# How Different are Language Models and Word Clouds?

Rianne Kaptein[1], Djoerd Hiemstra[2], and Jaap Kamps[1,3]

[1] Archives and Information Studies, University of Amsterdam, the Netherlands
[2] Database Group, University of Twente, Enschede, the Netherlands
[3] ISLA, Informatics Institute, University of Amsterdam, the Netherlands

**Abstract.** Word clouds are a summarised representation of a document's text, similar to tag clouds which summarise the tags assigned to documents. Word clouds are similar to language models in the sense that they represent a document by its word distribution. In this paper we investigate the differences between word cloud and language modelling approaches, and specifically whether effective language modelling techniques also improve word clouds. We evaluate the quality of the language model using a system evaluation test bed, and evaluate the quality of the resulting word cloud with a user study. Our experiments show that different language modelling techniques can be applied to improve a standard word cloud that uses a TF weighting scheme in combination with stopword removal. Including bigrams in the word clouds and a parsimonious term weighting scheme are the most effective in both the system evaluation and the user study.

## 1 Introduction

This paper investigates the connections between tag or word clouds popularised by Flickr and other social web sites, and the language models as used in IR. Fifty years ago Maron and Kuhns [14] suggested a probabilistic approach to index and search a mechanised library system. Back then, documents were indexed by a human cataloguer who would read a document and then assign one or several indexing terms from a controlled vocabulary. Problems with this approach were the ever increasing amount of documentary data and the semantic noise in the data. The correspondence between a document and its index terms is not exact, because the meaning of terms are a function of their setting. The meaning of a term in isolation is often quite different when it appears in an environment (sentence, paragraph, etc.) of other words. Also, word meanings are individual and can vary from person to person. Because of these problems, Maron and Kuhns [14] proposed to, instead of having a human indexer decide on a yes-no basis whether or not a given term applies for a particular document, assign weights to index terms to more accurately characterise the content of a document. Since then the information retrieval community has developed many models to automatically search and rank documents. In this paper we focus on the language modelling approach. We choose this approach because it is conceptually simple and it is based on the assumption that users have some sense of the frequency of words and which words distinguish documents from others in the collection [16].

The new generation of the Internet, the social Web, allows users to do more than just retrieve information and engages users to be active. Users can now add tags to categorise web resources and retrieve your own previously categorised information. By sharing


**Fig. 1.** Word cloud from 10 results for the topic "diamond smuggling"

these tags among all users large amounts of resources can be tagged and categorised. These generated user tags can be visualised in a tag cloud where the importance of a term is represented by font size or colour. Terms in a tag cloud usually link to a collection of documents that are associated with that tag. To generate tag clouds the tripartite network of users, documents and tags [10] can be exploited. Of course, the majority of documents on the web is not tagged by users. An alternative to clouds based on user-assigned tags, is to generate tags automatically by using statistical techniques. Clouds generated by automatically analysing the document contents are referred to as 'word clouds'. Word clouds have for example been generated for the inaugural speeches of American presidents [e.g., 13]. Word clouds can be used in the same way as tag clouds, but are especially useful to get a first impression of long documents, such as books, or parliamentary proceedings. Also word clouds can be used to summarize a collection of documents, such as clustered or aggregated search results. Figure 1 shows a word cloud summarising top 10 retrieved documents.

This paper investigates the connections between tag or word clouds and the language models of IR. Our main research question is: do words extracted by language modelling techniques correspond to the words that users like to see in word clouds? We discuss related work on tag clouds and language modelling in Section 2 with the goal of determining which specific techniques have been explored in both approaches. We decide to focus on four different features of word clouds, i.e. pseudo-relevance vs. relevance information, stemming, including bigrams, and term weighting schemes. Each of them is investigated in a separate section (Sections 4 through 7). In Section 3 we describe our experimental set-up. We use an IR test collection to evaluate the effectiveness of the technique for language models, and we conduct a user study establishing user preferences over the resulting word clouds as a means to convey the content of a set of search results. Finally, in Section 8 we draw our conclusions.

## 2 Related Work

In this section, we will discuss related work on tag/word clouds and language modelling, with the aim of determining a number of techniques applicable for both types of approaches. The first appearance of a tag cloud is attributed to Douglas Coupland's novel Microserfs [4]. In this novel the main character Daniel writes a program to take terms out of his journal entries and create snapshots of keywords, which are called 'subconscious files.' The first widespread use of tag clouds was on the photo-sharing site Flickr. Other sites that contributed to the popularisation of tag clouds were Del.ici.ous and Technorati. Nowadays tag clouds are often considered as one of the typical design elements of the social Web. Evaluation of tag clouds appears scarcely in scientific literature, in the blogosphere however there is a lot discussion on the usefulness of tag clouds [2]. Part of the evaluation of tag clouds are the effects of visual features such as font

size, font weight, colour and word placement [1, 6, 18]. Font size and font weight are considered the most important visual properties. Font sizes are commonly set to have a linear relationship to the log of the frequency of occurrence of a tag. Colour draws the attention of users, but the meaning of colours needs to be carefully considered. The position of the words is important, words in the top of the tag cloud attract more attention. An alphabetical order of the words helps users to find information quicker. Rivadeneira et al. [18] identify four tasks tag clouds can support. In our experiments we will evaluate our word clouds on the basis of these tasks:

- Search: locating a specific term that represents a desired concept.
- Browsing: casually explore the cloud with no specific target in mind.
- Impression Formation or Gisting: use the cloud to get a general idea on the underlying data.
- Recognition / Matching: recognise which of several sets of information the tag cloud is likely to represent.

Some recent information retrieval papers discuss the use of tag or word clouds for various applications. PubCloud uses clouds for the summarisation of results from queries over the PubMed database of biomedical literature [9]. A stopword list is used to remove common words, and a Porter Stemmer is applied [17]. Colours are used to represent recency, and font size represents frequency. Mousing over a tag displays a list of words that share the same prefix and a hyperlink links to the set of PubMed abstracts containing the tag. In Dredze et al. [5] summary keywords are extracted from emails. Common stopwords and e-mail specific stopwords such as 'cc,' 'to' and 'http' are removed. Latent semantic analysis and latent Dirichlet allocation outperform a baseline of TF-IDF on an automated foldering and a recipient prediction task.

On the Internet tools like Wordle [22] and ManyEyes [13] create visually pleasing word clouds from any document. To create word clouds these tools remove stopwords and use term frequencies to determine font sizes. Information retrieval systems mainly remove stopwords to reduce index space and speed up processing. Since the discrimination value of stop words is low, removing these terms will not have a large effect on retrieval performance. Modern IR systems and web search engines exploit the statistics of language and do not use stopword lists, or very small stopword lists (7-12 terms) [12]. For word clouds however it is essential to have a good stopword list. Both Wordle and ManyEyes also have an option to include multi-word phrases. Popular social tagging sites like Flickr and Technorati allow multi-word tags. Most first-generation tagging systems did not allow multi-word tags, but users find this a valuable feature.

Term frequencies are most commonly used to create tag clouds. For information retrieval term frequencies are also a commonly used method of term weighting, but in addition some alternative weighting schemes have been developed. It was recognised early that more weight should be given to query terms matching documents that are rare within a collection, and therefore the inverse document frequency (IDF) was introduced [20]. The idf factor varies inversely with the number of documents $n$ in which a term occurs in a collection of $N$ documents. Since then many variants with different normalisation steps have been developed to improve retrieval results. Several relevance feedback approaches attempt to filter out background noise from feedback documents. Zhai and Lafferty [23] apply an Expectation-Maximization model to concentrate on

words that are common in the feedback documents but are not very common in in the complete collection. This same idea is used to create parsimonious models of documents in [8].

This section aimed to determine a number of techniques applicable for both language modelling and word cloud generation. The innovative features of tag clouds lie in the presentation and the willingness of users to assign tags to resources. Considering other technical features of tag clouds, we have not found features in tag clouds that have not been explored in the language modelling approach to information retrieval. From the techniques in the literature we will investigate the four features we think are the most interesting for creating word clouds, i.e., using relevance or pseudo-relevance information, stemming, including bigrams and term weighting schemes. In Sections 4 to 7 each of these features will be discussed and evaluated using the set-up that is discussed in the next section.

## 3 Evaluation of Word Clouds

In this section, we will detail our experimental set-up. Since there is no standard evaluation method for word clouds, we created our own experimental test bed. Our experiments comprise of two parts, a system evaluation and a user study. For both experiments we use query topics from the 2008 TREC Relevance Feedback track.

### 3.1 System Evaluation

We test our approaches using the 31 topics that have been evaluated using Pool10 evaluation, which is an approximation of the normal TREC evaluation strategy, and allows for ranking of systems by any of the standard evaluation measures [3]. We execute two experiments that correspond to tasks tag clouds can support, as described in the previous section. In the first experiment we evaluate the tasks 'Impression Formation' and 'Recognition' by using the words of the clouds for query expansion. Our assumption is that the quality of the query expansion equates the quality of the used model. The weights that are used to determine font size, are now used to represent the weight of query expansion terms. Prominent words carry more weight, but less prominent items can still contribute to the performance of the complete cloud, which is also the case in the two tasks. Our query expansion approach is similar to the implementation of pseudo-relevance feedback in Indri [21]. We keep the original query, and add the expansion terms with their normalised probabilities. We use the standard evaluation measures MAP and P10 to measure performance.

In our second experiment we evaluate the 'Search' task. In this task you want to locate a specific term that represents a desired concept. In our experiment the desired concept is the topic, and all terms that represent this topic are therefore relevant. We consider a word representative of the topic if adding the word to the original query leads to an improvement of the retrieval results. We take the feedback sets and 31 queries that we also used in the previous experiment. We let each model generate a word cloud consisting of 25 terms. For each topic we generate 25 queries where in each query a word from the word cloud is added to the original query. No weights are assigned to the expansion terms. For each query we measure the difference in performance caused by

adding the expansion term to the original query. Our evaluation measure is the percentage of 'relevant' words in the word cloud, i.e. the percentage of words where adding them to the query leads to an improvement in retrieval results. Additionally, we also calculate the percentage of 'acceptable' words that can be added to the query without a large decrease (more than 25%) in retrieval results.

### 3.2 User Study

In addition to the system-based approach for evaluation, we evaluate the word clouds from a user's point of view. In this user study we are focusing on the question which words should appear in a word cloud. We set the size of the word cloud to 25 terms. We do not want to investigate the optimal size for word clouds, this size suffices to show users the differences between the different types of word clouds. The only visual feature we are considering is font size, other features, such as lay-out, colours etc. are not considered. We present a word cloud as a list of words in alphabetical order. The test persons first read a TREC topic consisting of the query title (keywords that are used for search), query description (one line clarification of the query title) and narrative (one paragraph that explains which documents are relevant). For each topic users rank four groups of word clouds. In each group we experiment with a different feature:

- Group 1: Pseudo relevance and relevance information
- Group 2: Stemming
- Group 3: Including bigrams
- Group 4: Term weighting scheme

Test persons may add comments to each group to explain why they choose a certain ranking. Each test person gets 10 topics. In total 25 topics are evaluated, each topic is evaluated by at least three test persons and one topic is evaluated by all test persons. 13 test persons participated in the study. The test persons were recruited at the university in different departments, 4 females and 9 males with ages ranging from 26 to 44.

### 3.3 Baseline

Each group of clouds includes the standard word cloud which acts as a baseline to which the other clouds are compared. Since stopwords have high frequencies, they are likely to occupy most places in the word cloud. We therefore remove an extensive stopword list consisting of 571 common English words. Only single words (unigrams) are included in the standard cloud. Stemming is applied and words are conflated as described later in Section 5. The standard word cloud uses a TF weighting scheme which equals term frequency counting. The probability of a word occurring in a document is its term frequency divided by the total number of words in the document. For all models we have a restriction that a word has to occur at least twice to be considered. To create a word cloud all terms in the document are sorted by their probabilities and a fixed number of the 25 top ranked terms are kept. Since this results in a varying probability mass depending on document lengths and word frequencies, we normalise the probabilities in order to determine the font size. The standard cloud uses pseudo-relevant documents to generate the word cloud. The top 10 documents retrieved by a language model run are

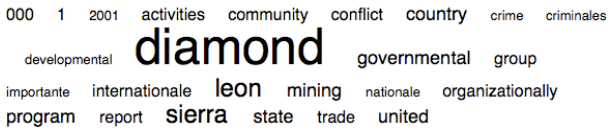



**Fig. 2.** Word cloud from 100 relevant results

**Table 1.** Effectiveness of feedback based on pseudo-relevance vs. relevance information. Evaluated after removing the used 100 relevant documents from runs and qrels

| Approach | MAP | P10 | % Rel. words | % Acc. words |
|---|---|---|---|---|
| Pseudo | 0.0985 | 0.1613 | 35 | 73 |
| Rel. docs | **0.1161** $^{-}$ | **0.2419** $^{-}$ | **50** | **85** |

concatenated and treated as one long document. Throughout this paper we will use the topic 766 'diamond smuggling' to show examples. In the earlier Figure 1 the standard TF word cloud of this topic was shown.

## 4 Clouds from Pseudo Relevant and Relevant Results

In this section, we look at the impact of using relevant or pseudo-relevant information to generate language models and tag clouds. In the first group a TF cloud made from 10 pseudo-relevant documents is compared to a cloud of 100 relevant documents. By making this comparison we want to get some insights on the question if there is a mismatch between words which improve retrieval performance, and the words that users would like to see in a word cloud. Our standard word cloud uses pseudo-relevant results because these are always available. The cloud in Fig. 2 uses 100 pages judged as relevant to generate the word cloud.

**Results** When we look at the query expansion scores, shown in Table 1,[4] a query based on the 100 relevant documents is on average better than the query based on 10 pseudo-relevant documents, and also there are more relevant and acceptable words in the clouds based on the 100 relevant documents. The test persons in our user study however clearly prefer the clouds based on 10 pseudo-relevant documents: 66 times the pseudo-relevant document cloud is preferred, 36 times the relevant documents cloud is preferred, and in 27 cases there is no preference (significant at 95% using a two-tailed sign-test).

There seem to be three groups of words that often contribute positively to retrieval results, but are not appreciated by test persons. First, there are numbers, usually low numbers from 0 to 5, which occur frequently in relevant documents. Without context these numbers do not provide any information to the user. Numbers that represent years can sometimes be useful. The second group are general and frequently occurring words which do not seem specific to the query topic. e.g. for the query 'hubble telescope repairs' adding the word 'year' ,'up' or 'back' results in improved retrieval results. The third group consists of words that test persons don't know. These can be for example abbreviations or technical terms. In this user study the test persons did not create the queries themselves, therefore the percentage of unknown words is probably higher than

[4] In all tables significance of increase/decrease over baseline according to t-test, one-tailed is shown: no significant difference($^{-}$), significance levels 0.05($^{\circ}$), 0.01($^{\bullet}$), and 0.001($^{\bullet}$).

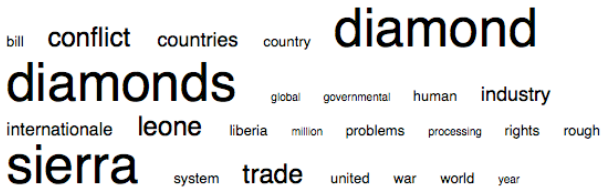



**Fig. 3.** Word cloud of 10 results using plain (non-stemmed) words

in a normal setting. In addition for most of the test persons English is not their first language. In some cases also the opposite effect takes place, test persons assume words they don't know (well) are relevant, while in fact the words are not relevant. Words appreciated by test persons and also contributing to retrieval performance are the query title words and keywords from the description and the narrative. The query description and narrative are in a real retrieval setting usually not available. Most of the informative words are either a synonym of a query word, or closely related to a query word.

These findings agree with the findings of a previous study, where users had to select good query expansion terms [19]. Also here reasons of misclassification of expansion term utility are: users often ignore terms suggested for purely statistical reasons, and users cannot always identify semantic relationships.

## 5 Non-Stemmed and Conflated Stemmed Clouds

In this section, we look at the impact of stemming to generate conflated language models and tag clouds. To stem, we use the most common English stemming algorithm, the Porter stemmer [17]. To visualize terms in a word cloud however, Porter word stems are not a good option. There are stemmers or lemmatizers that do not affect the readability of words, the simple S-removal stemmer for example conflates plural and singular word forms by removing the suffix -s according to a small number of rules [7]. The Porter stemmer is more aggressive, reducing for example 'immigrant' to 'immigr,' and 'political' to 'polit'. A requirement for the word clouds is to visualize correct English words, and not stems of words which are not clear to the user. Using word stems reduces the number of different terms in a document, because different words are reduced to the same stem. Since these words are very closely correlated, it is useful to aggregate them during the generation of terms for the word clouds. The question remains however which words should be visualised in the word cloud. In our experiments we consider non-stemmed word clouds and conflated word clouds where word stems are replaced by the most frequently occurring word in the collection that can be reduced to that word stem. The standard word cloud is conflated, in Figure 3 a non-stemmed word cloud is displayed. The non-stemmed cloud contains both 'diamond' and 'diamonds,' while the corresponding conflated cloud (see Fig. 1) only contains 'diamond'. The conflated cloud does bring up a small conflation issue. The non-stemmed cloud contains the word 'leone' (from Sierra Leone), but in the conflated cloud this is undesirably conflated to 'leon'. We opted for the collection-wise most frequent expansion since it is easy to process, but with hindsight choosing the most frequent word in the specific document(s) would have been preferred.


(a) Mixed uni- and bigrams


(b) Bigrams

**Fig. 4.** Word cloud of 10 results with unigrams and bigrams

**Results** The effect of stemming is only evaluated in the user study. We did not do a system evaluation, because we do not have a non-stemmed index available. Looking at pairwise preferences, we see that it often makes only a small difference to the word clouds to conflate words with the same stem: 38 times the conflated cloud is preferred, 20 times the non-stemmed cloud is preferred, and 71 times there is no preference (significant at 95% on a two-tailed sign-test). Often the difference is so small that it is not noticed by test persons. A disadvantage of the conflated cloud is that sometimes words are conflated, but then expanded to an illogical word. For example for the query 'imported fire arms' in the word cloud 'imported' is changed into 'importante'. A disadvantage of the non-stemmed cloud is that users do not like to see two words that are obviously reduced to the same stem, like 'ant' and 'ants'. These kind of words also appear next to each other, because of the alphabetical order of the words.

## 6 Bigrams

In this section, we look at the impact of adding bigrams to generate more informative language models and tag clouds. For users, bigrams are often easier to interpret than single words, because a little more context is provided. We have created two models that incorporate bigrams, a mixed model that contains a mix of unigrams and bigrams, and a bigram model that consists solely of bigrams. To incorporate bigrams, we use the TF model with some adjustments. In the bigram model each term now consists of two words instead of one word. Bigrams containing one or two stopwords are excluded. The most frequently occurring bigram will receive the highest probability. In the mixed model, a term can either consist of one or two words. Both unigrams and bigrams contribute to the total term count. Again all terms containing one or two stopwords are excluded from the model. The probability of occurrence of a term, either bigram or unigram, is its frequency count, divided by the total term count. We want to avoid however that unigrams which occur usually as part of a bigram, receive too much probability. Therefore, we subtract from each unigram that occurs as part of a bigram, the probability of the most frequently occurring bigram that contains the unigram. Since the probabilities of the unigrams and the bigrams are estimated using the same approach,

**Table 2.** Effectivenss of unigram, bigram, and mixed tokenizations evaluated over the full qrels

| Approach | MAP | P10 | % Rel. words | % Acc. words |
|---|---|---|---|---|
| Unigrams | 0.2575 | 0.5097 | **35** | **73** |
| Mixed | **0.2706**$^{-}$ | **0.5226**$^{-}$ | 31 | 71 |
| Bigrams | 0.2016$^{\circ}$ | 0.4387$^{-}$ | 25 | 71 |

**Table 3.** Pairwise preferences of test person over unigram, bigram, and mixed tokenizations

| Model 1 | Model 2 | # Preferences | | | Sign test 95% |
|---|---|---|---|---|---|
| | | Model 1 | Model 2 | Tied | |
| bigram | mixed | 49 | 54 | 26 | – |
| mixed | unigram | 71 | 33 | 25 | 0.95 |
| bigram | unigram | 62 | 46 | 21 | – |

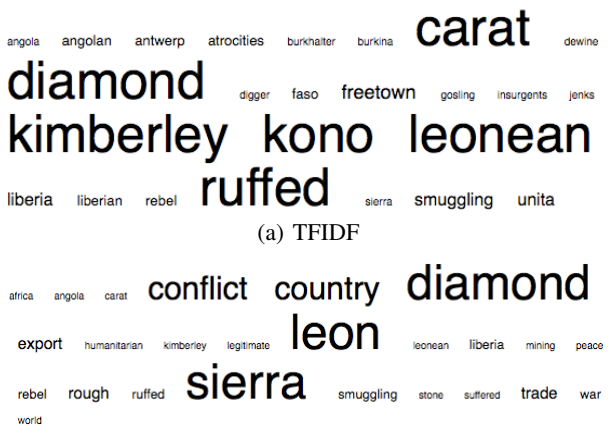
the resulting probabilities are comparable. So, we can create word clouds and query expansions that are a mix of unigrams and bigrams. To include a bigram as a query expansion term we make use of the proximity operator available in Indri [15]. The terms in the bigram must appear ordered, with no terms between them. For the user study we placed bigrams between quotes to make them more visible as can be seen in Figure 4, bigrams can also be differentiated by using different colours.

**Results** In Table 2 the system evaluation results are shown. For query expansion, the model that uses a mix of unigrams and bigrams performs best with a MAP of 0.2706. Using only bigrams leads to a significant decrease in retrieval results compared to using only unigrams. Looking at the percentages of relevant and acceptable words, the unigram model produces the most relevant words. The mixed model performs almost as good as the unigram model.

In the user study, the clouds with mixed unigrams and bigrams and the clouds with only bigrams are selected most often as the best cloud as can be seen in Table 3. There is no significant difference in preference between mixed unigrams and bigrams, and only bigrams. Users do indeed like to see bigrams, but for some queries the cloud with only bigrams contains too many meaningless bigrams such as 'http www'. An advantage of the mixed cloud is that the number of bigrams in the cloud is flexible. When bigrams occur often in a document, also many will be included in the word cloud.

## 7 Term Weighting

In this section, we look at the impact of term weighting methods to generate language models and tag clouds. Besides the standard TF weighting we investigate two other variants of language models to weigh terms, the TFIDF model and the parsimonious model. In the TFIDF algorithm, the text frequency (TF) is now multiplied by the inverse document frequency (IDF). Words with an inverse document frequency of less than 10 are excluded from the model. In Figure 5(a) the example word cloud of the TFIDF model is shown. The last variant of our term weighting scheme is a parsimonious model [8]. The parsimonious language model concentrates the probability mass on fewer words than a standard language model. Only terms that occur relatively more frequent in the document as in the whole collection will be assigned a non-zero probability. The model automatically removes both common stopwords and corpus specific




(a) TFIDF

(b) Parsimonious

**Fig. 5.** Word cloud of 10 results with TFIDF and parsimonious term weighting.

**Table 4.** Effectiveness of term weighting approaches evaluated over the full qrels

| Approach | MAP | P10 | % Rel. words | % Acc. words |
|---|---|---|---|---|
| TF | 0.2575 | 0.5097 | **35** | **73** |
| TFIDF | 0.1265$^{\bullet}$ | 0.3839$^{\circ}$ | 22 | 67 |
| Pars. | **0.2759**$^{\circ}$ | **0.5323**$^{\circ}$ | 31 | 68 |

stopwords, and words that are mentioned occasionally in the document. The parsimonious model estimates the probability $P(t|R)$ using *Expectation-Maximization*:

$$\text{E-step: } e_t = tf(t,R) \cdot \frac{(1-\lambda)P(t|R)}{(1-\lambda)P(t|R) + \lambda P(t|C)} \tag{1}$$

$$\text{M-step: } P(t|R) = \frac{e_t}{\sum_t e_t}\text{, i.e. normalize the model} \tag{2}$$

$\lambda$ determines the weight of the background model $P(t|C)$. There is no fixed number of terms that are kept, but in the M-step terms that receive a probability below a threshold of 0.001 are removed from the model. In the next iteration the probabilities of the remaining terms are again normalised. The algorithm stops after a fixed number of iterations. In Figure 5(b) the parsimonious word cloud of our example topic is shown. Compared to the standard TF cloud (Figure 1) , we see that frequently occurring words like 'year' and 'system' have disappeared, and are replaced by more specific words like 'angola' and 'rebel'.

**Results** To start with the system based evaluation, Table 4 shows the system evaluation results for the different term weighting schemes. The parsimonious model performs best on both early and average precision. The TFIDF model performs significantly worse than the TF and the parsimonious model. Our simplest model, the TF model, actually produces the highest number of relevant and acceptable words. The parsimonious model produces more relevant words than the TFIDF model, but the number of acceptable words is the same. The weighting scheme of the parsimonious model is clearly more

**Table 5.** Pairwise preferences of test person over term weighting approaches

| Model 1 | Model 2 | # Preferences | | | Sign test 95% |
|---|---|---|---|---|---|
| | | Model 1 | Model 2 | Tied | |
| TF | TFIDF | 76 | 33 | 20 | 0.95 |
| Pars. | TFIDF | 84 | 23 | 22 | 0.95 |
| Pars. | TF | 56 | 41 | 32 | – |

effective than the TF model, since for query expansion where weights were considered the parsimonious model performed better than the TF model.

The results of the user study can be found in Table 5. The parsimonious model is preferred more often than the TF model, and both the parsimonious and the TF model are significantly more often preferred over the TFIDF model. The parsimonious model contains more specific and less frequently occurring words than the TF model. In Section 4 we saw already that more general words are not appreciated by our test persons, but that they can be beneficial for retrieval. Although the TF model contains more relevant words according to our system evaluation, these words are less informative than the words in the parsimonious model. Indeed, both for query expansion and from the user's point of view the parsimonious model generates the best word clouds.

## 8 Conclusion

This paper investigated the connections between tag or word clouds popularised by Flickr and other social web sites, and the language models as used in IR. The main research question was: do words extracted by language modelling techniques correspond to the words that users like to see in word clouds?

We have investigated how we can create word clouds from documents and use language modelling techniques which are more advanced than only frequency counting and stopword removal. We found that different language modelling techniques can indeed be applied to create better word clouds. Including bigrams in the word clouds and a parsimonious term weighting scheme are the most effective improvements. We found there is some discrepancy between good words for query expansion selected by language modelling techniques, and words liked by users. This will be a problem when a word cloud is used for suggestion of query expansion terms. The problem can be partly solved by using a parsimonious weighting scheme which selects more specific and informative words than a TF model, but also achieves good results from a system point of view.

For future work, the parsimonious term weighting scheme can not only be useful to generate word clouds from documents, but also to create tag clouds from assigned tags. For example on LibraryThing [11] tag clouds are created for books. The resulting tag clouds feature generic tags like 'read,' 'own' and 'fiction,' which are non-distinctive for the book at hand. When the parsimonious model would be used to create the tag clouds, tags more specific to a book would feature more prominently in the cloud.

**Acknowledgments** This research was supported by the Netherlands Organization for Scientific Research (NWO, under project # 612.066.513).

## REFERENCES


[1] S. Bateman, C. Gutwin, and M. Nacenta. Seeing things in the clouds: the effect of visual features on tag cloud selections. In *Proceedings HT '08*, pages 193–202. ACM, 2008.

[2] C. H. Brooks and N. Montanez. Improved annotation of the blogosphere via autotagging and hierarchical clustering. In *Proceedings WWW'06*, pages 625–632. ACM, 2006.

[3] C. Buckley and S. Robertson. Relevance feedback track overview: TREC 2008. In *The Seventeenth Text REtrieval Conference (TREC 2008) Notebook*, 2008.

[4] D. Coupland. *Microserfs*. HarperCollins, Toronto, 1995.

[5] M. Dredze, H. M. Wallach, D. Puller, and F. Pereira. Generating summary keywords for emails using topics. In *Proceedings of the 2008 International Conference on Intelligent User Interfaces*, 2008.

[6] M. J. Halvey and M. T. Keane. An assessment of tag presentation techniques. In *Proceedings WWW '07*, pages 1313–1314. ACM, 2007.

[7] D. Harman. How effective is suffixing? *Journal of the American Society for Information Science*, 42:7–15, 1991.

[8] D. Hiemstra, S. Robertson, and H. Zaragoza. Parsimonious language models for information retrieval. In *Proceedings SIGIR'04*, pages 178–185. ACM Press, New York NY, 2004.

[9] B. Y.-L. Kuo, T. Hentrich, B. M. Good, and M. D. Wilkinson. Tag clouds for summarizing web search results. In *Proceedings WWW'07*, pages 1203–1204. ACM, 2007.

[10] R. Lambiotte and M. Ausloos. Collaborative tagging as a tripartite network. In *Computational Science – ICCS 2006*, pages 1114–1117, 2006.

[11] LibraryThing, 2009. URL `http://www.librarything.com/`.

[12] C. D. Manning, P. Raghavan, and H. Schutze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[13] ManyEyes, 2009. URL `http://manyeyes.alphaworks.ibm.com/manyeyes/page/Tag_Cloud.html`.

[14] M. E. Maron and J. L. Kuhns. On relevance, probabilistic indexing and information retrieval. *Journal of the ACM*, 7(3):216–244, 1960.

[15] D. Metzler and W. B. Croft. Combining the language model and inference network approaches to retrieval. *Information Processing & Management*, 40(5):735–750, 2004.

[16] J. Ponte and W. Croft. A language modeling approach to information retrieval. In *SIGIR'98*, pages 275–281, 1998.

[17] M. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[18] A. W. Rivadeneira, D. M. Gruen, M. J. Muller, and D. R. Millen. Getting our head in the clouds: toward evaluation studies of tagclouds. In *Proceedings CHI'07*, pages 995–998. ACM, 2007.

[19] I. Ruthven. Re-examining the potential effectiveness of interactive query expansion. In *SIGIR '03*, pages 213–220, New York, NY, USA, 2003. ACM.

[20] K. Spärck Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28:11–21, 1972.

[21] T. Strohman, D. Metzler, H. Turtle, and W. B. Croft. Indri: a language-model based search engine for complex queries. In *Proceedings of the International Conference on Intelligent Analysis*, 2005.

[22] Wordle, 2009. URL `http://wordle.net`.

[23] C. Zhai and J. Lafferty. Model-based feedback in the language modeling approach to information retrieval. In *Proceedings CIKM'01*, pages 403–410. ACM, 2001.