Improving Neighborhood-Based Collaborative Filtering by Reducing Hubness

Peter Knees,[1] Dominik Schnitzer,[2] and Arthur Flexer[2]
[1] Department of Computational Perception, Johannes Kepler University Linz, Austria
[2] Austrian Research Institute for Artificial Intelligence, Vienna, Austria
peter.knees@jku.at, dominik.schnitzer@ofai.at, arthur.flexer@ofai.at

ABSTRACT

For recommending multimedia items, collaborative filtering (CF) denotes the technique of automatically predicting a user's rating or preference for an item by exploiting item preferences of a (large) group of other users. In traditional memory-based (or neighborhood-based) recommenders, this is accomplished by, first, selecting a number of similar users (or items) and, second, combining their ratings into a single user's predicted rating for an item. Strategies for both defining similarity (i.e., to identify nearest neighbors) and for combining ratings (i.e., to weight their impact) have been extensively studied and even resulted in inconsistent findings.

In this paper, we investigate the effects of the high dimensionality of user×item matrices on the quality of memory-based movie rating prediction. By examining several publicly available real-world CF data sets, we show that the step of nearest neighbor selection is affected by the phenomena of *similarity concentration* and *hub occurrence* due to high-dimensional data spaces and the class of similarity measures used. To mitigate this, we adapt a normalization technique called *mutual proximity* that has been shown to reduce these effects in classification tasks. Finally, we show that removing hubs and incorporating normalized similarity values into the neighbor weighting step leads to increased rating prediction accuracy, observable on all examined data sets in terms of lowered error measure (RMSE).

Categories and Subject Descriptors

[**Information Systems**]: Recommender systems; [**Information Systems**]: Collaborative filtering

General Terms

Algorithms, Experimentation, Measurement

Keywords

collaborative filtering, memory-based, rating prediction, hubs, mutual proximity

ICMR 2014 Glasgow, Scotland, UK

1. INTRODUCTION

Personalized recommendation of media items to users by means of mining and exploiting preference data of many other users, also known as collaborative filtering (CF) [8], has become a central technique and everyday commodity on the Web. Today, recommender systems can be found for various types of media such as movies, music, books, or news and are a driving force in increasing online consumption. Despite the recent trend towards model-based CF methods, foremost methods based on matrix factorization, e.g., [10, 14], traditional memory-based (or neighborhood-based) approaches are still widespread due to their simplicity, explainability, and effectiveness [5].

1.1 Neighborhood-Based CF

In their most traditional form, *user-based CF*, a predicted rating $r'_{u,i}$ for a user u and an item i is calculated by finding the N most similar users according to their rating preferences (nearest neighbors) and combining their ratings for item i [12]:

$$r'_{u,i} = \frac{\sum_{j \in N} sim(u,j)(r_{j,i} - \bar{r}_j)}{\sum_{j \in N} sim(u,j)} + \bar{r}_u \qquad (1)$$

where $r_{j,i}$ denotes the rating of user j given to item i, $\bar{r}_u$ the average rating of user u, and $sim(u,j)$ a weighting factor that corresponds to the similarity of the neighboring user.

While user-based CF identifies neighboring users by examining the rows of the user×item matrix, *item-based CF*, on the other hand, operates on its columns to find similar items [19]. Predictions $r'_{u,i}$ are then made analogously to equation 1 by combining the ratings that user u has given to items similar to i. Since typically the number of rated items per user is small compared to the total number of items, item-to-item similarities can be pre-calculated and cached. In many real-world recommender systems, item-based CF is thus chosen over user-based CF for performance reasons [15].

For determining nearest neighbors (NNs), i.e., the k most similar vectors, in both user-based and item-based CF, *cosine similarity* or a "cosine-like measure" [17], such as *Pearson correlation* or *adjusted cosine similarity* is utilized. With these measures, similarity between two vectors p and q is calculated on the set of n co-rated entities via the angle enclosed.

$$sim_{\cos}(p,q) = \frac{p^T q}{\|p\| \, \|q\|} = \frac{\sum_{i=1}^{n} p_i q_i}{\sqrt{\sum_{i=1}^{n} p_i^2} \sqrt{\sum_{i=1}^{n} q_i^2}} \qquad (2)$$

Basing similarity on the angle rather than the absolute feature values, in the user-based case, makes the approach ag-

nostic to user rating biases or, in the item-based case, to item popularity biases. For selecting NNs, typically only neighbors with $sim > 0$, i.e., positively correlated entities, are considered [9]. Throughout this paper, we will adopt this strategy.[1]

1.2 Curse of Dimensionality

Since the size of the user×item rating matrix depends on the number of users and items, features in memory-based CF are known to be high-dimensional. When calculating distances in high-dimensional spaces, two phenomena emerge, as pointed out by Nanopolous et al. [17]: First, pairwise distances between all data points tend to become very similar, known as *distance concentration*. Second, as a consequence of this, *hubs*, i.e., data points which have a small distance (or high similarity) to disproportionally many other data points, appear. Both effects are considered to be a new aspect of the *curse of dimensionality* [18, 20, 23]. A number of recent publications have investigated the influence of hubs on several machine learning tasks, such as classification [18], clustering [25], and content-based item retrieval [6], and have shown that *(a)* they have a negative impact and *(b)* the mentioned tasks benefit from methods that decrease their influence.

For neighborhood-based CF, in which similarity of high-dimensional data points plays a central role, it can therefore be assumed that the same effects occur. Moreover, due to their nature of being "always similar," hubs are expected to get frequently selected as nearest neighbors, thus becoming overly influential and ultimately impacting the prediction of ratings. As a first step to prove the existence of the "hub problem" in CF, by means of theoretical and empirical analysis, Nanopolous et al. [17] show that the effects of distance concentration — in this case, similarity concentration — and hubness also result from cosine-like measures as frequently applied in CF.[2]

In this paper, we substantiate these findings by investigating several real-world rating data sets for hubness. We empirically confirm that the step of nearest neighbor selection in memory-based CF is affected by the existence of hubs and negatively impacts subsequent steps. To mitigate this, we propose an adapted normalization technique to rescale the similarity space and symmetrize the nearest neighbor relation. Ultimately, we show that removing hubs and incorporating normalized similarity values into the neighbor weighting step leads to increased rating prediction accuracy, observable on all examined data sets.

The remainder of this paper is structured as follows. First, we review existing approaches on nearest neighbor selection and weighting in collaborative filtering as well as work dealing with hubness in general and with regard to recommender systems. In section 3, we examine publicly available CF data collections containing movie rating data for the occurrence of hubs. We propose the adapted similarity space normalization methods in section 4 and investigate their effects on CF data in section 5. In section 6, we present systematic rating prediction experiments using our proposed method to show the advantage of hub reduction for traditional memory-based CF. Finally, we discuss our findings and implications for future work in section 7.

[1]Initial experiments we have conducted have also shown that incorporating negative correlations decreases prediction accuracy and thus confirmed this view.

[2]Note that neither of these phenomena is connected to the *data sparsity problem* ubiquitous in recommender systems, but manifest as a consequence of the high dimensionality and the applied similarity measure [17].

2. RELATED WORK

Since the basic rating aggregation schemes of memory-based systems all are conceptually quite similar (cf. equation 1), a lot of existing work deals with finding the best matching neighbors and determining the weights of their ratings. In this section, we review this work, followed by related work on hubness in the context of recommendation.

2.1 Nearest Neighbor Selection and Weighting in Memory-based Recommenders

Breese et al. [4] and Herlocker et al. [9] explore several similarity weighting methods for user-based CF, such as cosine vector similarity, Pearson correlation, Spearman rank correlation, entropy, and mean squared difference and yield partly inconsistent results. However, from their combined findings, it can be concluded that Pearson correlation and Spearman rank correlation are the preferred similarity measures for user-based CF. Furthermore, they find that selecting NNs based on techniques such as *similarity thresholding* and *top N selection* not only reduces the *data point coverage* of NNs considered but also tends to de-emphasize the contributions of the few highly correlated users in favor of noise coming from a large number of less correlated users. To alleviate this, Breese et al. propose a technique called *case amplification*, i.e., amplifying NN influence by exponentially weighting similarities [4].

In addition, correlations based on only a few co-rated items are less trustworthy than correlations based on many co-rated items. To address this, literature proposes neighborhood selection methods that take the number of co-rated vector dimensions (again denoted as n, cf. eq. 2) into account. Such a method is *significance weighting* [9], which lowers weights if less than 50 items are co-rated by multiplying user similarities with the factor $min(n/50, 1)$. Takács et al. [24] regularize the correlation estimate by applying Fisher's z-transformation and amplifying the result exponentially. Koren [13] suggests the use of the *correlation shrinkage factor* $n/(n + 100)$ for item-based approaches (cf. section 3). To deal with the small overlap in rating vectors due to data sparsity, *default voting* [4] assumes default values (e.g., the average rating $\bar{r}_u$) for ratings missing for one of the users to calculate correlations on the union of rated items rather than on the intersection.

Another direction in weighting functions is based on the idea that the agreement of users on items that are "controversial" , i.e., items that get many different ratings, should have more impact on the similarity than the agreement on items everybody likes. Following the information retrieval paradigm of inverse document frequency, *inverse user frequency* [4] reduces the influence of items rated by many users on the similarity function by weighting the corresponding item dimension with $\log \frac{|U|}{|U_i|}$, where $|U|$ denotes the total number of users and $|U_i|$ the number of users who rated item i. A similar idea is proposed by Xie et al. [26]. Their technique, *most-same opinion* (MSO), aims at selecting only users as NNs that are very consistent with the active user. In comparison with inverse user frequency, MSO is even stricter in that it considers the weighted item dimension only if the

corresponding item is rated equally by both users. *Variance weighting* [9] incorporates a variance weight term into similarity calculation, i.e., items with a high variance in their ratings gain more influence in the similarity weight calculation. However, it should be noted that variance weighting had no positive effect on the prediction accuracy according to [9].

Jin et al. [11] introduce an item weighting scheme that aims at bringing users closer to users with similar interests while separating them from users with different interests. To this end, they operate on a probabilistic description, resulting in an asymmetric similarity function that estimates the likelihood of a user to be similar to another from a conditional exponential model that incorporates item weights automatically learned from training data.

Baltrunas and Ricci [2] compare different feature (i.e., item) weighting strategies, namely random, item-based Pearson correlation, variance-based, mutual-information-based, and tag-based weighting. In tag-based weighting, external domain-specific information is utilized to calculate item similarity from an overlap in meta-data categories. For feature selection, top N selection according to overall weight, top N selection per user, and top N selection from overlapping items are explored. Experimental results show that there is no consistently superior weighting scheme and that selecting the highest scoring items from overlap can improve user-based CF methods.

External information is also utilized by Amatriain et al. [1]. Instead of deriving nearest neighbors from the user×item matrix, an independent source of ratings, so called expert neighbors, are used to make predictions. While this approach is successful in avoiding some of the common issues of CF systems such as noisy ratings, scalability, and privacy concerns, the performance in terms of prediction accuracy remains below traditional memory-based approaches.

2.2 Hubness in Recommender Systems

Despite principal considerations of hubness being a phenomenon in and affecting collaborative filtering [17], so far, the topic has received comparatively little attention in the recommender systems literature and mostly been addressed for the task of content-based item recommendation.

Seyerlehner et al. [21] identify hubs as a limitation in browsing top N item recommender systems. In the similar item recommendation scenario, hubness has a negative effect as hubs occur in many of the top N recommendations and thus reduce coverage and reachability, particularly of long-tail items. It is shown that this limiting factor is inherent in content-based and CF-based features alike.

Schnitzer et al. [20] show that hubness reduction increases nearest neighbor classification accuracy on a variety of machine learning data sets as well as for the task of audio-signal-based music recommendation. Flexer et al. [6] extend these investigations by conducting a meta-analysis of different content-based music recommendation approaches and show that distance normalization improves retrieval accuracy independently of the underlying method. In this paper, we propose to adapt the technique used in [20] and [6] for usage in collaborative filtering recommenders.

3. EXAMINING RATING DATA SETS

As first step to address the problem, in this section, we examine different data sets for the effect of hubness. To allow experimental reproducibility, we examine the following well-known and publicly available movie rating datasets:

- **MovieLens 100k** *(ml-100k)*: 100,000 ratings from 943 users on 1,682 items [9]
- **MovieLens 1m** *(ml-1m)*: 1,000,209 ratings from 6,040 users on 3,706 items
- **MovieLens 10M** *(ml-10M)*: 10,000,054 ratings from 69,878 users on 10,677 items[3]
- **Netflix Prize data set** *(netflix)*: 100,480,507 ratings from 480,189 users on 17,770 items[4] [3]

Note that *ml-100k*, *ml-1m*, and *netflix* contain integer ratings on a scale from 1 to 5, whereas *ml-10M* uses a scale from 0.5 to 5 with a step size of 0.5.

For similarity calculation, we explore two measures, namely *Pearson correlation* and *binary cosine similarity*, which are related to cosine similarity and therefore exhibit the same concentration tendency, as discussed in [17]. Pearson correlation is equivalent to calculating cosine similarity on the mean-centered data vectors, accounting for rating biases.

$$sim_{corr}(p,q) = \frac{\sum_{i=1}^{n}(p_i-\bar{p})(q_i-\bar{q})}{\sqrt{\sum_{i=1}^{n}(p_i-\bar{p})^2}\sqrt{\sum_{i=1}^{n}(q_i-\bar{q})^2}} \quad (3)$$

In addition, on the first two sets, we study the impact of the *correlation shrinkage factor* proposed by Koren [13] to reweigh Pearson correlation values for item-based approaches (cf. section 2.1)

$$sim_{shrunk}(p,q) = \frac{n}{n+\lambda} sim_{corr}(p,q) \quad (4)$$

with λ set to 100 as suggested in [13].

Binary cosine similarity is a simple overlap measure addressing co-ratings and equivalent to calculating cosine similarity on binarized data vectors, i.e., $sim_{\cos}(p_b, q_b)$ where p_b results from p by setting entries $\neq 0$ to 1. Thus, calculation of binary cosine similarity comes to

$$sim_{bcos}(p,q) = \frac{n}{\sqrt{n_p}\sqrt{n_q}} \quad (5)$$

where n is again the number of co-rated elements (overlap) and n_p the number of rated elements in p. For our experiments, we utilize the implementation of both measures provided by the *MyMediaLite*[5] framework [7].

To quantitize the strength of the hubness phenomenon in a data set we utilize the hubness measure defined by Radovanović et al. [18].

Hubness (S^k): To compute hubness we first define $O^k(x)$ as the k-occurrence of point x, that is, the number of times x occurs in the k-nearest neighbor lists of all other objects in the collection. Hubness is then defined as the skewness of the distribution of k-occurrences, O^k:

$$S^k = \frac{\mathrm{E}\left[(O^k-\mu_{O^k})^3\right]}{\sigma_{O^k}^3}. \quad (6)$$

A data set having high hubness produces few hub objects with very high k-occurrence and many anti-hubs with k-occurrence of zero. This makes the distribution of k-occurrences right-skewed with positive skewness indicating high

[3] ml-100k, ml-1m, and ml-10M are available at `http://www.grouplens.org/node/73`

[4] `http://www.netflixprize.com` (data set no longer made available)

[5] `http://mymedialite.net`, release version 3.08

Table 1: Hubness values S^k of CF data sets with Pearson correlation as similarity function. To the right of the dividing line hubness values after transformation with mutual proximity. The bottom part shows hubness values resulting from using the "shrunk" Pearson correlation (eq. 4).

			NN Pearson			$\mathbf{MP}_{emp}$			$\mathbf{MP}_{ig}$		
data set	data points	feature dim.	S^5	S^{10}	S^{20}	S^5	S^{10}	S^{20}	S^5	S^{10}	S^{20}
ml-100k — item	1,682	943	4.4	3.1	2.2	2.4	2.0	1.6	1.4	0.9	0.3
ml-100k — user	943	1,682	2.9	1.8	1.3	2.4	1.9	1.3	1.2	0.7	0.3
ml-1m — user	6,040	3,706	7.9	5.5	3.9	4.0	3.4	2.8	1.9	1.4	1.0
ml-1m — item	3,706	6,040	6.2	4.3	3.2	4.0	3.5	2.8	1.7	1.3	1.0
ml-10M — item	10,677	69,878	16.7	10.5	6.6	5.3	4.6	3.9	4.2	3.0	2.1
netflix — item	17,770	480,189	26.2	17.1	12.9	4.2	3.6	3.1	8.3	6.4	4.8
			NN $\mathbf{P}_{shrunk}$			$\mathbf{MP}_{emp}$			$\mathbf{MP}_{ig}$		
ml-100k — item	1,682	943	3.8	3.4	2.9	1.7	1.4	1.2	1.4	1.4	1.4
ml-100k — user	943	1,682	8.9	6.8	4.8	3.0	2.4	1.8	0.9	0.7	0.6
ml-1m — user	6,040	3,706	13.7	11.4	9.3	5.2	4.2	3.3	2.2	2.0	1.7
ml-1m — item	3,706	6,040	3.1	2.8	2.5	2.8	2.2	1.7	1.7	1.5	1.3

Table 2: Comparison of hubness S^k of CF data sets with binary cosine as similarity function (cf. table 1).

			NN BCosine			$\mathbf{MP}_{emp}$			$\mathbf{MP}_{ig}$		
data set	data points	feature dim.	S^5	S^{10}	S^{20}	S^5	S^{10}	S^{20}	S^5	S^{10}	S^{20}
ml-100k — item	1,682	943	4.7	3.5	2.7	3.4	1.5	0.4	2.1	1.1	0.5
ml-100k — user	943	1,682	2.9	2.6	2.1	0.7	0.6	0.4	1.3	0.9	0.7
ml-1m — user	6,040	3,706	4.6	4.0	3.5	1.4	1.2	1.0	1.8	1.5	1.3
ml-1m — item	3,706	6,040	5.2	4.0	2.7	1.9	0.9	0.5	1.2	0.8	0.6
ml-10M — item	10,677	69,878	7.5	7.1	6.2	3.0	1.7	1.0	2.7	1.4	0.7
netflix — item	17,770	480,189	45.4	30.1	17.1	0.5	0.4	0.3	11.0	7.7	4.8

hubness. Tables 1 (column *NN Pearson*) and 2 (col. *NN BCosine*) show the hubness S^k of the test data sets at $k = 5, 10, 20$ for Pearson correlation and binary cosine similarity, respectively (for the moment, the reader is asked to ignore the remaining columns in both tables). It can be seen that hubness increases with increasing dimensionality. For increasing k, hubness obviously decreases as more objects are included.

For Pearson correlation, we can study the effect of the shrinkage factor in terms of hubness by comparing rows from the upper and the lower part of table 1. In the item-based setting, for which the factor is originally intended, the simple reweighing scheme actually decreases hubness. However, conversely, for user-based settings, a negative effect, i.e., an increase in hubness, can be observed.

For both, Pearson and binary cosine, the observed hubness values are rather high. For reference, from other domains hubness values of up to 14.6 are reported and, as a guideline, values above 1.4 indicate a hub problem [20]. Here, values of up to 45.4 are observed (*netflix item-based, binary cosine*), showing the extent to which CF data is affected by the phenomenon of hubness and clearly demonstrating the need to address this issue.

4. REDUCING HUBNESS

After documenting the phenomenon of hubness in CF, in this section, we propose to adopt a recently introduced global distance scaling method [20]. Its general idea is to reinterpret the original distance space so that two objects sharing similar nearest neighbors are more closely tied to each other, while two objects with dissimilar neighborhoods are repelled from each other. This is done by reinterpreting the distance of two objects as a *mutual proximity* (MP) in terms of their distribution of distances. It was shown that by using MP, hubs are effectively removed from the data set while the intrinsic dimensionality of the data stays the same.

To apply MP to a distance matrix, it is assumed that the distances $D_{x,i=1..N}$ from an object x to all other objects in the data set follow a certain probability distribution, thus any distance $D_{x,y}$ can be reinterpreted as the probability of y being the nearest neighbor of x, given their distance $D_{x,y}$ and the probability distribution $P(X)$:

$$P(X > D_{x,y}) = 1 - P(X \leq D_{x,y}) = 1 - \mathcal{F}_x(D_{x,y}). \quad (7)$$

MP is then defined as the probability that y is the nearest neighbor of x given $P(X)$ and x is the nearest neighbor of y given $P(Y)$:

$$MP(D_{x,y}) = P(X > D_{x,y} \cap Y > D_{y,x}). \quad (8)$$

Due to the probabilistic foundation of MP, the resulting values range from 0 to 1 and can again be interpreted as similarity values.

We want to study the effects of two variants of MP, namely MP based on the empirical distribution of the similarities and a simplified and faster variant that assumes that similarities $S_{x,i=1..N}$ follow a Gaussian distribution. However, in order to compute MP in our CF experiments, we have to make adaptations to the original formulation. First, in CF we are dealing with similarities $S_{x,y}$ rather that distances $D_{x,y}$,[6] resulting in a modified definition of MP:

$$MP(S_{x,y}) = P(X \leq S_{x,y} \cap Y \leq S_{y,x}). \quad (9)$$

Second, in CF, due to the sparsity of the rating matrix, the similarity matrix tends to be also sparse. Thus, when empir-

[6]A simple conversion from distances to similarities is, e.g., $D = 1 - S$ which can be applied to cosine similarity and Pearson correlation alike and preserves the range of [0..1], assuming that negatively correlated objects are omitted in advance.

ically modeling the underlying distribution of similarities we have to adapt MP in order to deal with smaller NN spaces (see below). The following sections describe the two adapted variants that we apply in more detail.[7]

4.1 Adapted MP with Empirical Distribution

If the number of observations is large enough, the empirical distribution will tend to model the true underlying distribution of similarities closely. Computing MP for a given similarity $S_{x,y}$ using the empirical distribution boils down to simply counting the number of objects j having a similarity to x and y which is less or equal to $S_{x,y}$ and setting this in relation to the number of objects where a similarity $S(x,j)$ or $S(y,j)$ exists in the sparse CF similarity matrix.

$$MP_{emp}(S_{x,y}) = \frac{|\{j : S_{x,j} \leq S_{x,y}\} \cap \{j : S_{y,j} \leq S_{y,x}\}|}{|\{\exists j : S_{x,j}\} \cup \{\exists j : S_{y,j}\}|} \tag{10}$$

4.2 Adapted MP with Gaussian Distribution

We also use an MP variant that works under the assumption that the similarities $S_{x,i=1..N}$ follow independent Gaussian distributions, which simplifies and speeds up the computation of MP. To use this method, the parameters of the underlying Gaussian similarity distribution $P(X)$ of each object x need to be estimated. This is done by computing the sample mean ($\hat{\mu}_x$) and standard deviation ($\hat{\sigma}_x$):

$$\hat{\mu}_x = \frac{1}{n_x} \sum_{i=1}^{n_x} S_{x,i}, \tag{11}$$

$$\hat{\sigma}_x^2 = \frac{1}{n_x} \sum_{i=1}^{n_x} (S_{x,i} - \hat{\mu}_x)^2. \tag{12}$$

By assuming independence between the distance distributions $P(X)$ and $P(Y)$, the computation of MP simplifies in accordance with the product rule to

$$MP_{ig}(S_{x,y}) = P(X \leq s_{x,y}) \cdot P(Y \leq s_{y,x}) \tag{13}$$

which can be computed in a straightforward manner as all parameters of the Gaussians have been estimated.

5. EFFECTS OF MP

To show the effect of MP scaling on cosine measures, we first experiment with an artificial data set, cf. [17]. Second, we examine the impact of MP on the real-world data sets.

5.1 Synthetic Data Set

In this experiment we use uniformly distributed data randomly sampled from a d-dimensional unit cube. By using this strategy, the following basic effects should be observable with increasing data dimensionality: (i) according to the effect of similarity concentration for cosine-like similarity measures (see [17]), the average measured (expected) similarity should remain constant while the standard deviation of similarities decreases, and (ii) with the concentration of similarities the hubness phenomenon should emerge.

Figure 1 shows these general effects of high dimensional data. We start by generating two dimensional data ($d = 2$) and gradually increase the data dimensionality to $d = 100$, sampling $N = 2{,}000$ data points and averaging our measurements of similarity concentration and hubness over 100 repetitions. In figures 1(a) and 1(b) we can see that, with increasing dimensionality, the similarities clearly concentrate. At the same time, the measured hubness increases steadily up to a value of 5.5, which already indicates a strong skewness of the O^k-occurrences, cf. fig. 1(c). The figure also shows the impact of using MP on the similarities. It is indicated by the gray line in all three plots. MP seems to effectively de-concentrate the similarity space while at the same time reducing hubness to very low values if compared to the original data space.

5.2 Real-World Data Sets

Apart from a synthetic data set, we investigate the effects on the examined real-world test sets. Looking again at tables 1 and 2 (this time columns MP_{emp} and MP_{ig}), we can see the effects of both MP variants on Pearson correlation and binary cosine similarity, respectively. It can clearly be seen that hubness is drastically reduced by both approaches on all sets and for all examined values of k. However, it should be noted that for some sets, especially the high-dimensional data sets that exhibit high native hubness, such as *netflix item-based*, hubness is still high even after MP transformation. An exception to this is MP_{emp} applied to the binary cosine measure on the *netflix* data, that reduces hubness from 45.4 to 0.5 (S^5).

Another aspect of the application of MP concerns *data point coverage*. We define data point coverage as the fraction of objects that are considered as NNs and thus incorporated into rating prediction. Removing hubs should therefore increase the overall coverage. Figure 2 shows the data point coverage for $k = 5$ using Pearson and binary cosine.

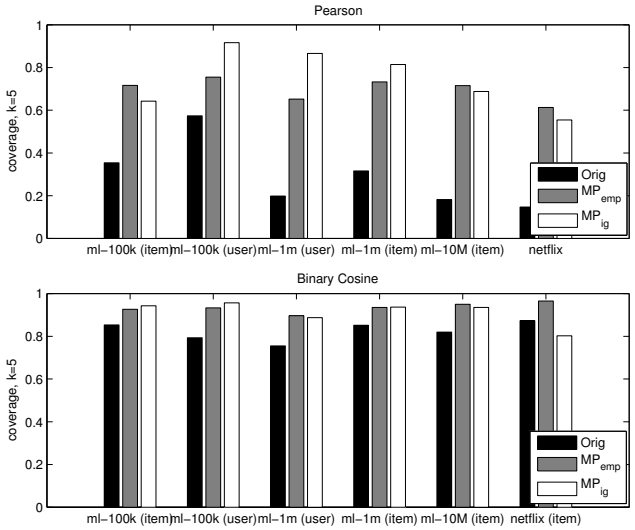

Figure 2: Data point coverage for $k = 5$ before and after MP transformations using Pearson (top) and binary cosine (bottom).

[7]Implementations of both adapted MP variants, along with example scripts to integrate MP into *MyMediaLite*'s memory-based recommendation pipeline are available online at `http://www.ofai.at/research/impml/projects/hubology.html`.

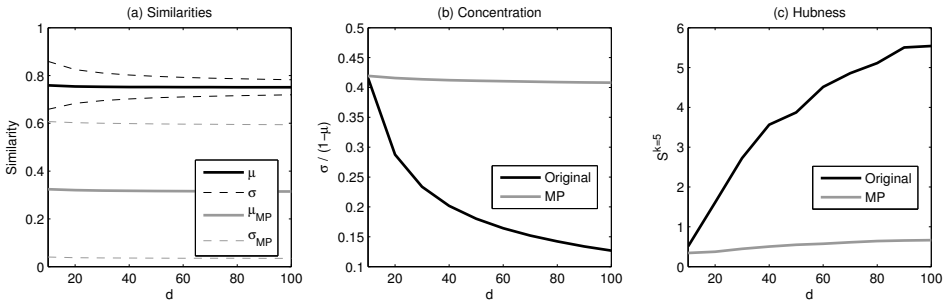

Figure 1: With increasing dimensionality, the measured cosine similarities concentrate and the hubness effect becomes more pronounced. Mutual Proximity (gray line) seems to be a viable method to remove hubness (here only the empirical version is shown).

With the exception of MP_{ig} on the *netflix* set using binary cosine, MP improves data point coverage as expected. In this context, also the differences of the two similarity measures become apparent. While Pearson correlation leads to a rather low neighbor coverage that is drastically increased by MP (e.g., we observe an increase of around 350% for user-based *ml-1m* and MP_{ig}), binary cosine already provides high coverage. This can be connected to the simplicity of the binary cosine measure that mostly relies on the existence of co-rated items, irrespective of the actual ratings, whereas when using Pearson correlation, negatively correlated objects are omitted. For the binary cosine measure on the *netflix* data, hubness seems to be such a dominant effect that hub removal can actually decrease data point coverage (cf. table 2).

6. EVALUATING THE EFFECTS ON CF

Finally, we investigate the effects of MP transformed similarity measures on the quality of the task of rating prediction. In addition to reselecting the NNs for an object under consideration, MP has the advantage of being directly applicable as a similarity weighting function, leaving the prediction scheme unaltered. Hence, for rating prediction according to equation 1, perform NN selection based on the MP values and define

$$sim(u, j) := MP(S_{u,j}) \tag{14}$$

to incorporate the transformed similarity space.

For the sake of reproducible experiments, for *ml-100k*, we conduct a 5-fold cross validation according to the folds `u1-u5` provided in the data set. For *ml-1m*, for the lack of a predefined partitioning, we conduct a 5-fold cross validation with random assignments of ratings to folds. For *netflix*, we split the data into a training and a test set according to the *probe set* that was defined for the Netflix Prize. We evaluate the performance over different numbers of k-NNs, with k varying from 5 to 100 in intervals of 5. As performance measure and evaluation criteria we utilize *root mean squared error (RMSE)*, e.g., [22], defined as

$$RMSE = \sqrt{\frac{1}{|T|} \sum_{(u,i) \in T} \left(r'_{u,i} - r_{u,i}\right)^2} \tag{15}$$

where T denotes the test set. When evaluating via cross validation, every rating in the data set will serve as test data in one of the folds, thus, for calculating RMSE in those cases, T comprises the complete data set.

Figures 3-5 illustrate the positive impact of both MP variants on prediction quality. With the exception of item-based CF using binary cosine on *netflix*, MP_{emp} produces consistently smaller or equal errors in comparison to the unscaled measures. MP_{ig} generally also improves prediction quality, however, the improvement is not as consistently observable as with MP_{emp}. For every data set, the lowest RMSE value found is produced by one of the MP variants (for proper selection of k).

Examining the Pearson correlation results in figures 3 and 4, we see that the hub-reducing correlation shrinkage factor has a very positive impact on prediction accuracy and can rival (*ml-1m*, user-based) or outperform MP applied to standard Pearson correlation (all others). As expected, this effect is more pronounced for item-based experiments. However, even starting from the advanced results of shrunk Pearson correlation, MP is capable of further reducing hubness and RMSE, suggesting that correlation shrinkage and MP address complementary weaknesses of memory-based CF and hub removal.

On the *ml-100k* set, using the Pearson correlation, MP_{ig} error even increases in comparison to Pearson. Since MP_{ig} is based on estimating the parameters of the similarity distribution, model quality is directly connected to size of the underlying "sample" of similarities, i.e., the number of objects that have (positively correlated) co-ratings with and are thus connected to the object under consideration. For sparse data sets, however, also the neighborhood matrix tends to be sparse, rendering parameter estimates unreliable. In principle, the same sparsity effect could also influence MP_{emp}, however MP_{emp} behaves more stable due to its unparametrized formulation. Despite the differences in MP variants, item-based approaches seem to benefit from MP to a larger extent than user-based approaches.

For binary cosine on item-based *netflix*, results when using MP get worse than with the classic approach for $k > 25$. We assume that this effect can be connected to the decrease in data point coverage when applying MP, i.e., hub removal has such a strong impact, that NN selection resorts to fewer and less informative data points using binary cosine (cf. fig. 2).

7. DISCUSSION AND FUTURE WORK

In this paper, we examined CF rating data for the effect of hubness, a common phenomenon in high-dimensional data spaces. We found that CF data is indeed very prone to hubness, imposing a problem on memory-based recommender

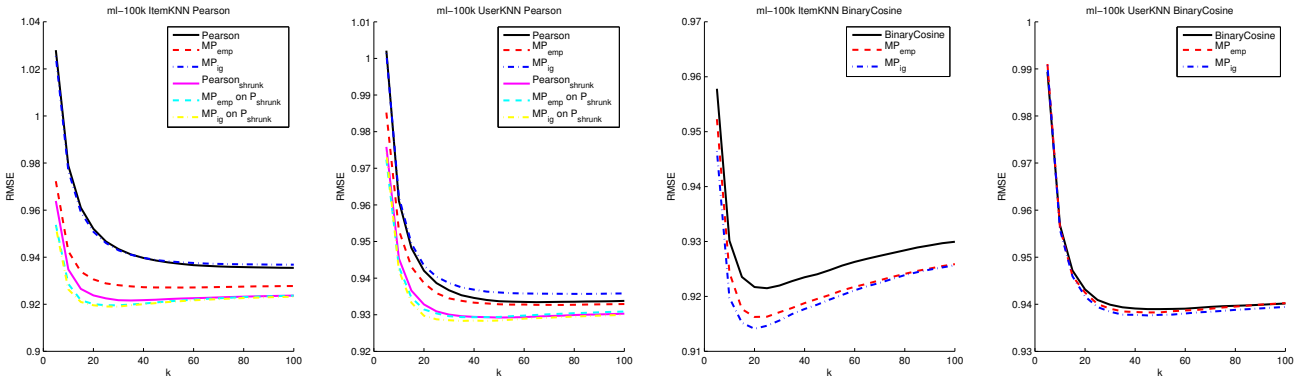

Figure 3: RMSE of original similarity functions vs. MP-transformed similarities on the *ml-100k* set.

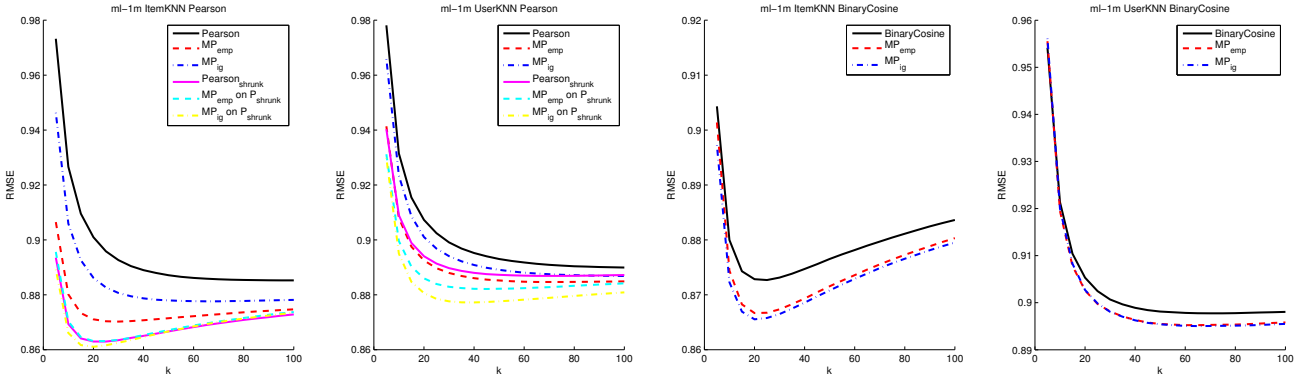

Figure 4: RMSE of original similarity functions vs. MP-transformed similarities on the *ml-1m* set.

systems which extensively make use of NN information. By adapting MP, a technique to rescale the similarity space such that NN sets become more balanced, to the task of CF, we could reduce hubs and increase data point coverage among NNs. Furthermore, it could be shown that similarities calculated by MP are favorable over standard cosine-like measures for usage in neighborhood-based rating prediction, suggesting that MP selects better suited NNs and weights them accordingly. In addition, we investigated a similarity correction function from the literature and demonstrated its effect in terms of hub removal. The results obtained suggest that different approaches mitigate different problematic characteristics of rating data and that a combination of different countermeasures (that includes hub removal) is capable of improving prediction accuracy.

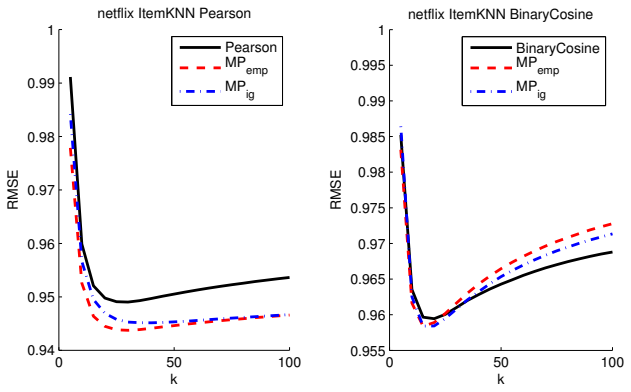

Figure 5: RMSE of original similarity functions vs. MP-transformed similarities on the *netflix* set using item-based CF.

In contrary to existing approaches for calculating similarities and weighting NNs, mutual proximity is a parameter free approach. While defining threshold parameters such as lower bounds for the number of co-rated items might be justified and applicable in specific scenarios, a statistical and data-driven approach is to be preferred. However, for future work it is a goal to compare MP to further existing weighting heuristics from the literature. In particular, it will be interesting to see which other existing weighting schemes implicitly address a hub problem in the underlying data and how effective those methods are in reducing hubness.

While the values obtained can not rival the state-of-the-art of model-based approaches for recommendation, they show that CF data itself is prone to effects such as hubness, and that more research into this direction is necessary. Furthermore, more similarity and correlation measures and the impact of MP on them need to be explored in detail and evaluations should be extended to a larger pool of CF data sets from different domains. Future work will also cover fur-

ther evaluation strategies (such as diversity and precision and recall) to see the effects of MP on different aspects of recommenders.

Additional future directions cover experimentation with MP directly on the rating matrix, researching coverage on the final item recommendations, and investigating the effects of MP on the robustness of recommender systems. One of the main approaches to attack a system consists in injecting biased profiles in order to change the system's recommendation behavior [16]. To have a larger impact, injected profiles are typically designed to be similar to many profiles. These profiles can be considered artificial hubs. A key aspect of MP in that context might be its extended coverage as this could lead to a lowered impact of malicious profiles and thus less susceptibility for attacks and profile injection.

We can conclude that investigations for hubness seem crucial — for recommender systems and in general. When dealing with high-dimensional data, an analysis of hubness should be carried out, as a simple countermeasure can easily improve the following results.

8. ACKNOWLEDGMENTS

This research is supported by the Austrian Science Fund (FWF): P24095 and the European Union Seventh Framework Programme FP7/2007-2013 under grant agreement no. 610591 (GiantSteps).

9. REFERENCES

[1] X. Amatriain, N. Lathia, J. M. Pujol, H. Kwak, and N. Oliver. The wisdom of the few – a collaborative filtering approach based on expert opinions from the web. In *Proc 32nd SIGIR*, pp. 532–539, 2009.

[2] L. Baltrunas and F. Ricci. Dynamic item weighting and selection for collaborative filtering. *Web mining*, 2:135–146, 2007.

[3] J. Bennett and S. Lanning. The Netflix prize. In *Proc KDD Cup and Workshop*, p. 35, 2007.

[4] J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proc 14th conference on Uncertainty in artificial intelligence (UAI'98)*, pp. 43–52, 1998.

[5] C. Desrosiers and G. Karypis. A comprehensive survey of neighborhood-based recommendation methods. In F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, editors, *Recommender Systems Handbook*, pp. 107–144. Springer, 2011.

[6] A. Flexer, D. Schnitzer, and J. Schlüter. A mirex meta-analysis of hubness in audio music similarity. In *Proc 13th int'l society for Music information retrieval conference*, pp. 175–180, 2012.

[7] Z. Gantner, S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. MyMediaLite: A free recommender system library. In *Proc 5th ACM conference on Recommender systems*, 2011.

[8] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Comm ACM*, 35(12):61–70, 1992.

[9] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *Proc 22nd SIGIR*, pp. 230–237, 1999.

[10] T. Hofmann. Latent semantic models for collaborative filtering. *ACM Trans. Inf. Syst.*, 22(1):89–115, 2004.

[11] R. Jin, J. Y. Chai, and L. Si. An automatic weighting scheme for collaborative filtering. In *Proc 27th SIGIR*, pp. 337–344, 2004.

[12] J. A. Konstan, B. N. Miller, D. Maltz, J. L. Herlocker, L. R. Gordon, and J. Riedl. Grouplens: applying collaborative filtering to usenet news. *Comm ACM*, 40(3):77–87, Mar. 1997.

[13] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proc 14th ACM SIGKDD*, pp. 426–434, 2008.

[14] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[15] G. Linden, B. Smith, and J. York. Amazon.com recommendations: item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1):76–80, 2003.

[16] B. Mobasher, R. Burke, R. Bhaumik, and C. Williams. Toward trustworthy recommender systems: An analysis of attack models and algorithm robustness. *ACM Trans. Internet Technol.*, 7(4):23, 2007.

[17] A. Nanopoulos, M. Radovanović, and M. Ivanović. How does high dimensionality affect collaborative filtering? In *Proc 3rd ACM conference on Recommender systems*, pp. 293–296, 2009.

[18] M. Radovanović, A. Nanopoulos, and M. Ivanović. Hubs in space: Popular nearest neighbors in high-dimensional data. *Journal of Machine Learning Research*, 11:2487–2531, 2010.

[19] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *Proc 10th int'l conference on World Wide Web (WWW)*, pp. 285–295, 2001.

[20] D. Schnitzer, A. Flexer, M. Schedl, and G. Widmer. Local and global scaling reduce hubs in space. *Journal of Machine Learning Research*, 13:2871–2902, 2012.

[21] K. Seyerlehner, A. Flexer, and G. Widmer. On the limitations of browsing top-n recommender systems. In *Proc 3rd ACM conference on Recommender systems*, pp. 321–324, 2009.

[22] G. Shani and A. Gunawardana. Evaluating recommendation systems. In F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, editors, *Recommender Systems Handbook*, pp. 257–297. Springer, 2011.

[23] I. Suzuki, K. Hara, M. Shimbo, Y. Matsumoto, and M. Saerens. Investigating the effectiveness of laplacian-based kernels in hub reduction. In *Proc 26th conference on Artificial Intelligence (AAAI)*, pp. 1112–1118, 2012.

[24] G. Takács, I. Pilászy, B. Németh, and D. Tikk. Major components of the gravity recommendation system. *ACM SIGKDD Explor. Newsl.*, 9(2):80–83, 2007.

[25] N. Tomašev, M. Radovanović, D. Mladenić, and M. Ivanović. The role of hubness in clustering high-dimensional data. In *Advances in Knowledge Discovery and Data Mining*, pp. 183–195, 2011.

[26] B. Xie, P. Han, F. Yang, R.-M. Shen, H.-J. Zeng, and Z. Chen. Dcfla: A distributed collaborative-filtering neighbor-locating algorithm. *Information Sciences*, 177(6):1349 – 1363, 2007.