Hybrid Reciprocal Recommender Systems: Integrating Item-to-User Principles in Reciprocal Recommendation

James Neve
Computer Science Department
University of Bristol
Bristol, United Kingdom
james.neve@bristol.ac.uk

Iván Palomares
Andalusian Research Institute on Data Science and Computational Intelligence (DaSCI), University of Granada
Spain
ivanpc@ugr.es

ABSTRACT

Reciprocal Recommender Systems (RRS) recommend users to other users in a personalised manner, in scenarios where both sides of the preference relation must be considered. Existing RRS approaches based on collaborative filtering or content-based filtering, have been used for enhancing user experience in online dating and other online services aimed at connecting users with each other. However, some of these services e.g. skill sharing platforms, are still pervaded by content published, shared and consumed by users, consequently there is a valuable source of item-to-user preferential information not captured by existing RRS models. We present a novel hybrid RRS framework that integrates user preferences towards content in reciprocal recommendation, and we instantiate and evaluate it using data from Cookpad, a recipe sharing social media platform. As part of our model, we also implement a novel content-based extension of Jaccard similarity measure that operates on word embeddings.

KEYWORDS

Reciprocal Recommender Systems, Latent Factor Models, Word Embeddings

ACM Reference Format:
James Neve and Iván Palomares. 2020. Hybrid Reciprocal Recommender Systems: Integrating Item-to-User Principles in Reciprocal Recommendation. In *Proceedings of The 2020 ACM Web Conference, Taipei, Taiwan, 2020.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3366424.3383295

1 INTRODUCTION

Recommender Systems (RS) are personalisation [2] tools that identify suitable or desirable items for users based on information gathered by the system about them. They were first used on online shopping and movie watching web services. The scientific basis for these algorithms has grown significantly since the release of open data sets such as the *Netflix* and *MovieLens* datasets [6]. RS are broadly divided into a number of categories, with two of them being widely investigated. 1) *Content-Based Filtering* (CB) uses explicit or implicit information about the user's preferences and item's properties to make recommendations, whereas 2) *Collaborative Filtering* (CF) uses correlations between similar users to this end [21]. CF is often more generalisable and allows for recommending items with more diverse properties [2, 13]. However, CF suffers from the *Cold Start Problem* [20], where new users to the system who expressed few or no preferences cannot be given meaningful recommendations. CB approaches can make recommendations immediately based on the information given by the user. These two techniques are therefore most effective when used together. Indeed, various prizewinning RS algorithms in contests such as the *Netflix Prize* are usually hybrid systems based on CB and CF [6].

In *Reciprocal Recommender Systems* (RRSs), users are recommended to each other, therefore unlike classical RS where preference relations are unidirectional (user-to-item), in RRS preference relations among pairs of users need to be considered. RRSs are most often used in online dating websites [10] [14], where explicit indicators of positive and negative preference are gathered from users. However, they are noticeably having emergent applications in areas such as recruiting [22] and social networks [9].

Despite ongoing RRS primarily focus their application on online dating, different algorithms may be effective on different types of online services for connecting users. For instance, a fundamental characteristic in online dating websites is that they often have two distinct classes of user - male and female - whereas social websites such as *Twitter* and *Facebook* have only a single class of user where any one user can be recommended to any other. To the best of our knowledge, there is a clear shortfall of research on single-class RRSs as of now. Moreover, the rise of skill sharing and social platforms such as *Meetup.com*, in which contents published and shared among users play an important role, raises the need for new or extended RRS models that accommodate user-user recommendations in these scenarios. Extant RRS research in general lags a long way behind traditional RS research, with broad areas such as hybrid RRSs still completely unexplored in a reciprocal context.

To address the aforesaid challenges, in this paper we present a novel Hybrid RRS (HRRS) framework for recommending users to connect with each other socially in content/skill sharing platforms where: (i) unlike most online dating services, there exists a single class of users and (ii) both user-to-item and user-to-user preference information coexist. We propose a HRRS model that employs a CF-based RRS algorithm based on latent factor models recently proposed in [1], and combines it with classical RS principles relying on users' preferences towards content. For this, a novel method that exploits free text content information using word embeddings is also introduced, for calculating similarities between users predicated on their implicit preferences towards content. We evaluate the results of using our proposed HRRS model on *Cookpad*, a popular recipe sharing website in countries such as Japan, Taiwan and Indonesia.

WebConf'20, April, 2020, Taipei, Taiwan

ACM ISBN 978-1-4503-7024-0/20/04.
https://doi.org/10.1145/3366424.3383295

The contributions of this paper are fourfold. (1) The first hybrid RRS framework and model in the literature, combining reciprocal CF with principles from classical CF and CB. Importantly, many item-to-user RS services use hybrid techniques to produce better and more robust recommendations, but these techniques have not been yet explored in the field of reciprocal recommendation. (2) Our system is also the first RRS model in the literature that operates on a single class of users, i.e. recommending users to each other within the same class, unlike e.g. opposite-sex online dating. (3) A novel similarity metric based on word embeddings modeled after free text information associated to content shared and/or liked by users. (4) A preliminary offline evaluation that includes an experimental study on real data, with a real time implementation of the algorithm. It has been demonstrated that standard RS metrics such as precision and recall based on cross-validation are not always representative of the real effectiveness of RRS.

2 RELATED WORK

In this section, we overview the state-of-the-art advances made on RRS research, broadly divided into CB and CF-based approaches.

In general, CB filtering attempts to match either explicit or implicit preferences of users with the properties of items [2] [5]. However, in many modern systems, item properties are present only in freetext. In this case, word embeddings have been used in a number of content-based user-item recommender systems to represent and deal with the freetext properties of items e.g. [8]. CF has seen a large number of applications in recommender systems. Early systems were based on nearest neighbours with respect to correlations decided by correlation coefficients such as *Pearson-Cosine* [6]. Modern recommender systems are often based on matrix factorisation [15], known as *Latent Factor Models*, have been successfully deployed into numerous practical applications [21].

One of the earliest attempts at building RRS is RECON, a content-based algorithm that determines the preferences of users implicitly [11]. It bases its recommendations on the attributes of users whom they have indicated preference for in the past. It then recommends users with the desired attributes being sought. To the best of our knowledge, no RRS models have been designed based on word embeddings nor incorporating information about content (items) associated with users. The earliest CF-based RRS is RCF [14]. Designed with online dating in mind, RCF compares the preferences of users of one gender with other users of the same gender who have shown preference for the same users of opposite gender, and uses these implicit preferences to make recommendations. RCF performed well on cross-validation and has subsequently been used as the baseline for several other RRSs [3]. More recently, latent factor models were applied to CF-based RRSs, with promising results [5].

Hybrid recommender system approaches attempt to combine the strengths of more than one class of recommender system, which might include content-based, collaborative, knowledge-based etc. In practice, hybrid systems most commonly combine content-based and collaborative filtering, often outperforming both approaches compared to their performance in isolation [7]. In particular, collaborative filtering by itself often suffers from the Cold Start Problem [20], where users with no history of preference indication cannot be given accurate recommendations. Combining elements of content-based filtering until users have indicated enough preferences that collaborative filtering can be used is a common way to address this problem [4]. Despite the successful efforts in applying CB and CF to RRSs, there are still no studies on incorporating hybrid approaches in user-to-user domains. This is an important aspect to consider, because some platforms have rich relevant information about both user actions and content.

3 HYBRID SINGLE-CLASS RECIPROCAL RECOMMENDATION

This section firstly introduces a novel HRRS framework. It then describes our proposed HRRS algorithm, characterised by:

(1) Incorporating principles from classical item recommendation in the process of recommending users to each other. Despite the numerous hybrid CF-CB models devised in the literature for *item-to-user* recommendation, this is the first hybrid model of its kind in a *user-to-user* setting.
(2) Calculating reciprocal preferences among pairs of users who belong to the same class. This is a notable difference from most existing RRS algorithms that rely on predicting matching scores on pairs of users belonging to two classes, e.g. male and female in an online dating domain.

A novel framework for Hybrid RRS or HRRS, where both inter-user preferences and user-to-content preferences coexist, is formulated as follows:

- There exist a set of users U in the system, with $a, b \in U$ denoting any two users. We consider a framework where, unlike opposite sex online dating, all users belong to the same class and therefore any two users in U can be potentially recommended to each other.
- There is a set of content items X. In skills sharing platforms, items $r_a \in X$ are associated to a user a who published them, hence preferences towards such content can be taken as an indicator of potential preference from one user to another.
- There exist indicators of user-user preference $Pref(a \rightarrow b)$, e.g. likes or follows towards users.
- There exist indicators of user-item preference $Pref(a \rightarrow r_b)$, e.g. liked on content posted by other users.
- The HRRS recommendation problem consists in recommending users b to a target user a taking both types of preference indicators, $Pref(a \rightarrow b)$ and $Pref(a \rightarrow r_b)$, into consideration.

Figure 1 shows the general pipeline followed by our model to predict the level of matching between two users a and b, consisting of three main stages: (i) item-to-user based matching or *non-reciprocal* matching, (ii) reciprocal CF matching, and (iii) aggregation of item-to-user and reciprocal predicted matching scores. Our model also introduces in (i) a novel extension of the Jaccard index formula to calculate pairwise similarities between users' preferences on content shared among users, predicated on word embeddings associated with content descriptions.

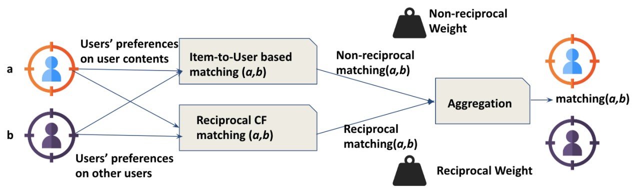

Figure 1: General scheme of the HRRS Model

3.1 Item-to-User (Non-reciprocal) Matching

This component of our model considers users' preferences towards content published by other users and then we calculate pairwise user similarities based on content liked by both users. Without loss of generality, in the application domain of Web skill-sharing platform considered in our study (Sections 4 and 5), the basic unit of *content* posted by a user $a \in U$ is a recipe $r_a \in R$, Therefore, our aim is to assign a higher matching to pairs of users whose preferences on content items, i.e. recipes, are similar. Preference-based similarities among users are known to be challenging to calculate in domains where only implicit rating information (e.g. liked or seen items) are available. A classical solution for this is to identify recipes commonly liked by two users a and b, and use the Jaccard Index to calculate their similarity:

$$\frac{R_a \cap R_b}{R_a \cup R_b}$$

with $R_a, R_b \subset R$ the subsets of recipe items liked by a and b, respectively. This presents however an important limitation in skillsharing platforms where many instances of content published by different users can be highly similar to each other, because the Jaccard index only detects co-occurrences of *same* items in any two users' preferences. Consider for instance that user a liked four types of risotto recipes, and user b liked another four risotto recipes different from those liked by a. If R_a and R_b contain only these risotto recipes for each user, their Jaccard similarity would be zero (no risotto recipes in common). Users who have liked similar recipes but none of the same recipes would be identified as being dissimilar to each other, which is undesirable in a content-based system. We therefore introduce a modified form of Jaccard similarity to account for similarity between non-identical content items that users may have liked. In essence, we integrate a non-identical content similarity measure into a user-user similarity metric in terms of their preferences towards content.

In order to quantify recipes and thus calculate their similarity, we use Word2Vec [18]. Word2Vec is a series of models trained on shallow neural networks that produce word embeddings such that words with similar meanings are represented by vectors with a short Euclidean distance between them. The training is based on proximity between words in large document corpuses such as Wikipedia [16]. Many pre-trained word embeddings are available, and for this project we used the *Google News Vectors*[1], which contain 300-dimensional vectors representing a dictionary of 3 billion words, and have been tried and tested in a number of previous projects [19]. We then modify the Jaccard Index, as follows. We introduce an adjustment term, to smooth the (typically pessimistic) similarity degree obtained by the classic Jaccard Index.

$$\frac{|R_a \cap R_b| + \lambda}{|R_a \cup R_b| + \mu} \tag{1}$$

where,

$$\lambda = \sum_{r_a \in R_a - R_b} \sum_{r_b \in R_b - R_a} \delta(r_a, r_b) \tag{2}$$

$$\mu = |R_a - R_b| \cdot |R_b - R_a| \tag{3}$$

and λ is a sum over soft (non-strict) similarities $\delta(r_a, r_b)$ between non-identical pairs of recipes r_a, r_b found in R_a or R_b, respectively, but not in both. μ is the total number of such recipe pairs. Let $|r_a|$ be the number of vector representations of words associated to recipe r_a (obtained for instance by applying a *word2vec* algorithm). Then, by looking at pairs of word vectors in r_a and r_b, we calculate a similarity degree between these two recipes as follows:

$$\delta(r_a, r_b) = \sum_{l=1}^{|r_a|} \sum_{k=l+1}^{|r_b|} sum(w_l, w_k) \tag{4}$$

assuming we chose r_a and r_b such that $|r_a| \leq |r_b|$. Here, $w_l \in r_a$ and $w_k \in r_b$ are vector representations of words present in both recipes, e.g. ingredients in common, and *sim* is a vector similarity metric between both vectors.

3.2 Reciprocal Matching

RRS approaches normally rely on indicators of preference between users. In the case of the Cookpad skillsharing platform, preference indicator data consist of *Follows*. Users can follow other users in order to be notified of their new recipe content whenever they post it. The data also contains a number of indirect preference indicators, such as *Bookmarks* (users can bookmark others' recipes to find them easily in future) and *Cooksnaps* (when a user makes another user's recipe, they can post their results along with a short review). We decided to use *Follows* $F(a, b)$ and *Bookmarks* $B(a, b)$ to construct a unidirectional preference score that represents a user a's preference for a user b, $P(a, b)$, as follows:

$$P(a, b) = F(a, b) + \sum B(a, b) \tag{5}$$

With this, we could construct a two-dimensional square preference matrix representing the preference of every user for every other user, and use this as the core of our reciprocal matching part.

The reciprocal matching process relies on two indicators of preference associated with each user a:

- Followed users by a.
- Users $b \neq a$ whose associated content has been liked by a.

Let $U_{m \times N}$ and $V_{m \times N}$ be two matrices. Each row in U, denoted $u_a = (u_{a1}\ u_{a2}\ \ldots\ u_{aN})$ contains N preference latent factors associated with user a, which represent what a likes. Each row in V, denoted $v_a = (v_{a1}\ v_{a2}\ \ldots\ v_{aN})$ contains N attribute latent factors associated with user a, which represents the properties of a. Both matrices have been obtained by applying a Stochastic Gradient Descent (SGD) algorithm to reduce the dimensionality of the original $|U| \times |U|$ matrices built upon the aforesaid preference indicators. Calculating the preference or affinity level from user a

[1]https://code.google.com/archive/p/word2vec/

towards b, boils down to computing the similarity between a's preferences and b's attributes. Concretely, a unidirectional preference score from a to b is determined by applying the vector product, $p(a \rightarrow b) = u_a \cdot v_b^T$. Conversely, the preference score from b to a is similarly determined as $p(b \rightarrow a) = u_b \cdot v_a^T$. Both unidiretional preference scores are combined into a reciprocal preference score or matching score $m_{CF}(a, b) \in [0, 1]$ using the harmonic mean operator [11, 14]:

$$m(a,b) = \frac{2\, p(a \rightarrow b)\, p(b \rightarrow a)}{p(a \rightarrow b) + p(b \rightarrow a)} \tag{6}$$

It is noteworthy that the harmonic mean operator has been typically used in previous user-to-user recommendation approaches to calculate reciprocal preference scores, due to its tendency to generate a lower aggregation output when none of the inputs are high enough, compared to other classical mean operators. This is convenient in reciprocal recommendation domains where a match between two users should be identified only when both users have potential preference towards each other to some extent.

3.3 Aggregation of User Matchings

The outputs of the non-reciprocal and reciprocal matching processes are finally aggregated into an overall matching between a pair of users a and b. Without loss of generality, in this work we propose using a weighted average for this aggregation (within the scope of the experiments in this paper we consider equal weights). Notwithstanding, a recent study on the effect of using other averaging and mixed behavior aggregation operators in RRS can be accessed in [12], and we plan to investigate the effect of these aggregation strategies in HRRS approaches for our future work.

4 DATASET

In this section, we describe the dataset used for the experiments and evaluation of our HRRS model's effectiveness.

The data for validating our system was provided by the international recipe sharing website Cookpad Inc., based in Japan [2]. On Cookpad, users share recipes with titles, pictures and textual information describing the ingredients and descriptions of those recipes. Other users can demonstrate their results when making those recipes via "Cooksnaps", which are mini reviews of the recipes with a picture of their own results.

Users have a number of ways for indicating preference for each other on the site. They can *Follow* each other, where the follower is notified of the followee's public actions. They can also bookmark other users' recipes (an implicit indicator of preference used in the non-reciprocal part of your HRSS approach) and send messages to each other. Users and recipes share very little information about themselves in quantifiable form - for instance, users do not give their age, nationality or explicit preferences such as ratings on other users' recipes, therefore only implicit preferences are used. However, the recipes shared and bookmarked by users in the form of a title, list of ingredients and steps for making the dish, provide a wealth of freetext information about the users taste.

Cookpad's data is quite different to, and in many ways more complex than, the data from a dating service, where users often have a list of attributes and demonstrate clear, direct preferences. However, we feel that this data is more representative of many general social networks - including skillsharing platforms - than online dating sites in two ways:

(1) The data includes only a single class of users who have to be matched with each other. Dating site data is generally divided into two distinct classifications (male and female), and to the best of our knowledge, no research work has been done on RRSs for single sex dating as of yet.
(2) Most of the attribute data for users is unstructured freetext data, as opposed to well structured datasets that have been used in existing RRS models.

5 EXPERIMENTAL EVALUATION

5.1 Evaluation Metrics

The evaluation of our model effectiveness was performed offline. There were two reasons for this. Firstly, this is the first implementation of a HRRS algorithm for connecting users to each other in skill-sharing services, therefore we considered important to firstly test its effectiveness without taking the risk of exposing real users to any possibly unsatisfactory results. The second reason is that offline evaluation allowed us to run our tests more quickly and gather a wealth of data on different options. However, there is some evidence that offline evaluation is not always representative of real user behaviour or some niches of them [13]. For this reason, we intend to follow up with a user study based on real users in our immediate future work.

We ran cross-validation on a sample containing 500 pairs of users, of which 250 pairs had already indicated preference for each other via *Follows*, and the other 250 pairs had shown no specific preference for each other. It is worth noting that Cookpad had no indicators of negative preference, and that a lack of preference information is not necessarily an indicator of negative preference. The effectiveness of the algorithm may therefore not be as clearly evident in the results as if it had been performed with data containing negative preference indications.

The offline evaluation was performed using a threshold, where a result above the theshold is considered an positive recommendation i.e. a match between two users, and a score below that threshold is considered no recommendation. This threshold $\mu \in]1, 0[$ represents the level of bidirectional preference required to make a recommendation. The closer this threshold is to zero, the larger the absolute number of recommendations made, and therefore the higher the expected recall and lower the expected precision is to be. By varying the threshold, we can model the recommendation as a binary classification problem and therefore draw an ROC curve by considering the number of true and false positives and negatives at each threshold.

In order to show the effectiveness of the hybrid algorithm when compared against a non hybrid strategy, we evaluated the non-reciprocal and reciprocal algorithms individually, and subsequently evaluated the hybrid version that uses both of them. We ran the evaluation on the same pairs of users for all three algorithms and different prediction prediction thresholds between zero and one.

It has been demonstrated that accuracy is not always a satisfactory metric for offline evaluation of RSs [13]. We therefore focus on

[2] https://cookpad.com/

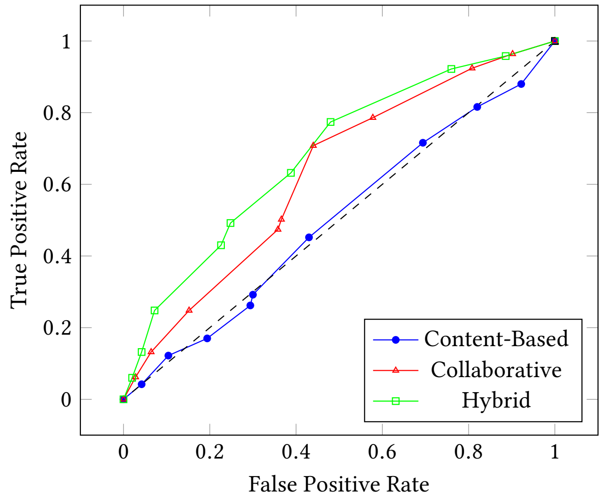

Figure 2: ROC curve obtained for the content-based, collaborative and hybrid models.

using precision, recall and F1 score as our main evaluation metrics. Precision is the proportion of true positives relative to the total number of positives. Where TP is the number of true positives, and FP is the number of false positives, precision is defined as

$$Precision = \frac{TP}{TP + FP} \tag{7}$$

A high precision is critical for a successful RS, as it is important that a user sees a high proportion of desirable items in their recommendation list. We also look at recall, which is the quantity of positive recommendations as a fraction of all positive items (i.e. pairs of users with positive preference indicators) defined as:

$$Recall = \frac{TP}{TP + FN} \tag{8}$$

Finally, F1 score is a machine learning metric that represents the balance of precision and recall, defined as the harmonic mean of the two:

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} \tag{9}$$

5.2 Results and Discussion

We evaluated the two components of our HRRS model individually, and then evaluated the hybrid model. The ROC curve for each of the three models is shown in Figure 2. The neutral 0-1 line is also given on the figure as a dashed black line for reference. As there are no other examples of RRSs used in the context of a recipe sharing service, or on a single class of users, we did not feel that it was appropriate to evaluate the system against other current RRS, which were designed for online dating.

The non-reciprocal algorithm where similarities among users are calculated upon user-content preference information, performed relatively poorly by itself, only slightly better than random filtering for some threshold setting. This was largely a result of a large number of false positives which in turn leads to a lower precision and F1 score. As intuitively, users with similar terms in their liked and created recipes would be more likely to like each other, we attribute this to the offline testing procedure. As discussed, the data provided us with no negative preference indicators among users, and so we used users who had not shown preference indicators towards each other as our 'negative' test.

In contrast, the reciprocal algorithm where both directions of preference between users are analysed, produced good results, with a positive ROC curve. The improvement seen in the hybrid model that incorporates the (poorly performing by itself) non-reciprocal algorithm, in spite of its neutral ROC curve, is not surprising. The reciprocal matching algorithm's positive results are able to be further refined by the non-reciprocal counterpart, with significantly dissimilar users in terms of their created content being filtered out by the reciprocal process.

Threshold	Rec. F1 Score	NonRec. F1 Score	HRRS F1 Score
0	**0.666**	**0.666**	**0.666**
0.1	0.672	0.628	**0.673**
0.2	0.676	0.619	**0.687**
0.3	0.664	0.594	**0.686**
0.4	**0.659**	0.480	0.625
0.5	0.537	0.366	**0.565**
0.6	0.517	0.336	**0.519**
0.7	0.354	0.249	**0.375**
0.8	0.220	0.199	**0.224**
0.9	**0.113**	0.077	0.111
1.0	0	0	0

Table 1: Results obtained by varying the threshold for bidirectional preference score-based recommendation

For reference, the F1 scores for each of the model versions is displayed in the table. The highest F1 score occurred at a threshold of 0.2. However, as is evident from the ROC curve, the precision of the system is significantly higher at thresholds closer to 0.8. As RS approaches normally value precision more highly than recall (users are more likely to trust a system with a few positive recommendations), higher thresholds may be more beneficial to a live system.

Based on the cross validation experimental study conducted, we consider that the precision and recall of our proposed hybrid approach is high enough to convince us that it will produce good recommendations for end users, and that the hybrid system is more useful than either the non-reciprocal or the purely reciprocal algorithm by itself. However, we feel that due to the lack of negative data for cross-validation, and the complexity of the recommendation domain in general, this should be considered as a promising yet preliminary result only, and that further real time testing is required to determine the system's effectiveness.

6 CONCLUSIONS AND FUTURE WORK

The main goal of this work was to create an effective hybrid reciprocal recommender system for a social networking service focused on skill sharing, and demonstrate its effectiveness on a representative dataset. We successfully designed and developed this system, and demonstrated through cross-validation that the hybrid system was more effective than either the item-to-user based or the purely reciprocal collaborative filtering systems by themselves. As iterated in the introduction, this work is novel in a number of ways, most significantly that it is the first hybrid reciprocal recommender system, and the first RRS to operate on a single class of users.

Cross-validation is an effective method of evaluating the results, but it has been demonstrated that it is not always representative of the system's use in real world systems [13]. We therefore aim to conduct and present an evaluation of the system in real time.

Unidirectional hybrid recommender systems are a well researched field [17], and there are a number of different ways of performing the hybridisation, including weighted hybrids (assigning different weights to the content-based and collaborative filtering based on the user) and cascading hybrids (using one type of filtering to shortlist for the other) [7]. It would be interesting to experiment with these types of filtering, and to find out which was effective for different types of reciprocal recommender system.

In this case, we used our judgement to create a function to calculate the preference score from a user a to a user b. However, we feel that there is more research to be done in this area. Many online services that would benefit from RRS implementation such as social networks do not have systems for users to indicate direct preference for one another as easily as online dating websites, and in this case preference scores must be calculated through indirect methods such as preference for another user's content. More research could usefully be done into which indirect indicators of preference are useful, and whether a complex function is more effective than a simple one.

To the best of our knowledge, RECON is the only content-based filtering system to be used in a reciprocal recommender system. RECON is based on the similarity between a user's preferences and their partner's attributes, whereas our system is based on the similarity between different users' preferences. It would be valuable to compare the two directly, and investigate which is more effective. This would be especially interesting to investigate in a social group recommender system, where similarity between a user's preferences and the group's preferences might determine whether or not a user is inclined to join the group.

ACKNOWLEDGMENTS

This research has been supported by the EPSRC Doctoral Training Programme (DTP). The authors would also like to thank Cookpad Inc. for providing the data for testing the system, and in particular Mr Aki Sono, Mr Miles Woodroffe and Mr Kentaro Takeguchi for their support and expertise in data acquisition and the recipe recommendation domain.

REFERENCES

[1] Deepak Agarwal and Bee-Chung Chen. 2009. Regression-based latent factor models. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '09)*. ACM, New York, NY, 19–28. https://doi.org/10.1145/1557019.1557029

[2] Charu Aggarwal. 2016. *Recommender Systems: The Textbook* (1st ed.). Springer, London, England.

[3] Francesco Ricci Akiva Kleinerman, Ariel Rosenfeld and Sarit Kraus. 2018. Optimally balancing receiver and recommended users' importance in reciprocal recommender systems. In *Proceedings of the 12th ACM Conference on Recommender Systems (RecSys '18)*. ACM, New York, NY, 131–139. https://doi.org/10.1145/3240323.3240349

[4] Lyle H. Ungar Andrew I. Schein, Alexandrin Popescul and David M. Pennock. 2002. Methods and metrics for cold-start recommendations. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '02)*. ACM, New York, NY, 253–260. https://doi.org/10.1145/564376.564421 Hybrid recsys for cold start problem.

[5] Justin Basilico and Thomas Hofmann. 2004. Unifying collaborative and content-based filtering. In *Proceedings of the twenty-first international conference on Machine learning (ICML '04)*. ACM, New York, NY, 09–09. https://doi.org/10.1145/1015330.1015394 For content-based filtering references.

[6] Robert Bell and Yehuda Koren. 2007. Lessons from the Netflix prize challenge. *ACM SIGKDD Explorations Newsletter - Special issue on visual analytics* 9, 2 (Dec. 2007), 75–79. https://dl.acm.org/citation.cfm?id=1345465

[7] Robin Burke. 2002. Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-Adapted Interaction* 12, 4 (Nov. 2002), 331–370. https://link.springer.com/article/10.1023 Hybrid recommender systems review.

[8] Marco de Gemmis Cataldo Musto, Giovanni Semeraro and Pasquale Lops. 2016. Learning Word Embeddings from Wikipedia for Content-Based Recommender Systems. In *Advances in Information Retrieval (ECIR 2016)*. Springer, Cham, 729–734. https://doi.org/10.1007/978-3-319-30671-1_60 Word embeddings in recsys.

[9] Jianming He and Wesley Chu. 2010. A Social Network-Based Recommender System (SNRS). *Data Mining for Social Network Data* 12 (May 2010), 47–74. https://link.springer.com/chapter/10.1007/978-1-4419-6287-4_4

[10] Joshua Akehurst Irena Koprinska Kalina Yacef Luiz Pizzato, Tomasz Rej and Judy Kay. 2013. Recommending people to people: the nature of reciprocal recommenders with a case study in online dating. *User Model User-Adap Inter* 23, 5 (Nov. 2013), 447–488. https://link.springer.com/article/10.1007/s11257-012-9125-0

[11] Thomas Chung Irena Koprinska Luiz Pizzato, Tomek Rej and Judy Kay. 2010. RECON: a reciprocal recommender for online dating. In *Proceedings of the fourth ACM conference on Recommender systems (RecSys '10)*. ACM, New York, NY, 207–214. https://doi.org/10.1145/1864708.1864747

[12] James Neve and Ivan Palomares. 2018. Aggregation Strategies in User-to-User Reciprocal Recommender Systems. In *Proceedings of the 2018 IEEE International Conference on Systems, Man and Cybernetics (IEEE SMC 2018)*. IEEE, New York, NY, 2299–2304. Reciprocal recommender systems aggregation.

[13] Yehuda Koren Paolo Cremonesi and Roberto Turrin. 2010. Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the fourth ACM conference on Recommender systems (RecSys '10)*. ACM, New York, NY, 39–46. https://doi.org/10.1145/1864708.1864721 Recommender systems evaluation real time.

[14] Yizhou Sun Peng Xia, Benyuan Liu and Cindy Chen. 2015. Reciprocal Recommendation System for Online Dating. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM '15)*. ACM, New York, NY, 234–241. https://doi.org/10.1145/2808797.2809282

[15] Yehuda Koren Robert Bell and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer* 42, 8 (Aug. 2009), 30–37. https://dl.acm.org/citation.cfm?id=1608614

[16] Xin Rong. 2011. word2vec Parameter Learning Explained. *axIv* 0, 0 (Nov. 2011). https://archive.org/details/arxiv-1411.2738

[17] Badrul Sarwar and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International World Wide Web Conference (WWW '01)*. ACM, New York, NY, 285–295. https://doi.org/10.1145/371920.372071

[18] Gregory S. Corrado Tomas Mikolov, Kai Chen and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *ICLR* 0, 0 (Jan. 2013), 0. https://arxiv.org/abs/1301.3781

[19] Alan W. Black Wang Ling, Chris Dyer and Isabel Trancoso. 2015. Two/Too Simple Adaptations of Word2Vec for Syntax Problems. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL '15)*. Association for Computational Linguistics, Denver, Colorado, 1299–1304. https://doi.org/10.3115/v1/N15-1142 Word2vec example problems.

[20] Trong Duc Le Xuan Nhat Lam, Thuc Vu and Anh Duc Duong. 2008. Addressing cold-start problem in recommendation systems. In *Proceedings of the 2nd international conference on Ubiquitous information management and communication (ICUIMC '08)*. ACM, New York, NY, 208–211. https://doi.org/10.1145/1352793.1352837

[21] Robert Schreiber Yunhong Zhou, Dennis Wilkinson and Rong Pan. 2008. Large-Scale Parallel Collaborative Filtering for the Netflix Prize. In *Proceedings of the 4th international conference on Algorithmic Aspects in Information and Management (AAIM '08)*. Springer-Verlag, Berlin, Heidelberg, 337–348. https://doi.org/10.1007/978-3-540-68880-8_32

[22] Zhang Ning Zheng Siting, Hong Wenxing and Yang Fan. 2012. Job recommender systems: A survey. In *Proceedings of the 7th International Conference on Computer Science & Education (ICCSE '12)*. IEEE, Melbourne, VIC, Australia, 920–924. https://doi.org/10.1109/ICCSE.2012.6295216