Dimensionality Reduction for Information Retrieval using Vector Replacement of Rare Terms

Tobias Berka [*] Marian Vajteršic [*†]

Abstract

Dimensionality reduction by algebraic methods is an established technique to address a number of problems in information retrieval. These methods are known to alleviate synonymy and polysemy, but they convert the highly sparse corpus matrices into dense matrix format, although with a reduced dimensionality. However, we can use fast, dense matrix arithmetic instead of the sparse matrix programs, which are unsuitable for modern multi-scalar processors due to the prevalence of conditional branching. In this paper, 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.

A version of this report has been published as "Tobias Berka and Marian Vajteršic: Dimensionality Reduction for Information Retrieval using Vector Replacement of Rare Terms. Proceedings of the Ninth Text Mining Workshop (TM'11), 2011."

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. In some applications the reduction is used to discover and account for latent dependencies between features which are not reflected in the original data [4]. 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 (see e.g. [25] [6]). 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

[*]Department of Computer Sciences, University of Salzburg, Austria, tberka@cosy.sbg.ac.at, marian@cosy.sbg.ac.at.
[†]Department of Informatics, Mathematical Institute, Slovak Academy of Sciences, Bratislava, Slovakia

term weighting approach (see e.g. [20]). 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 paper is structured as follows. In Section 2 we will briefly examine the state-of-the-art in literature and how it relates to our approach. We give a mathematical description of our method in Section 3, discuss the algorithmic aspects in Section 4 and evaluate the performance on real-world data sets in Section 5. Lastly, we summarize our findings in Section 6.

2 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. [4], and the principal component analysis (PCA), see e.g. [17]. 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 [10]. In multivariate statistics, factor analysis [16] and more recently independent component analysis (ICE), see [14], 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 [2] have successfully been applied to extend the ICE to account for non-linear data dependencies [27]. Non-negative matrix factorizations (NMF), see e.g. [24], 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. [7], determines a projection onto a lower dimensional space while preserving pair-wise distances. Fastmap [11] is a modern technique for computing such a projection. However, it should be noted that the SVD is an optimal variant of MDS [3]. 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. [5], or explicitly with the ISOMAP algorithm [28] or local linear embedding method (LLE) [26]. Moreover, clustering algorithms can also be used for dimensionality reduction by projecting onto the representative vectors of the clusters [9] or in a supervised variant using the centroid vectors of category-specific centroids according to a labeled set of examples [18]. But representatives for the projection can also be chosen from the document collection, see e.g. [1] for an evolutionary approach to the optimized selection.

In information retrieval, latent semantic indexing (LSI), see [8], is the straightforward application of the SVD to the task at hand. The PCA has also been applied in the COV approach [19]. Factor analysis based on the SVD applied to automated indexing has been reported as probabilistic latent semantic analysis (PLSA) in [12]. NMF methods are often used in various text classification tasks [15] [21]. The reduction by projection onto the representative vectors of a feature clustering has in fact been developed specifically for text retrieval

applications [9]. 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.

However, our own dimensionality reduction technique 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), see [30], 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, see e.g. [13] [29]. 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.

3 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 $\text{col}(d) \in \{1, ..., n\}$. As a convenient notation, we use d_j to denote $\text{col}(d) = j$. Analogously, every feature $f \in F$ has a row index $\text{row}(f) \in \{1, ..., m\}$, using the short form $f_i :\Leftrightarrow \text{row}(f) = i$. We can thus form a corpus matrix $C \in \mathbb{R}^{m \times n}$, which contains the documents' feature vectors as column vectors

$$C \quad = \quad \left[\begin{array}{cccc} d_1 & d_2 & \cdots & d_n \end{array} \right] \in \mathbb{R}^{m \times n}.$$

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

$$\mathcal{D}(f_i) := \{d_j \in D \mid C_{i,j} \neq 0\}.$$

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

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

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

$$E := \mathcal{N}(t),$$

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

$$k := m - \|E\|.$$

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 $E = \mathcal{N}(t)$.

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 $C' = RC \in \mathbb{R}^{k \times n}$.

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
$\mathcal{D}(f)$	Documents containing feature f,
$\mathcal{F}(d)$	Features occurring in document d,
$\mathcal{N}(t)$	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,
$\mathcal{R}_E$	Replacement operator (applies replacement vectors),
R_E	Replacement matrix (equivalent to $\mathcal{R}_E$).

Table 1: Mathematical Notation

3.1 Basic Formulation

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

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

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

$$\tau_{\emptyset}(v) := v, \quad \tau_{\{A,b\}}(v) := \tau_{\{b\}}\left(\tau_A\left(v\right)\right).$$

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 $\mathcal{D}(f_i)$. We will truncate all document vectors $d_j \in \mathcal{D}(f_i)$, 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:

$$\begin{aligned} \rho_E(f_i) &:= \frac{1}{\lambda_E(f_i)} \sum_{d_j \in \mathcal{D}(f_i)} C_{i,j}\, \tau_E\left(C_{\star,j}\right), \quad \text{where} \\ \lambda_E(f_i) &:= \sum_{d_j \in \mathcal{D}(f_i)} |C_{i,j}|. \end{aligned}$$

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* $\mathcal{R}_E$. 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:

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

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

$$\mathcal{R}_E(C) := \left[\mathcal{R}_E(d_1) \quad \cdots \quad \mathcal{R}_E(d_n)\right].$$

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.

3.2 Matrix Representation

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

$$e_i^T = (0 \cdots 0 \underset{i}{1} 0 \cdots 0),$$

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

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

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

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

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

$$C' := R_E C,$$

because it is easily verified that

$$R_E C = [\mathcal{R}_E(d_1) \ \mathcal{R}_E(d_2) \cdots \mathcal{R}_E(d_n)] = \mathcal{R}_E(C).$$

Therefore, R_E is the matrix representation of the linear replacement operator $\mathcal{R}_E$ 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

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

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.

3.3 Queries and Index Updating

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.

3.4 Subsequent Rank Reduction

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 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

$$C'C'^T \approx P_h S_h P_h^T.$$

The choice of h is as difficult a decision as the selection of the rank of the SVD in an LSI 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. We can now 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.

4 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 ∈ F do
    if ‖D(f_i)‖ ≤ t then E := E ∪ {f_i};
    else π(i) := k,   k += 1;
k -= 1;
for f_i ∈ E do
    l := 0;
    for d_j ∈ D(f_i) do
        R(1 : k, i) += C(i, j) * τ_E(C(1 : m, j));
        l += |C_{i,j}|;
    if l ≠ 0 then R(1 : k, i) /= l;
for f_i ∉ E do  R(1 : k, i) := e_{π(i)};
```

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

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 $O(m^2 n)$. 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. $c = \underset{d \in D}{\operatorname{argmax}} \| \{j \in \{1, ..., m\} \mid d_j \neq 0\} \|$. 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 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 perform no worse than $O(m^2 n)$.
- For text retrieval applications, we can assume that it is bounded by $O(\|E\|^2 t)$.

In this serial form, our method does not necessarily provide an improvement 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. We plan to investigate this issue in our future research.

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 $\mathcal{F} : D \to F$ that determines which features occur in a given document, formally $\mathcal{F}(d_j) := \{f_i \in F \mid C_{i,j} \neq 0\}$. Using this function, Algorithm 2 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.

Unfortunately, we cannot present a full performance comparison at this early stage of our research. But we would like to point out some initial observations. The simultaneous accumulation of all replacement vectors using vectorized processing incurs a small overhead, which can lead to a performance degradation for small data sets. However, we have observed a speed-up on larger data sets.

5 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. [23]. 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 and more realistic test, we have used the Reuters Corpus Volume I in its corrected second version (RCV1-v2) [22]. 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.

We have conducted a preliminary evaluation of the sensitivity of our reduction with respect to the choice of parameters, the threshold t and the reduced rank k of the subsequent PCA. Currently, we believe that the performance of our method is potentially quite stable under a relatively wide range of values for t, but that some care must be taken when choosing k. Overzealous attempts at reducing the dimensionality with the spectral rank reductions are met with a drastic drop in retrieval performance.

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 seconds using Algorithm 1 or 2.6 seconds 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

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 ∈ D do
    for f_i ∈ F(d_j) do
        N(i) += 1;
k := 1;
for f_i ∈ F do
    if N(i) ≤ t then
        E := E ∪ {f_i};
    else
        π(i) := k;
        k += 1;
k -= 1;
for d_j ∈ D do
    G := E ∩ F(d_j);
    for f_i ∈ G do
        R(1 : k, i) += C(i, j) * τ_E(C(1 : m, j));
        l_i += |C(i, j)|;
for f_i ∈ F do
    if f_i ∈ E then
        R(1 : k, i) := (l(i))^{-1} * R(1 : k, i);
    else
        R(1 : k, i) := e_{π(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.

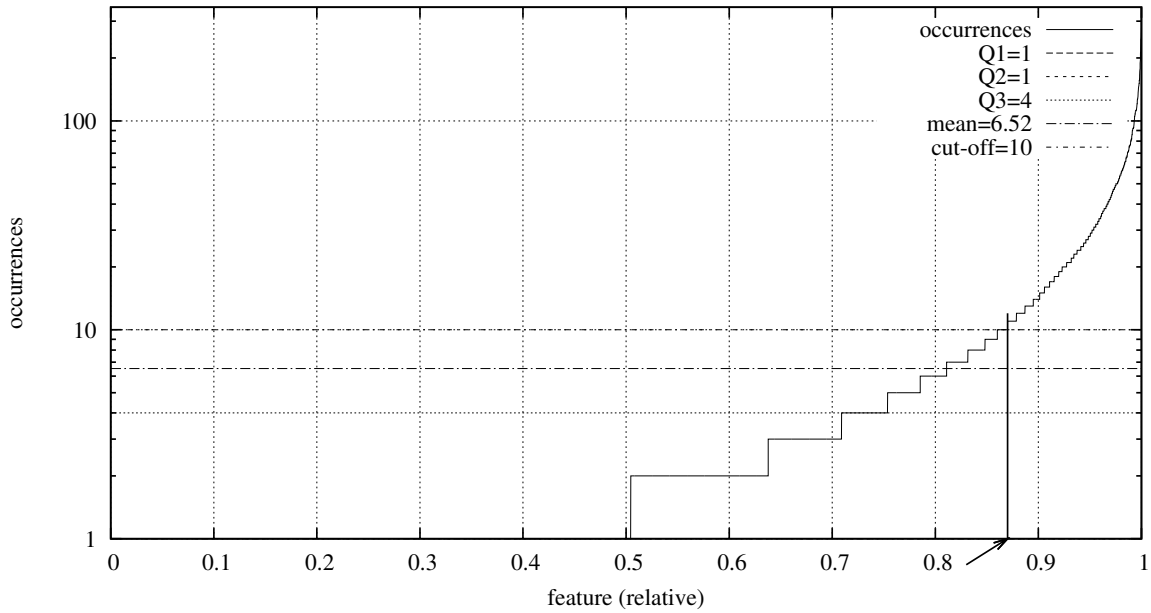

Figure 1: Feature occurrence counts for the MEDLARS corpus – depicted along with the quartiles, the sampling mean and the cut-off threshold for rare features. The vertical line indicated by the arrow shows the dividing line between rare and common features.

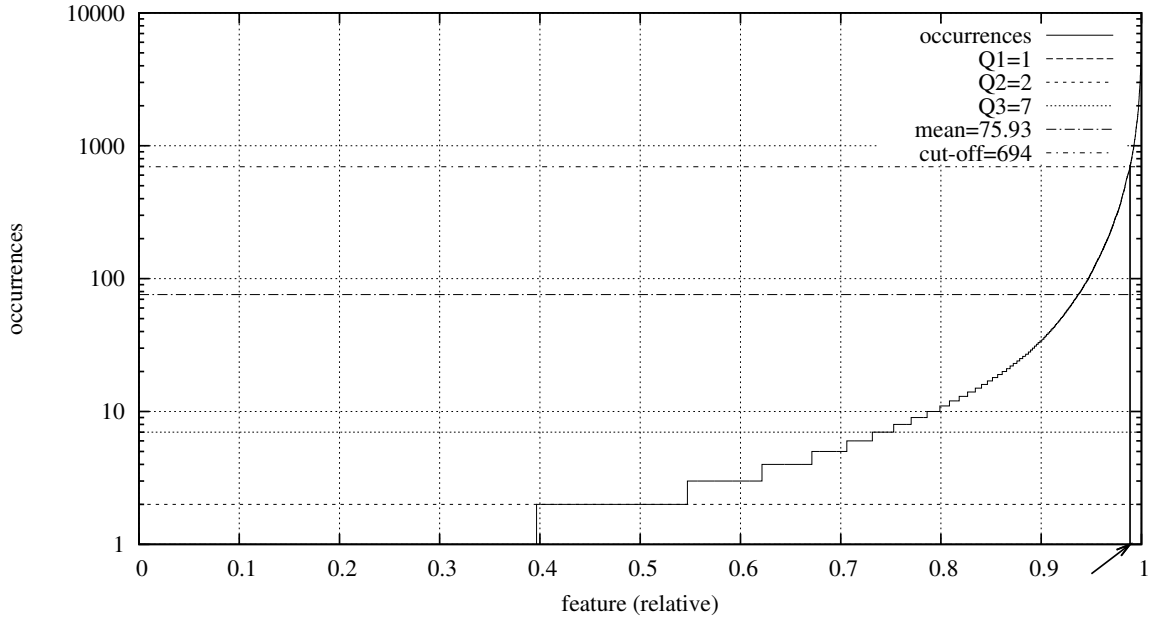

Figure 2: Feature occurrence counts for the Reuters Corpus Volume I Version 2 – depicted along with the quartiles, the sampling mean and the cut-off threshold for rare features. The vertical line indicated by the arrow shows the dividing line between rare and common features.

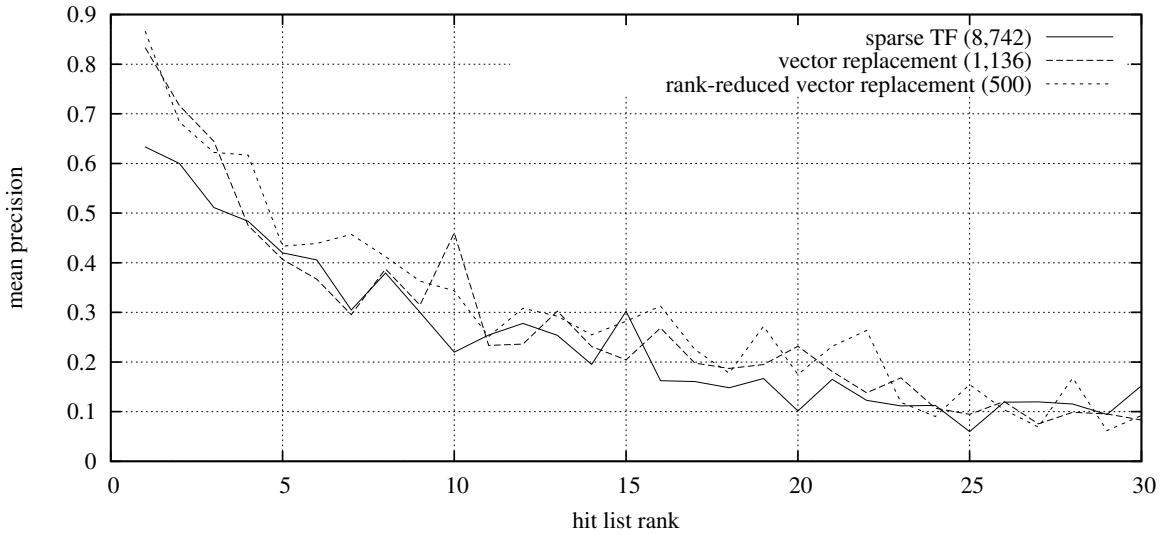

Figure 3: Precision-at-k retrieval performance evaluation on the MEDLARS benchmark corpus with 1,033 documents and 30 sample queries. The hit list rank k was sampled from 1 to 30. This measurement has been conducted with (1) the raw, sparse term-frequency vectors, (2) the replacement vector approach 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.

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 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 Figure 3. 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 minutes using Algorithm 1 and just under 6 minutes with Algorithm 2. While we cannot give a serious performance comparison at this stage, 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 performed a subsequent rank-reduction creating 392 features as a third representation for our evaluation. Figure 4 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.

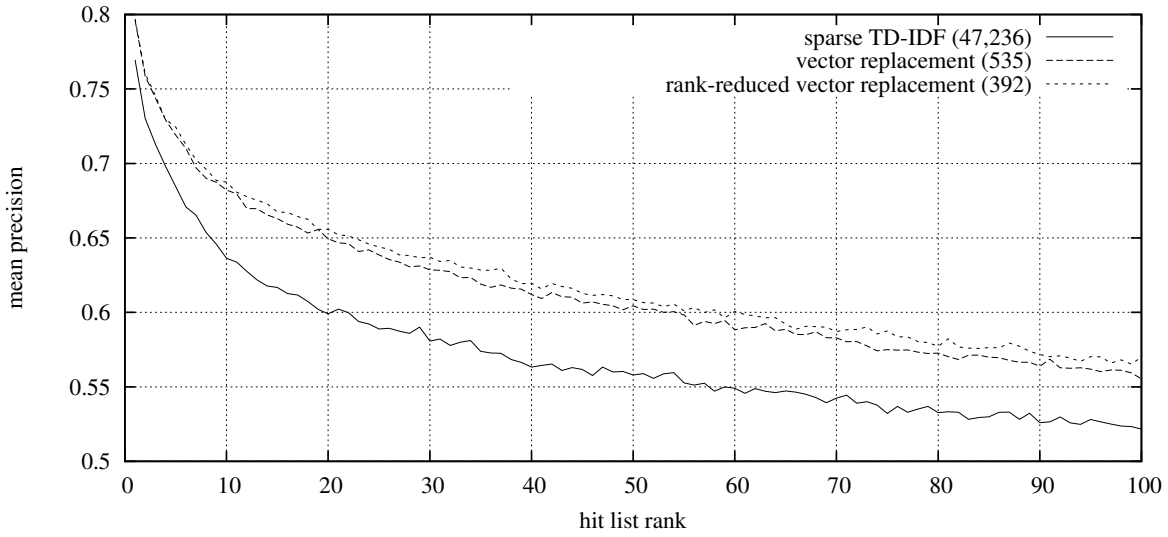

Figure 4: All-documents all-categories evaluation of the precision-at-k retrieval performance on the 23,149 training documents of the Reuters Corpus Volume I Version 2 with the topic categorization. Queries have been conducted using every document as a query example. For all categories of the query example, 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.

6 Summary & Conclusions

In this paper, 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] Charu Aggarwal. The Generalized Dimensionality Reduction Problem. In *Proc. SDM'10*, pages 607–618. SIAM, 2010.

[2] A. Aizerman, E. M. Braverman, and L. I. Rozoner. Theoretical Foundations of the Potential Function Method in Pattern Recognition Learning. *Automat. Rem. Contr.*, 25:821–837, 1964.

[3] Brian T. Bartell, Garrison W. Cottrell, and Richard K. Belew. Latent Semantic Indexing is an Optimal Special Case of Multidimensional Scaling. In *Proc. SIGIR'92*, pages 161–167, USA, 1992. ACM.

[4] Michael W. Berry, Zlatko Drmac, and Elizabeth R. Jessup. Matrices, Vector Spaces, and Information Retrieval. *SIAM Review*, 41(2):335–362, 1999.

[5] Pascual Campoy. Dimensionality Reduction by Self Organizing Maps that Preserve Distances in Output Space. In *Proc. IJCNN'09*, pages 2976–2982, USA, 2009. IEEE Press.

[6] Ramon Ferrer i Cancho and Ricard V. Solé. Least Effort and the Origins of Scaling in Human Language. *Proc. US National Academy of Sciences*, 100(3):788–791, 2003.

[7] Trevor F. Cox and Michael A. A. Cox. *Multidimensional Scaling*. Chapman & Hall, 2001.

[8] Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. Indexing by Latent Semantic Analysis. *JASIS*, 41(6):391–407, 1990.

[9] Inderjit S. Dhillon and Dharmendra S. Modha. Concept Decompositions for Large Sparse Text Data using Clustering. *Mach. Learn.*, 42(1/2):143–175, 2001.

[10] Carl Eckart and Gale Young. The Approximation of One Matrix by Another of Lower Rank. *Psychometrika*, 1(3):211–218, 1936.

[11] Christos Faloutsos and King-Ip Lin. FastMap: A Fast Algorithm for Indexing, Data-Mining and Visualization of Traditional and Multimedia Datasets. In *Proc. SIGMOD'95*, pages 163–174. ACM, 1995.

[12] Thomas Hofmann. Probabilistic Latent Semantic Indexing. In *Proc. SIGIR'99*, pages 50–57, USA, 1999. ACM.

[13] Syed Fawad Hussain and Gilles Bisson. Text Categorization Using Word Similarities Based on Higher Order Co-occurrences. In *Proc. SDM'10*, pages 1–12. SIAM, 2010.

[14] Aapo Hyvarinen, Juha Karhunen, and Erkki Oja. *Independent Component Analysis*. J. Wiley, 2001.

[15] Andreas G.K. Janecek and Wilfried N. Gansterer. Utilizing Nonnegative Matrix Factorization for E-mail Classification Problems. In Michael W. Berry and Jacob Kogan, editors, *Survey of Text Mining III: Application and Theory*. Wiley, 2010.

[16] Richard A. Johnson and Dean W. Wichern. *Applied Multivariate Statistical Analysis*. Prentice Hall, 6th edition, 2007.

[17] Ian T. Jolliffe. *Principal Component Analysis*. Springer, 2nd edition, 2002.

[18] George Karypis and Eui-Hong (Sam) Han. Fast Supervised Dimensionality Reduction Algorithm with Applications to Document Categorization & Retrieval. In *Proc. CIKM '00*, pages 12–19, USA, 2000. ACM.

[19] M. Kobayashi, M. Aono, H. Takeuchi, and H. Samukawa. Matrix Computations for Information Retrieval and Major and Outlier Cluster Detection. *J. Comput. Appl. Math.*, 149(1):119 – 129, 2002.

[20] Man Lan, Chew-Lim Tan, Hwee-Boon Low, and Sam-Yuan Sung. A Comprehensive Comparative Study on Term Weighting Schemes for Text Categorization with Support Vector Machines. In *Proc. WWW'05*, pages 1032–1033, New York, NY, USA, 2005. ACM.

[21] Amy N. Langville and Michael W. Berry. Nonnegative Matrix Factorization for Document Classification. In *NGDM*, chapter 17, pages 339–356. Chapman & Hall/CRC, 2008.

[22] David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li. RCV1: A New Benchmark Collection for Text Categorization Research. *JMLR*, 5:361–397, December 2004.

[23] Wenlei Mao and Wesley W. Chu. The Phrase-based Vector Space Model for Automatic Retrieval of Free-Text Medical Documents. *Data Knowl. Eng.*, 61:76–92, April 2007.

[24] Pentti Paatero and Unto Tapper. Positive Matrix Factorization: A Non-negative Factor Model With Optimal Utilization of Error Estimates of Data Values. *Environmetrics*, 5(2):111–126, 1994.

[25] David M. W. Powers. Applications and Explanations of Zipf's Law. In *Proc. NeMLaP3/CoNLL '98*, pages 151–160, USA, 1998. ACL.

[26] Sam T. Roweis and Lawrence K. Saul. Nonlinear Dimensionality Reduction by Locally Linear Embedding. *Science*, 290(5500):2323–2326, 2000.

[27] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear Component Analysis as a Kernel Eigenvalue Problem. *Neural Comput.*, 10(5):1299–1319, 1998.

[28] J. B. Tenenbaum, V. Silva, and J. C. Langford. A Global Geometric Framework for Nonlinear Dimensionality Reduction. *Science*, 290(5500):2319–2323, 2000.

[29] George Tsatsaronis and Vicky Panagiotopoulou. A Generalized Vector Space Model for Text Retrieval based on Semantic Relatedness. In *Proc. EACL '09*, pages 70–78, USA, 2009. ACL.

[30] S. K. M. Wong, Wojciech Ziarko, and Patrick C. N. Wong. Generalized Vector Spaces Model in Information Retrieval. In *Proc. SIGIR'85*, pages 18–25, USA, 1985. ACM.