Learning Bag-of-Embedded-Words Representations for Textual Information Retrieval

Nikolaos Passalis*, Anastasios Tefas

Department of Informatics
Aristotle University of Thessaloniki
Thessaloniki 54124, Greece

Abstract

Word embedding models are able to accurately model the semantic content of words. The process of extracting a set of word embedding vectors from a text document is similar to the feature extraction step of the Bag-of-Features (BoF) model, which is usually used in computer vision tasks. This gives rise to the proposed Bag-of-Embedded Words (BoEW) model that can efficiently represent text documents overcoming the limitations of previously predominantly used techniques, such as the textual Bag-of-Words model. The proposed method extends the regular BoF model by a) incorporating a weighting mask that allows for altering the importance of each learned codeword and b) by optimizing the model end-to-end (from the word embeddings to the weighting mask). Furthermore, the BoEW model also provides a fast way to fine-tune the learned representation towards the information need of the user using relevance feedback techniques. Finally, a novel spherical entropy objective function is proposed to optimize the learned representation for retrieval using the cosine similarity metric.

Keywords: Word Embeddings, Bag-of-Words, Bag-of-Features, Dictionary Learning, Relevance Feedback, Information Retrieval

1. Introduction

The textual *Bag-of-Words* (BoW) representation [1], is among the prevalent techniques used for textual Information Retrieval (IR). In the textual BoW model a set of predefined words, called *dictionary*, is selected and then each document is represented by a histogram vector that counts the number of appearances of each word in the document. Its great success in IR tasks has led a great deal of research to be devoted to improve the textual BoW model. For example, some techniques focused on pruning the dictionary [2], while other methods on improving the extracted histograms by applying a weighting scheme, such as the tf-idf (term frequency/inverse document frequency) method [1, 3, 4]. Furthermore, other more advanced

*Corresponding author
Email addresses: passalis@csd.auth.gr (Nikolaos Passalis), tefas@aiia.csd.auth.gr (Anastasios Tefas)

techniques, such as the Latent Dirichlet Allocation (LDA) [5], also use word occurrence statistics to model each document as a *mixture of topics*.

Word embedding models are capable of extracting semantically-enriched representations of words. However, it is not straightforward to use them to encode whole documents. Perhaps the most commonly used technique to overcome this limitation is to simply calculate the average word embedding vector of a document [6, 7, 8, 9, 10]. Also, a few sophisticated techniques, such as the Paragraph Vector [11, 12], have been proposed to directly calculate embeddings of documents. However, the averaging process ignores part of the information that the document contains, while the paragraph embedding requires a computationally intensive inference step to provide out-of-sample embeddings, which limits its applicability.

In this paper we propose a method that is capable of overcoming the aforementioned limitations by using an *efficient end-to-end trainable text representation* scheme that exploits the representation power of semantic-enriched word embeddings and is inspired by the well known Bag-of-Features (BoF) model. The proposed method also aims at providing a link between the textual Bag-of-Words model, mainly used by the natural language processing and information retrieval communities, and the feature-based Bag-of-Features model, mainly used by the computer vision community. That way, this work paves the way for developing powerful text representation machines for information retrieval building upon the extensive existing research on the BoW-based techniques [1, 2, 3, 4, 5], as well as on the BoF-based methods [13, 14, 15, 16, 17, 18].

To better understand the link between the BoW and BoF methods, note that the process of extracting a word embedding (feature) vector for each word of a document is similar to the feature extraction step that is used in order to represent multimedia objects, such as images and videos [19, 20]. For example, for image recognition/retrieval tasks it is common to extract multiple SIFT vectors [21], from an image and then use the Bag-of-Features technique, also known as Bag-of-Visual Words (BoVW), to extract a constant dimensionality vector from each image [22, 13]. Thus, a text document comprises of a set of feature vectors (word embeddings) in a similar way to an image that comprises of a set of visual feature vectors (e.g., SIFT vectors). The pipeline of the BoF model can be summarized as follows:

1. *feature extraction*, in which multiple features, such as SIFT descriptors [21], are extracted from each object, e.g., image. That way, the *feature space* is formed where each object is represented as a set of features.
2. *codebook learning*, in which the extracted features are used to learn a *codebook* of representative features (also called *codewords*),
3. *feature quantization* and *encoding*, in which each feature vector is represented using a codeword from the learned codebook and a histogram is extracted for each object. That way, the *vector space* (also referred to as *representation space* or *histogram space* in BoF-based dictionary learning literature) is formed, where each object is represented by a constant dimensionality term/histogram vector, similarly

to the vector space model [1].

The similarity between extracting a set of word embedding vectors from a text document and extracting a set of feature vectors from a multimedia object was noticed quite recently [7, 23, 24]. Exploiting this similarity allows us to use the BoF model to extract representations from text documents using the extracted word embedding vectors as feature vectors. The application of the BoF model in the context of text representation is called Bag-of-Embedded Words (BoEW) model. In [24] the BoF model is used to encode text documents using the extracted word embedding vectors, while in [23], and [7], Fisher vector encoding was used instead to represent each document.

Also, it has been well established that using unsupervised algorithms, such as k-means [25], to learn the codebook of the BoF/BoEW representation leads to suboptimal results [14, 26]. Therefore, the codebook of the BoF/BoEW model must be optimized towards the task at hand. Although a wide range of methods exist for learning discriminative dictionaries, e.g., [14, 27, 28, 29], many of them produce highly discriminative representations that are not always optimal for retrieval tasks. This phenomenon was studied and explained in [13], where an entropy-based retrieval-oriented objective function was proposed. In the case of [13], and [24], the Euclidean distance was used to calculate the entropy. Therefore, the learned representation was optimized for retrieval using the Euclidean distance. However, in most cases using the cosine similarity instead of the Euclidean distance significantly increases the retrieval precision. Motivated by this observation a new type of entropy is proposed in this work, the *spherical entropy*, that optimizes the representation for retrieval using the cosine similarity. In Section 4 it was experimentally demonstrated that this can lead to significant improvements in the retrieval precision.

Furthermore, the BoEW model is extended using a weighting mask that allows us to alter the importance of each codeword. Note that this is similar to the weighting schemes used in the classical BoW schemes, such as the tf-idf. The purpose of using the proposed weighting mask is two-fold: a) it allows for further optimizing the learned representation towards the task at hand and b) allows for quickly fine-tuning the representation towards the information need of the user using relevance feedback techniques [30, 31, 32]. The latter is especially important, since a) it provides a very fast way to adjust the representation using the user's feedback without having to re-encode the whole database and b) allows for optimizing the representation when only a few annotated documents are available.

The main contributions of this paper are briefly summarized below. First, the BoF model is adjusted towards representing text documents using word embeddings leading to the proposed BoEW model. The proposed model utilizes a histogram-space weighting mask, inspired by the weighting schemes used in the BoW models, that increases the flexibility of the model and allows for further fine-tuning the representation towards different tasks. Also, the proposed BoEW model is optimized end-to-end, i.e., all the parameters of the model (the word embedding, the codebook, the scaling factor and the weighting mask) are simultane-

ously learned using the proposed spherical entropy objective, which optimizes the learned representation for retrieval using the cosine similarity. Furthermore, two different optimization algorithms are proposed for the BoEW model: a) an offline algorithm for optimizing the representation using an annotated set of documents and b) a relevance feedback algorithm that allows for quickly optimizing the representation and re-querying the database using the feedback from the user. Finally, both algorithms are evaluated using three collections of text documents from a diverse range of domains and it is demonstrated that they can both increase the retrieval precision and reduce the size of the extracted representation (increasing the retrieval speed and reducing the storage requirements) for both in-domain and out-of-domain retrieval tasks.

The rest of the paper is structured as follows. The related work is discussed in Section 2 and the proposed method is presented in Section 3. The experimental evaluation of the proposed method is presented in Section 4 and conclusions are drawn in Section 5. Finally, note that a reference implementation of the proposed method will be provided at `http://github.com/passalis/boew` to enable other researcher easily use and extend the proposed technique.

2. Related Work

Early text retrieval approaches used the term frequency/inverse document frequency (tf-idf) method to represent documents as vectors. Then, relevant documents can be retrieved by measuring the similarity between a query vector and document vectors stored in the database (vector space model) [1]. Several methods were subsequently developed building upon this model, ranging from tf-idf variants and extensions, such as [3, 4], to more advanced topic-based analysis techniques, e.g,. Latent Semantic Indexing (LSI) [33], Probablistic Latent Semantic Indexing (PLSI) [34], and Latent Dirichlet Allocation (LDA) [5].

Even though the aforementioned techniques were used with great success for several information retrieval tasks [1], they ignore part of the semantic relationships between the words that compose the dictionary (the term vectors are equidistant to each other and, thus, fail to capture the semantic similarity between different words). This problem gave rise to word embedding models, such as [35] and [36], where each word is mapped to a dense real-valued vector that captures the semantic properties of the corresponding word and encode the underlying linguistic patterns. That is, vectors that correspond to words with similar meaning are closer to each other than vectors for words with irrelevant semantic content. Among the most well known word embedding models is the word2vec model [35], and the GloVe model [36]. Both use unsupervised algorithms to learn word embeddings either by predicting the words in a given window or by using word co-occurrence statistics. In contrast to these methods, the proposed approach concerns document representation instead of learning embeddings of individual words.

However, it is not straightforward to use word embeddings to represent a document that is composed of multiple words. Among the most commonly used, yet naive, approaches is to average all the word embedding

vectors that correspond to the words that a document contains. Even though this approach is widely used in many natural language processing tasks [6, 7, 8, 9, 10], the averaging process leads to loss of valuable information that a document contains. To overcome this limitation, Paragraph Vector [11, 12, 37], and word embedding-based document-level distance metrics [38], have been proposed. Paragraph Vector is a state-of-the-art document representation technique that is capable of directly extracting *document* embeddings. Word Mover's Distance (WMD) [38], provides a way to calculate the distance between two documents that are composed of multiple word embedding vectors. However, note that paragraph embedding requires a computationally intensive inference step to provide out-of-sample embeddings, which often limits its practical applicability. Similar computational constrains also exist for the WMD, for which usually approximate solutions are used. In contrast to these techniques, the proposed method uses trainable quantization to *learn* how to represent a document, instead of just averaging the word embedding vectors, and does not require any optimization during the inference for providing out-of-sample embeddings or the calculation of computationally demanding distance metrics.

The proposed method also concerns dictionary learning for the BoF representation, where a rich literature exists. The related methods can be classified into two categories according to the used optimization objective. The methods of the first category assume that the feature vectors of an object carry the same label as the object and set the optimization objective in the feature space [27, 39, 40]. The methods of the second category tie the classifier and the codebook learning and rely on the classifier's decisions to optimize the codebook. In [14], [41], and [28], max-margin formulations are used to learn the codebook, while in [42] a multilayer perceptron (MLP) is used to backpropagate the error to the dictionary. In [43] multiple dictionaries with complementary discriminative information are learned. Some other approaches used a discriminative criterion in the histogram space instead of a classifier to optimize the codebook. These methods focused on learning class-specific dictionaries [29], [44], or adjusting the dictionary in order to increase a specific objective such as the mutual information [45], or the ratio of intra/inter class variation [16, 26, 46].

However, these methods focus on learning highly discriminative dictionaries optimized towards classification tasks. In [13] the use of an entropy-based retrieval-oriented objective function was proposed. This objective was also successfully used to optimize a simple BoEW model towards retrieval using the Euclidean distance [24]. In this paper the entropy objective is extended to allow us to optimize the representation for retrieval using the cosine similarity. Also, to the best of our knowledge, this is the first work that incorporates a weighting mask into the BoEW model that alters the importance of each codeword and allows for a) the end-to-end optimization of the the parameters of model towards information retrieval and b) providing a way to quickly fine-tune the representation using relevance feedback.

3. Proposed Method

In this Section the proposed Bag-of-Embedded-Words (BoEW) model is described. Then, the concept of entropy, which is used as a retrieval-oriented objective, is briefly introduced and the proposed spherical entropy is described. Finally, the proposed method for the optimization of the BoEW model using the spherical entropy as well as a BoEW-based relevance feedback technique are derived.

3.1. BoEW Model

Let $\mathcal{D}$ be the textual dictionary, i.e., a list of the unique words that appear in a collection of text documents. Let $\mathbf{x}_i \in \mathbb{R}^D$ be the word embedding vector of the i-th word of the dictionary $\mathcal{D}$, where D is the dimensionality of the word embedding. The set of all word embedding vectors is denoted by $\mathcal{E} = \{\mathbf{x}_i,\ i = 1...|\mathcal{D}|\}$. Also, let N be the number of the text documents that are to be encoded using the proposed BoEW model. The i-th document is described by N_i word embedding (feature) vectors: $\mathbf{x}_{ij} \in \mathcal{E}$ ($i = 1...N$, $j = 1...N_i$), where N_i is the number of words of the i-th document. As described before, each feature vector corresponds to a word of the document, i.e., the feature vector $\mathbf{x}_{ij}$ is the embedding vector for the j-th word of the i-th text document. The number of the extracted feature vectors may vary according to the number of the words that appear in each text document. However, the BoEW model is capable of representing each document using a fixed-length vector of its quantized feature vectors regardless the number of the extracted word embedding vectors.

To quantize the word embedding vectors a set of "prototype" word embedding vectors, is learned. This set is corresponds to the codebook that is used in the regular BoF model. When hard assignment is used each feature vector is quantized to its nearest codeword, while when soft-assignment is used every feature vector contributes, by a different amount, to each codeword [17]. In this work only soft-assignments are considered, since hard-assignments does not allow for the end-to-end optimization of the model. The codebook is learned by clustering the set of all word embedding vectors $\mathcal{E}$ into N_K clusters. Then, the corresponding centroids (codewords) $\mathbf{v}_k \in \mathbb{R}^D (k = 1...N_K)$ are used to form the codebook $\mathbf{V} \in \mathbb{R}^{D \times N_K}$, where each column of $\mathbf{V}$ is a centroid vector. These centroids are used to quantize the word embedding vectors. It should be noted that the codebook is learned only once and then it can be used to encode any new document.

To encode the i-th document, the similarity between each word embedding vector $\mathbf{x}_{ij}$ and each codeword $\mathbf{v}_k$ is computed as:

$$d_{ijk} = exp(-\frac{||\mathbf{v}_k - \mathbf{x}_{ij}||_2}{\sigma^2}) \in \mathbb{R}. \quad (1)$$

The parameter σ (scaling factor) controls the quantization process: for harder assignment a small value, e.g., $\sigma < 0.1$, is used, while for softer assignment larger values, e.g., $\sigma > 0.1$, are used. Note that in contrast to other soft BoF formulations, e.g., [13, 14, 16, 17], the square of the quantization parameter is used. Even though this is equivalent to the other soft BoF formulations (simply by setting $\sqrt{\sigma}$ as the quantization

parameter) it allows for learning the optimal scaling factor without having to guard for negative values (since the scaling factor must be always a positive number).

Then, the l^1 normalized membership vector of each feature vector $\mathbf{x}_{ij}$ is obtained by:

$$\mathbf{u}_{ij} = \frac{\mathbf{d}_{ij}}{||\mathbf{d}_{ij}||_1} \in \mathbb{R}^{N_K}. \tag{2}$$

This vector describes the similarity of feature vector $\mathbf{x}_{ij}$ to each codebook vector. Note that l^1 normalization is used to ensure that a histogram distribution over the codewords is obtained for each feature vector. Then, the histogram representation for the i-th document $\mathbf{z}_i$ is extracted as:

$$\mathbf{z}_i = \frac{1}{N_i} \sum_{j=1}^{N_i} \mathbf{u}_{ij} \in \mathbb{R}^{N_K}. \tag{3}$$

The histogram $\mathbf{z}_i$ has unit l^1 norm, since $||\mathbf{u}_{ij}||_1 = 1$ for every j. These histograms describe each document and they can be used to classify or retrieve relevant documents. Furthermore, a weighting mask $\mathbf{r} \in \mathbb{R}^{N_K}$ is used (similar to the weighing schemes used in, e.g., the tf-idf model) to alter the importance of each codeword and to form the final *representation space*:

$$\mathbf{s}_i = \mathbf{r} \odot \mathbf{z}_i \in \mathbb{R}^{N_K}. \tag{4}$$

The weighting mask is initialized to 1, i.e., $\mathbf{z}_{ij} = \mathbf{1}$ and it is learned during the retrieval optimization of the BoEW model.

3.2. Spherical Entropy

To calculate the entropy of a representation a set of annotated documents is used, where the i-th document is annotated with a label $l_i \in \{1, ..., N_C\}$ and N_C is the number of different annotations (labels). Intuitively, the entropy in the representation space is minimized when the document vectors are gathered in pure clusters, i.e., each cluster contains documents annotated with the same label.

In order to measure the entropy the vectors $\mathbf{s}_i$ are clustered into N_T clusters. The centroid of the k-th cluster is denoted by $\mathbf{c}_k$ ($k = 1...N_T$). Note that these centroids are different from the codewords $\mathbf{v}_k$ that were used in the previous Subsection for extracting the representation of a document. Each of these centroids lies in the representation space, instead of the feature space, and can be considered as a *representative* query that expresses a specific information need. According to the cluster hypothesis, low-entropy clusters, i.e., clusters that contain mostly vectors from documents of the same class, are preferable for retrieval tasks to high-entropy clusters, i.e., clusters that contain vectors from documents that belong to several different classes.

A smooth cluster membership vector $\mathbf{q}_i \in \mathbb{R}^{N_T}$ is defined to measure the similarity between a centroid and a histogram vector:

$$q_{ik} = exp(-\frac{dist(\mathbf{s}_i, \mathbf{c}_k)}{m}), \tag{5}$$

where $dist(\mathbf{s}_i, \mathbf{c}_k)$ is a distance metric used for the retrieval process. The parameter m controls the fuzziness of the assignment process: for $m \to 0$ each histogram is assigned to its nearest cluster (hard entropy), while larger values allow for more fuzzy membership (soft entropy). In this work soft entropy ($m > 0$) is used, since it is not computationally tractable to directly optimize the hard entropy. Then the corresponding smooth l^1 normalized membership vector $\mathbf{w}_i$ is defined as:

$$\mathbf{w}_i = \frac{\mathbf{q}_i}{||\mathbf{q}_i||_1} \in \mathbb{R}^{N_T}. \tag{6}$$

Note the similarity between Eq. (1), which refers to the quantization of word embedding vectors using the codewords, and the previously defined Eq. (5), which refers to the assignment of the extracted histogram vectors to the clusters that are used for calculating the entropy. Even though in both equations a soft quantization approach is used, they refer to different spaces, i.e., Eq. (5) is used in the original feature space, while Eq. (1) is used in the vector space learned using the BoEW model. The same is also true for Eq. (2) and Eq. (6).

If the Euclidean distance is used in Eq. (5):

$$dist(\mathbf{s}_i, \mathbf{c}_k) = ||\mathbf{s}_i - \mathbf{c}_k||_2, \tag{7}$$

the classical entropy is derived [13]. A cosine based distance metric can be defined similarly (since the cosine similarity ranges from -1 to 1):

$$dist(\mathbf{s}_i, \mathbf{c}_k) = 1 - \cos\theta = 1 - \frac{\mathbf{s}_i^T \mathbf{c}_k}{||\mathbf{s}_i||_2 ||\mathbf{c}_k||_2}, \tag{8}$$

where θ is the angle between the $\mathbf{s}_i$ and the $\mathbf{c}_k$ vectors. That leads to the definition of the spherical entropy that optimizes the representation for retrieval using the cosine similarity. In Section 4, it is demonstrated that using the proposed spherical entropy for the optimization process leads to significant improvements of the retrieval precision. Then, the entropy of the k-th cluster can be calculated as:

$$E_k = -\sum_{j=1}^{N_C} p_{jk} \log p_{jk} \tag{9}$$

where p_{jk} is the probability that a document of the k-th cluster is annotated with the j-th label. This probability is estimated as $p_{jk} = h_{jk}/n_k$, where

$$n_k = \sum_{i=1}^{N} w_{ik}, \tag{10}$$

and

$$h_{jk} = \sum_{i=1}^{N} w_{ik} \pi_{ij}, \tag{11}$$

where π_{ij} is 1 if the i-th document belongs to class j and 0 otherwise.

The aim of the proposed method is to learn an optimized BoEW representation that minimizes the total entropy of a clustering configuration, which is defined as:

$$E = \sum_{k=1}^{N_T} r_k E_k, \tag{12}$$

where $r_k = n_k/N$ is the proportion of documents in cluster k. By substituting Eq. (9) into Eq. (12), the final objective function is obtained:

$$E = -\frac{1}{N} \sum_{k=1}^{N_T} \sum_{j=1}^{N_C} h_{jk} \log p_{jk}. \tag{13}$$

3.3. Retrieval Optimization of the BoEW

In this Subsection an algorithm for optimizing the BoEW representation, i.e., the word embedding vectors, the codebook, the scaling factor and the weighting mask, towards retrieval using the proposed spherical entropy objective is presented. Since both the model and the objective function are continuous and differentiable the simple gradient descent technique can be used for the optimization process:

$$\Delta(\mathbf{V}, \mathbf{X}_E, \mathbf{r}, \sigma) = -(\eta_V \frac{\partial E}{\partial \mathbf{V}}, \eta_{X_E} \frac{\partial E}{\partial \mathbf{X}_E}, \eta_r \frac{\partial E}{\partial \mathbf{r}}, \eta_\sigma \frac{\partial E}{\partial \sigma}) \tag{14}$$

where $\mathbf{X}_E$ is the matrix that contains all the word embedding vectors of $\mathcal{E}$ and η_V, η_{X_E}, η_r and η_σ are the learning rates for the corresponding parameters. In this work, instead of using the gradient descent algorithm, the adaptive moment estimation algorithm, also known as Adam [47], is used, since it provides faster and more stable convergence. For both the gradient descent and the Adam algorithm the partial derivatives of the objective function with the respect to each parameter must be calculated. The calculation of these derivatives is provided in Appendix A. The optimized representation is called Retrieval Optimized BoEW (RO-BoEW) to be distinguished from the baseline BoEW representation.

The proposed learning algorithm for the BoEW representation is described in Algorithm 1. First, the dictionary is constructed (line 2, Algorithm 1) and the word embedding is initialized (line 3, Algorithm 1). This can be done either using a randomly initialized embedding or using a pretrained embedding, e.g., the GloVe vectors [36]. In this work, the 300-dimensional GloVe vectors trained on the Wikipedia 2014 + Gigaword 5 dataset are used. If a word in the dictionary does not exist in the GloVe corpus, then a random 300-d vector is generated (using a Gaussian distribution with $\mu = 1$ and $\sigma = 1$). This does not negatively impact the learned representation (since the word embeddings are finetuned during the learning process) and allows for learning embedding vectors for terms that were not initially in the corpus. Then, an initial dictionary is learned by clustering the word embedding vectors (line 4, Algorithm 1) and the weighting mask is initialized to 1 (line 5, Algorithm 1). After the model is initialized, the training documents are encoded (line 6, Algorithm 1).

In order to calculate the entropy, the centers in the representation space ($\mathbf{c}_k$) must be selected and appended to the list of cluster centers C (lines 7 and 10). It is reasonable to assume that each class should be represented by at least one cluster. Instead of using k-means to select the centers, the mean vector of each class is used (line 9, Algorithm 1) and a total of N_C centers are utilized (lines 7-10, Algorithm 1). If the distribution of some classes is multimodal, then it is reasonable to select more than one center for each class, e.g., by running k-means over the vectors of this class. However, in the conducted experiments it was established that selecting one center per class yields similar results in terms of retrieval precision, while reducing the optimization time. Therefore, one center for each class is selected (a unimodal distribution is assumed).

As stated before, the Adam algorithm is used to optimize the RO-BoEW using the supplied training set $\mathcal{X}$ (lines 11-13, Algorithm 1). Note that the entropy is calculated in batches of N_B training documents. The training dataset is shuffled before each iteration to ensure that different batches are used. Also, the following learning rates were used for all the conducted experiments: $\eta_V = \eta_{X_E} = \eta_r = 0.01$ and $\eta_\sigma = 0.001$, since it was established that they provide stable and smooth convergence. The learning rate for the scaling parameter σ is lower, since the gradients from all the codewords are accumulated. The RO-BoF optimization procedure runs for 10 iterations for all the conducted experiments, except otherwise stated.

In order to better demonstrate the concept of the entropy optimization of the BoEW representation a toy example is provided. To this end, a toy dataset consisting of 500 documents of the 20 Newsgroup dataset is used [48, 49]. The sampled documents belongs to the following three categories: *politics*, *religion* and *politics.guns*. For the conducted experiments 8 codewords were used and the resulting vectors were projected into a 2-d space using the PCA technique [50]. In Figure 1 the initial BoEW representation is compared to the RO-BoEW. In Figure 1b only the codebook and the scaling factor are optimized, in Figure 1c the embedding is also optimized, while in Figure 1d all the parameters of the model are optimized (as described in Algorithm 1). It is evident that the optimization of the embedding (1c) is crucial for the performance of the RO-BoEW model, since the three classes are gathered in pure clusters. Also optimizing the weighting mask further improves the separability between the documents that are annotated with different labels. The greater learning capability of the proposed RO-BoEW formulation is also confirmed in Figure 2, where the learning curves for the three optimization variations are plotted. Optimizing all the parameters of the RO-BoEW allows for faster and smoother training. Finally, in Figure 3 the representation space is plotted during the optimization process (all the parameters are optimized), where the gradual transformation of the representation space to better discriminate the documents that serve different information needs is illustrated.

Algorithm 1 RO-BoEW Learning Algorithm

Input: A set $\mathcal{X} = \{\mathbf{x}_1, ..., \mathbf{x}_N\}$ of N training document and their class labels $\mathcal{L} = \{l_1, ..., l_N\}$

Parameters: N_K, N_{iters}, N_B, m and σ

Output: The optimized parameters of the RO-BoEW model

procedure RO-BoEW Learning

    Scan the documents and create the dictionary $\mathcal{D}$

    Initialize the word embedding $\mathcal{E}$ using $\mathcal{D}$

    Initialize V by running k-means on $\mathcal{E}$

    Initialize the elements of vector $\mathbf{r}$ to 1

    $S \leftarrow$ ENCODE(X)

    $C \leftarrow []$

    for $i \leftarrow 1; i \leq N_C; i++$ **do**

        Calculate the mean vector of the documents that belongs to class i

        Append this vector to cluster centers C

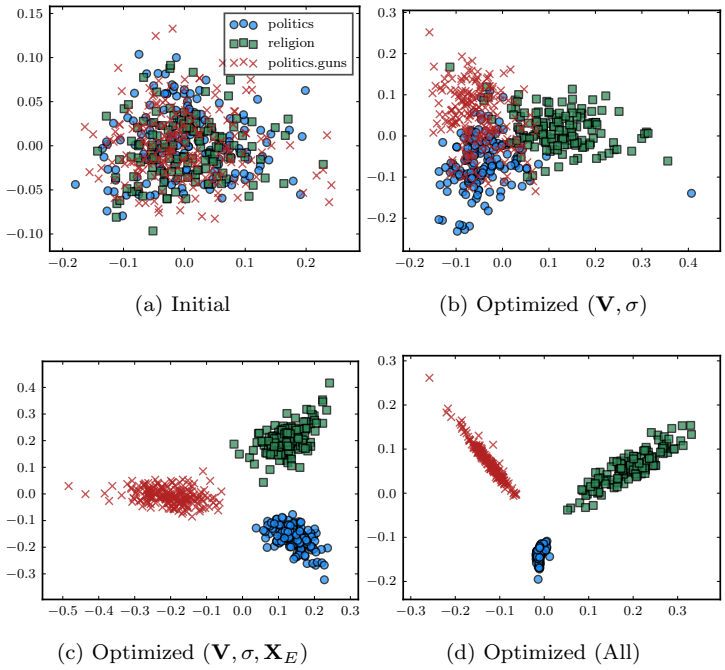
    for $i \leftarrow 1; i \leq N_{iters}; i++$ **do**

        for each batch $\in \mathcal{X}$ **do**

            Apply the Adam algorithm to update the parameters $\mathbf{V}$, $\mathbf{X}_E$, $\mathbf{r}$ and σ using the corresponding learning rates (as given in Eq. (14))

    return $\mathbf{V}$, $\mathbf{X}_E$, $\mathbf{r}$ and σ

procedure encode(X) **return** the document vectors according to Eq. (4)



(a) Initial (b) Optimized $(\mathbf{V}, \sigma)$

(c) Optimized $(\mathbf{V}, \sigma, \mathbf{X}_E)$ (d) Optimized (All)

Figure 1: Toy Example: Optimizing different parameters of the RO-BoEW model

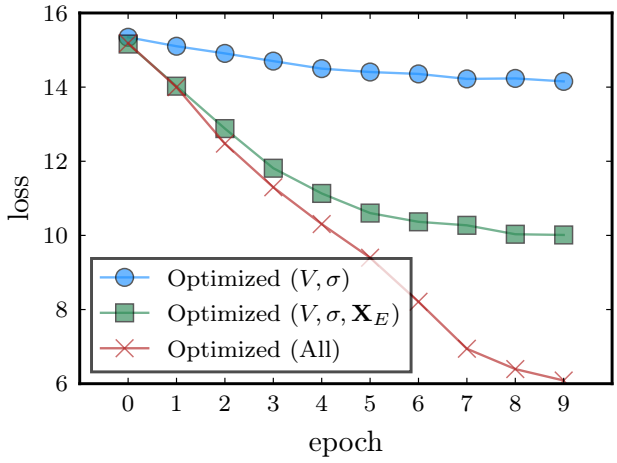


Figure 2: Toy Example: Learning curves for the RO-BoEW model

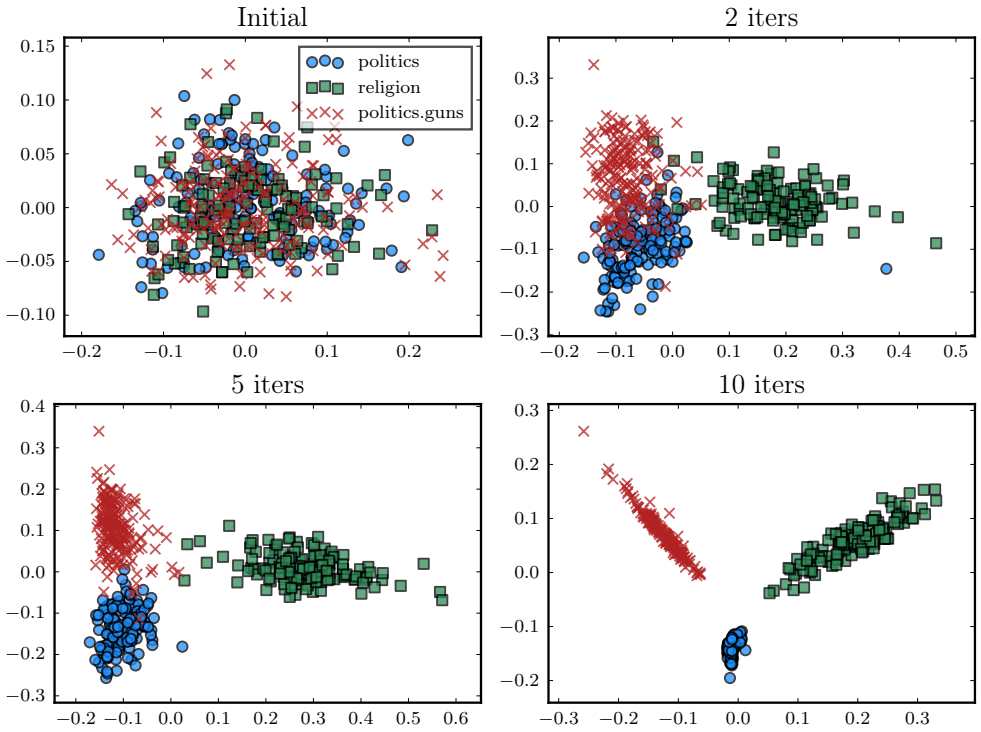


Figure 3: Toy Example: Representation space during the optimization

3.4. Relevance Feedback using the RO-BoEW

In some cases it is not possible to have large collections of annotated documents to optimize the RO-BoEW representation. In that case the RO-BoEW can be optimized online using the feedback supplied by the user. However, optimizing the whole BoEW representation, including the codebook $\mathbf{V}$, the word embedding vectors $\mathcal{E}$ and the scaling factor σ, requires re-encoding the whole database, which is impractical in most cases. The proposed RO-BoEW model provides a way to avoid this costly recalculation by optimizing only the weighting mask $\mathbf{r}$ using the feedback provided by the user. Since adjusting the weighting mask alters only the distance calculations, this method allows for fast adaptation of the representation to the needs of the users.

The relevance feedback algorithm is provided in Algorithm 2. First, the user annotates some of the retrieved documents (line 2, Algorithm 2) as either relevant or irrelevant to his information need. Then, two entropy centers are calculated (one for the relevant and one for the irrelevant documents, line 3 of Algorithm 2). Again, more centers can be used if the distribution of one class is multi-modal, which is expected to be the case for the irrelevant documents. However, in the conducted experiments it was established that using just one center for each category yields satisfactory results. Finally, the Adam algorithm is used to optimize the weighting mask (lines 4-5, Algorithm 2) and the query is repeated (line 6, Algorithm 2). The proposed relevance feedback technique can be also combined with any other relevance feedback technique

that adjusts the query vector, e.g., the Rocchio technique [1], to further improve the retrieval precision. This is also experimentally confirmed in Section 4. Finally note that the feedback provided by the users can be also stored in order to update the learned representation offline using the algorithm described before (Algorithm 1).

Algorithm 2 RO-BoEW Relevance Feedback Algorithm

Input: A set $\mathcal{X} = \{\mathbf{x}_1, ..., \mathbf{x}_N\}$ of N retrieved document and their annotations $\mathcal{L} = \{l_1, l_2\}$ (supplied by the user)

Parameters: N_{iters}, η_r, m

Output: The optimized weighting mask $\mathbf{r}$

procedure RO-BoEW Relevance Feedback Learning
  Get the annotations l_i from the user
  Calculate the two entropy centers (relevant and irrelevant documents)
  for $i \leftarrow 1; i \leq N_{iters}; i++$ **do**
    Apply the Adam algorithm to update the parameter $\mathbf{r}$ using learning rate η_r (as given in Eq. (14))
    Query the database using the updated weighting mask $\mathbf{r}$
  return the updated results

4. Experiments

The proposed method was evaluated using three text datasets and the results are presented in this Section. First, the used datasets and the evaluation metrics are briefly described. Then, the evaluation protocol and the experimental evaluation of the proposed method are provided. Note that the parameter selection procedure and the selected parameters for the evaluated methods are presented in Appendix B.

4.1. Datasets

Three datasets were used for the evaluation of the proposed method: the 20 Newsgroups dataset, Reuters newswire dataset and WebKB dataset. The 20 Newsgroups dataset [48, 49], contains 18,846 documents that belong to 20 different newsgroup categories. The train split (11,314 documents) and test split (7,532 documents) are predefined. The Reuters dataset [49], contains articles from the Reuters newswire. In this work, the R8 split of the Reuters dataset is used which contains 5,485 train documents and 2,189 test documents that belong to 8 different classes. The WebKB dataset contains webpages from various computer science departments that were classified into seven different classes: *student, faculty, stuff,* etc. The preprocessed WebKB dataset, provided by [49], was used in this work. This preprocessed split contains 4 different classes: *project, course, faculty, student.* There is a total of 2,803 train documents and 1,396 test documents.

4.2. Evaluation Procedures

Two different evaluation procedures were used in this work. For the first one, which is called *Retrieval Evaluation*, the train split was used to build the database and the test split was used to query the database and evaluate the retrieval precision. For optimizing the RO-BoEW all the documents in the database were used (using the procedure described in Section 3). Two different setups were utilized for the retrieval evaluation [13]: a) regular (in-domain) evaluation, where the representation was optimized using documents that belong to the same classes as the queries and b) *out-of-domain* evaluation, where the representation was optimized using documents that were annotated with different labels than the query documents.

However using large collections of annotated documents to optimize the representation is not always possible, since acquiring document annotations is a costly process. Furthermore, the information need of the user might not always match the existing annotations. To this end, the proposed method is also evaluated using a *Relevance Feedback* evaluation procedure, where the learned representation is adapted to the information need of the user. Again, the train split is used to build the database and 100 randomly selected queries are used to query the database and measure the retrieval precision. Then, the top-30 results are returned and (at most) 5 relevant and 5 irrelevant documents are used to provide the feedback and re-evaluate the performance of the technique.

4.3. Evaluation Metrics

Throughout this paper, three evaluation metrics are used: precision (also abbreviated as 'prec.'), recall and mean average precision (mAP). The precision is defined as:

$$Pr(q,k) = \frac{rel(q,k)}{k}, \tag{15}$$

where k is the number of retrieved objects and $rel(q,k)$ is the number of retrieved objects that belong to the same class as the query q. The recall is similarly defined as:

$$Rec(q,k) = \frac{rel(q,k)}{nclass(q)}, \tag{16}$$

where $nclass(q)$ is the total number of database objects that belong to the same class as q. The interpolated precision, $Pr_{interp}(q,k) = \max_{k',\ k' \geq k} Pr(q,k')$, is used instead of the raw precision since it reduces the precision-recall curve fluctuation [1]. The average precision (AP) for a given query is computed at eleven equally spaced recall points (0, 0.1, ..., 0.9, 1) and the mean average precision as the mean of APs for all queries. Two types of curves are plotted: the precision-recall curve and the precision-scope curve. The scope refers to the number of objects returned to the user and the precision-scope curve allow us to evaluate the precision at lower recall levels.

4.4. Retrieval Evaluation

In this Subsection the proposed method is evaluated using the retrieval evaluation setup and it is compared to other five baseline and state-of-the-art text representation methods:

1. **tf-idf**: The well-known tf-idf scheme was used to extract a representation from each document after pruning dictionary terms with frequency less than 5 [1].
2. **tf-idf (stop)**: Same as the "tf-idf' method, but after removing the stop-words (the scikit-learn library was used for the term extraction and stop-word removal [51]).
3. **LSI**: The Latent Semantic Indexing (LSI) [33], also known as Latent Semantic Analysis, was used to extract a compact representation (200 topics) (the scikit-learn library was used to perform SVD on the "tf-idf (stop)" representation).
4. **Mean WE**: The mean word embedding vector of all the words that appear in a document was used. For all the conducted experiments, 300-dimensional GloVe vectors trained on the Wikipedia 2014 + Gigaword 5 dataset were used [36].
5. **Paragraph Vector** (also abbreviated as **PV**): The state-of-the-art paragraph vector method was also evaluated [11]. The gensim library was used to compute the paragraph embedding vectors using the *distributed bag-of-word* (PV-DBOW) method and window size equal to 8 [52]. Instead of using random initialization for the word embedding vectors (which is the standard gensim's approach), they were initialized using the corresponding GloVe vectors and simultaneously trained (skip-gram word training) during the optimization of the document embeddings. This approach led to significantly better results than the other PV variants that were evaluated.

First, the evaluated methods were compared using the 20 Newsgroups dataset. The results are shown in Table 1. Two variants of the proposed BoEW representation were evaluated: the unsupervised BoEW representation (abbreviated as "*BoEW*") and the optimized BoEW representation (abbreviated as "*RO-BoEW*"). Finally, two different retrieval distance metrics were used, the Euclidean distance and the cosine similarity, as well as two different optimization objectives, the regular entropy and the proposed spherical entropy. Several conclusions can be drawn from the results shown in Table 1. First, removing the stop-words before compiling the td-idf representation leads to significant improvements in the retrieval precision. Using LSI further improves the mAP from 24.48% ("tf-idf (stop)") to 31.89%. The Mean WE achieves lower retrieval precision from both the tf-idf and LSI representations. The same is also true for the (unoptimized) BoEW representation. Finally, using the Paragraph Vector representation further improves the performance over all the other baseline methods to 32.09%. Every evaluated representation leads to better retrieval precision when combined with the cosine similarity metric.

Even though the unoptimized BoEW representation performs worse than most of the other baseline and state-of-the-art representations, the proposed RO-BoEW technique significantly outperforms all the other

Table 1: 20 Newsgroups: Retrieval Evaluation Results

Method	**Optim. Obj.**	**Retr. Distance**	**Repr. Length**	**mAP**	**top-20 prec.**	**top-50 prec.**
tf-idf	-	Cosine	21698	17.45	49.29	38.28
tf-idf (stop)	-	Cosine	21393	24.48	59.34	50.66
LSI	-	Euclidean	200	16.81	46.54	38.10
LSI	-	Cosine	200	31.89	62.80	56.76
Mean WE	-	Euclidean	300	19.32	46.09	39.00
Mean WE	-	Cosine	300	20.66	47.95	40.97
Paragraph Vector	-	Euclidean	300	28.11	60.13	53.83
Paragraph Vector	-	Cosine	300	32.09	62.01	56.14
BoEW	-	Euclidean	64	16.91	39.39	33.57
RO-BoEW	Entropy	Euclidean	64	40.70	53.93	51.28
BoEW	-	Cosine	64	17.20	39.69	33.92
RO-BoEW	Entropy	Cosine	64	41.52	55.29	52.57
RO-BoEW	Spherical Ent.	Cosine	64	**51.79**	**63.16**	**60.93**

methods (the mAP increases by more than 61.39% over the next best performing method). This is partially due to using the proposed spherical entropy, since the spherical entropy optimization increases the mAP by more than 20% over using the regular entropy. This is also true for the top$-k$ precision. Note that even though the RO-BoEW outperforms all the other evaluated techniques it uses the smallest representation (only 64 dimensions).

The precision-recall and the precision-scope curves for the best performing methods (LSI, Paragraph Vector (PV), BoEW and RO-BoEW) are shown in Figures 4a and 5a respectively. The cosine similarity is used as the retrieval metric for all the evaluated methods. Again, the RO-BoEW outperforms all the other evaluated methods. The higher precision of the LSI and PV for the first few top-k results can be attributed mostly to regularized nature of the entropy optimization objective [13], that increases the precision for lower recall levels, instead of learning a highly-discriminative representation that can possibly ovefit the data. Nonetheless, the RO-BoEW quickly matches and outperforms all the other evaluated methods.

To evaluate the performance of the learned representation for out-of-domain retrieval tasks another set of experiments was conducted using the 20 Newsgroups dataset. Five labels (*comp.graphics, comp.sys.ibm.-pc.hardware, rec.autos, talk.politics.guns* and *talk.religion.misc*) were selected and any document annotated with them was excluded from the training set. The rest of the training documents (annotated with the following labels: *alt.atheism, comp.os.ms-windows.misc, comp.sys.mac.hardware, comp.windows.x, misc.forsale, rec.motorcycles, rec.sport.baseball, rec.sport.hockey, sci.crypt, sci.electronics, sci.med, sci.space, soc.religion.christian, talk.politics.mideast,* and *talk.politics.misc*) were used for optimizing the representation (20

Table 2: 20 Newsgroups: Out-of-domain Retrieval Evaluation

Method	**Optim. Obj.**	**Retr. Distance**	**mAP**
BoEW	-	Euclidean	13.83
RO-BoEW	Entropy	Euclidean	**22.71**
BoEW	-	cosine	14.02
RO-BoEW	Sph. Entropy	cosine	**23.07**

optimization iterations were used). The test documents that were annotated with the first set of the five labels were used to query the database. This setup evaluates the ability of the learned representation to generalize on relevant out-of-domain data, i.e., data that belong to classes that were not seen during the optimization process. The results are shown in Table 2. The retrieval optimization of the representation significantly improves the precision over the BoEW representation, even though it has been optimized for fulfilling different information needs.

Next, the performance of the evaluated methods was measured using the Reuters (R8) dataset. The results are shown in Table 3. Again, removing the stop-words leads to better retrieval precision for the tf-idf representation, while LSI further improves the retrieval results. For this dataset the Mean WE representation works slightly better than the tf-idf representation. However, the Paragraph Vector further improves the precision over both the Mean WE, LSI, and tf-idf methods. Optimizing the BoEW representation leads to significant precision improvements (the mAP increases from 68.67% to 87.70%) as before. Again using the spherical entropy as optimization objective improves the retrieval precision, but by a smaller amount. Similar observations can be also made from the precision-recall and the precision-scope curves that are shown in Figures 4b and 5b respectively. The RO-BoEW outpeforms all the other evaluated methods, especially for higher recall/scope levels.

Finally, the proposed method was evaluated using the WebKB dataset (Table 4). Since the WebKB dataset is already preprocessed and stemmed removing the stop-words does not increase the precision of the tf-idf method. The tf-idf scheme performs better than the MeanEW and the unoptimized BoEW representation. As before, both the Paragraph Vector and LSI leads to better retrieval precision than the MeanEW and tf-idf representations. However, the proposed RO-BoEW combined with the spherical entropy optimization objective outperforms all the other evaluated methods, even though it uses a significantly smaller representation (16 dimensions). The precision-recall and the precision-scope curves of Figures 4c and 5c confirm the previous findings.

Table 3: Reuters (R8): Retrieval Evaluation Results

Method	**Optim. Obj.**	**Retr. Distance**	**Repr. Length**	**mAP**	**top-20 prec.**	**top-50 prec.**
tf-idf	-	Cosine	5283	67.28	87.45	83.55
tf-idf (stop)	-	Cosine	5030	69.74	88.28	85.35
LSI	-	Euclidean	200	60.25	87.79	83.54
LSI	-	Cosine	200	72.36	92.13	89.40
Mean WE	-	Euclidean	300	69.95	91.50	88.33
Mean WE	-	Cosine	300	69.96	91.45	88.30
Paragraph Vector	-	Euclidean	300	69.20	92.04	89.27
Paragraph Vector	-	Cosine	300	75.80	93.37	91.15
BoEW	-	Euclidean	64	68.54	89.21	85.87
RO-BoEW	Entropy	Euclidean	64	83.97	92.72	91.70
BoEW	-	Cosine	64	68.67	89.28	85.97
RO-BoEW	Entropy	Cosine	64	86.33	**93.75**	**92.91**
RO-BoEW	Spherical Ent.	Cosine	64	**87.70**	93.31	92.63

Table 4: WebKB: Retrieval Evaluation Results

Method	**Optim. Obj.**	**Retr. Distance**	**Repr. Length**	**mAP**	**top-20 prec.**	**top-50 prec.**
tf-idf	-	Cosine	4837	47.50	68.01	62.85
tf-idf (stop)	-	Cosine	4790	47.43	67.92	62.80
LSI	-	Euclidean	200	40.78	58.65	52.11
LSI	-	Cosine	200	48.48	70.11	64.85
Mean WE	-	Euclidean	300	41.70	65.05	59.52
Mean WE	-	Cosine	300	43.27	66.55	61.39
Paragraph Vector	-	Euclidean	300	54.28	75.84	72.01
Paragraph Vector	-	Cosine	300	56.98	77.14	73.06
BoEW	-	Euclidean	16	37.63	50.46	46.51
RO-BoEW	Entropy	Euclidean	16	70.68	80.07	78.78
BoEW	-	Cosine	16	37.62	50.49	46.49
RO-BoEW	Entropy	Cosine	16	70.56	80.05	78.78
RO-BoEW	Spherical Ent.	Cosine	16	**71.54**	**81.02**	**79.39**

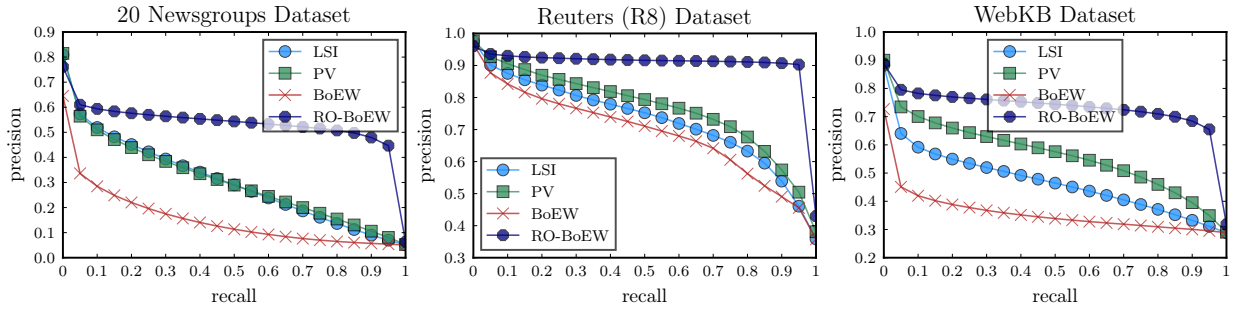


Figure 4: Retrieval Evaluation: Precision-Recall Curves

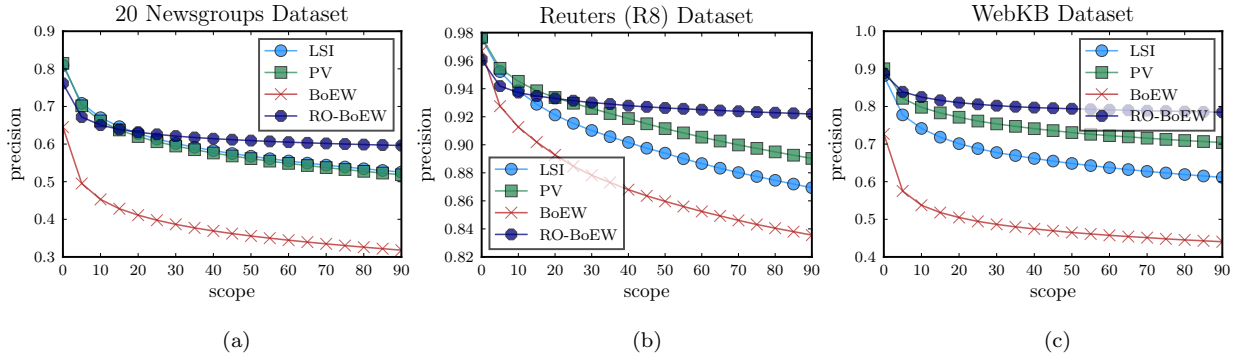


Figure 5: Retrieval Evaluation: Precision-Scope Curves

4.5. Relevance Feedback Evaluation

As it was already mentioned, the proposed method RO-BoF method is capable of optimizing only the weighting mask using the feedback provided by the user (using the algorithm that was described in Section 3). That allows for quickly updating the representation of the objects (since the weighting mask only alters the distance calculations) without re-encoding the whole database. The proposed relevance feedback technique was evaluated using the baseline BoEW representation to demonstrate its ability to improve the retrieval results even when the rest of the BoEW representation is not optimized towards the specific information needs.

The results of the relevance feedback evaluation are provided in Table 5. The following four techniques are compared:

1. Initial (also abbreviated as 'Init'), where no relevance feedback is used,
2. Rocchio (also abbreviated as 'Roc.'), where the standard Rocchio technique is used [1],
3. Proposed (also abbreviated as 'Prop'), where the proposed method is used,
4. Proposed + Rocchio (also abbreviated as 'Roc.+Prop'), where the proposed method is combined with the Rocchio technique.

The default parameters for the Rocchio technique were used: the weight of the query was set to $a = 1$, the weight of the relevant documents was set to $b = 0.8$ and no weight was used for the irrelevant documents ($c = 0$). The Rocchio technique can be also combined with the proposed approach by simply using the weighted vectors during the Rocchio's calculations. Note that at most 5 positive and 5 negative samples were selected and used during the relevance feedback procedure. These samples were selected according to their similarity to the query, i.e., the first 5 most similar to they query positive/negative samples were selected. This scenario corresponds to the typical behavior of a user, who usually provides the feedback examining only the first few results.

The proposed relevance feedback technique greatly improves both the mAP and the top$-k$ precision for all the used datasets (the mAP increases by over 30%). On the other hand, using only the Rocchio method only slightly improves (or even harms, in the case of the Reuters dataset) the mAP. Also, combining the proposed method with the Rocchio always further improves the retrieval precision. This is also confirmed by the precision-recall and the precision-recall scope of Figures 6 and 7.

5. Conclusions

In this paper, the similarity between the process of extracting a set of word embedding vectors from a text document and the feature extraction step of the BoF model was exploited to develop a novel text representation model, inspired from the traditional BoW and BoF models. The proposed Bag-of-Embedded

Table 5: Relevance Feedback Evaluation Results

Dataset	Method	mAP	top-10 prec.	top-20 prec.	top-50 prec.
20 Newsgroups	Initial	9.55	14.35	13.60	12.77
20 Newsgroups	Rocchio	9.73	16.43	15.03	13.13
20 Newsgroups	Proposed	14.56	34.75	30.98	26.91
20 Newsgroups	Proposed + Rocchio	**17.40**	**43.28**	**37.06**	**30.17**
Reuters (R8)	Initial	65.17	85.01	83.50	81.82
Reuters (R8)	Rocchio	63.59	86.93	85.45	83.20
Reuters (R8)	Proposed	85.42	96.12	95.59	94.53
Reuters (R8)	Proposed + Rocchio	**85.67**	**97.09**	**96.15**	**94.74**
WebKB	Initial	35.11	40.34	39.05	37.46
WebKB	Rocchio	35.34	40.80	39.26	38.00
WebKB	Proposed	46.16	67.32	64.89	61.04
WebKB	Proposed + Rocchio	**47.08**	**69.12**	**65.68**	**61.79**

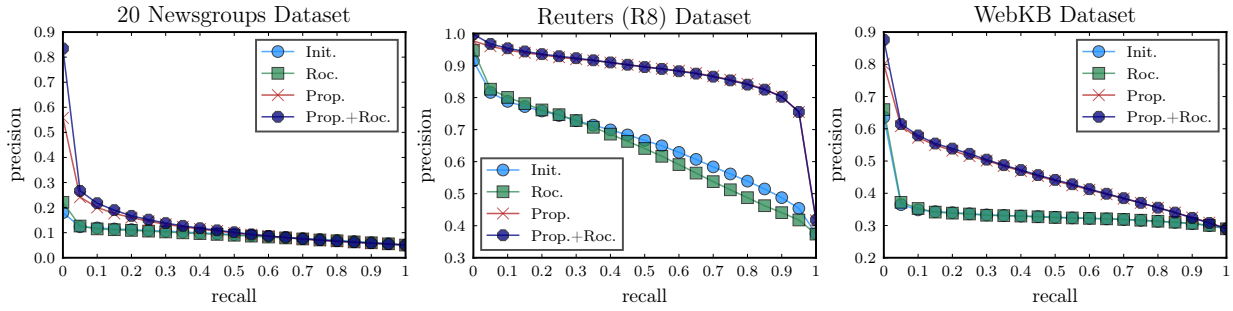


Figure 6: Relevance Feedback Evaluation: Precision-Recall Curves

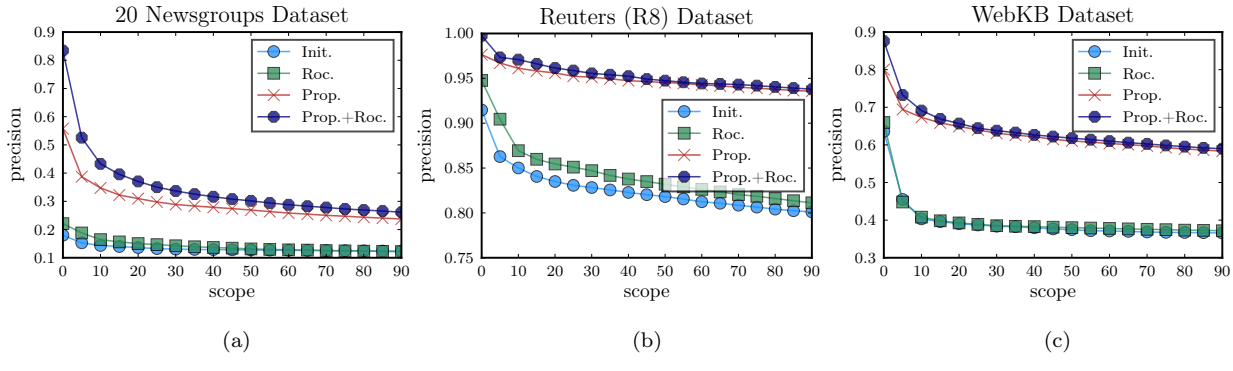


Figure 7: Relevance Feedback Evaluation: Precision-Scope Curves

Words (BoEW) model extends the regular BoF model by a) incorporating a weighting mask that allows for altering the importance of each learned codeword and b) by optimizing the model end-to-end (from the used word embeddings to the weighting mask). When enough training data are available, the BoEW model can be optimized using the proposed spherical entropy objective, which optimizes the learned representation for retrieval using the cosine similarity. When such kind of annotations are not available, the proposed approach can use relevance feedback to quickly fine-tune the representation and re-query the database to fulfill the information needs of the user. Finally, the ability of the proposed method to improve the retrieval performance, both in terms of retrieval precision and speed, was demonstrated using three text datasets from a diverse range of domains.

Appendix A - RO-BoEW Derivatives

The derivative of E with respect to $\mathbf{v}_m$ can be calculated as the product of two other partial derivatives:

$$\frac{\partial E}{\partial \mathbf{v}_m} = \sum_{l=1}^{N} \sum_{\kappa=1}^{N_K} \frac{\partial E}{\partial s_{l\kappa}} \frac{\partial s_{l\kappa}}{\partial \mathbf{v}_m}. \tag{17}$$

In order to reduce the entropy in the representation space the vectors $\mathbf{s}_l$ that, in turn, depend on the codebook $\mathbf{V}$, must be shifted. The partial derivative $\frac{\partial E}{\partial \mathbf{s}_l}$ provides the direction in which the vector $\mathbf{s}_l$ must be moved. Since each codeword $\mathbf{v}_m$ lies in the feature space, the derivative $\frac{\partial s_{l\kappa}}{\partial \mathbf{v}_m}$ projects the previous direction into the codebook. Therefore, the representation space derivative is computed as:

$$\frac{\partial E}{\partial s_{l\kappa}} = -\frac{1}{N} \sum_{k=1}^{N_T} \sum_{j=1}^{N_C} \log p_{jk} \pi_{lj} \frac{\partial w_{lk}}{\partial s_{l\kappa}}, \tag{18}$$

where:

$$\frac{\partial w_{lk}}{\partial s_{l\kappa}} = -\frac{w_{lk}}{m} (dist(\mathbf{s}_l, \mathbf{c}_k) \frac{dist(\mathbf{s}_l, \mathbf{c}_k)}{\partial s_{l\kappa}} - \sum_{k'=1}^{N_T} w_{lk'} dist(\mathbf{s}_l, \mathbf{c}_{k'}) \frac{dist(\mathbf{s}_l, \mathbf{c}_{k'})}{\partial s_{l\kappa}}). \tag{19}$$

If the regular (Euclidean) entropy is used, the derivative $\frac{dist(\mathbf{s}_l, \mathbf{c}_{k'})}{\partial s_{l\kappa}}$ is calculated as:

$$\frac{dist(\mathbf{s}_l, \mathbf{c}_k)}{\partial s_{l\kappa}} = \frac{s_{l\kappa} - c_{k\kappa}}{||\mathbf{s}_l - \mathbf{c}_k||_2^2}, \tag{20}$$

while for the spherical entropy (assuming, without loss of generality, that $s_{li} > 0$) as:

$$\frac{dist(\mathbf{s}_l, \mathbf{c}_k)}{\partial s_{l\kappa}} = -\sum_{i=1}^{N_K} \frac{\delta_{i\kappa} c_{k\kappa}}{||\mathbf{s}_l||_1 ||\mathbf{c}_k||_1} - \frac{s_{li} c_{ki}}{||\mathbf{s}_l||_1^2 ||\mathbf{c}_k||_1}, \tag{21}$$

where δ_{ab} is 1 if $a = b$ and 0 otherwise.

The codebook projection derivative (last term of (17)) is derived as:

$$\frac{\partial s_{l\kappa}}{\partial \mathbf{v}_m} = -\frac{r_\kappa}{\sigma^2 N_l} \sum_{j=1}^{N_l} u_{ljm} (\delta_{\kappa m} - u_{lj\kappa}) \frac{\mathbf{v}_m - \mathbf{x}_{lj}}{||\mathbf{v}_m - \mathbf{x}_{lj}||_2}. \tag{22}$$

Then, the derivative of E with respect to each $\mathbf{x}_{ij}$ is similarly derived as:

$$\frac{\partial E}{\partial \mathbf{x}_{ij}} = \sum_{l=1}^{N} \sum_{\kappa=1}^{N_K} \frac{\partial E}{\partial s_{l\kappa}} \frac{\partial s_{l\kappa}}{\partial \mathbf{x}_{ij}}. \tag{23}$$

The word embedding projection derivative, i.e., the last term of (23), is calculated as:

$$\frac{\partial s_{l\kappa}}{\partial \mathbf{x}_{ij}} = \frac{1}{\sigma^2 N_l} \sum_{j=1}^{N_l} \sum_{t=1}^{N_K} u_{ljt}(\delta_{\kappa t} - u_{lj\kappa}) \frac{\mathbf{v}_t - \mathbf{x}_{lj}}{||\mathbf{v}_t - \mathbf{x}_{lj}||_2}. \tag{24}$$

Since different feature vectors $\mathbf{x}_{ij}$ might correspond to the same word vector $\mathbf{x}_i$, the gradients are accumulated for the same words, i.e., the vectors $\mathbf{x}_{ij}$ are tied if they correspond to the same word.

The derivatives of E with respect to the weighting mask r, i.e., $\frac{\partial E}{\partial \mathbf{r}}$, and the derivative of E with respect to the scaling factor σ, i.e, $\frac{\partial E}{\partial \sigma}$, can be similarly derived by observing that:

$$\frac{\partial s_{lk}}{\partial r_k} = z_{lk}, \tag{25}$$

and

$$\frac{\partial d_{ljk}}{\partial \sigma} = 2\frac{d_{ljk}}{\sigma^3} ||\mathbf{v}_k - \mathbf{x}_{lj}||_2. \tag{26}$$

Finally, note that the representation space derivative does not exist when a representation space vector and an entropy centroid coincide (when the Euclidean distance is used for the entropy). The same holds for the codebook and the word embedding derivatives when a codebook center and a feature vector also coincide. When that happens, the corresponding derivatives are set to 0.

Appendix B - Parameter Selection

In this Appendix the effect of several parameters on the quality of the learned (RO)-BoEW representation is examined. For all the conducted experiments 1000 randomly chosen training documents were used to build the database and another 1000 randomly chosen training documents were used as validation set (to query the database). First, the effect of the scaling parameter σ on the learned representation is evaluated in Figure 8. The abbrevation '(e)' is used to denote that the Euclidean distance was used for the retrieval process, while the abbreviation '(c)' is used for the cosine similarity. For the BoEW representation the effect of σ on the mAP is minimal. On the other hand, a good initial value of σ is crucial for the optimization of the RO-BoEW representation. In most cases a value around $\sigma = 0.4 - 0.7$ provides the best results. The importance of the supervised optimization of the representation is evident, since the RO-BoEW greatly outperforms the BoEW in any case.

Next, the effect of the entropy fuzziness parameter m is evaluated in Figure 9. Large values of m make every document to belong to every cluster, while very small values leads to vanishing gradients and numerical stability problems. Therefore it is important to select an appropriate value to allow for smooth optimization

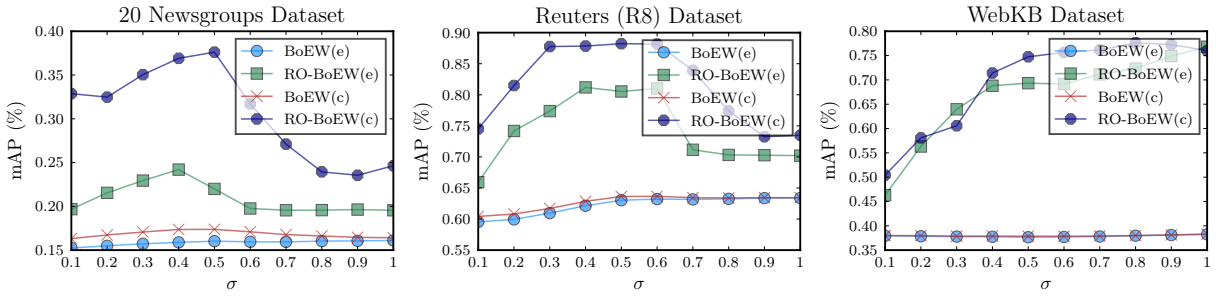


Figure 8: Effect of the scaling parameter σ on the learned representation

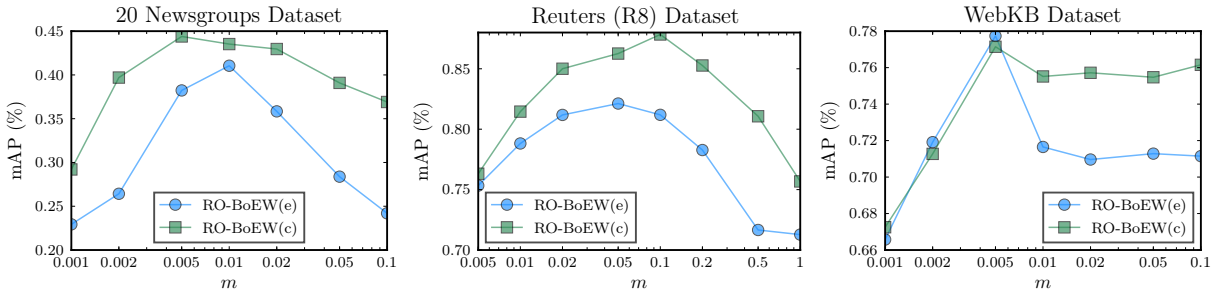


Figure 9: Effect of the scaling parameter m on the learned representation

of the RO-BoEW representation. As shown in Figure 9 the optimal value depends of the used dataset ($0.005 \leq m \leq 0.1$).

Since the entropy is computed in mini-batches, the batch size can also effect the quality of the learned representation as shown in Figure 10. Batch sizes around 50-100 lead to the best results in terms of mAP. Note that using large batch size leads to slower convergence and that the optimization runs for a fixed number of epochs (10 epochs over the all training data).

The effect of the number of codewords is examined in Figure 11. Using more codewords only slightly improves the retrieval precision for the BoEW. On the other hand, the precision is constantly increasing with the number of codewords (except for the WebKB dataset where it is kept constant) when the optimized

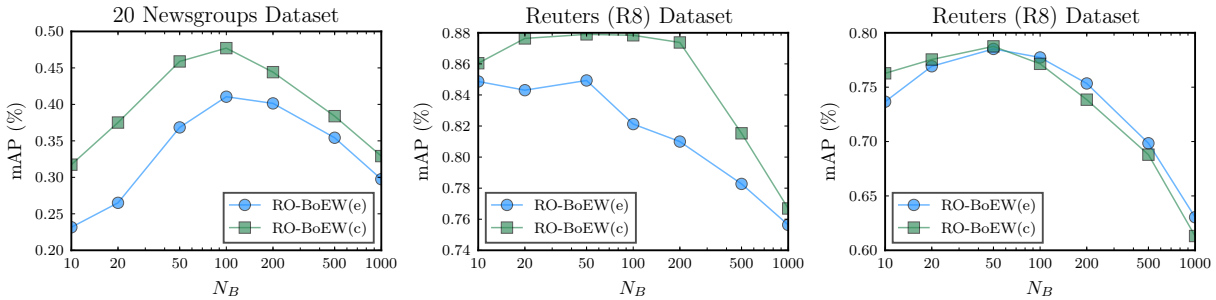


Figure 10: Effect of the scaling parameter N_B on the learned representation

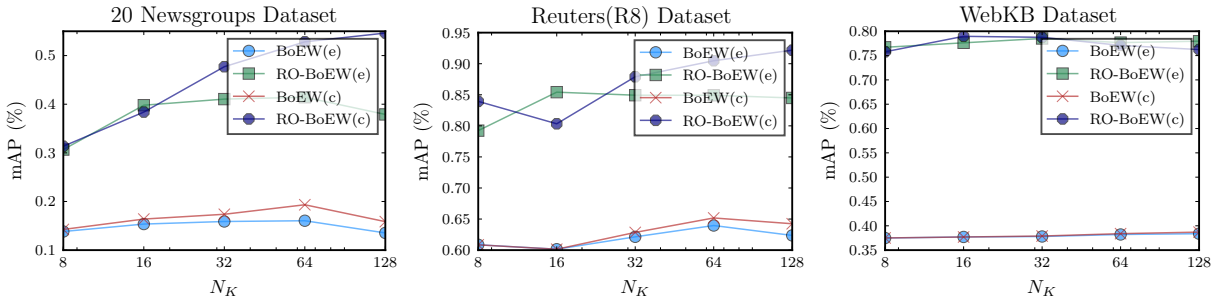


Figure 11: Effect of the number of codewords N_K on the learned representation

Table 6: Selected Parameters

Dataset	**Distance**	σ	m	N_B	N_K
20 Newsgroups	Euclidean	1/0.4	0.01	100	64
20 Newsgroups	Cosine	0.5/0.5	0.005	100	64
Reuters (R8)	Euclidean	1/0.4	0.05	50	64
Reuters (R8)	Cosine	1/0.4	0.1	50	64
WebKB	Euclidean	1/0.7	0.005	50	16
WebKB	Cosine	1/0.7	0.005	50	16

RO-BoEW representation is used. Also, note that using just 8 codewords with the RO-BoEW representation greatly outperforms the BoEW, even when one order of magnitude more codewords are used. That highlights the ability of the proposed technique to reduce the storage requirements and increase the retrieval speed (since both depend on the size of the used representation).

Finally, the selected parameters are summarized in Table 6. For the σ parameter two values are reported: the first one is used for the BoEW model, while the second one for the RO-BoEW model. Note that when the mAP was relatively stable for a wide range of parameters, the median value was chosen for the specific parameter. The proposed method was implemented using the Theano library [53] and for technical reasons the maximum length of any document was restricted to 1000 words. However, this restriction only affects the optimization process and it does not affect the deployment of the method.

Acknowledgments

Nikolaos Passalis was supported by the General Secretariat for Research and Technology (GSRT) and the Hellenic Foundation for Research and Innovation (HFRI) (PhD Scholarship No. 1215).

References

[1] C. D. Manning, P. Raghavan, H. Schütze, Introduction to Information Retrieval, 1st Edition, Cambridge University Press, Cambridge, 2008.

[2] R. E. Madsen, S. Sigurdsson, L. K. Hansen, J. Larsen, Pruning the vocabulary for better context recognition, in: Proceedings of the International Conference on Pattern Recognition, Vol. 2, 2004, pp. 483–488.

[3] J. H. Paik, A novel tf-idf weighting scheme for effective ranking, in: Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval, 2013, pp. 343–352.

[4] C.-H. Chen, Improved tfidf in big news retrieval: An empirical study, Pattern Recognition Letters 93 (2017) 113–122.

[5] D. M. Blei, A. Y. Ng, M. I. Jordan, Latent dirichlet allocation, The Journal of Machine Learning Research 3 (2003) 993–1022.

[6] A. Jabri, A. Joulin, L. van der Maaten, Revisiting visual question answering baselines, in: Proceedings of the European Conference on Computer Vision, Springer, 2016, pp. 727–739.

[7] S. Clinchant, F. Perronnin, Aggregating continuous word embeddings for information retrieval, in: Proceedings of the Workshop on Continuous Vector Space Models and their Compositionality, 2013, pp. 100–109.

[8] H. Zamani, W. B. Croft, Estimating embedding vectors for queries, in: Proceedings of the ACM on International Conference on the Theory of Information Retrieval, 2016, pp. 123–132.

[9] H. Amiri, H. D. III, Short text representation for detecting churn in microblogs, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2016, pp. 2566–2572.

[10] X. Chen, Z. Liu, M. Sun, A unified model for word sense representation and disambiguation, in: Proceedings of the Conference on Empirical Methods on Natural Language Processing, 2014.

[11] Q. Le, T. Mikolov, Distributed representations of sentences and documents, in: Proceedings of the International Conference on Machine Learning, 2014, pp. 1188–1196.

[12] A. M. Dai, C. Olah, Q. V. Le, Document embedding with paragraph vectors, arXiv preprint arXiv:1507.07998.

[13] N. Passalis, A. Tefas, Entropy optimized feature-based bag-of-words representation for information retrieval, IEEE Transactions on Knowledge and Data Engineering 28 (7) (2016) 1664–1677.

[14] X.-C. Lian, Z. Li, B.-L. Lu, L. Zhang, Max-margin dictionary learning for multiclass image categorization, in: Proceedings of the European Conference on Computer Vision, 2010, pp. 157–170.

[15] B.-D. Liu, Y.-X. Wang, Y.-J. Zhang, B. Shen, Learning dictionary on manifolds for image classification, Pattern Recognition 46 (7) (2013) 1879–1890.

[16] A. Iosifidis, A. Tefas, I. Pitas, Multidimensional sequence classification based on fuzzy distances and discriminant analysis, IEEE Transactions on Knowledge and Data Engineering 25 (11) (2013) 2564–2575.

[17] N. Passalis, A. Tefas, Neural bag-of-features learning, Pattern Recognition 64 (2017) 277–294.

[18] A. Shrivastava, J. K. Pillai, V. M. Patel, Multiple kernel-based dictionary learning for weakly supervised classification, Pattern Recognition 48 (8) (2015) 2667–2675.

[19] L. Zhou, Z. Zhou, D. Hu, Scene classification using a multi-resolution bag-of-features model, Pattern Recognition 46 (1) (2013) 424–433.

[20] A. Hernández-Vela, M. Á. Bautista, X. Perez-Sala, V. Ponce-López, S. Escalera, X. Baró, O. Pujol, C. Angulo, Probability-based dynamic time warping and bag-of-visual-and-depth-words for human gesture recognition in rgb-d, Pattern Recognition Letters 50 (2014) 112–121.

[21] D. G. Lowe, Object recognition from local scale-invariant features, in: Proceedings of the IEEE international conference on Computer vision, Vol. 2, 1999, pp. 1150–1157.

[22] J. Sivic, A. Zisserman, Video google: A text retrieval approach to object matching in videos, in: Proceedings of the IEEE International Conference on Computer Vision, 2003, pp. 1470–1477.

[23] S. Clinchant, F. Perronnin, Textual similarity with a bag-of-embedded-words model, in: Proceedings of the Conference on the Theory of Information Retrieval, 2013, pp. 25:117–25:120.

[24] N. Passalis, A. Tefas, Bag of embedded words learning for text retrieval, in: Proceedings of the International Conference on Pattern Recognition, 2016.

[25] S. Lloyd, Least squares quantization in pcm, IEEE Transactions on Information Theory 28 (2) (1982) 129–137.

[26] A. Iosifidis, A. Tefas, I. Pitas, Discriminant bag of words based representation for human action recognition, Pattern Recognition Letters 49 (2014) 185–192.

[27] S. Lazebnik, M. Raginsky, Supervised learning of quantizer codebooks by information loss minimization, IEEE Transactions on Pattern Analysis and Machine Intelligence 31 (7) (2009) 1294–1309.

[28] X. Wang, B. Wang, X. Bai, W. Liu, Z. Tu, Max-margin multiple-instance dictionary learning, in: Proceedings of the 30th International Conference on Machine Learning, 2013, pp. 846–854.

[29] F. Perronnin, C. Dance, G. Csurka, M. Bressan, Adapted vocabularies for generic visual categorization, in: Proceedings of the European Conference on Computer Vision, 2006, pp. 464–475.

[30] R. Fernandez-Beltran, F. Pla, Latent topics-based relevance feedback for video retrieval, Pattern Recognition 51 (2016) 72–84.

[31] S. R. Bulo, M. Rabbi, M. Pelillo, Content-based image retrieval with relevance feedback using random walks, Pattern Recognition 44 (9) (2011) 2109–2122.

[32] M. Arevalillo-Herráez, F. J. Ferri, J. Domingo, A naive relevance feedback model for content-based image retrieval using multiple similarity measures, Pattern Recognition 43 (3) (2010) 619–629.

[33] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, R. Harshman, Indexing by latent semantic analysis, Journal of the American society for Information Science 41 (6) (1990) 391.

[34] T. Hofmann, Probabilistic latent semantic analysis, in: Proceedings of the Conference on Uncertainty in Artificial Intelligence, 1999, pp. 289–296.

[35] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality, in: Proceedings of the Advances in Neural Information Processing Systems, 2013, pp. 3111–3119.

[36] J. Pennington, R. Socher, C. D. Manning, Glove: Global vectors for word representation., in: Proceedings of the Conference on Empirical Methods on Natural Language Processing, Vol. 14, 2014, pp. 1532–43.

[37] Q. Ai, L. Yang, J. Guo, W. B. Croft, Analysis of the paragraph vector model for information retrieval, in: Proceedings of the International Conference on the Theory of Information Retrieval, 2016, pp. 133–142.

[38] M. Kusner, Y. Sun, N. Kolkin, K. Weinberger, From word embeddings to document distances, in: Proceedings of the International Conference on Machine Learning, 2015, pp. 957–966.

[39] Y. Kuang, K. Åström, L. Kopp, M. Oskarsson, M. Byröd, Optimizing visual vocabularies using soft assignment entropies, in: Proceedings of the Asian Conference on Computer Vision, 2011, pp. 255–268.

[40] Y. Kuang, M. Byrod, K. Astrom, Supervised feature quantization with entropy optimization, in: IEEE International Conference on Computer Vision Workshops, 2011, pp. 1386–1393.

[41] H. Lobel, R. Vidal, D. Mery, A. Soto, Joint dictionary and classifier learning for categorization of images using a max-margin framework, in: Proceedings of the 6th Pacific-Rim Symposium, 2014, pp. 87–98.

[42] M. Jiu, C. Wolf, C. Garcia, A. Baskurt, Supervised learning and codebook optimization for bag-of-words models, Cognitive Computation 4 (4) (2012) 409–419.

[43] W. Zhang, A. Surve, X. Fern, T. Dietterich, Learning non-redundant codebooks for classifying complex objects, in: Proceedings of the 26th Annual International Conference on Machine Learning, 2009, pp. 1241–1248.

[44] F. Perronnin, Universal and adapted vocabularies for generic visual categorization, IEEE Transactions on Pattern Analysis and Machine Intelligence 30 (7) (2008) 1243–1256.

[45] B. Fulkerson, A. Vedaldi, S. Soatto, Localizing objects with smart dictionaries, in: Proceedings of the European Conference on Computer Vision, 2008, pp. 179–192.

[46] J. Winn, A. Criminisi, T. Minka, Object categorization by learned universal visual dictionary, in: Proceedings of the IEEE International Conference on Computer Vision, Vol. 2, 2005, pp. 1800–1807.

[47] D. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980.

[48] J. Rennie, The 20 Newsgroups data set, `http://qwone.com/~jason/20Newsgroups` (2008).

[49] A. Cardoso-Cachopo, A. L. Oliveira, Semi-supervised single-label text categorization using centroid-based classifiers, in: Proceedings of the Symposium on Applied Computing, 2007, pp. 844–851.

[50] I. T. Jolliffe, Principal Component Analysis, 2nd Edition, Springer Series in Statistics, Springer-Verlag, New York, 2002.

[51] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine learning in Python, Journal of Machine Learning Research 12 (2011) 2825–2830.

[52] R. Řehůřek, P. Sojka, Software Framework for Topic Modelling with Large Corpora, in: Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks, ELRA, Valletta, Malta, 2010, pp. 45–50, `http://is.muni.cz/publication/884893/en`.

[53] Theano Development Team, Theano: A Python framework for fast computation of mathematical expressions, arXiv e-prints abs/1605.02688.
URL `http://arxiv.org/abs/1605.02688`