Dimensionality Reduction for Information Retrieval Using Vector Replacement of Rare Terms

Tobias Berka and Marian Vajteršic

Abstract Dimensionality reduction by algebraic methods is an established technique to address a number of problems in information retrieval. In this chapter, we introduce a new approach to dimensionality reduction for text retrieval. According to Zipf's law, the majority of indexing terms occurs only in a small number of documents. Our new algorithm exploits this observation to compute a dimensionality reduction. It replaces rare terms by computing a vector which expresses their semantics in terms of common terms. This process produces a projection matrix, which can be applied to a corpus matrix and individual document and query vectors. We give an accurate mathematical and algorithmic description of our algorithms and present an initial experimental evaluation on two benchmark corpora. These experiments indicate that our algorithm can deliver a substantial reduction in the number of features, from 8,742 to 500 and from 47,236 to 392 features, while preserving or even improving the retrieval performance.

1 Introduction

Dimensionality reduction techniques reduce the number of components of a data set by representing the original data as accurately as possible with fewer features and/or instances. The goal is to produce a more compact representation of the data with only limited loss of information in order to reduce the storage and runtime requirements.

T. Berka (✉) · M. Vajteršic
Department of Computer Sciences, University of Salzburg,
Jakob-Haringer-Str. 2, 5020 Salzburg, Austria
e-mail: tberka@cosy.sbg.ac.at

M. Vajteršic
Department of Informatics, Mathematical Institute, Slovak Academy of Sciences,
Stefanikova 49, 81473 Bratislava, Slovakia
e-mail: marian@cosy.sbg.ac.at

In some applications the reduction is used to discover and account for latent dependencies between features which are not reflected in the original data [1]. Here, the lower dimensionality not only reduces the computational costs, but also improves the retrieval performance. In the area of text retrieval, low-rank matrix approximations have long been utilized in order to deal with polysemy and synonymy—words with multiple meanings (e.g. bank or light) or different words with the same meaning (e.g. drowsy and sleepy).

But the problem with information retrieval applications is that these systems operate on tens of thousands of features and millions of documents or more. The key issues that warrant the investigation of new dimensionality reduction techniques are performance and scalability. While traditional methods focus on theoretic properties of the underlying mathematical method, the ultimate measure of success for any dimensionality reduction technique is the impact on the retrieval performance. The development of new methods for dimensionality reduction beyond the traditional, algebraic or statistical methods is therefore fair game, as long as the retrieval performance is equal or better than on the unprocessed raw data.

Our approach is based on the fact that according to Zipf's law, most terms occur only in a very limited number of documents [2, 3]. This makes them interesting candidates for any form of dimensionality reduction or compression. It is also a well-known fact that features with a low document frequency play an important role in the query processing. This phenomenon forms the basis of the inverse-document frequency term weighting approach (see e.g. [4]).

Deletion of such *rare terms* is therefore out of the question. In our approach, we attempt to replace rare terms with a signature feature vector, which preserves the semantics of the rare term by expressing it in more frequent terms. This vector is then scaled by the frequency of the rare term in this document. By forming the linear combination of all rare term replacement vectors and adding it to the original document vector we obtain a vector representation without the rare terms. The hypotheses for our approach to dimensionality reduction can be stated as follows:

- Rare terms with a low document frequency can be replaced by a *replacement vector* that expresses their semantics in feature vector form.
- A suitable replacement vector for a rare term can be obtained by forming the weighted average vector of all documents containing this rare term.
- For any document, the rare terms can be eliminated by forming a linear combination of the corresponding replacement vectors, scaled by the weight of the corresponding features in the original feature vector.
- Performing such a replacement operation on the document and query vectors will not lead to a reduction in retrieval quality.
- In agreement with Zipf's law, eliminating rare terms will lead to a considerable reduction in the number of features.

The rest of this chapter is structured as follows. We give a mathematical description of our method in Sect. 2 and discuss the algorithmic aspects in Sect. 3. The retrieval performance on real-world data sets is discussed in Sect. 4 and we analyze the computational performance both formally and empirically in Sect. 5. In Sect. 6

we will briefly examine the state-of-the-art in literature and how it relates to our approach. Lastly, we summarize our findings in Sect. 7.

2 The Vector Replacement Approach

In order to systematically devise an algorithm, we first define our approach mathematically. Table 1 presents a summary of all symbols and notation used here.

We have a set D of $\|D\| = n$ documents and a set F of $\|F\| = m$ features. Every document $d \in D$ is represented as a feature vector $d \in \mathbb{R}^m$ to which we assign a column index $\mathrm{col}(d) \in \{1, \ldots, n\}$. As a convenient notation, we use d_j to denote $\mathrm{col}(d) = j$. Analogously, every feature $f \in F$ has a row index $\mathrm{row}(f) \in \{1, \ldots, m\}$, using the short form $f_i :\Leftrightarrow \mathrm{row}(f) = i$. Again, we use a corpus matrix $C \in \mathbb{R}^{m \times n}$, which contains the documents' feature vectors as column vectors

$$C = \left[d_1 \ d_2 \cdots d_n \right] \in \mathbb{R}^{m \times n}. \tag{1}$$

Since we are interested in the occurrence of features within documents we define a function $\mathscr{D} : F \to D$ to determine which documents contain any particular feature f_i, formally

$$\mathscr{D}(f_i) := \left\{ d_j \in D \mid C_{i,j} \neq 0 \right\}. \tag{2}$$

We select the rare features through a function $\mathscr{N} : \mathbb{N} \to F$ that determines the set of features occurring in at most t documents,

$$\mathscr{N}(t) := \{ f \in F \mid \|\mathscr{D}(f)\| \leq t \}. \tag{3}$$

After choosing an *elimination threshold* t, which was experimentally determined to be 1 and 3 % of all documents for our experiments, we can now define the *set of elimination features* $E \subseteq F$ as

$$E := \mathscr{N}(t), \tag{4}$$

which will ultimately lead to a reduced-dimensional feature space consisting of k common terms, where

$$k := m - \|E\|. \tag{5}$$

The sensitivity of our method with respect to the choice of k is evaluated empirically and discussed in Sect. 4.

Our objectives can now be formulated as follows:

1. We have a corpus matrix $C \in \mathbb{R}^{m \times n}$ for m features and n documents, which is *sparse*, i.e. C contains mostly zero components.
2 We seek to eliminate all features which occur in t or fewer documents. Formally, this means that we seek to replace all features in the set of elimination features $E = \mathscr{N}(t)$.

Table 1 Mathematical notation

Symbol	Description
m	Number of features
n	Number of documents
k	Number of common features
t	Maximum occurrences for rare features
C	Corpus matrix
F	Set of features
D	Set of documents
$E \subseteq F$	Elimination (rare) features
$\mathscr{D}(f)$	Documents containing feature f
$\mathscr{F}(d)$	Features occurring in document d
$\mathscr{N}(t)$	Features with t or less documents
τ_E	Vector truncation operator (eliminates indices E)
$\rho_E(f)$	Replacement vector for feature f and rare features E
$\mathscr{R}_E$	Replacement operator (applies replacement vectors)
R_E	Replacement matrix (equivalent to $\mathscr{R}_E$)

3. We want to replace every feature $f \in E$ by a vector formed as a linear combination of common features that co-occur with f. We will refer to this replacement vector as $\rho_E(f)$ (rho).
4. This replacement operator should be computed in such a way, that it can be applied to the original corpus matrix C, any new documents d and query vectors $q \in \mathbb{R}^m$. We therefore compute a replacement matrix $R \in \mathbb{R}^{k \times m}$ which maps vectors from the original to the reduced feature space.
5. Finally, we apply this replacement to the original corpus matrix C to obtain a reduced dimensional corpus matrix $C' = RC \in \mathbb{R}^{k \times n}$.

To eliminate an index i of a vector v, we define a truncation operator τ as a formal mechanism

$$\tau_{\{i\}}(v) := (v_1, \ldots, v_{i-1}\, v_{i+1} \ldots v_m)^T, \tag{6}$$

and generalize it to the elimination of a set of indices with the recursive definition

$$\tau_{\emptyset}(v) := v, \tag{7}$$

$$\tau_{\{A,b\}}(v) := \tau_{\{b\}}(\tau_A(v)). \tag{8}$$

We use a linear combination of common features to replace rare features $f_i \in E$. Formally, we will initially determine the set of documents that contain the feature $\mathscr{D}(f_i)$. We will truncate all document vectors $d_j \in \mathscr{D}(f_i)$, eliminate all rare features by taking $\tau_E(d_j)$ and scale them by the quantification $C_{i,j}$ of feature f_i in document d_j. We compute the sum of these vectors and apply a scaling factor λ to normalize the length of the resulting replacement vector. Formally, we define the function ρ_E to compute the replacement vector:

$$\rho_E(f_i) := \frac{1}{\lambda_E(f_i)} \sum_{d_j \in \mathscr{D}(f_i)} C_{i,j}\, \tau_E\left(C_{\star,j}\right), \tag{9}$$

where the scaling factor $\lambda_E(f_i)$ is the absolute sum of all contributions,

$$\lambda_E(f_i) := \sum_{d_j \in \mathscr{D}(f_i)} |C_{i,j}|. \tag{10}$$

If a rare feature has a high weight in a particular document, the co-occurrences in this document have a higher impact on the replacement vector. Conversely, a low weight would marginalize the contribution of this document.

Our next goal is to obtain a linear *replacement operator* $\mathscr{R}_E$. First, we truncate the original document vector d_j to eliminate all unwanted features $f_i \in E$ by computing $\tau_E(d_j)$. Then we take the replacement vectors $\rho_E(f_i)$ for these features and scale them by their relevance $C_{i,j}$ in the document d_j. The reduced document vector is formed as the linear combination of the truncated source document and the scaled replacement vectors:

$$\mathscr{R}_E\left(d_j\right) := \tau_E\left(d_j\right) + \sum_{f_i \in E} C_{i,j}\, \rho_E\left(f_i\right). \tag{11}$$

We can extend this operator to the corpus matrix $C \in \mathbb{R}^{m\times n}$ by applying it to the column vectors,

$$\mathscr{R}_E\left(C\right) := \left[\mathscr{R}_E\left(d_1\right) \cdots \mathscr{R}_E\left(d_n\right)\right]. \tag{12}$$

Since this operator is a linear function we can represent it as a matrix, and we will indeed use this representation to implement our algorithm.

If we use the notation $e_1, \ldots, e_m \in \mathbb{R}^m$ to denote the standard base vectors,

$$e_i^T = (0, \ldots, 0\,\underset{i}{1}\,0, \ldots, 0), \tag{13}$$

we can define replacement vectors r_E, which either preserve features we do not wish to eliminate, or perform the vector replacement on features $f \in E$,

$$r_E(f_i) := \begin{cases} \rho_E(f_i) & \ldots\ f_i \in E \\ \tau_E\left(e_i\right) & \ldots\ f_i \notin E. \end{cases} \tag{14}$$

We now assemble these vectors column-wise to form a replacement matrix R_E for the elimination features E, formally

$$R_E := \left[\, r_E(f_1)\; r_E(f_2) \cdots r_E(f_m) \,\right] \in \mathbb{R}^{k\times m}. \tag{15}$$

An improved, dimensionality-reduced corpus matrix C' can now be obtained by taking

$$C' := R_E C, \tag{16}$$

because it is easily verified that

$$R_E C = [\mathscr{R}_E(d_1), \dots, \mathscr{R}_E(d_n)] = \mathscr{R}_E(C). \tag{17}$$

Therefore, R_E is the matrix representation of the linear replacement operator $\mathscr{R}_E$ as defined above.

Theoretically, we must assume that some replacement vectors could be zero. Any particular feature may co-occur exclusively with other infrequent features that are removed during the process. For a feature f in a set of elimination candidates E it may hold that

$$(\forall d \in \mathscr{D}(f))\,(\tau_E(d) = 0). \tag{18}$$

A consequence of this phenomenon is that the replacement vector obtained by the approach outlined above is zero, i.e. $\rho_E(f) = 0$. If it occurs, we may need to retain the affected feature(s) to avoid loosing them. However, thus far we have not observed this behavior in practice.

Now that we can map our original corpus matrix C into the reduced dimensional space by taking $C' := R_E C$, we have to consider the on-line query processing. For any incoming query $q \in \mathbb{R}^m$, we compute $q' := R_E\, q$ and evaluate all similarities on the augmented corpus C'. For practical reasons, we will normalize the column vectors of the augmented corpus C' and all query vectors q' to unit length prior to query processing.

An equally important question is the maintenance of the index. New documents must be added, existing documents modified and old documents retired and deleted. Currently, our support for index maintenance is very crude. New documents d are mapped to the reduced dimensional space by computing $d' := R_E\, d$ and added by extending both the original and augmented corpus matrix with an additional column. The projection matrix R_E must be re-evaluated periodically from the sparse raw corpus. Improved support for updates is one of our future research objectives.

During our experiments, we found that we can get a greater reduction in the number of features with a higher retrieval performance if we apply a spectral dimensionality reduction to the augmented corpus matrix C'. We compute a rank-reduced principal component analysis (PCA) with a biased covariance, i.e. we do not shift the data to the mean. Due to the fact that C' is already dimensionality reduced, it is advisable to solve the underlying eigenvalue problem using a one-sided eigensolver to compute the first h left eigenvectors of C'. These eigenvectors are identical to the left singular vectors of a singular value decomposition, but the computation is more effective. More specifically, we compute the rank-reduced factorization

$$C' C'^T \approx P_h S_h P_h^T. \tag{19}$$

The choice of h is similar decision as the selection of the rank of the singular value decomposition in a latent semantic indexing system. A practical means to determine a cut-off threshold is to compute a larger number of singular values and plot them on a logarithmic scale. In such a plot, the threshold is much more visible than on a linear scale. However, as we will see in Sect. 4, the retrieval quality is relatively stable for a range of values of h and k. Based on our current experience, we advise to pick h according to the formula

$$h \approx 0.85\ k. \tag{20}$$

We can compute a joint left factor matrix $Q = P_h^T R_E \in \mathbb{R}^{h \times m}$ as the final projection matrix, which we can use to map the original feature vectors into a rank reduced feature space. In any case, it is important to note that a significant reduction in the number of features has already been achieved before computing the factorization.

3 Implementation Details

Our mathematical definition for the construction of a replacement matrix R easily translates into an algorithmic formulation, as given in Algorithm 1.

```
Input: The corpus matrix $C \in \mathbb{R}^{m \times n}$, the sets of documents $D$, the set of features $F$ and the
       threshold $t \in \mathbb{N}$.
Data: The occurrence count $N \in \mathbb{N}$, the elimination features $E \subseteq F$, the permutation
      $\pi : \mathbb{N} \to \mathbb{N}$, a floating point variable $l \in \mathbb{R}$, the feature $f_i \in F$, the document $d_j \in D$
      and an integer $k \in \mathbb{N}$.
Output: The replacement matrix $R \in \mathbb{R}^{k \times m}$.
$k := 1$;
for $f_i \in F$ do
    if $\|\mathcal{D}(f_i)\| \leq t$ then
        $E := E \cup \{f_i\}$;
    else
        $\pi(i) := k$;
        $k \mathrel{+}= 1$;
$k \mathrel{-}= 1$;
for $f_i \in E$ do
    $l := 0$;
    for $d_j \in \mathcal{D}(f_i)$ do
        $R(1:k, i) \mathrel{+}= C(i, j) * \tau_E(C(1:m, j))$;
        $l \mathrel{+}= |C_{i,j}|$;
    if $l \neq 0$ then
        $R(1:k, i) /= l$;
for $f_i \notin E$ do
    $R(1:k, i) := e_{\pi(i)}$;
```

Algorithm 1: The naive implementation proceeds feature-wise and computes all replacement vectors individually

The algorithm suffers a serious drawback in this form: since many documents contain more than one rare term, most documents have to be read from main memory more than once, leading to poor memory and cache performance. However, it is possible to vectorize the naive version and rearrange the loops so that every document is accessed only once. We introduce a new function $\mathscr{F}$,

$$\mathscr{F} : D \to F, \tag{21}$$

that determines which features occur in a given document, formally

$$\mathscr{F}(d_j) := \left\{ f_i \in F \mid C_{i,j} \neq 0 \right\}. \tag{22}$$

Using this function, Algorithm 1 describes an optimized variant of our construction method which uses the same number of computational steps, but requires only a single sweep over all documents in the corpus.

```
Input: The corpus matrix $C \in \mathbb{R}^{m \times n}$, the set of documents $D$, the set of features $F$ and the
        maximum occurrence count for rare features $t \in \mathbb{N}$.
Data: The occurrence count vector $N \in \mathbb{N}^m$ for all features, the elimination features $E \subseteq F$,
      the elimination features present in a document $G \subseteq F$, the permutation $\pi : \mathbb{N} \to \mathbb{N}$,
      the feature $f_i \in F$, the document $d_j \in D$ and an integer $k \in \mathbb{N}$.
Output: The replacement matrix $R \in \mathbb{R}^{k \times m}$.
for $d_j \in D$ do
    for $f_i \in \mathscr{F}(d_j)$ do
        $N(i)$ += 1;
$k := 1$;
for $f_i \in F$ do
    if $N(i) \leq t$ then
        $E := E \cup \{f_i\}$;
    else
        $\pi(i) := k$;
        $k$ += 1;
$k$ -= 1;
for $d_j \in D$ do
    $G := E \cap \mathscr{F}(d_j)$;
    for $f_i \in G$ do
        $R(1:k, i)$ += $C(i, j) * \tau_E(C(1:m, j))$;
        $l_i$ += $|C(i, j)|$;
for $f_i \in F$ do
    if $f_i \in E$ then
        $R(1:k, i) := (l(i))^{-1} * R(1:k, i)$;
    else
        $R(1:k, i) := e_{\pi(i)}$;
```

Algorithm 2: This optimized variant of the naive algorithm proceeds document-wise and accumulates all replacement vectors simultaneously in a single sweep over all documents. Since every document vector is read only once, this optimization results in a more cache and memory friendly algorithm

4 Retrieval Performance Evaluation

To test the performance of our approach in a small-scale environment with a low number of documents, we have used the classic MEDLARS collection, see e.g. [5]. With merely 1,033 documents this collection is indeed quite small. But it also contains 30 evaluation queries, along with a hand-generated set of relevant documents for each query, allowing us to conduct a standard precision-at-rank k retrieval performance evaluation. The documents have been converted into (unweighted) term frequency vectors with 8,742 features. The resulting corpus matrix C contained 57,013 non-zero components, making this corpus matrix merely 0.63 % dense, i.e. less than one percent of all components are non-zero.

For a larger test, we have used the Reuters Corpus Volume I in its corrected second version (RCV1-v2) [6]. We used the pre-vectorized TF-IDF version with 47,236 features in a corpus matrix which is 0.16 % dense. Since this is a benchmark corpus for text categorization, we used the available class labels for the 23,149 training documents to discriminate between relevant and irrelevant search results. All documents in the official set of training documents were used as sample queries and evaluated against all other vectors. Every sample query has been evaluated for *all of its categories*, which have been counted as though they were independent queries with the same vector.

Figures 1 and 2 depict the occurrence counts for all features in both document collections used in our evaluation. The power distribution observed by Zipf's law is clearly visible in both plots, and we have included the cut-off threshold and the resulting division between rare and common features for a more intuitive understanding of our reduction.

For both data sets, we have experimentally determined the threshold t and the reduced rank k of the subsequent PCA. To compare the retrieval performance with different parameters we used the mean average precision (MAP), the mean of the precision-at-k averaged over all hit list ranks $k \in \{1, \ldots, 100\}$, as a measure of quality. Figure 3 shows our experimental data for the RCV1-v2 data set. We can clearly see that the primary choice is the threshold t, which determines the trade-off between the retrieval quality and the number of features, i.e. the computational performance. We observed that a combination of vector replacement and limited PCA performed best. However, the retrieval performance decreases dramatically for a stronger spectral dimensionality reduction.

For the MEDLARS collection, we have computed the replacement vector matrix for all features occurring in less than 1 % of all documents on an Intel Xeon E5520 CPU clocked at 2.27 GHz in just under 1.9 s using Algorithm 1 or 2.6 s with Algorithm 2. As we have previously indicated, we can expect a slight performance degradation with the single-sweep algorithm on such a small document collection because of the overhead of the vectorized processing. Our reduction produced a reduced corpus matrix C' with 1,136 features containing 750,903 non-zero features, now being 63.99 % dense. Lastly, we computed a rank-500 dimensionality reduction of C' via a principal component analysis on the biased 1,136 by 1,136 feature

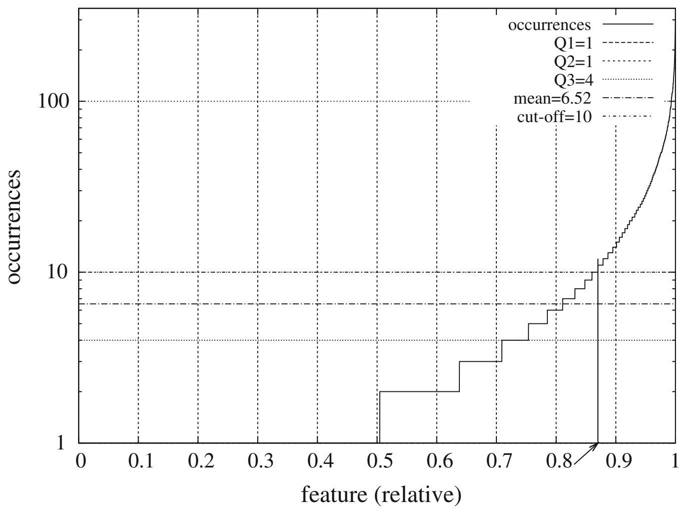

Fig. 1 Feature occurrence counts for the MEDLARS corpus—depicted along with the quartiles, the sampling mean and the cut-off threshold for rare features. The vertical line indicated by the arrow shows the dividing line between rare and common features

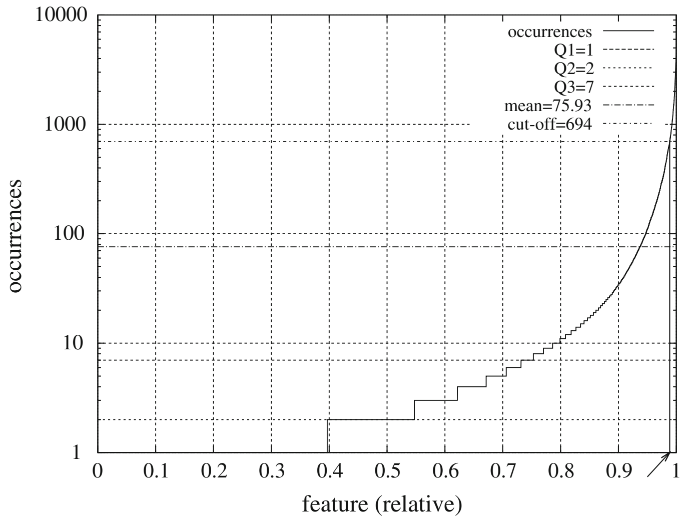

Fig. 2 Feature occurrence counts for the Reuters Corpus Volume I Version 2—depicted along with the quartiles, the sampling mean and the cut-off threshold for rare features. The vertical line indicated by the arrow shows the dividing line between rare and common features

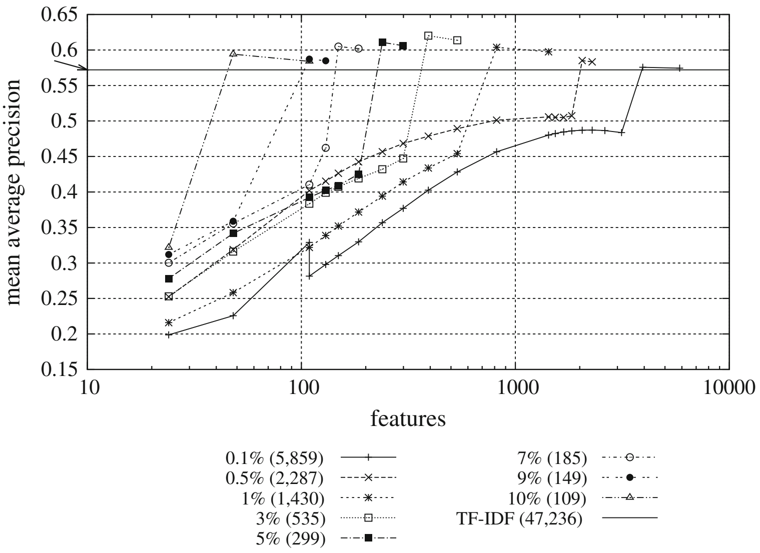

Fig. 3 Impact of the Subsequent Rank Reduction—visualized with the mean average precision (MAP) computed for the top 100 search results on the RCV1-v2 corpus (see above). Every curve depicts a vector replacement reduction with the threshold t given as the relative number of documents. The rightmost measurement of every curve shows the retrieval performance without a spectral dimensionality reduction, all others include the subsequent PCA. The horizontal line indicated by the arrow shows the performance baseline

covariance matrix, producing a third corpus matrix C'' with 500 features. We conducted three runs on these three versions of the corpus: (1) retrieval using the vector space model without term weighting on the sparse vectors in the corpus C, (2) on the vector replacement dimensionality reduction $C' = R_E C$, and (3) on a the rank-reduced corpus $C'' = QC$.

The data obtained during these measurements are depicted in Fig. 4. These figures indicate that the vector replacement approach succeeds in its objective of creating a reduced-dimensional representation which preserves or improves the retrieval performance. Even the limited number of documents available in the MEDLARS collection provided enough information to construct replacement vectors that stand up to a performance evaluation with the raw, sparse term frequency vectors. The subsequent rank reduction does not provide a decisive advantage in terms of retrieval performance, but it does succeed in cutting the final number of features in half without a significant loss in terms of accuracy.

Replacing all features which occur in less than 3 % of all documents of the RCV1-v2 was performed on the same CPU in under 8 min using Algorithm 1 and just under 6 min with Algorithm 1. Our preliminary measurements indicate that the single-sweep strategy does provide a speed-up for larger document collections. The reduction produced 535 features and a corpus matrix which is 99.98 % dense. We again

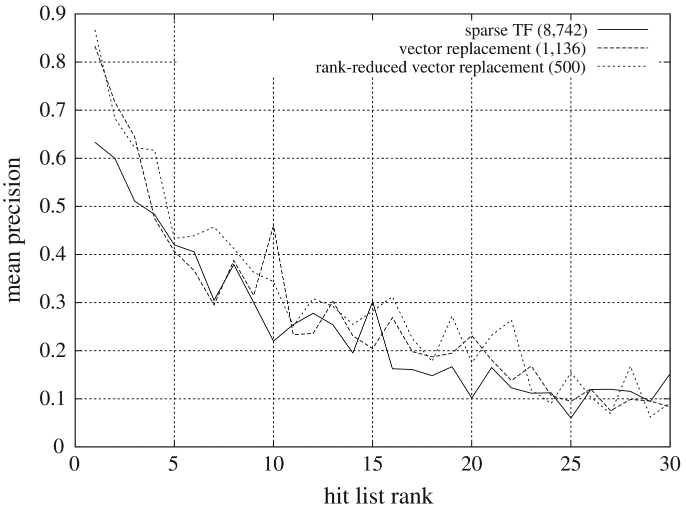

Fig. 4 Precision-at-k retrieval performance evaluation on the MEDLARS benchmark corpus with 1,033 documents and 30 sample queries. The hit list rank k was sampled from 1 to 30. This measurement has been conducted with (1) the raw, sparse term-frequency vectors, (2) the replacement vector approach and (3) the replacement vector approach with subsequent rank reduction. The data indicates that the replacement vector approach can deliver a dimensionality reduction which succeeds to preserve or improve the retrieval effectiveness on small scale document collections

performed a subsequent rank-reduction creating 392 features as a third representation for our evaluation.

Figure 5 illustrates our results, which provide a clear indication that our approach succeeds in reducing the dimensionality and improving the retrieval performance on this substantially larger data set. Here, the subsequent rank reduction manages to both cut the number of features by 50 % and further improve the precision of the vector replacement approach.

By replacing a rare term with a (weighted) centroid, we make a reasonably good effort to recover said term in a nearest neighbor search. In fact, the empirical evaluation shows that we make an effort which is even better—good enough to outperform the retrieval performance of the original representation. While we cannot provide any formal evidence to that end, our current best estimate is that the centroid replacement strategy performs a function similar to smoothing in natural language models, i.e. it evens out problems caused by the extreme sparsity of the raw term frequency or TF-IDF representation and improves the recoverability of the related documents by providing a much better placement under the cosine measure.

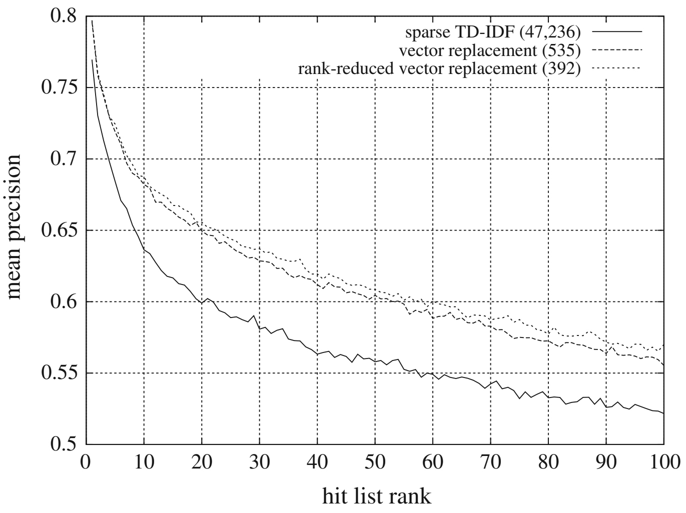

Fig. 5 All-documents all-categories evaluation of the precision-at-k retrieval performance on the 23,149 training documents of the Reuters Corpus Volume I Version 2 with the topic categorization. Queries have been conducted using every document as a query example. For all categories of the query example, we have scanned the hit list and considered only those documents relevant that featured the same category. The graph depicts the mean average accuracy over all documents and categories for (1) the precomputed sparse TF-IDF vectors as available in the RCV1-v2 collection, (2) the replacement vector approach and (3) the replacement vector approach with subsequent rank reduction. This exhaustive measurement indicates the ability of the replacement vector approach to preserve and improve the similarities on a large text categorization collection

5 Performance Analysis

We will now discuss the computational performance of our algorithm. We begin with a formal analysis of the complexity, before we discuss the measurements of the observed performance on real data.

The algorithmic complexity of this algorithm depends heavily on the distribution of non-zero components in the corpus matrix. In general, the upper bound for the complexity is $O(m^2n)$. But since our method has been specifically designed for text retrieval, we can consider the specific properties of text index data, notably Zipf's law, and derive tighter bound for our application domain. Let c be the maximum number of non-zero components in any document vector, i.e.

$$c = \underset{d \in D}{\operatorname{argmax}} \| \left\{ j \in \{1, \ldots, m\} \mid d_j \neq 0 \right\} \|. \tag{23}$$

Then the truncation operator τ can be implemented has a complexity of $O(\|E\| + c)$ by keeping the non-zero entries of the sparse vector implementation sorted by the component index. But for text retrieval we may assume that $\|E\| > c$, and so we

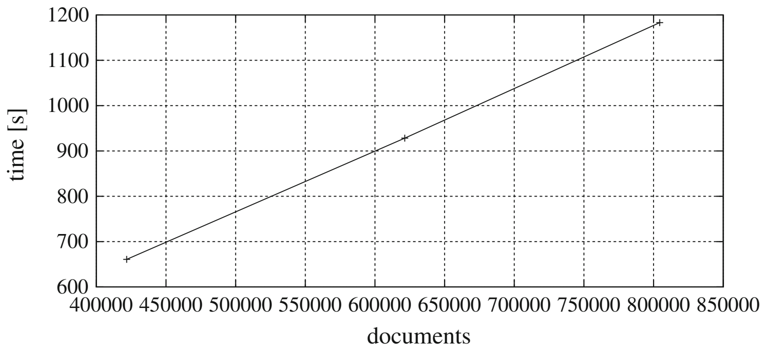

Fig. 6 Execution time for serial rare term vector replacement—the algorithm displays nearly linear complexity on one, two and three parts of the Reuters corpus

can simplify the complexity to $O(\|E\|)$. In order to compute a single replacement vector, one has to process at most t documents, because if the feature occurred in more documents it would not be eliminated. Consequently, an individual vector can be computed in $O\ (t\|E\|)$. The complexity of constructing the replacement matrix R consists of two parts: building the replacement vectors for rare features and setting a single feature to one for all others. This leads to a complexity of $O(\|E\|^2 t + k)$. Due to Zipf's law, we can assume that $k < \|E\|$, and so we obtain a practical complexity of $O(\|E\|^2 t)$ for text retrieval applications. Summarizing, we can state two bounds for the algorithmic complexity of our vector replacement algorithm:

- In general, the algorithm will never peform worse than $O(m^2 n)$.
- For text retrieval, we can assume that it is bounded by $O(\|E\|^2 t)$.

In this serial form, our method does not necessarily provide an improvement over the singular value decomposition in the algorithmic complexity, especially if $\|D\| \ll \|F\|$. But since the replacement vectors can be computed independently of each other, we have a great potential for parallel scalability.

The actual execution time still depends greatly on the non-zero elements in the corpus matrix C. Since this is difficult to evaluate, we have measured the execution time of our algorithm for 421,816, 621,392 and 804,414 documents of one, two and three parts of the Reuters corpus. We have used an E5520 Intel Nehalem Xeon running at 2.27 GHz with 4 cores equipped with 48 GiB of RAM with Hyperthreading disabled. The operating system was CentOS 64 bit version 5.2, Linux kernel version 2.6.18–128.el5 using OpenMPI version 1.3.3 and the GNU Compiler Collection version 4.1.2. The results of our measurements are depicted in Fig. 6.

Our measurements indicate that, due to the actual non-zero structure of a real-world text collection, the actual growth in the execution time appears to be linear in the size of the corpus. At present, we estimate that there is a dependency between the number of elimination features and the size of the corpus matrix, which can explain this linear behavior. Unfortunately, we cannot present any conclusive facts about this relationship at present.

6 Related Work

There are several well known dimensionality reduction techniques in the fields of numerical linear algebra and statistics. The two foremost are the singular value decomposition (SVD), see e.g. [1], and the principal component analysis (PCA), see e.g. [7]. Both methods are strongly connected and share some theoretically desirable properties such as determinism and uniqueness. Furthermore, they have been formally shown to produce the best linear approximation for any given rank, i.e. the effective dimensionality of the data matrix, as shown in [8]. In multivariate statistics, factor analysis [9] and more recently independent component analysis (ICE), see [10], attempt to determine latent statistical factors, which can provide a linear approximation of the original data. And indeed, the latter is again based on the PCA. Kernel methods [11] have successfully been applied to extend the ICE to account for non-linear data dependencies [12]. Non-negative matrix factorizations (NMF), see e.g. [13], are a more recent development that is motivated by factor analysis, where non-negativity may be necessary to interpret the factors. Multidimensional scaling (MDS), see e.g. [14], determines a projection onto a lower dimensional space while preserving pair-wise distances. Fastmap [15] is a modern technique for computing such a projection. However, it should be noted that the SVD is an optimal variant of MDS [16]. A classic geometric approach to the dimensionality reduction problem is the fitting of a mesh of grid points to produce a map, onto which the individual data points are projected. Such a map can be constructed implicitly by self-organization, see e.g. [17], or explicitly with the ISOMAP algorithm [18] or local linear embedding method (LLE) [19]. Moreover, clustering algorithms can also be used for dimensionality reduction by projecting onto the representative vectors of the clusters [20] or in a supervised variant using the centroid vectors of category-specific centroids according to a labeled set of examples [21]. But representatives for the projection can also be chosen from the document collection, see e.g. [22] for an evolutionary approach to the optimized selection.

In information retrieval, latent semantic indexing (LSI), see [23], is the straightforward application of the SVD to the task at hand. The PCA has also been applied in the COV approach [24]. Factor analysis based on the SVD applied to automated indexing has been reported as probabilistic latent semantic analysis (PLSA) in [25]. NMF methods are often used in various text classification tasks [26, 27]. The reduction by projection onto the representative vectors of a feature clustering has in fact been developed specifically for text retrieval applications [20]. The use of kernel methods can lead to a square increase in the number of features and is therefore unsuitable for sparse, high-dimensional text data.

It is difficult to compare our own dimensionality reduction technique to previous methods. It is based on an intuition about documents in the vector space model rather than statistical, numerical or geometric properties. Due to the use of linear combination of vectors scaled by relevance scores, the generalized vector space model (GVSM) [28], is much more comparable to our own method than any of the canonical methods for dimensionality reduction. In its principal form it uses term similarities as weights for the linear combination, but it has been modified in a number of ways [29, 30]. However, modifications to the query processing such as the GVSM should be considered complementary techniques that can be used in conjunction with our dimensionality reduction method.

The index vector representation has recently gained popularity in the data mining community due to the random index vector representation [31]. Random projections based on random index vectors also play a role in very fast dimensionality reduction [32, 33]. The random index vector representation can also be improved with a bag-of-concept representation [34]. Our own method is indeed an index vector representation, but it is far from random. It is a co-occurrence index vector representation, which is restricted to the common terms.

The rare term vector replacement strategy is also related to the random index vector representation. Random indexing constructs a new vector representation from random vectors based on (document-level) word co-occurrence. Analogously, the rare term index vector representation builds new vectors based on co-occurrence relationships, but the vectors used in the construction process are anything but random. The basis for this synthesis is nothing less than a part of the original document vectors, namely the common term frequency or TF-IDF scores.

If we consider rare term vector replacement as a dimensionality reduction method for text, it is essential that we compare it to latent semantic indexing. The key to this comparison is a concise understanding of the underlying singular value decomposition. There are two ways we can interpret the factor matrices: analytically and synthetically.

Let us first consider three analytical interpretations based on linear subspaces, affine transformations and statistics. In terms of linear spaces, the singular vectors are orthonormal basis vectors for the left and right subspaces spanned by the rows and columns of C. By computing a rank-reduced representation of the original matrix, the row and column spaces are truncated to subspaces by trimming the numerical row and column kernel spaces of C. Theoretically, truncating the null space has no impact on the span. But the magnitude of the singular values associated with the basis vectors of the numerical kernel of a matrix can be far from zero. The SVD therefore augments the span of C by truncating it to proper subspaces. If we interpret C geometrically as a linear transformation, then U and V can be interpreted as rotations and reflections which, together with the axis-aligned scaling matrix Σ, can represent every affine transformation in a canonical manner. The sequence of the factorization $U \Sigma V^T$ reveals the canonical three-step process: reflect / rotate, scale along the main axes and reflect / rotate again. Thirdly, there is also the statistical interpretation which is inherited from the principal component analysis. For data with a non-zero mean, the left and right singular vector correspond to the principal axes obtained by using biased, uncentered notions of covariance and correlation. Correspondingly, a projection onto these axes yields an optimal linear reconstruction of the distances with respect to the Frobenius norm.

The synthetic interpretations follow from the constructive principles of the algorithms for computing the SVD. The geometric synthetic interpretation is based on the Householder or Givens reflections and rotations, that can be used to compute the diagonalization. Synthetically, the SVD therefore corresponds to a series of reflections and rotations that are applied as left and right factors. The convergence behavior of these algorithms tells us that these reflections and rotations will always succeed in diagonalizing the matrix. Statistically, this means that the data can eventually be decoupled and presented in (numerically) uncorrelated form. Thirdly, the Lanczos algorithm allow us to interpret the SVD as a fixed point iteration with the most significant, largest fixed point as the first singular vector, and decreasing orthogonal complements following on the subsequent vectors.

For our method, we only have two interpretations. The first interpretation follows immediately from the construction principle: We form the linear combination of the centroids of the rare terms and combine them with the common terms. Alternatively, we can also interpret the product $R_E E$ as a projection of the document vectors, expressed as term frequencies, onto the index vectors for the common terms, which are expressed as the rare and common terms they co-occur with.

Both of these interpretations clearly indicate that our method is substantially different from LSI. It does not consist of reflections or rotations, and it makes no attempt to decorrelate or whiten the data. And quite unlike an unsymmetric, general singular value decomposition, it is based on a one-sided matrix product.

There is yet another formal argument for this difference. Rare term vector replacement is based on a linear combination of existing vectors. Consequently, it does not change the space spanned by the common terms in C. The truncated SVD eliminates the numerical kernel space of the corpus matrix. This is a substantial difference and makes any similarities unlikely.

A more recent technique that we need to consider in our formal, structural analysis are non-negative matrix factorizations. Typically, the goal is to approximate a positive matrix X using two factor matrices W and H, such that

$$X \approx WH. \tag{24}$$

The first observation we can make is that if we omit the subsequent spectral dimensionality reduction with the PCA, then our method produces a positive augmented corpus matrix C' for any positive input corpus C. With this in mind, we can make two important assertions about RTVR and NMFs:

1. For any positive corpus matrix C we can compute a positive augmented corpus matrix C' using the RTVR method, i.e. by taking

$$C' = RC. \tag{25}$$

Since C' is positive, we can compute a non-negative factorization

$$C' \approx WH, \tag{26}$$

and use it to obtain a positive factor analysis of the augmented corpus.

2. Since both C, R and C' are all positive, the equality in Statement 25 can be relaxed to
$$C' \approx RC, \tag{27}$$
which suggests that the replacement matrix R and the original corpus matrix C are in fact a non-negative factorization of the augmented corpus matrix C'. We can therefore "explain" the data in our augmented matrix based on the original corpus matrix and the replacement vectors in R. This formal observation brings us to a very obvious interpretation of the augmented corpus C', which states that "the (i, j)-th component in the replacement matrix R states the *membership* of an original feature in the j-th row of C with the associated common term in the i-th row of C'."

Regarding NMFs, we can therefore state that (1) RTVR can be used in conjunction with NMFs if the corpus matrix C is positive and if the subsequent rank reduction is replaced by a positive factorization, and (2) that RTVR can itself be interpreted as a particular NMF, which servers as an alternative interpretation of the formal principles of the RTVR algorithm.

7 Summary and Conclusions

In this chapter, we have introduced a novel approach to dimensionality reduction in text retrieval, which is based on the replacement of rare terms with linear combinations of their co-occurring terms. We have given a detailed description of the mathematical formulation of the corresponding linear replacement operator. Furthermore, we have given a detailed report of the algorithmic formulation in pseudo-code. We analyzed the algorithmic complexity, which is $O(m^2n)$ for m features and n documents in the most general case. For text retrieval applications, we can refine this bound to $O(\|E\|^2t)$ where t is the maximum number of containing documents for a rare feature and E the corresponding set of elimination features.

We have evaluated our approach on two standard benchmark collections, the small MEDLARS collection with 1,033 documents and the Reuters Corpus Volume I Version 2 with 23,149 documents. For both corpora, we eliminated all features which occurred in less than 1 or 3 % of all documents. The MEDLARS collection was thus reduced from 8,752 to 1,136 features using rare vector replacement and to 500 features with a subsequent conventional rank reduction. Our experiments show that both dimensionality reduced versions are competitive with the sparse vector format in terms of retrieval accuracy. For the Reuters corpus we conducted an extensive cross-evaluation using all topics as indicators for related and unrelated results and all documents as sample queries. We reduced the original 47,236 features to 525 features using vector replacement and to 392 terms using a subsequent rank reduction. This transformation consistently increased the average precision for all result list ranks. While these experiments are still preliminary, we believe that they do deliver an initial proof-of-concept for our reduction method.

In our future research, we plan to extend our experiments to a wider range of test corpora, especially large-scale text collections, to improve the empirical evidence for the utility of our method and to conduct a full-scale performance evaluation. In addition, we will investigate how we can efficiently update an existing reduction to account for new, changed and deleted documents as well as new features. Lastly, our method shows great potential for parallel implementation, because the replacement vectors can be computed independently. We hope that this will allow us to outperform the SVD in terms of scalability in future experiments.

References

1. Berry, M.W., Drmac, Z., Jessup, E.R.: Matrices, vector spaces, and information retrieval. SIAM Rev. **41**(2), 335–362 (1999)
2. Cancho, R.F.i., Solé, R.V.: Least effort and the origins of scaling in human language. Proc. US Nat. Acad. Sci. **100**(3), 788–791 (2003)
3. Powers, D.M.W.: Applications and explanations of Zipf's law. In: Proceedings of NeM-LaP3/CoNLL, pp. 151–160. ACL Stroudsburg, PA, USA (1998)
4. Lan, M., Tan, C.L., Low, H.B., Sung, S.Y.: A Comprehensive comparative study on term weighting schemes for text categorization with support vector machines. In: Proceedings of WWW'05, pp. 1032–1033. ACM, New York (2005)
5. Mao, W., Chu, W.W.: The phrase-based vector space model for automatic retrieval of free-text medical documents. Data Knowl. Eng. **61**, 76–92 (2007)
6. Lewis, D.D., Yang, Y., Rose, T.G., Li, F.: RCV1: a new benchmark collection for text categorization research. JMLR **5**, 361–397 (2004)
7. Jolliffe, I.T.: Principal Component Analysis, 2nd edn. Springer, New York (2002)
8. Eckart, C., Young, G.: The approximation of one matrix by another of lower rank. Psychometrika **1**(3), 211–218 (1936)
9. Johnson, R.A., Wichern, D.W.: Applied Multivariate Statistical Analysis, 6th edn. Prentice Hall, Upper Saddle River (2007)
10. Hyvarinen, A., Karhunen, J., Oja, E.: Independent Component Analysis. Wiley, New York (2001)
11. Aizerman, A., Braverman, E.M., Rozoner, L.I.: Theoretical foundations of the potential function method in pattern recognition learning. Automat. Rem. Contr. **25**, 821–837 (1964)
12. Schölkopf, B., Smola, A., Müller, K.R.: Nonlinear component analysis as a kernel eigenvalue problem. Neural Comput. **10**(5), 1299–1319 (1998)
13. Paatero, P., Tapper, U.: Positive matrix factorization: a non-negative factor model with optimal utilization of error estimates of data values. Environmetrics **5**(2), 111–126 (1994)
14. Cox, T.F., Cox, M.A.A.: Multidimensional Scaling. Chapman and Hall, London (2001)
15. Faloutsos, C., Lin, K.I.: FastMap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. In: Proceedings of SIGMOD'95, pp. 163–174. ACM, New York (1995)
16. Bartell, B.T., Cottrell, G.W., Belew, R.K.: Latent semantic indexing is an optimal special case of multidimensional scaling. In: Proceedings of SIGIR'92, pp. 161–167. ACM, New York (1992)
17. Campoy, P.: Dimensionality reduction by self organizing maps that preserve distances in output space. In: Proceedings of IJCNN'09, pp. 2976–2982. IEEE Press, Hoboken (2009)

18. Tenenbaum, J.B., Silva, V., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. Science **290**(5500), 2319–2323 (2000)
19. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. Science **290**(5500), 2323–2326 (2000)
20. Dhillon, I.S., Modha, D.S.: Concept decompositions for large sparse text data using clustering. Mach. Learn. **42**(1/2), 143–175 (2001)
21. Karypis, G., Han, E.H.S.: Fast supervised dimensionality reduction algorithm with applications to document categorization and retrieval. In: Proceedings of CIKM, pp. 12–19. ACM, New York (2000)
22. Aggarwal, C.: The generalized dimensionality reduction problem. In: Proceedings of SDM'10, pp. 607–618. SIAM (2010)
23. Deerwester, S.C., Dumais, S.T., Landauer, T.K., Furnas, G.W., Harshman, R.A.: Indexing by latent semantic analysis. JASIS **41**(6), 391–407 (1990)
24. Kobayashi, M., Aono, M., Takeuchi, H., Samukawa, H.: Matrix computations for information retrieval and major and outlier cluster detection. J. Comput. Appl. Math. **149**(1), 119–129 (2002)
25. Hofmann, T.: Probabilistic latent semantic indexing. In: Proceedings of SIGIR'99, pp. 50–57. ACM, New York (1999)
26. Janecek, A.G., Gansterer, W.N.: Utilizing nonnegative matrix factorization for e-mail classification problems. In: Berry, M.W., Kogan, J. (eds.) Survey of Text Mining III: application and Theory. Wiley, New York (2010)
27. Langville, A.N., Berry, M.W.: Nonnegative matrix factorization for document classification. In: NGDM, chap. 17, pp. 339–356. Chapman and Hall/CRC, London (2008)
28. Wong, S.K.M., Ziarko, W., Wong, P.C.N.: Generalized vector spaces model in information retrieval. In: Proceedings of SIGIR'85, pp. 18–25. ACM, New York (1985)
29. Hussain, S.F., Bisson, G.: Text categorization using word similarities based on higher order co-occurrences. In: Proceedings of SDM'10, pp. 1–12. SIAM (2010)
30. Tsatsaronis, G., Panagiotopoulou, V.: A generalized vector space model for text retrieval based on semantic relatedness. In: Proceedings of EACL'09, pp. 70–78. ACL Stroudsburg, PA, USA (2009)
31. Kanerva, P., Kristoferson, J., Holst, A.: Random indexing of text samples for latent semantic analysis. In: Proceedings of Cognitive Sciences, pp. 103–6. Erlbaum, Hillsdale (2000)
32. Bingham, E., Mannila, H.: Random projection in dimensionality reduction: applications to image and text data. In: Proceedings of KDD, pp. 245–250. ACM, New York (2001)
33. Sakai, T., Imiya, A.: Fast Spectral clustering with random projection and sampling. In: Machine Learning and Data Mining in Pattern Recognition, pp. 372–384. Springer, New York (2009)
34. Carrillo, M., Eliasmith, C., López-López, A.: Combining text vector representations for information retrieval. In: Proceedings of TDF, pp. 24–31. Springer, New York (2009)