Goal-Oriented Annotation Recommenders

Malte Kiesel[1] and Injy Hamed[2]

[1]DFKI GmbH, Kaiserslautern, Germany
[2]German University in Cairo, Cairo, Egypt
malte.kiesel@dfki.de,injy.hamed@guc.edu.eg

Abstract. *With the rise of Web 2.0, the amount of information available to the users has grown tremendously. Recommendation systems have emerged as successful tools that look into the users' perspectives and accordingly provide users with information presumed to be of interest to them. Early generations of recommendation systems have achieved great success. However, in order to obtain more precise recommendations and improve the overall user experience, the need for fine-grained annotations that accurately describe the items has become essential. This gave rise to annotation recommenders that motivate users to annotate more. In this paper, two goal-oriented annotation recommenders are proposed, that aim at supporting different functionalities of the recommender system.*

Keywords: recommender systems, annotation recommender, knowledge management, social networks, personalization, personalized recommendation

1 Introduction

We are living in an era of information overload. Users are constantly confronted with many choices and massive amounts of information. Recommender systems have emerged as tools that support the users in their search for relevant information and discovery of subjectively interesting items (such as books, movies, etc.). The great success achieved by recommender systems has drawn researchers' attention to further improve and fully exploit the information available in such systems to reach their full potential. In recent years, recommendations provided to users were more of a "black box" model. The user had no control over it, and no explanations were provided for the recommended items. Nowadays, researchers strive to make recommendations an interactive process that includes the user, and build a social network where users interact and discuss. Among other goals, researchers aim at giving users the control over the recommendation process, making recommendation logic transparent by providing insights through explanations and providing further social functionality.

To achieve those functionalities, fine-grained annotations are needed that provide information on the content of items. Feature extraction methods are available that extract keywords or features from resources' contents. However, these are laborious and costly. Consequently, we rely on users to enter content annotations. This, along with other benefits of annotations, lead to the widespread success of social annotation systems. Several social annotation services have emerged in the last years covering different media types. Among others, BibSonomy [1], del.icio.us [2] and Flickr [3] are social annotation systems that have achieved great success. In these systems, users share resources and assign descriptors to resources that are of interest to them—this is referred to as the act of annotation. However, the success of social annotation systems is tied to the frequency of annotations provided by users. The number of annotations given by users to items depends on their motivation to annotate. It is usually the case in any recommender system that this number is small, as annotation is a tedious task.

In this paper, two annotation recommenders will be introduced that motivate the users to enter annotations that would be of benefit to the system. The work done in this paper was built on an existing

[1] http://www.bibsonomy.org/
[2] http://www.del.icio.us.com/
[3] http://www.flickr.com/

recommender system Skipforward[4]. Skipforward is a multi-user annotation and recommendation system. It allows users to create and associate feature instances with items. Two goal-oriented annotation recommenders have been implemented in Skipforward. The first one, *annotation recommender (item)*, is aimed at aiding the user when providing annotations for an item. The recommender suggests user-based as well as item-based annotations. This eases the process of annotation, and motivates users to provide more information on items, and thus consequently improves item recommendations. The other recommender, the *annotation recommender (expert)*, is aimed at gathering more information on the difference between users' opinions with regard to specific topics. This helps the systems to better determine the similarities between users, and thus improve the competence metric Skipforward uses for its expert recommender. In the long run, this also improves item recommendations in the collaborative filtering approach.

This paper is organized as follows. In Section 2 Skipforward is introduced. The ontologies used to formalize users' opinions are explained. It is then briefly described how Skipforward identifies similar users. Section 3 defines the notations used throughout the paper. Section 4 gives an overview on the first annotation recommender, *annotation recommender (item)*, developed in this work. In Section 5, the second annotation recommender, *annotation recommender (expert)*, is discussed. In Sections 6 and 7, the methods used to evaluate each of the proposed annotation recommenders are outlined, and the results are shown. Finally, Section 8 concludes and gives suggestions for future work.

2 Skipforward Overview

Skipforward is an ontology-based distributed annotation system that allows users to formalize their opinions about items in an organized way. Users can express liking or disliking of an item, such as stories, board games, or video games. Users can also describe an item by assigning a feature type to it, such as *Fast story pace* for a story. The act of a user annotating an item with a feature type is represented by a feature. When annotating, features are given two values, applicability and confidence. Applicability is a measure of how much the feature type applies to an item. The value of applicability ranges from -1 for "does not apply" to +1 "totally applies". Confidence is a measure of how much the user is certain about his/her opinion. It ranges from 0 as in "completely unsure" to 1 as in "certain". A screenshot for the item view is given in Figure 1.

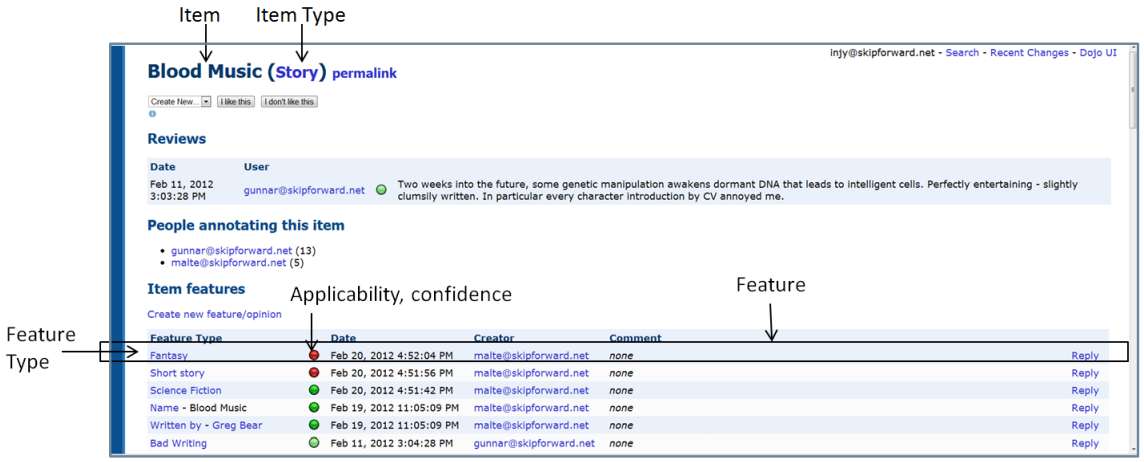


Fig. 1. Item view for the item *Blood Music*.

Skipforward uses a hybrid approach [4] for item recommendation. Similar items are identified by comparing items with regards to the features assigned to them. For an item, there could exist annotations by multiple users for a certain feature type. The idea is to aggregate all the values given to each feature type, and then compare items on basis of the aggregated value given to each feature type. This aggregation takes into consideration the similarity between users, such that the users similar to the current user

[4] http://skipforward.net/

should have a greater impact on the final value. The similarity between two users regarding a feature type is calculated using competence metric that is based on the constrained Pearson correlation. The competence metric, discussed in Section 2.1, is the foundation of many functionalities in Skipforward. Besides item recommendation, for any stated opinion, Skipforward shows the user that made that statement, and provides a scale on how much that user is similar to the current user regarding that feature type. This way, the current user can decide what to believe and what not to. For an overview of Skipforward components and how the user's view of the data is completely personalized, also see [1].

Skipforward uses a closed vocabulary of feature types that are defined as RDF statements using domain ontologies. Currently, within Skipforward multiple domains are covered. Skiptrax and Ludopinions ontologies define the feature types for songs and board games, respectively. Apart from those, DBTropes[5] is mainly used as a source for feature types as well as the hierarchy within the feature types. DBTropes [2] extracts data from a huge database containing thousands of items (fiction works) and feature types ("Tropes"), called TV Tropes[6], which is a wiki-based community project. TV Tropes also provides a hierarchy to its Tropes, which can be used in inferencing. DBTropes parses TV Tropes pages and outputs machine-interpretable RDF having the same format as the Skipforward ontologies, and thus can be directly used in Skipforward. As of December 2012, the complete DBTropes data consists of about 12.000.000 RDF statements, 32.000 items, 22.400 feature types and 2.000.000 feature instances.

2.1 Calculating users' similarity

The similarity between two users regarding a certain feature type is identified by comparing their opinions given to that feature type across items.

Competence metric The Pearson correlation is used to calculate the similarity between two users regarding a certain feature type. The Pearson correlation coefficient is a measure of the correlation (linear dependence) between two variables X and Y. The correlation value lies in the range [-1;+1]. The Pearson correlation is applied on two variables, which in the case of Skipforward, represent the applicability values given by each user to an item for a certain feature type. However, for a feature, there is an additional value that should be taken into consideration, which is the confidence, where features with high confidence values should have higher impact on the users' similarity. This is not handled in the standard Pearson correlation, and thus skipforward uses the weighted Pearson product-momentum correlation coefficient, where the confidence value is the weighting. Similarity between two users with regards to a feature type is calculated as follows:

$$sim_t(u_x, u_y) = \frac{\sum\limits_{i \in I_{xy}} w_{xy,i} r_{x,i} r_{y,i}}{\sqrt{\sum\limits_{i \in I_{xy}} w_{xy,i} (r_{x,i})^2 \sum\limits_{i \in I_{xy}} (r_{y,i})^2}} \tag{1}$$

where u_x and u_y denote users x and y, I_{xy} is the set of co-rated items of both users (Items rated by both users), $r_{x,i}$ denotes the applicability value given by user x to item i for feature type t. $w_{x,i}$ is the confidence value given by user x to item i for feature type t. $w_{xy,i}$ is the combined confidence of the statements concerning the feature type t of users x and y for item i.

Confidence in similarities The similarity value calculated between two users is more reliable when there are many co-ratings. In other words, the similarity between users x and y regarding feature type t is more reliable when x and y both annotated a hundred items in common with that feature type, rather that just two items. The confidence for a similarity value is calculated as follows:

$$conf_{sim}(u_x, u_y) = \frac{2|I_{xy}|}{|I_x| + |I_y|} \tag{2}$$

Where I_x and I_y are the items annotated by users x and y, respectively, and I_{xy} denotes the co-rated items.

[5] http://dbtropes.org/
[6] http://tvtropes.org/

Feature Aggregation In order to compare two items, all statements concerning each feature type assigned to this item are aggregated to give a single value. This is done taking into consideration the competence metric, such that the opinion of more similar users have greater influence on the result. The applicability value for an aggregated feature is calculated as a weighted sum of the applicability values given to each feature assigned to this item for that feature type. The weight is the respective user's competence (or similarity) regarding the feature type, taking into consideration the confidence in the similarity.

2.2 Expert recommender

The expert recommender gives the current user a list of the users that usually express similar opinions to his/hers concerning a certain feature type. It shows the similarity between both users with regard to that feature type ($sim(x, y)$ calculated in equation (1)), and how much the system is confident about this similarity value ($conf(u_x, u_y)$ calculated in equation (2)). For the expert recommender to work, there has to be a good number of overlapping annotations between the two users for that feature type. Figure 2 shows that the current user agrees with the user Durham, while he/she disagrees with TV Tropes for the feature type *Battle Butler*.

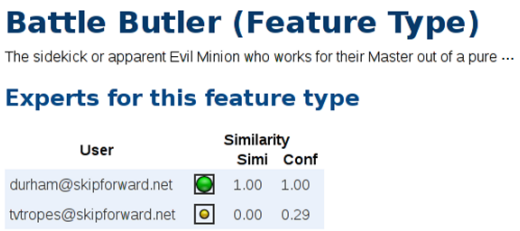


Fig. 2. Expert recommender for the feature type *Battle Butler*.

3 Notations

In this section, we define the notations used in this paper.

Symbol	Description
I	the set of all items
i	an instance of an item
U	the set of all users
u	an instance of a user
uc	the current user
T_f	the set of all feature types
ft	an instance of a feature type
F	the set of all features
$f(u, i, ft)$	an instance of a feature

Table 1. Notations.

4 Annotation Recommender (item)

In this section we provide a detailed description of the *annotation recommender (item)*. This recommender aims at helping and guiding the user in the process of annotation by suggesting a list of feature types that are both user- and item-related. The user then can choose what feature type he wants to instanciate, and what applicability value should be assigned. This allows to express explicit dissent with other people's opinions: Creating a feature with applicability value -1.0 in case there is a feature of the same type by another user with value $+1.0$ represents an opinion conflict. So, while annotation recommenders increase item and user *comparability*, they do not necessarily increase item and user *similarity*.

The *annotation recommender (item)* reduces the annotation task from the act of generation to recognition. Along with its other benefits, it most importantly encourages the user to annotate more frequently,

thus decreasing data sparsity. In this work, we proposed an interactive approach that exploits the knowledge gathered from different sources. The composite annotation recommender system incorporates four individual evidence sources: (1) *Personal*, (2) *Global*, (3) *Web* and (4) *Co-occurrence*. The recommendations obtained from the four evidences are combined to maximize performance. This is done using two aggregation methods, normalization and Borda count. Moreover, the users are given control over the impact each evidence source will have on the final recommendation. With the use of sliders, user set the weights applied to each evidence source in the aggregation step. The following sections provide detailed description of each individual evidence source. Finally, the aggregation step is discussed. Users are also provided with an explanation on why the feature types were recommended to them. Figure 7 shows the item view for the item *A Tale of Two Cities*.

4.1 Personal: feature types known to user

In this evidence source, feature types known to the user are recommended. These feature types fall under two categories: (1) Feature types previously used by the user, (2) feature types viewed by the user (whose pages have been visited). A list of feature types is first acquired from each category, and then merged to give one recommended list of feature types that are known to the user.

Feature types used by user User profile reflects user's interests and activities, as it contains the feature types previously used by the user. The intuition behind this evidence source is that usually users use a limited set of feature types when annotating items. Consequently, it is safe to assume that when a user annotates an item, there's a high probability the user will reuse some of the feature types available in the user profile that are associated to the type of the current item. The score of a feature type is defined as its number of occurrences in the user profile:

$$\underset{used}{Score}(ft) = \Big|\{f(\mathbf{uc}, i, ft) | f(\mathbf{uc}, i, ft) \in F, i \in I\}\Big|$$

Feature types viewed by user When a user visits a feature type page, it is assumed that the description of the feature type is read, and the user knows what the feature type stands for, and can thus use it. It can also be assumed that the user is interested in this feature type and looked it up. Logging is done for the feature types viewed by each user. All viewed feature types are given an equal score of 1 (as it makes no difference whether a feature type is viewed once or more, in both cases, it is assumed that the user is familiar with the feature type).

$$\underset{viewed}{Score}(ft) = 1 \tag{3}$$

Merging method The weight applied to the source *Personal* is adjusted by the user. It is also necessary to give the user the control over the weight of the sub-source *Feature types viewed by user*. Recommendations from each of the sub-sources are merged to give the recommendation list of this source as follows:

$$\underset{personal}{Score}(ft) = \underset{used}{Score}(ft) + \underset{viewed}{w} \times \underset{viewed}{Score}(ft) \tag{4}$$

Where w_{viewed} is the weight assigned by the user to the evidence source *Feature types viewed by user*.

4.2 Global: Feature types most commonly used

Skipforward is a centralized system, where each user has a node in the network, and the information available on a node is limited to the user's roster (friends). A user's roster contains the users that share co-rated item(s). In this evidence source, the most popular feature types in the system are recommended. The score of a feature type is given as the frequency of usage of the feature type by all users.

$$\underset{global}{Score}(ft) = |\{f(u, i, ft) | f(u, i, ft) \in F, u \in U, i \in I\}| \tag{5}$$

This source offers high recall, which comes at the expense of precision. However, it is a good option in the case of new users with no or few previous annotation history. User-related evidence sources recommend existing popular feature types which can lead to the echo chamber effect [6] when popular tags are recommended thus becoming even more popular and it doesn't allow for distinction between items. For this reason, item-related evidence sources are introduced.

4.3 Web: feature types found in web

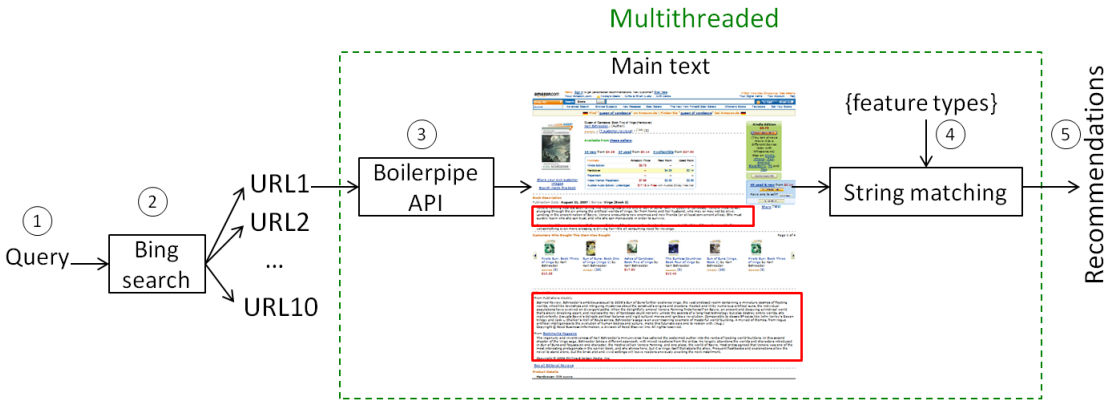


Fig. 3. Pipeline for extracting feature types recommendation list in *Web* evidence source.

The main aim of this source is to find feature types that are specific for an item. This is achieved by searching relevant web pages for the feature types available in the system. The sequence of operations held are shown in Figure [3] and are as follows:

1. A query that defines the item is constructed as the concatenation of item title and author (if available). Query = "Item title" {"Author"}$^+$
2. Bing search engine is queried and the first 10 URLs are found.
   Steps 3-4 are executed for each URL.
3. To improve precision, the main article in the web page is extracted using boilerpipe API. Boilerpipe offers several extraction strategies, KeepEverythingWithMinKWordsExtractor(20) is used which extracts all blocks of text with at least 20 words.
4. After crawling the web page and extracting the main content, case conversion and string matching are used to find the feature types occurring in the text. The matched feature types along with their number of occurrences are stored.
5. Feature types found in all web pages are merged to form the list of recommended feature types in this evidence source, where the score of a feature type is the number of its occurrences in all ten web pages.

$$\underset{web}{Score}(ft) = \sum_{i=1}^{10} |occurrences(ft, webPage_i)| \tag{6}$$

4.4 Co-occurrences

TV Tropes contains a vast and huge collection of tropes (idioms that describe items) covering a wide range of media, including television, film, literature, comics, video-games, tabletop games, music and sound effects, and theater. In this source, we utilize the rich database provided by TV Tropes, and find the tropes that co-occur. This source is based on the assumption that the more often two tropes are used in combination, the stronger the relationship between them is. The co-occurrence technique is one of the famous graph-based approaches, known from Social Network Analysis [49]. The idea here is to define a set of feature types that are very precise and related to an item, referred to as "seeds". This set is then expanded by finding the feature types that co-occur with the seeds in TV Tropes.

In order to obtain the final list of recommended feature types, this is done in a 3-step process: Defining the seeds, building the co-occurrence graph, then merging the feature types that co-occur with the seeds to get the final recommendation list.

Defining Seeds It is essential for the seeds to be specific for this item, so that the co-occurring feature types are also related to the item. Such item-related feature types are obtained from two sources:

1. Feature types previously annotated to the item with high applicability values.
2. Feature types found in the web which are obtained from the *Web* evidence source.

Co-occurrence graph A co-occurrence graph is built that represents the co-occurrences of the tropes (feature types) in TV Tropes. The graph is defined as an undirected graph $G = (V, E)$. V is the set of nodes in the graph, where each node corresponds to a feature type, and E is the set of edges, where $E(ft1, ft2)$ exists if both feature types co-occur in at least one item. The weight of each edge $w(ft1, ft2)$ is defined as the number of items in which the two feature types co-occur. A subset of the co-occurrence graph obtained from DBTropes is shown in Figure 4.

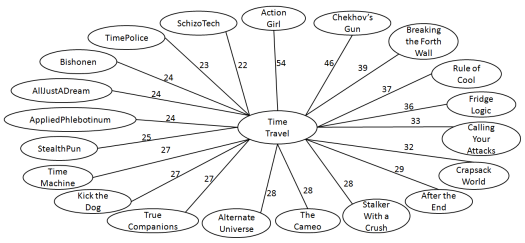


Fig. 4. A subset form the co-occurrence graph obtained from DBTropes.

Merging co-occurring feature types for all seeds After the seeds are obtained, and the final co-occurrence graph is built, the recommendation list for this evidence source is generated. The score given to a recommended feature type is equal to the total number of co-occurrences of that feature type with all the seeds.

$$\underset{co-occurrences}{Score}(ft) = \sum_{ft_{seed} \in seeds} (w(ft, ft_{seed})) \tag{7}$$

4.5 Aggregation methods

The *annotation recommender (item)* incorporates the output of the four individual evidence sources to generate a unified ranked recommendation list.

The definition of score assigned to the recommended feature types differs between the evidence sources. First, it is required to map the scores in recommendation lists given by individual sources such that they all have the same scale to be able to combine them. For this purpose two techniques are used, normalization and Borda count. Each technique is used separately and later evaluated. After that, the scores for each feature type across all evidence sources are summed with a weighting factor which is the evidence source weight, that is given by the user.

Normalization In this method, the scores of the feature types given by a source are normalized such that all the scores lie in the range $\{0, 1\}$, as shown in Figure 5. The normalized score is defined in equation (8), where *max* is the maximum score given to a feature type for that evidence source.

Feature Type	Score
Fantasy	20
Science Fiction	15
Shout Out	5

Feature Type	Normalization Score
Fantasy	1.0
Science Fiction	0.75
Shout Out	0.25

Fig. 5. Applying normalization on scores in recommendation lists.

$$NScore(ft) = \frac{Score(ft)}{max} \tag{8}$$

Borda count Borda count is a simple yet effective method for combining a ranked set of results (voting ranks). There exists several variants for Borda count [5]. The method used in this approach assigns descending consecutive integer scores to each element of each evidence source recommendation list as shown in Figure 6. The Borda count score of a feature type is defined in equation (9), where *size* is the size of the recommendation list for the evidence source and $Position(ft)$ is the index of the feature type ft in the recommendation list.

Feature Type	Score
Fantasy	20
Science Fiction	15
Shout Out	5

Feature Type	Borda Count Score
Fantasy	3
Science Fiction	2
Shout Out	1

Fig. 6. Applying Borda count on scores in recommendation lists.

$$BCScore(ft) = size - Position(ft) \tag{9}$$

Aggregation The final step is to combine the normalized or Borda count score for each feature type over all evidence sources. In this step, the weight of each evidence source, as assigned by the user, is taken into account when calculating the final score of a recommended feature type. The final score is calculated as follows:

$$\underset{final}{Score}(ft) = \begin{cases} \sum\limits_{evidences} weight_{evidence} * NScore(ft) \\ or \\ \sum\limits_{evidences} weight_{evidence} * BCScore(ft) \end{cases} \tag{10}$$

Fig. 7. Screenshot for *annotation recommender (item)*.

5 Annotation Recommender (expert)

In this section, the second part of the work is introduced, which is the *annotation recommender (expert)*. Skipforward, as well as other recommender systems using collaborative filtering, suffers from the sparsity problem when identifying similar users. The competence metric is a fundamental step in Skipforward. The accuracy of this metric is dependent on the amount of collaboratively available data in the system. It is not only dependent on the number of statements, but rather on the number of overlaps between users' statements. In this work, a recommender referred to as the *annotation recommender (expert)*, is implemented to tackle this problem. It aims at improving the performance of the competence metric by introducing more information on the extent of agreement between users. Not every annotation provided by a user would benefit the competence metric. This recommender searches for such information that would be of most usefulness when calculating the competence metric.

Consider the example shown in Figure 8, which shows some annotations for four items by the users (Malte, Florian, Gunner and Injy), for the feature types *Fast story pace*, *Bad writing*, *Action girl*, *Humor* and *Science fiction*. The annotations in a cell are given by the same order as the users shown in the lower right corner. For example, for *Ghost Moon*, the feature type *Fast story pace* is annotated by Injy and Florian, while *The Beautiful Land* is annotated by all users for *Fast story pace*. The colors of annotations shown are green, white and red, corresponding to applicability values of 1, 0, and -1 respectively. However for this recommender, the applicability values given to features do not affect the output.

In this example, assume Gunnar is the current user. There exist annotation overlaps between Injy, Gunnar and Florian regarding the feature type *Science fiction* in the items *Still Summer* and *Pearl in the Sand*. Skipforward will be able to calculate the similarity between Gunnar and Injy as well as between Gunnar and Florian regarding this feature type. Consequently, it will be able to calculate the extent to which Gunnar will identify the item *Ghost Moon* as *Science fiction*. However, it will fail to interpret the extent to which Gunnar would find the item *Ghost Moon* as *Humor*. Even though Injy and Florian provided their opinions, since there are no annotation overlaps between Gunnar and the other users regarding the feature type *Humor*. The question is which annotations, if provided by Gunnar, would provide the system with the most missing information and give the highest increase in overlap between Gunnar and the other users?

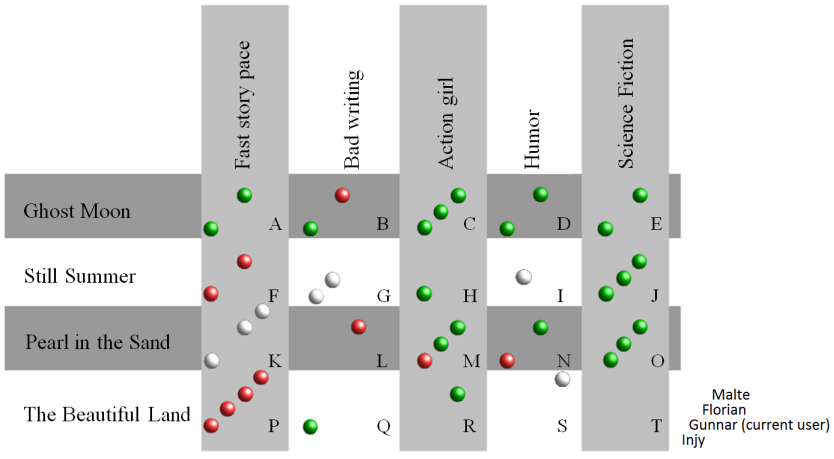


Fig. 8. Example of an *Item*×*Feature Type*×*User* matrix showing some user annotations.

5.1 Introducing the criteria

In order to identify the usefulness of annotating a feature, three criteria are defined. This is represented by three respective matrices, where each value in a matrix corresponds to the usefulness of annotating that feature in terms of the corresponding criterion. In this section, the criteria are introduced, and it is shown how to construct the matrices.

A. Feature missing correlation:
Feature missing correlation represents the number of users missing correlation with the current user for an item-feature type pair. *FeatureMissingCorrelation(Item i, feature type ft)* holds the number of users that previously annotated item i with the feature type ft and have no overlap with the current user for that feature type. For example, *FeatureMissingCorrelation(The Beautiful Land,Bad writing)* $= 0$ and *FeatureMissingCorrelation(Pearl in the Sand, Bad writing)* $= 1$. The matrix is built as follows:

$$FeatureMissingCorrelation(i, ft) = \Big|\{u|u \in U, f(u, i, ft) \in F, AO(\mathbf{uc}, u, ft) = 0\}\Big| \tag{11}$$

Where AO stands for AnnotationOverlap. $AO(\mathbf{uc}, u, ft)$ is the number of overlaps between users u and the current user **uc** for the feature type ft across all items.

B. Feature type missing correlation:
Feature type missing correlation identifies the feature types to which no correlation is known between the current user and all other users. *FeatureTypeMissingCorrelation(feature type ft)* is assigned a value of 1 if feature type ft contains no overlaps between the current user u and the other users' annotations, and 0 otherwise. For example *FeatureTypeMissingCorrelation(Bad writing)*$= 0$ and *FeatureTypeMissingCorrelation(Humor)*$= 1$. The matrix is built as follows:

$$FeatureTypeMissingCorrelation(ft) = \begin{cases} 1 & \text{If } \sum_{u \in U} AO(\mathbf{uc}, u, ft) = 0 \\ 0 & \text{Otherwise} \end{cases} \tag{12}$$

C. Overlap:
One direct way to calculate the increase in overlap would be the number of users that annotated that feature. For example for the feature F it would be 2, while for the feature K it would be 3. This method for calculating the change in overlap is not very accurate, which can be seen in the case of the features A and E. Both features if annotated would give an overlap with 2 users, Injy and Florian. However, since there exists more information about those two users regarding *Science fiction* than *Fast story pace*, the feature A should be preferred. In other words, the users having more known correlations should be penalized.
The *Overlap* matrix is built as follows:

$$.\text{Overlap}(i, ft) = \sum_{u \in U'} \frac{1}{2^{CoRatings(u,ft)}} \tag{13}$$

where U' denotes the set of users that annotated the item i with feature type ft, and $CoRatings(u, ft)$ is the number of overlaps between users u and **uc** for feature type ft.

$$CoRatings(u, ft) = \Big|\{f(u, i, ft)|i \in I, ft \in T_f, (u, i, ft) \in F, f(\mathbf{uc}, i, ft) \in F\}\Big| \tag{14}$$

The values given by *Overlap(item i, feature type ft)* are then mapped to integers. Each annotation is assigned a rank according to its overlap value, such that the annotation with the minimum value is assigned a rank of 0.

5.2 Aggregation

After the three matrices are built, their values need to be aggregated, as shown in Figure 9. In order to get the usefulness of annotating a feature, the three values corresponding to that feature in each matrix are aggregated, while keeping the priority within the criteria. For example, in Figure 8, the feature D is given a value 2 for the criterion *Feature missing correlation*, 1 for *Feature type missing correlation*, and 5 for *OverlapRank*, therefore the score for this feature would be 215. This is done using the following formula:

$$\begin{aligned} Score(i, ft) = &(10^x) * FeatureMissingCorrelation(i, ft) \\ &+ (10^y) * FeatureTypeMissingCorrelation(ft) \\ &+ (10^0) * Overlap(i, ft) \end{aligned}$$

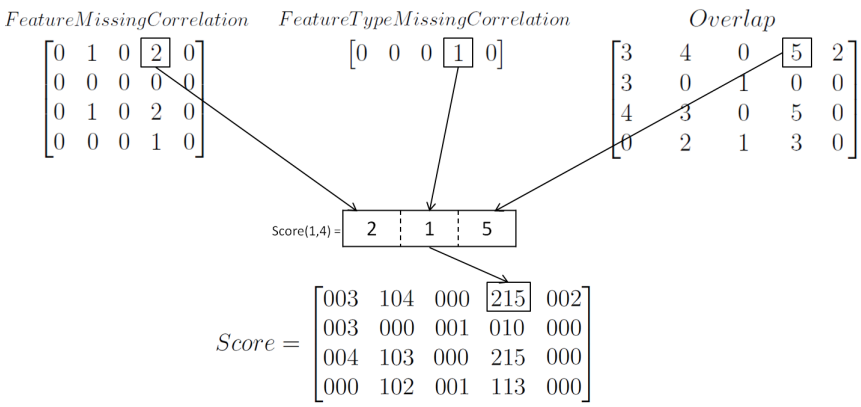


Fig. 9. Overview on approach.

y: the maximum number of digits in the cells of *Overlap*
x: the maximum number of digits in the cells of $FeatureMissingCorrelation + y$

6 Evaluation - Annotation recommender (item)

This section presents the evaluation done for the *annotation recommender (item)*.

6.1 Methodology

A survey was performed where participants were asked to view certain items and annotate some of the recommended feature types. The experiment included two user groups. The participants of user group 1 were familiar with Skipforward and were asked to annotate interzone stories which are short and non-popular. In user group 2, the participants were new to the system and were asked to annotate well-known novels. Feedback was collected from the participants using 3 sources: (1) the applicability values assigned to the annotated feature types. (2) A four-point scale (very good, good, don't know and not good) used by the user to evaluate the relevance of each recommended feature type that is annotated. (3) A questionnaire. The aim of the experiment was to evaluate the overall performance of the recommender with different kinds of users and items. The evaluation also included a comparison between both aggregation methods, normalization and Borda count. Moreover, the performance of the individual evidence sources was also held in comparison.

6.2 Results

Overall performance To evaluate the overall performance of the recommender, three metrics are adopted that reflect the performance at different aspects. For the three metrics, "good recommendations" are defined as those which are annotated by the user and given a judgement of "good" or "very good" on the four-point scale. The three metrics are defined as follows:

- Mean Reciprocal Rank (MRR): MRR measures the rank of the first feature type to be considered as a "good recommendation", averaged over all annotations.
- Success at rank k (S@K): S@K provides the probability that a "good recommendation" is found among the first k-ranked recommendations. In this evaluation S@1 and S@5 are measured.

– Precision at rank k (P@K): P@K is defined as the proportion of the first k-ranked recommendations that are considered to be "good recommendations". In this evaluation, P@5 is used.

Evaluation results for the three metrics for both aggregation methods are shown in Table 2. The MRR shows that a "good recommendation" is found in the first rank for most of the cases. In terms of S@k, S@1 shows that for the vast majority of recommendations, a "good recommendation" is found in the first rank. S@5 shows that the participants always find at least one relevant feature type in the first 5 ranked recommended feature types. The third metric, P@5 shows that nearly 50% of the first 5-ranked feature types are considered to be "good recommendation". Both aggergataion methods give nearly the same results.

	Normalization	**Borda count**
MRR	1.073	1.146
S@1	97.6%	95.1%
S@5	100%	100%
User group 1		
P@5	52.6%	52.7%
User group 2		
P@5	57.2%	61.9%

Table 2. Shows the results of the three metrics.

The overall performance of both aggregation methods is further investigated. The recommendations given "very good" on the four-point scale and annotated with an applicability value not less than 0.5 were examined. Out of the total 681 annotations provided by the participants throughout the experiment, the number of such recommendations is 207, which is equivalent to 30.4%.

Normalization Vs. Borda count In order to further compare the performance of both aggregation methods, another measurement is introduced. The recommendations given "very good" on the four-point scale are examined. The mean and median ranks of those recommendations are calculated. Results , as presented in Table3 , show that normalization gives slightly better performance.

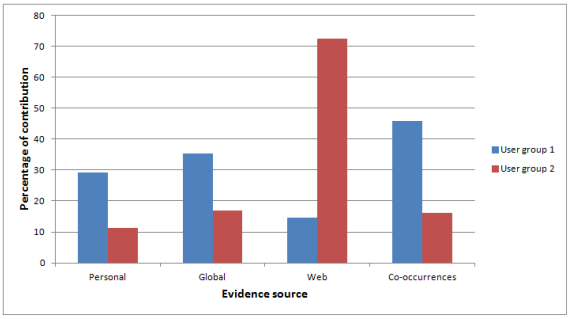
	Normalization	**Borda count**
mean	11.06	13.87
median	9	11

Table 3. Shows the mean and median ranks for the recommendations given a "very good" feedback by users in the four-point scale, as obtained by normalization and Borda count.

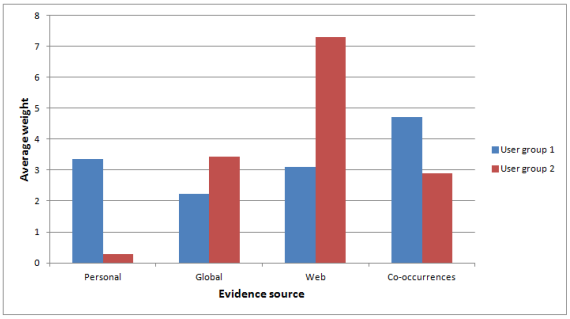
Individual evidence sources' performance The previous evaluations give an assessment on the performance of all four evidence sources' recommendations combined. In this evaluation, the recommendations given "very good" are examined and the percent to which each evidence source contributes to such recommendations is calculated. Figure 10(a) shows the results.

A second method for evaluating the performance of each of the individual evidence source is to inspect the weight values assigned to each by the participants. Figure 10(b) shows the average weight assigned to each evidence source by the participants from each of the two user groups.

It was shown that the performance of each evidence source is dependent on several factors including the type of user and the item being viewed. Firstly, the *Web* evidence source is highly dependent on the popularity of the item. It was given a much better feedback from user group 2 who were asked to annotate well-known novels than user group 1 who annotated non-popular short stories. Secondly, the performance of the *Personal* evidence source is greatly limited by the user's annotation history. It is



(a) The plot shows the percentage of contribution of each evidence source in the recommendations given "very good" on the four-point scale for the two user groups.



(b) The plot shows the average weight values assigned to the evidence sources by the participants from each user group.

Fig. 10. Evaluations for performance of individual evidence sources.

shown that the *Personal* evidence source received a better feedback from user group 1. Thirdly, the performance of the *Co-occurrence* evidence source is dependent of the availability of "seeds" to which the co-occurring feature types are obtained.

Questionnaire results The questionnaire included several sections each providing user feedback on different aspects of the annotation recommender. Users were asked to answer the questions on a scale 1-5. The ratings given by users to some of the questions are shown in Table 4.

	Average		
Question	**User group 1**	**User group 2**	**Overall**
The recommender helps you annotate items easier	3.00	4.40	3.88
The recommended feature types affect how you describe an item	4.33	4.80	4.63
Explanations given to recommended feature types are clear	3.67	3.80	3.75
The usefulness of controlling individual evidences' weights in improving recommendation	3.00	3.80	3.50
Overall, I am satisfied with the tag recommender	3.00	3.80	3.50
I am inclined to use this recommender again when annotating items	4.00	3.40	3.625

Table 4. Shows the ratings given by users to some of the questionnaire questions.

7 Evaluation - Annotation recommender (expert)

This section presents the evaluation done for the *annotation recommender (expert)*. In this section, the ability of the *annotation recommender (expert)* to provide good recommendations in the correct order will be evaluated. The purpose of this recommender is to increase the overlap between the annotations of the current user and other users. For any two users u_x and u_y, the more the number of overlapping annotations between them is, the more accurately the system can calculate the similarity between them, $sim(u_x, u_y)$ in equation (1). The accuracy of that calculation is given by the $conf_{sim}(u_x, u_y)$ value, which is defined in equation (2).In other words, $conf_{sim}(u_x, u_y)$ serves as an indicator to the amount of overlap in annotations between both users for a certain feature type.

In order to assess the performance of the *annotation recommender (expert)*, the total change in $conf_{sim}$ values resulting from the current user annotating each of the recommended features is observed.

The total change in $conf_{sim}(u_x, u_y)$ values, referred to as Δ_{conf},is calculated as the summation of the change in $conf_{sim}(\mathbf{uc}, u')$ between the current user and all other users. Δ_{conf} can be formally defined as:

$$\Delta_{conf} = \sum_{u \in U} conf'_{sim}(\mathbf{uc}, u) - conf_{sim}(\mathbf{uc}, u) \tag{15}$$

Where $conf_{sim}$ and $conf'_{sim}$ are the confidence in similarities values defined before and after the user annotated a feature.

The Δ_{conf} value is calculated against each of the recommended features. The first recommendation should result in the highest Δ_{conf} value, followed by the second recommendation and so on. For each of the three test cases shown in Figure 7, the recommendation list provided by the *annotation recommender (expert)* is obtained and the Δ_{conf} value is calculated for each of the recommended features. Table 5 shows the Δ_{conf} values of each recommended feature ordered in ascending order of their rank for the three test cases. Rank 0 is occupied by the first feature recommended.

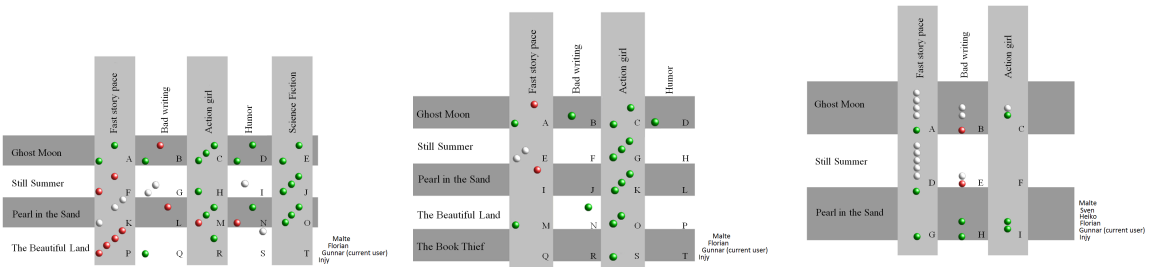


Fig. 11. Annotation recommender (expert) evaluation test cases.

Rank	Feature label	Δ_{conf}
0	D	0.80
1	N	0.80
2	S	0.50
3	B	0.57
4	L	0.40
5	K	0.48
6	F	0.34
7	A	0.34
8	E	0.11
9	P	0.17
10	R	0.06
11	H	0.06

Rank	Feature label	Δ_{conf}
0	K	0.50
1	A	0.57
2	G	0.40
3	J	0.17
4	C	0.12

Rank	Feature label	Δ_{conf}
0	B	1.07
1	G	0.90
2	A	0.93
3	H	0.57

Table 5. Shows the list of recommended features and the Δ_{conf} values resulting from the current user annotating each feature for the three test cases, respectively.

8 Conclusion and Future Work

In this paper, we have introduced two goal-oriented annotation recommenders. The *annotation recommender (item)*, aims at helping and guiding the user in the process of annotation by suggesting a list of feature types that are both user- and item-related. Results show that the *annotation recommender*

(item) is successful at providing the user with relevant feature types and would be a useful asset for users annotating items. It was also shown that when comparing the quality of recommendations provided by normalization and Borda count, no significant difference was observed. However, normalization gave slightly better results. The performance of the individual evidence sources was dependant on the type of users and items.

The second recommender, the *annotation recommender (expert)* aims at asking the user to annotate some item-feature type pairs that would help the system better calculate users' similarities in regards to a certain feature type. It was shown that the system was able to provide a correct ranking of the features according to their usefulness. As the rank of a feature decreases, so does its Δ_{conf}. This is proof that the *annotation recommender (expert)* recommends features to the users that, when annotated, would improve the accuracy of similarity calculation done by the system. Consequently, this recommender can improve the item recommendations provided by collaborative approaches in general.

Several issues should be considered for future improvement. The following ideas can be applied to the *annotation recommender (item)* to improve its performance. Firstly, the performance of the *Web* evidence source can be further improved by backing the algorithm with stemming and a thesaurus such as WordNet. Several web services are available that offer information on resources such as tags and reviews. For example LibraryThing[7] can be used for books and Internet Movie Database[8] for movies. Such specific sites that are targeted at the type of resource under consideration can be searched for feature types. Secondly, TV Tropes provides a textual description on the resources and the tropes. This can be utilized by machine learning techniques to increase recall. The descriptions of the resources that apply to each feature type can be used to train a machine learning algorithm to automatically recognize the text obtained from the web pages and relate it to the correct feature types. Finally, additional evidence sources can be added to explore different facets of the system. For example, Feature types that partition the items best can be recommended. Tag non-obviousness and discrimination are discussed in [3].

References

1. Kiesel, M., and Mittag, F. Personalization in Skipforward, an Ontology-Based Distributed Annotation System. In: Marco de Gemmis, Ernesto William De Luca, Tommaso Di Noia, Aldo Gangemi, Michael Hausenblas, Pasquale Lops, Thomas Lukasiewicz, Till Plumbaum, and Giovanni Semeraro, editors, Proceedings of the Second Workshop on Semantic Personalized Information Management: Retrieval and Recommendation. Workshop on Semantic Personalized Information Management (SPIM-11), 2nd, located at International Semantic Web Conference, October 23-27, Bonn, Germany (2011)
2. Kiesel, M., and Grimnes, G. A. DBTropes—a linked data wrapper approach incorporating community feedback. In Johanna Volker; Oscar Corcho, editor, EKAW 2010 Demo and Poster Abstracts. International Conference on Knowledge Engineering and Knowledge Management (EKAW-10), 17th International Conference on Knowledge Engineering and Knowledge Management, Lisbon, Portugal (2010)
3. Farooq, U., Kannampallil, T. G., Song, Y., Ganoe, C. H., Carroll, J. M., and Giles, L. Evaluating Tagging Behavior in Social Bookmarking Systems: Metrics and design heuristics. In Proceedings of the 2007 international ACM conference on Supporting group work, GROUP '07, pages 351—360, New York, NY, USA (2007)
4. Peis, E., del Castillo, J. M. M., and Delgado-Lopez, J. A. Semantic recommender systems. analysis of the state of the topic. online (2008)
5. Erp van, M., and Schomaker, L. Variants Of The Borda Count Method For Combining Ranked Classifier Hypotheses. In the seventh international workshop on frontiers in handwriting recognition. Amsterdam learning methodology inspired by human intelligence, pages 443—452 (2000)
6. Jamieson, K. H., and Cappella, J. N. Echo Chamber: Rush Limbaugh and the Conservative Media Establishment. Oxford University Press (2009)

[7] http://www.librarything.com/
[8] http://www.imdb.com/