad Sen, Jesse Vig, and John Riedl. 2009. Tagommenders: Connecting Users to Items Through Tags. In *Proc. of WWW'09*. 671–680.

[43] Sungyong Seo, Jing Huang, Hao Yang, and Yan Liu. 2017. Interpretable Convolutional Neural Networks with Dual Local and Global Attention for Review Rating Prediction. In *Proc. of RecSys'17*. 297–305.

[44] Mohit Sharma, F. Maxwell Harper, and George Karypis. 2017. Learning from Sets of Items in Recommender Systems. In *Proc. of eKNOW'17*. 59–64.

[45] Harald Steck. 2010. Training and Testing of Recommender Systems on Data Missing Not at Random. In *Proc. of KDD'10*. 713–722.

[46] Harald Steck. 2013. Evaluation of Recommendations: Rating-prediction and Ranking. In *Proc. of RecSys'13*. 213–220.

[47] Nava Tintarev and Judith Masthoff. 2007. Effective Explanations of Recommendations: User-centered Design. In *Proc. of RecSys'07*. 153–156.

[48] Nava Tintarev and Judith Masthoff. 2012. Evaluating the Effectiveness of Explanations for Recommender Systems. *User Modeling and User-Adapted Interaction* 22, 4-5 (Oct. 2012), 399–439.

[49] Nava Tintarev and Judith Masthoff. 2015. Explaining Recommendations: Design and Evaluation. In *Recommender Systems Handbook* (2nd ed.), Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor (Eds.). Springer US, Chapter 10, 353–382.

[50] Koen Verstrepen, Kanishka Bhaduriy, Boris Cule, and Bart Goethals. 2017. Collaborative Filtering for Binary, Positiveonly Data. *SIGKDD Explor. Newsl.* 19, 1 (Sept. 2017), 1–21.

[51] Jesse Vig, Shilad Sen, and John Riedl. 2009. Tagsplanations: Explaining Recommendations using Tags. In *Proc. of IUI'09*. 47–56.

[52] Nan Wang, Hongning Wang, Yiling Jia, and Yue Yin. 2018. Explainable Recommendation via Multi-Task Learning in Opinionated Text Data. In *Proc. of SIGIR'18*. 165–174.

[53] Yongfeng Zhang. 2015. Incorporating Phrase-level Sentiment Analysis on Textual Reviews for Personalized Recommendation. In *Proc. of WSDM'15*. 435–440.

[54] Yongfeng Zhang and Xu Chen. 2018. Explainable Recommendation: A Survey and New Perspectives. *CoRR* abs/1804.11192 (2018). arXiv:1804.11192

[55] Yongfeng Zhang, Guokun Lai, Min Zhang, Yi Zhang, Yiqun Liu, and Shaoping Ma. 2014. Explicit Factor Models for Explainable Recommendation Based on Phrase-level Sentiment Analysis. In *Proc. of SIGIR'14*. 83–92.