On the Impact of Text Similarity Functions on Hashtag Recommendations in Microblogging Environments

Eva Zangerle, Wolfgang Gassler, Günther Specht

Databases and Information Systems
Institute of Computer Science
University of Innsbruck, Austria
firstname.lastname@uibk.ac.at

Abstract. Microblogging applications such as Twitter are experiencing tremendous success. Microblog users utilize hashtags to categorize posted messages which aim at bringing order to the myriads of microblog messages. However, the percentage of messages incorporating hashtags is small and the used hashtags are very heterogeneous as hashtags may be chosen freely and may consist of any arbitrary combination of characters. This heterogeneity and the lack of use of hashtags lead to significant drawbacks in regards to the search functionality as messages are not categorized in a homogeneous way. In this paper we present an approach for the recommendation of hashtags suitable for the message the user currently enters which aims at creating a more homogeneous set of hashtags. Furthermore, we present a detailed study on how the similarity measures used for the computation of recommendations influence the final set of recommended hashtags.

1 Introduction

Microblogging has become immensely popular throughout the last years. Platforms like Jaiku, Tumblr and Twitter are experiencing tremendous popularity on the web. Essentially, microblogging allows users to post short messages, on the Twitter platform these are at most 140 characters long. These posted messages—also known as tweets—are available to the public. Users are able to "follow" other users, which basically means that if user A follows user B (the followee), user A subscribes to the feed of tweets of user B. Messages are then added to the user's timeline (a chronological overview about his own tweets and the tweets of his followees) which enables him to always be up-to-date with his followee's tweets. Users may also re-broadcast tweets of other users (so-called retweets). Furthermore, users are also able to address tweets to certain users (so-called direct messages) or to simply mention users in their tweets. Considering the fact that currently about 340,000,000 Twitter messages are posted every day[1] at a peak

[1] according to http://business.twitter.com/en/basics/what-is-twitter/

level of 8,868 messages per second[2], it becomes clear that searching through this data can be a tedious task. Therefore, Twitter users themselves started to manually categorize and classify their tweets—they started to use so-called *hashtags* as a part of the message. The only requirement for a hashtag is that it has to be preceded by a hash symbol #, like e.g. in the hashtags `#apple`, `#elections` or `#obama`. There are no further restrictions in regards of the syntax or semantics of hashtags, which makes them a very convenient, easy-to-use way of categorizing tweets. Most importantly, hashtags can be used for searching messages, following a certain thread or topic and therefore mark a set of tweets focusing on a certain topic described by the hashtag. Hence, the use of appropriate hashtags is crucial for the popularity of a message in regards to how quickly messages concerning a certain topic can be found. Therefore, hashtags can also be seen as a way to give a certain amount of "context" to a tweet. However, choosing the best hashtags for a certain message can be a difficult task. Hence, users often feel forced to use multiple hashtags having the same meaning (synonyms), like e.g. for tweets regarding the Tour de France (a world-famous bicycle race in France), one could use `#tdf`, `#tourdefrance`, `#cycling` or `#procycling`. The usage of multiple synonymous hashtags decreases the possible length of the actual content of the tweet as only 140 characters including hashtags are allowed per tweet. Furthermore, the usage of synonyms also motivates other users to cram their messages with hashtags to cover as many searches as possible. To avoid such a proliferation of hashtags, hashtags concerning a certain event are often predefined and propagated to all its participants in order to ensure that the hashtags used for tweets regarding this event are homogeneous. This often leads event organizers (e.g. of conferences) to announce an "official" hashtag. E.g., Tim O' Reilly (@ timoreilly) posted on 2011-03-05: `At Wired Disruptive by Design conference, no hashtag announced. Hmmm.`. Such scenarios could easily be avoided if the tag vocabulary of the folksonomy is kept homogeneous which basically implies that no synonymous hashtags are used.

In this paper we present an approach aiming at supporting the user and creating a more homogeneous set of hashtags within the Twittersphere by facilitating a recommender system for the suggestion of suitable hashtags to the users. We show how the computation of hashtags can be facilitated and prove that this approach is able to provide the user with suitable hashtag recommendations. Furthermore, we focus on how the similarity function used during the computation of recommendations influences the quality of the final recommendations.
The remainder of this paper is organized as follows. Section 2 outlines the characteristics of the data set underlying our evaluations. Section 3 is concerned with the proposed recommendation algorithms. Section 4 features the evaluation of our approach and Section 5 describes related work. The paper concludes with final remarks in Section 6.

[2] according to `http://yearinreview.twitter.com/de/tps.html`

2 Twitter Data Set

The approach presented in this paper and its evaluation are based on an underlying data set of tweets which is used to compute the hashtag recommendations. As there are no large Twitter data sets publicly available, we crawled tweets in order to build up such a data set. The data set was crawled by using the Twitter streaming API[3]. In particular, we made use of the `GET statuses/sample` API function which returns a random sample of all public statuses (tweets) published. However, the API only allows for crawling about 1% of all public statuses (formerly called Spritzer access). Between June 2011 and May 2012 we were able to crawl about 386,000,000 tweets using this crawling strategy. Details about the crawled data set can be found in Table 1.

Characteristic	**Number**	**Percentage**
Crawled messages total	386,917,626	100%
Messages containing one or more hashtags	49,696,615	12.84%
Messages containing no hashtags	337,221,011	87.16%
Retweets	67,995,905	17.57%
Retweets containing one or more hashtags	14,395,494	3.72%
Direct messages, mentions	212,651,505	54.96%

Table 1. Basic Data Set Characteristics

Only 12.84% of the crawled messages contained hashtags. Hence, we were only able to use about 50,000,000 tweets out of the data set for our needs. Table 2 contains information about the hashtagging behavior of users within the data set. Figure 1 depicts the longtail-distribution of hashtag usages. This chart clearly shows that only a very small fraction of hashtags are used with a high frequency whereas the long tail of hashtags are only used very few times. As can be seen in Table 2, the majority of hashtags occur exactly once. This is the case for 5,765,835 hashtags which amounts to a total of 74.14%. This number indicates that the hashtag vocabulary is highly heterogeneous and as such, restricts the search capabilities within Twitter data in regards to hashtag-based search.

3 Hashtag Recommendations

The recommendation of hashtags supports the user during the process of creating a new message. While the user is typing, hashtags appropriate for the already entered message are computed on the fly. With every new keystroke, the recommendations are recomputed and get refined. The top-k recommendations are shown to the user, where k denotes the size of the set of recommended hashtags.

[3] `https://dev.twitter.com/docs/streaming-apis`

Characteristic	Value
Messages containing one or more hashtags	49,696,615
Hashtags usages total	65,612,803
Average number of hashtags per message	0.16
Average number of hashtags per message (within set of tweets containing at least one hashtag)	1.32
Maximum number of hashtags per message	47
Median of hashtags per message	1
Hashtags distinct	7,777,194
Hashtags occurring ≥ 5 times in total	757,832
Hashtags occurring < 5 times in total	7,135,627
Hashtags occurring < 3 times in total	6,841,523
Hashtags occurring once	5,765,835
Average number of usages per hashtag	8.43
Median number of usages per hashtag	1

Table 2. Overview Hashtags in Data Set

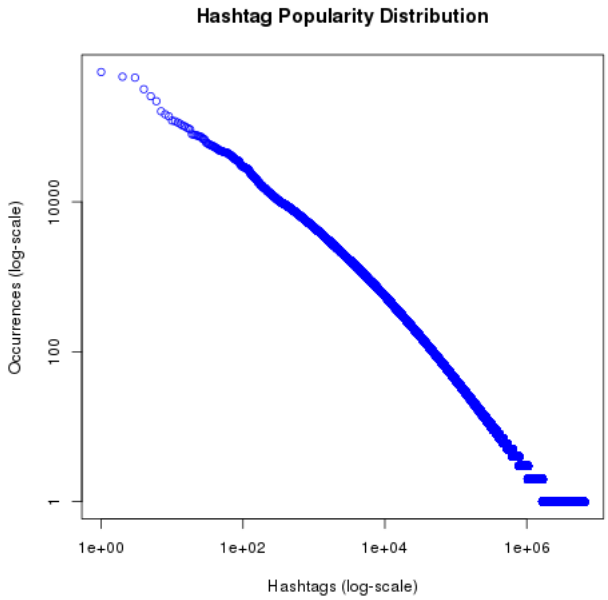

Fig. 1. Longtail Distribution of Hashtag Usages

Due to the fact that both the cognition of the user and the space available for displaying the recommendations is limited, the size of the set of recommended hashtags is restricted. In most cases a set of 5–10 recommendations is most appropriate which also corresponds to the capacity of short-term memory [28] and furthermore, can be perceived very quickly in a user interface. The problem of choice overload has also been addressed by Bollen *et al.* [5] who state that top-5 recommendations are easy to choose from by the user. For a given message (or part of it), the computation of these recommendations based on the underlying data set comprises the following steps which are also illustrated in Figure 2.

1. For a given input message (or a part of it) which is entered by the user:
2. Retrieve the most similar messages featuring hashtags from the data set.
3. Extract the hashtags contained in the top-n similar messages. These hashtags constitute the *hashtag recommendation candidate set*.
4. Rank the recommendation candidates computed in step 2 according to the ranking methods proposed in this paper.
5. Present the ranked top-k hashtags to the user.

These steps are described in detail in the following sections.

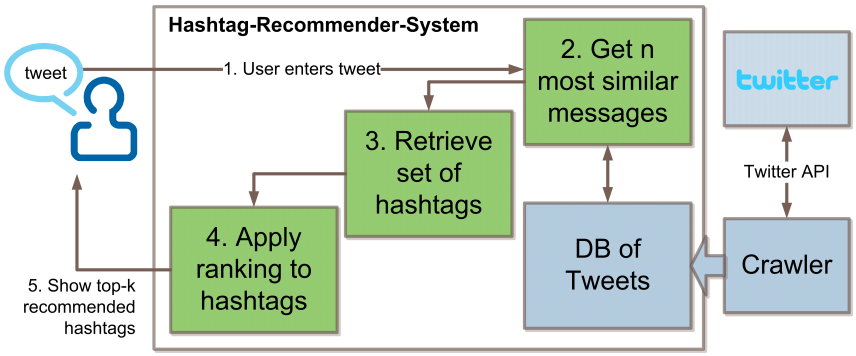

Fig. 2. Workflow: Hashtag Recommendation Computation

3.1 Similarity of Messages

The similarity of the input message and the messages contained in the data set is crucial for the further computation of recommendation candidates and also the ranking of these candidates. Hence, we evaluated the following text similarity functions:

- Cosine similarity on TF-IDF weighted vectors [19, 25]
- Cosine similarity on BM25 Okapi weighted vectors [25, 34]
- Dice coefficient [9]
- Jaccard coefficient [16]
- Levenshtein distance [22]

Cosine Similarity The cosine similarity function is one of the predominant measures in Information Retrieval. It is based on the (weighted) term vectors of both the query and the respective document which is to be compared to the query. In the case of searching for the best matching message for a certain input message, the query is defined by the input message and the cosine similarity is computed for the input message and every single message contained in the database. The definition of cosine similarity between vectors (i.e. the angle between these two vectors) is defined as shown in Equation 1 where v_i is the vector representing the input message (the query) and v_j is vector representing the reference message from the database.

$$cos(v_i, v_j) = \frac{v_i \cdot v_j}{\|v_i\| \, \|v_j\|} \tag{1}$$

Closely related to the cosine similarity of vectors is the weighting of terms [36] which allows for a weighting of each single term. We chose to evaluate the cosine similarity of weighted term vectors computed by two different weighting schemes: TF-IDF and the BM25 Okapi weighting scheme.

The traditional *term frequency-inverse document frequency* (TF-IDF) weighting scheme aims at estimating the relevance of a certain term in relation to the whole document corpus (in our case the reference data set). This function is based on two components: term frequency (TF) and inverse document frequency (IDF). TF can be defined as the number of occurrences of the given term t_i within the current document d. The IDF component basically defines how relevant a term is in relation to the whole set of documents, as can be seen in Equation 2 where $|D|$ is the total number of documents within the reference database and $n(t_i)$ is the number of documents which contain the given term t_i.

$$IDF(t_i) = log \frac{|D|}{n(t_i)} \tag{2}$$

These two components are then combined to the TF-IDF weight of a given term as can be seen in Equation 3.

$$TF - IDF(t_i, d) = TF(t_i, d) \cdot IDF(t_i) \tag{3}$$

The second weighting scheme we made use of is the BM25 Okapi weighting scheme [34] which additionally also incorporates the average length of a document and two tuning factors. BM25 is computed based on a similar definition of the inverse document frequency as can be seen in Equation 4, where $n(t_i)$ again is the number of documents which contain the given term t_i and $|D|$ is the total number of documents within the reference data set.

$$IDF(t_i) = log \, \frac{|D| - n(t_i) + 0.5}{n(t_i) + 0.5} \tag{4}$$

The BM25 Okapi weight of a given term in relation to the reference data set can subsequently be computed as in Equation 5, where $|D|$ is the total number of documents in the reference data set, $f(t_i, d)$ is the number of occurrences of term

t_i in the document d, $|d|$ is the length of document d and $avgLen$ is the average length of all documents within the reference data set. Furthermore, two tuning parameters are used: l_1 weights the influence of the document term frequency, whereas the factor b determines the weight length scaling. [34] propose to set l_1 to 1.2 and to set b to 0.75.

$$BM25(t_i, d) = IDF(t_i) \cdot \frac{f(t_i, d) \cdot (l_1 + 1)}{f(t_i, d) + l_1 \cdot (1 - b + b \cdot \frac{|d|}{avgLen})} \tag{5}$$

Dice Coefficient The Dice coefficient is a set-based similarity function. It models string similarity as the similarity of the set-interpretation of both the query string and the actual document. The function is defined as the fraction between the overlap of terms contained in both the query and the sum of the lengths of the two documents. The coefficient can be computed as in Equation 6, where t_i and t_j are the respective sets of words contained in the corresponding message.

$$dice(t_i, t_j) = \frac{|t_i \cap t_j|}{|t_i| + |t_j|} \tag{6}$$

Jaccard Similarity Coefficient The Jaccard Similarity Coefficient is also based on the bag-of-words approach. Equation 7 shows the computation of the coefficient, where t_i and t_j are the respective sets of words contained in the corresponding documents (messages).

$$jaccard(t_i, t_j) = \frac{|t_i \cap t_j|}{|t_i \cup t_j|} \tag{7}$$

Levenshtein Distance The Levenshtein distance [23] (also known as edit-distance) is a lexical measure which is defined as the number of edit operations (edit, deletion or insertion) required for a certain string t_i to be turned into another string t_j. Hence, identical strings feature a Levenshtein distance of 0, whereas e.g. the strings "tweet" and "tweak" feature an edit distance of 2 as two characters have to be edited.

3.2 Ranking

The ranking of the hashtag recommendation candidates is a crucial part of the hashtag recommendation process as only the top-k (with k between 5 and 10) hashtags are shown to the user. Therefore, we propose three ranking methods for the recommendation of hashtags. In the following, $\mathcal{T}$ denotes the crawled data set containing all messages and $\mathcal{C}_T$ is the candidate set consisting of the n messages which are most similar to the input message t_{input}. The function $contains(t, h)$ returns 1 if the specified hashtag h is present in the specified message t and 0 if it cannot be found in the message text.

– **ScoreRank** is based on the similarity values of the input message t_{input} and the messages containing the hashtag recommendation candidates $\mathcal{C}_T$. The similarity scores computed by the different measurements described in Section 3.1 are directly used for the ranking of the recommendation candidates. If the hashtag candidate is present in more than one similar message in the candidate set, the highest similarity score is used for further computations.

$$\begin{aligned} score(h) &= max(\{sim(t_c, t_{input}) \mid t_c \in \mathcal{C}_T \wedge h \in t_c\}) \\ \text{where } sim &\in \{cos\_tfidf, cos\_BM25, jaccard, dice, levenshtein\} \end{aligned} \tag{8}$$

– **RecCountRank** is based on the popularity of hashtags within the hashtag recommendation candidate set $\mathcal{C}_T$. This basically means that the more entries in the result set of similar messages contain a certain hashtag, the more suitable the hashtag might be.

$$score(h) = \sum_c contains(t_c, h) \qquad \text{where } \; t_c \in \mathcal{C}_T \tag{9}$$

– **GlobalPopularityRank** is based on the global popularity of hashtags within the whole underlying data set. As only a few hashtags are used at a high frequency, it is likely that such a popular hashtag matches the message entered by the user. Therefore, ranking the overall most popular hashtags from within the candidate set higher is also a suitable approach for the ranking of hashtags. We consider this method as a baseline ranking.

$$score(h) = \sum_i contains(t_i, h) \qquad \text{where } \; t_i \in \mathcal{T} \tag{10}$$

The ranking is performed based on a ranking score which is computed for each hashtag in the candidate set. After the computation of the ranking scores, all suitable hashtag candidates of set $\mathcal{C}_H$ are subsequently ranked in descending order of the score to compute the final ranking.

4 Evaluation

The evaluation was conducted based on an evaluation framework we implemented in Java and was based on the data set described in Section 2. The evaluation was performed on a 8-core machine with 32 GB of RAM on CentOS release 5.1.
Essentially, we performed leave-one-out tests [8] on the collected tweets in order to evaluate our approach. For our tests we only used tweets which contain less than 6 hashtags to exclude possible spam messages. Furthermore, we did not use any retweets for the evaluation as these would lead to hashtag recommendations based on identical messages and would therefore distort our evaluation.

The performed leave-one-out tests are sketched in Algorithm 1 and each of these steps was performed for 1,000 test messages.

```
Data: Set τ of all tweets within the data set
Result: Evaluation of Recommendation Algorithm
begin
    // Initialisation
    randomTweet, inputText := null
    hashtagRecommendations, evaluationResults, hashtags := { }
    numberOfCorrectRecommendations := 0
    // Get random tweet from τ
    randomTweet = getRandomTweet(τ)
    hashtags = extractHashtags(randomTweet)
    inputText = removeHashtags(randomTweet)
    // Get recommendations (see Section 3 for details)
    hashtagRecommendations = getRecommendations(inputText)
    // Evaluate Recommended Hashtags
    foreach r within hashtagRecommendations do
        if r ∈ hashtags then
            numberOfCorrectRecommendations++
        end
    end
    evaluationResult = computeMetrics(
        inputText,
        hashtagRecommendations,
        numberOfCorrectRecommendations)
    return evaluationResult
end
```

Algorithm 1: Basic Evaluation Algorithm

As for the parameters of the BM25 Okapi weighting function, we chose to use $k = 0.75$ and $b = 1.2$ as proposed in [34]. The ranking of the top-n most similar tweets was performed based on the proposed similarity functions and n was set to 500. I.e., the hashtag recommendation candidates are extracted from the 500 most similar messages.

In order to determine the quality and suitability of the recommendations of hashtags provided to the users, we chose to apply the traditional IR-metrics recall and precision. As a hashtag recommendation system should be aiming at providing the user with an optimal number of correct tags, the recall value is the most important quality measure for our approach.

Figures 3, 4 and 5 show the recall@k ($k = 1, 3, 5, 10, 15, 20$) plot of the recall values of the basic ranking methods and the different similarity functions. The performance of the Dice and Jaccard coefficients is equal as these two similarity measures are monotonically related. I.e., the ranking of hashtags computed based upon these two similarity measures is always equal. Therefore, we chose to only

incorporate the Jaccard similarity coefficient in our evaluations throughout the remainder of this chapter.

The good performance of the ScoreRank (as can bee seen in Figure 3) can be explained by the fact that the message in which the hashtag recommendation candidate is embedded in is directly related to the relevance of the hashtag. The other ranking methods are based on the popularity of hashtags (either locally or globally) which are only loosely coupled to the message it is contained in. It can be seen that already five shown hashtags are sufficient to get a reasonable recall value of about 22% and therefore allow to build a lightweight recommendation interface without overwhelming the user by too many recommendations. Figure 4 shows the recall@k values for the RecCountRank ranking method. ScoreRank performs better and hence proves that the similarity of the messages in which the hashtag recommendation candidates are embedded in are more relevant for the quality of recommendations as simply the number of tweets such recommendation candidates stem from. As can be seen in Figure 5, GlobalPopularityRank is not very suitable as the major ranking function, however the popularity of hashtags can be used as a boosting score which we already showed in [44]. Furthermore, if no similar tweets for an input tweet can be retrieved (hence, no customized recommendations can be provided), the overall most popular hashtags can be recommended as a baseline set of recommendations. A time-sensitive ranking aiming at resembling hashtags about trending topics would furthermore enhance this baseline strategy.

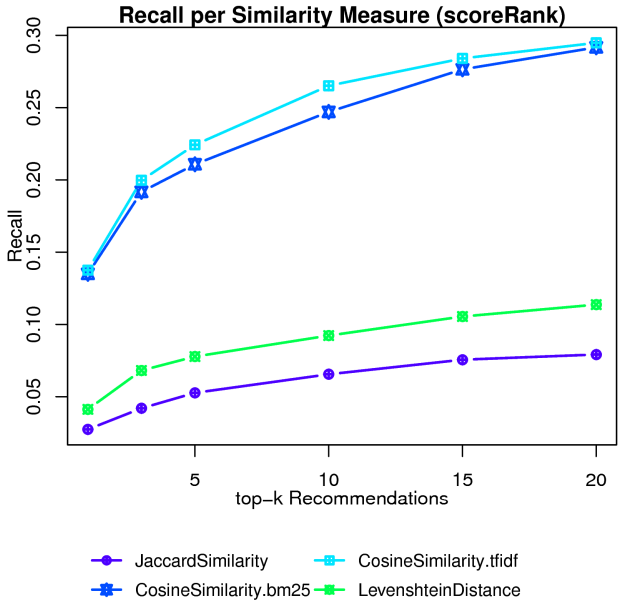

Fig. 3. ScoreRank Recall

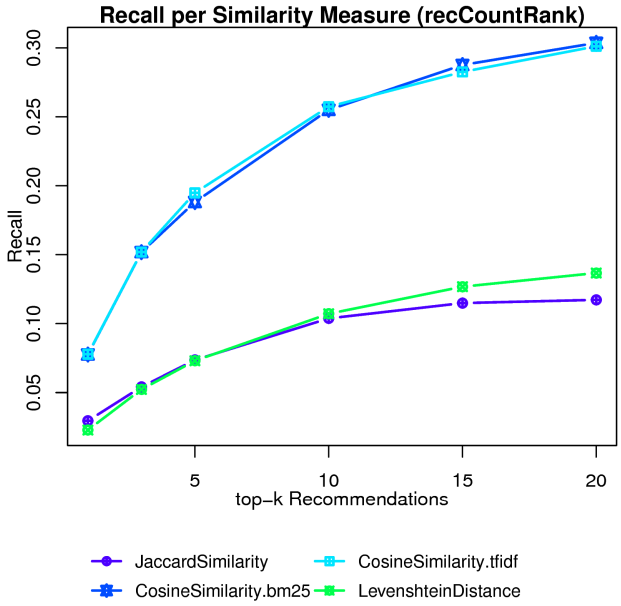

Fig. 4. RecCountRank Recall

From these figures we can observe that the best performing similarity measure is the cosine similarity function regardless of which weighting schema was applied to the term vectors. This can be lead back to the weighting of terms as this allows for a lowering of the influence of very frequently occurring terms which presumably are not that relevant while at the same time increasing the influence of less frequently occurring terms assuming that these are more relevant. Cosine similarity with BM25 and TF-IDF weighting using ScoreRank achieved a recall value of 26.52% (TF-IDF) respectively 24.41% (BM25) for $k = 10$. The bag-of-words based approaches Dice and Jaccard are able to reach recall values of 6.20% for $k = 10$. As for the recall values of the Levenshtein distance, the recall values of e.g. 8.72% at $k = 10$ can be explained by the fact that Levenshtein is a positional character-based distance metric and as such, two texts are equal if all characters are equal at the same position. I.e. the tweets "armstrong for gold" and "gold for armstrong" feature a high Levenshtein distance as the positions of the characters inspected are also featured in the distance metric. In contrast, these two tweets would have been assessed as equal by the bag-of-words based approaches Dice and Jaccard. The advantage of the Levenshtein distance is the flexibility regarding typos, alternative spellings and word forms which cannot be handled by bag-of-words based approaches as they need full accordance between words. Especially in microposts—due to the limitation of the length of messages—abbreviations and other mutilation of words are used

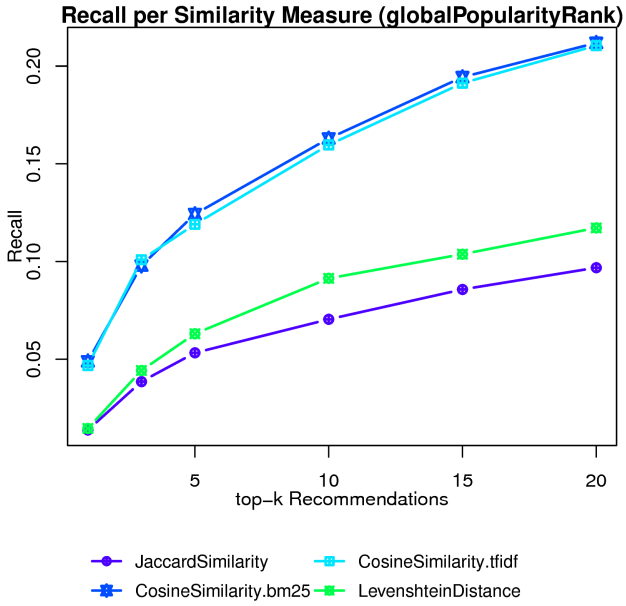

Fig. 5. GlobalPopularityRank Recall

which is one mayor reason for the better performance of Levenshtein distance based approach.

The ranking function performing best in regards to the precision values is again ScoreRank as can be seen in Figure 6. For ScoreRank, the highest precision was achieved at $k = 1$ by the cosine similarity measures achieving precision values of about 17%. The low precision values for $k > 2$ can be lead back to the fact that the Twitter messages in our reference data set contain 1.32 hashtags on average per message. Assume that a tweet originally contained two hashtags. Even when recommending five or more hashtags and the two original hashtags were correctly recommended, the precision value naturally is very low as three of the recommendations did not match the given hashtags. Hence, the precision value drops significantly with $k > 2$. However, it is important to note that the recommended hashtags not matching the original hashtags are not necessarily unsuited as the original hashtags only constitute a baseline for the quality of recommendations.

Furthermore, in [44] we discussed the hybridization of ranking functions which can lead to improved recommendations. The results showed that the combination of ScoreRank and GlobalPopularityRank performed the best. Such a combined approach is also feasible with all described similarity measures in this paper.

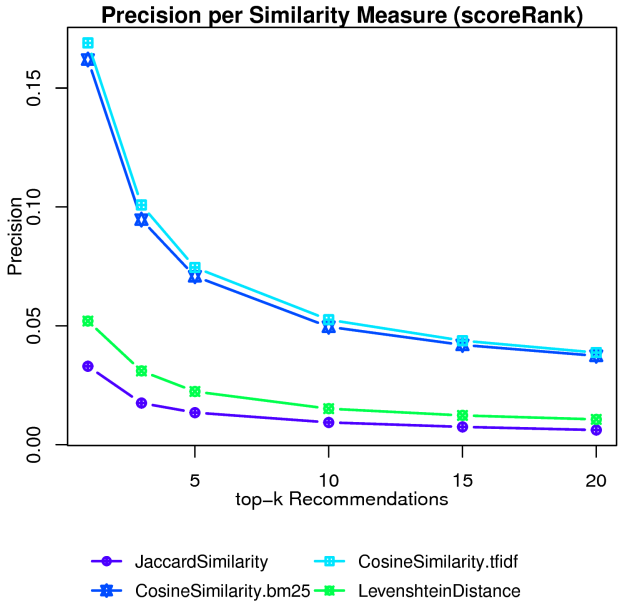

Fig. 6. ScoreRank Precision

5 Related Work

The recommendation of hashtags within the Twittersphere is closely related to the field of microblogging, tagging in Web 2.0 applications and the field of recommender systems as a whole. Tagging of online resources has become popular with the advent of Web 2.0 paradigms. However, the task of recommending traditional tags differs considerably from recommending hashtags. Our recommendation approach is solely based on 140 characters whereas in traditional tag recommender systems, much more data is taken into consideration for the computation of tag recommendations. Furthermore, tweets, hashtags and trends within the Twittersphere are changing at a fast pace and are very dynamic. New hashtags may evolve around trending topics and therefore, the recommendations have to consider this dynamic nature of Twitter. Nevertheless, the described ScoreRank based on vector based similarity measures already performs very well and computes highly suitable hashtag recommendations to guide and support the users.

Sigurbjörnsson *et al.* [41] presented an approach for the recommendation of tags within Flickr which was based on the co-occurrence of tags (also used in [12,24]). Two different tags co-occur if they are both used for the same photo. Based on this information about the co-occurrence of tags for Flickr photos, the authors developed a prototype which is able to recommend hashtags for photos which have been partly tagged. This recommendation is computed by finding those tags which have been used together with the tag the user already specified for a certain photo. These tags are subsequently ranked and recommended to

the user. It is important to note that such an approach is not feasible if a photo has not been tagged at all. Partly based on this work, Rae *et al.* [32] proposed a method for Flickr tag recommendations which is based on different contexts of tag usage. Rae distinguishes four different context which are used for the computation of recommendations: (i) the user's previously used tags, (ii) the tags of the user's contacts, (iii) the tags of the users which are members of the same groups as the user and (iv) the most popular tags within the whole community. A similar approach has also been facilitated by Garg and Weber in [11]. Furthermore, on the BibSonomy platform which basically allows its users to add bibliographic entries the users are provided with recommendations for suitable tags annotating these entries [24]. This approach extracts tags which might be suitable for the entry from the title of the entry, the tags previously used for the entry and tags previously used by the current user. Based on these resources, the authors propose different approaches for merging these sets of tags. The resulting set is subsequently recommended to the user. Tag recommendations based on Moviebase data has been presented in [40]. Jäschke *et al.* [17] propose a collaborative filtering approach for the recommendation of tags. The authors therefore construct a graph based on the users, the tags and the tagged entities. Within these graphs, the recommendations are computed and ranked based on a PageRank-like ranking algorithm for folksonomies. Recommendations based on the content of the entity which has to be tagged have been studied in [42]. Additionally, there have been numerous papers concerned with the analysis of the tagging behavior and motivation of users, e.g. in [2, 26].

The social aspects within social online media, such as the Twitter platform, has been analyzed heavily throughout the last years. These analysis were concerned with the motivations behind tweeting, like e.g. in [18]. Boyd *et al.* [6] showed how users make use of the retweet function and why users retweet at all. Honeycutt and Hering examined how direct Twitter messages can be used for online collaboration [14]. Recently, the work by Romero *et al.* [35] analyzed how the exposure of Twitter users to hashtags affects their hashtagging behavior and how the use of certain hashtags is spread within the Twittersphere. The authors found that the adoption of hashtags is dependent on the category of the tweet. E.g. hashtags concerned with politics or sports are adopted faster than hashtags concerned with any other topic category. Further analysis of Twitter data and the behavior of Twitter users can be found in [15, 20, 21, 43].

As for the recommendation of items within Twitter or based on Twitter data, there have been numerous approaches dealing with these matters. Hannon *et al.* [13] propose a recommender system which provides users with recommendations for users who might be interesting to follow. Chen *et al.* present an approach aiming at recommending interesting URLs to users [7]. The work by Phelan, McCarthy and Smyth [31] is concerned with the recommendation of news to users.

Traditionally, recommender systems are used in e-commerce where users are provided with recommendations for interesting products, like e.g. on the Amazon website. Recommendations are typically computed based on one of the follow-

ing two approaches: (i) a collaborative filtering approach [1, 33] which is based on finding similar users with a similar behavior for the recommendation of e.g. tags used by these users and (ii) a content-based approach [4, 30] which aims at finding items having the most similar characteristics as the items which have already been used by the user.
However, to the best of our knowledge, there is currently no other approach aiming at the recommendation of tags in microblogging platforms and hashtags for a certain Twitter message. As for the similarity of messages and hence (possibly short) texts, various approaches have been facilitated which often are closely related to a classification task of texts. Surveys about this task and the related similarity measures can be found in [3, 39]. Besides the syntactic similarity of texts, also the semantic similarity of texts has been investigated. The work by Mihalcea *et al.* [27] provides a good overview and comparison about various different approaches for computing word semantic similarity. Also, semantic similarity has been computed based on the incorporation of external sources such as Wikipedia, as e.g. in [10]. The authors make use of semantic concepts extracted from Wikipedia (and represented as vectors). Subsequently, the vectors of both a query and each document in the corpus are compared and used for the estimation of the similarity of these both documents. Furthermore, Schedl *et al.* provide a thorough overview about different string similarity functions in regards to their suitability for the automatic extraction of named entities especially music artist's names from tweets [37] and web pages [38]. Nishida *et al.* [29] model the problem of tweet classification as a data compression problem. Their approach is based on the compressibility of a given tweet in relation to a negative and a positive corpus. The higher the compressibility for a tweet is for the respective corpus, the more likely it is that the tweet is similar to the given corpus and hence can be classified into the given category.

6 Conclusion

In this paper we presented an approach aiming at the recommendation of hashtags to microblogging users. Such recommendations help the user to (i) use more appropriate hashtags and therefore to homogenize the set of hashtags and (ii) encourage the users to use hashtags as suitable hashtags recommendations are provided. The approach is based on analyzing messages similar to the message the user currently enters and deducing a set of hashtag recommendation candidates from these microblog messages. We evaluated multiple similarity measures and showed that vector-based similarity functions, such as cosine similarity with TF-IDF weighted entries, are the most appropriate similarity measure for the task of retrieving similar microposts. We furthermore presented different ranking techniques for the hashtags of the recommendation candidates. In combination with a ranking function based on the previously computed similarity score, the best results in terms of precision and recall of the recommended hashtags were achieved. The evaluations we conducted showed that our approach is capable of providing users with suitable recommendations for hashtags reaching

recall values of about 22% when presenting only 5 recommendations. Thus, this results are the basis to build a lightweight recommendation interface without overwhelming the user by too many recommendations. The presented approach was already used to implement a first prototype of such a hashtag-recommender-system which will be released under an open source license in the future. Thus, future work will also include user studies to evaluate the user acceptance of such a system. Further optimizations regarding the similarity of messages, i.e. the incorporation of semantic distance measures and the exploitation of public knowledge bases are planned. As such, for example Wikipedia concepts could be added in order to find more semantically similar messages. Also, the extraction of named entities might add to a better performance of recommendations as messages could be compared in regards to overlapping named entities which would better resemble the semantics of the message. Furthermore, the resolution of synonymous terms (e.g. via WordNet) is also part of future work. Another very valuable source are links to websites which are used very often in microblog messages. By analyzing the targeted websites and their topics, further information about the messages can be gathered. In regards to Twitter, future work also features incorporating the social graph of Twitter users into the process of computing recommendations for hashtags to optimize the presented hashtag recommendations, e.g. by ranking hashtags used by followers or followees higher. Furthermore, as users tend to re-use hashtags they already made use of, an analysis of the hashtags previously used by the user and a subsequent incorporation of these hashtags into the recommendation process are also future tasks.

References

1. G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE transactions on knowledge and data engineering*, 17(6):734–749, 2005.
2. M. Ames and M. Naaman. Why we tag: motivations for annotation in mobile and online media. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '07, pages 971–980, New York, NY, USA, 2007. ACM.
3. R. Baeza-Yates, B. Ribeiro-Neto, et al. *Modern information retrieval*, volume 82. Addison-Wesley New York, 1999.
4. M. Balabanović and Y. Shoham. Fab: content-based, collaborative recommendation. *Commun. ACM*, 40:66–72, March 1997.
5. D. Bollen, B. P. Knijnenburg, M. C. Willemsen, and M. Graus. Understanding choice overload in recommender systems. In *Proceedings of the fourth ACM conference on Recommender systems*, RecSys '10, pages 63–70, New York, NY, USA, 2010. ACM.
6. D. Boyd, S. Golder, and G. Lotan. Tweet, tweet, retweet: Conversational aspects of retweeting on twitter. In *hicss*, pages 1–10. IEEE Computer Society, 1899.
7. J. Chen, R. Nairn, L. Nelson, M. Bernstein, and E. Chi. Short and tweet: experiments on recommending content from information streams. In *Proceedings of the 28th international conference on Human factors in computing systems*, pages 1185–1194. ACM, 2010.

8. P. Cremonesi, R. Turrin, E. Lentini, and M. Matteucci. An evaluation methodology for collaborative recommender systems. In *Automated solutions for Cross Media Content and Multi-channel Distribution, 2008. AXMEDIS'08. International Conference on*, pages 224–231. IEEE, 2008.
9. L. Dice. Measures of the amount of ecologic association between species. *Ecology*, 26(3):297–302, 1945.
10. E. Gabrilovich and S. Markovitch. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *Proceedings of the 20th international joint conference on artificial intelligence*, volume 6, page 12. Morgan Kaufmann Publishers Inc., 2007.
11. N. Garg and I. Weber. Personalized, interactive tag recommendation for flickr. In *Proceedings of the 2008 ACM conference on Recommender systems*, RecSys '08, pages 67–74, New York, NY, USA, 2008. ACM.
12. W. Gassler, E. Zangerle, and G. Specht. The Snoopy Concept: Fighting Heterogeneity in Semistructured and Collaborative Information Systems by using Recommendations. In *The 2011 International Conference on Collaboration Technologies and Systems (CTS 2011)*, Philadelphia, PE, 05/2011 2011.
13. J. Hannon, M. Bennett, and B. Smyth. Recommending twitter users to follow using content and collaborative filtering approaches. In *RecSys '10: Proceedings of the fourth ACM conference on Recommender systems*, pages 199–206, New York, NY, USA, 2010. ACM.
14. C. Honeycutt and S. C. Herring. Beyond Microblogging: Conversation and Collaboration via Twitter. In *HICSS*, pages 1–10. IEEE Computer Society, 2009.
15. B. Huberman, D. Romero, and F. Wu. Social networks that matter: Twitter under the microscope. *First Monday*, 14(1):8, 2009.
16. P. Jaccard. Étude Comparative de la Distribution Florale dans une Portion des Alpes et des Jura. *Bulletin del la Société Vaudoise des Sciences Naturelles*, 37:547–579, 1901.
17. R. Jaeschke, L. Marinho, A. Hotho, L. Schmidt-Thieme, and G. Stumme. Tag Recommendations in Folksonomies. In J. Kok, J. Koronacki, R. Lopez de Mantaras, S. Matwin, D. Mladenic, and A. Skowron, editors, *Knowledge Discovery in Databases: PKDD 2007*, volume 4702 of *Lecture Notes in Computer Science*, pages 506–514. Springer Berlin / Heidelberg, 2007.
18. A. Java, X. Song, T. Finin, and B. Tseng. Why we twitter: understanding microblogging usage and communities. In *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis*, pages 56–65. ACM, 2007.
19. K. Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 28(1):11–21, 1972.
20. B. Krishnamurthy, P. Gill, and M. Arlitt. A few chirps about twitter. In *Proceedings of the first workshop on Online social networks*, pages 19–24. ACM, 2008.
21. H. Kwak, C. Lee, H. Park, and S. Moon. What is Twitter, a social network or a news media? In *Proceedings of the 19th international conference on World wide web*, pages 591–600. ACM, 2010.
22. V. Levenshtein. Binary Codes with Correction for Deletions and Insertions of the Symbol 1. *Problemy Peredachi Informatsii*, 1(1):12–25, 1965.
23. V. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710, 1966.
24. M. Lipczak and E. Milios. Learning in efficient tag recommendation. In *Proceedings of the fourth ACM conference on Recommender systems*, RecSys '10, pages 167–174, New York, NY, USA, 2010. ACM.

25. C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, Cambridge, UK, 2008.
26. C. Marlow, M. Naaman, D. Boyd, and M. Davis. HT06, tagging paper, taxonomy, Flickr, academic article, to read. In *Proceedings of the seventeenth conference on Hypertext and hypermedia*, HT '06, pages 31–40, New York, NY, USA, 2006. ACM.
27. R. Mihalcea, C. Corley, and C. Strapparava. Corpus-based and knowledge-based measures of text semantic similarity. In *Proceedings of the national conference on artificial intelligence*, volume 21, page 775. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2006.
28. G. Miller. The magical number seven, plus or minus two: some limits on our capacity for processing information. *Psychological review*, 63(2):81, 1956.
29. K. Nishida, R. Banno, K. Fujimura, and T. Hoshide. Tweet classification by data compression. In *Proceedings of the 2011 international workshop on DETecting and Exploiting Cultural diversiTy on the social web*, pages 29–34. ACM, 2011.
30. M. Pazzani and D. Billsus. Content-Based Recommendation Systems. In P. Brusilovsky, A. Kobsa, and W. Nejdl, editors, *The Adaptive Web*, volume 4321 of *Lecture Notes in Computer Science*, pages 325–341. Springer Berlin / Heidelberg, 2007.
31. O. Phelan, K. McCarthy, and B. Smyth. Using twitter to recommend real-time topical news. In *Proceedings of the third ACM conference on Recommender systems*, pages 385–388. ACM, 2009.
32. A. Rae, B. Sigurbjörnsson, and R. van Zwol. Improving tag recommendation using social networks. In *Adaptivity, Personalization and Fusion of Heterogeneous Information*, RIAO '10, pages 92–99, Paris, France, France, 2010. Le Centre de Hautes Etudes Internationales d'Informatique Documentaire.
33. P. Resnick and H. Varian. Recommender systems. *Communications of the ACM*, 40(3):58, 1997.
34. S. E. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, and M. Gatford. Okapi at TREC-3. In *Proceedings of the Text Retrieval Conference (TREC)*, pages 109–126, Gaithersburg, MD, USA, 1994. National Institute of Standards and Technology.
35. D. M. Romero, B. Meeder, and J. M. Kleinberg. Differences in the mechanics of information diffusion across topics: idioms, political hashtags, and complex contagion on twitter. In S. Srinivasan, K. Ramamritham, A. Kumar, M. P. Ravindra, E. Bertino, and R. Kumar, editors, *WWW*, pages 695–704. ACM, 2011.
36. G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513 – 523, 1988.
37. M. Schedl. On the use of microblogging posts for similarity estimation and artist labeling. In J. S. Downie and R. C. Veltkamp, editors, *ISMIR*, pages 447–452. International Society for Music Information Retrieval, 2010.
38. M. Schedl. # nowplaying madonna: a large-scale evaluation on estimating similarities between music artists and between movies from microblogs. *Information Retrieval*, pages 1–35, 2012.
39. F. Sebastiani. Machine learning in automated text categorization. *ACM computing surveys (CSUR)*, 34(1):1–47, 2002.
40. S. Sen, J. Vig, and J. Riedl. Tagommenders: connecting users to items through tags. In *Proceedings of the 18th international conference on World wide web*, WWW '09, pages 671–680, New York, NY, USA, 2009. ACM.
41. B. Sigurbjörnsson and R. Van Zwol. Flickr tag recommendation based on collective knowledge. In *Proceeding of the 17th international conference on World Wide Web*, pages 327–336. ACM, 2008.

42. M. Tatu, M. Srikanth, and T. D'Silva. *RSDC'08: Tag Recommendations using Bookmark Content*. Workshop at 18th Europ. Conf. on Machine Learning (ECML'08) / 11th Europ. Conf. on Principles and Practice of Knowledge Discovery in Databases (PKDD'08), 2008.
43. S. Ye and S. Wu. Measuring Message Propagation and Social Influence on Twitter. com. In *Social Informatics: Second International Conference, Socinfo 2010, Laxenburg, Austria, October 27-29, 2010, Proceedings*, page 216. Springer-Verlag New York Inc, 2010.
44. E. Zangerle, W. Gassler, and G. Specht. Using Tag Recommendations to Homogenize Folksonomies in Microblogging Environments. In L. Bolc, M. Makowski, and A. Wierzbicki, editors, *Social Informatics - Third International Conference, SocInfo 2011, Singapore, Singapore, October 6-8, 2011. Proceedings*, volume 6430 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2011.