Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2019) 27: 1028 – 1040

doi:10.3906/elk-1805-138

Selective word encoding for effective text representation

Savaş ÖZKAN[1,*], **Akın ÖZKAN**[2]

[1]Department of Electrical and Electronics Engineering, Middle East Technical University, Ankara, Turkey
[2]Department of Electrical and Electronics Engineering, Atılım University, Ankara, Turkey

Received: 20.05.2018 • **Accepted/Published Online:** 10.12.2018 • **Final Version:** 22.03.2019

Abstract: Determining the category of a text document from its semantic content is highly motivated in the literature and it has been extensively studied in various applications. Also, the compact representation of the text is a fundamental step in achieving precise results for the applications and the studies are generously concentrated to improve its performance. In particular, the studies which exploit the aggregation of word-level representations are the mainstream techniques used in the problem. In this paper, we tackle text representation to achieve high performance in different text classification tasks. Throughout the paper, three critical contributions are presented. First, to encode the word-level representations for each text, we adapt a trainable orderless aggregation algorithm to obtain a more discriminative abstract representation by transforming word vectors to the text-level representation. Second, we propose an effective term-weighting scheme to compute the relative importance of words from the context based on their conjunction with the problem in an end-to-end learning manner. Third, we present a weighted loss function to mitigate the class-imbalance problem between the categories. To evaluate the performance, we collect two distinct datasets as Turkish parliament records (i.e. written speeches of four major political parties including 30731/7683 train and test documents) and newspaper articles (i.e. daily articles of the columnists including 16000/3200 train and test documents) whose data is available on the web. From the results, the proposed method introduces significant performance improvements to the baseline techniques (i.e. VLAD and Fisher Vector) and achieves 0.823% and 0.878% true prediction accuracies for the party membership and the estimation of the category of articles respectively. The performance validates that the proposed contributions (i.e. trainable word-encoding model, trainable term-weighting scheme and weighted loss function) significantly outperform the baselines.

Key words: Text representation, orderless feature aggregation, trainable relative importance weights

1. Introduction

In the last decades, text classification and retrieval have attracted substantial interests in the literature, and various successful applications have been introduced with the success of representation learning from unstructured data [1–3]. Indeed, the techniques used in these methods frequently rely on the representations of vocabulary terms by their distribution characteristics as word vectors [1, 4]. Word encoding is a leading technique for various natural language-processing (NLP) tasks such as sentiment analysis [5], machine translation [3, 6], text similarity [7], part-of-speech tagging [8], and text summarization [9] and classification [10]. To this end, these methods accomplish substantial performance improvements by transforming individual terms, i.e. words, to low-level compact vectors in a latent space. Moreover, due to robustness to the overuse of affixes in the

*Correspondence: ozkan.savas@metu.edu.tr

vocabulary terms, these methods ease deploying effective models for agglutinative languages such as Turkish without the need of extra linguistic tools (i.e. complex parsing techniques [11]).

In essence, these methods offer two critical improvements. First, the methods are able to transform individual words to more compact representations than high dimensional sparse ones so that the sparsity of the representations is degraded on purpose and the ideal solution becomes more applicable (i.e., this assumption is feasible for classification tasks as explained in [2, 12, 13]). Second, they maintain the semantic similarity between coherent words as well as their robustness to the syntactic affixes since the distribution characteristics of words are uncovered. Inevitably, more consistent results can be obtained by these methods in the vector space [14].

Even though high performance can be achieved for individual words, it is still an active research field for NLP [15] to transform the semantic content of an entire text to a single representation. The current restriction for the problem is that the text has some inherent ambiguities so that the aggregation of word vectors highly depends on several factors such as the complexity of language models and the size of datasets (i.e. if a learning-based method is used). Hence, this makes it quite difficult to uncover the true model for the languages.

In this paper, we propose an effective text representation technique to obtain the best-performing text classification results for different tasks. In particular, we utilized a word aggregation-based method to encode word vectors for each text document with additional modifications. However, this base technique has several drawbacks which will be discussed in the following sections in detail. Briefly, word embeddings are computed in an unsupervised setup by not considering the current objectives for a classification problem. Thus, the true relations for the problem that are needed to be in the word vectors cannot be conserved. To this end, the aggregation of word vectors can be misleading for the classification problems by which the contributions of all words are regarded equally and the use of unnecessary words can adversely affect the performance. Hence, an effective aggregation method should estimate the importance of words from the text content automatically and consider this information in the encoding stage.

Throughout the manuscript, our contributions can be summarized as threefold. First, we adapt a trainable orderless encoding technique [16] to NLP domain in order to encode word vectors and describe the semantic content of texts effectively. Compared to the other techniques [13, 17], the main advantage is that it allows to partition the vector space to distinct clusters by exploiting the supervision exhibited from data. To this end, the clusters used for text encoding can be truly obtained for an ideal solution. Moreover, the classification error calculated for the tasks can be propagated to the initial layers due to a differentiable framework to compute high-level representations. Hence, the assumption of a trainable model can be extended for the initial layers. Second, we define trainable term weights for each word based on a trainable neural network model to boost the selectivity of word vectors in the encoding process. The model determines the relative importance of words from the context (i.e. from the word sequence) in an end-to-end manner and imposes this information onto words in the encoding step. Lastly, we formulate a weighted loss function to establish accurate and stable results even if imbalance data samples exist for the categories. Practically, it reduces the chances of overfitting to the categories with larger numbers of samples. To evaluate the performance, we conduct extensive experiments on various text classification tasks.

In the following sections, we will first describe the related works in the literature. Later, the proposed method for text representation will be explained in detail. Lastly, we will summarize the conducted experiments and conclude the manuscript.

Table 1. Top coherent words retrieved for the given key words according to the distance between word-embedding vectors. As observed, the retrieved words have both semantic similarity and robustness to significant syntactic differences compared to key words.

	#1	#2	#3	#4	#5	#6	#7
Key word: makale	dergide	makaleler	makaleyi	kitap	hakemli	yazı	indekslerde
Key word: yaz	kış	eylülekim	sonbahar	kasımaralık	ilkbahar	yazın	sezonu
Key word: insanlar	insanlarımız	insanların	insanları	vatandaşlar	insanlara	insanlara	onlar

2. Related works

In this section, we will summarize the literature in the context of representation learning for both words and texts.

2.1. Word-level representation

The underlying objective for learning the word-level semantic representation is to estimate compact vectors by leveraging the common distribution characteristics existed among the words. Latent semantic analysis (LSA) [18] utilizes this objective by using the cooccurrence of word–document and a full cooccurrence matrix is decomposed into constituent parts (i.e. orthogonal parts) to obtain compact word vectors. The main drawback is that the decomposition of the matrix becomes impractical when corpus entities are large; this makes it harder to reach the complexity of natural language models and practically causes the underestimation of the solutions.

GloVe method[1] [4] shares similar notions as in LSA by only exploiting word–word cooccurrence in the corpus than the word–document. Moreover, the critical contribution presented in the method is that the computation load is noticeably reduced by not decomposing the full matrix but learning it from random initializations with least-square variants [4]. Since the learning scheme overcomes the complexity of the method, it reduces the limitation of the underestimation for the word vectors.

Beyond the cooccurrence-based methods, Word2Vec (W2V) method[2] [1] learns the representation from the context by making prediction from its surroundings. More intuitively, the method predicts either the word from its surrounding words (CBOW) or multiple surroundings from the word (Skip-gram) with a neural network model. From their experiments, a critical observation is that CBOW gives extra robustness to syntactic tasks, i.e. affixes, while Skip-gram yields superior performance to conserve the semantic content of the words.

Table 1 shows several informative examples (in Turkish) that illustrate the power of these three methods in which the coherency between words are perfectly preserved. Moreover, to observe the effects of word-level representations for the text classification problem (especially in Turkish), we report results by using both W2V (i.e. CBOW and Skip-gram) and GloVe in the experiments.

2.2. Text-level representation

Although the word-level representations have the ability to partially capture the content from the short fragments of the text as explained, for the entire text, their vocabulary terms and content need to be used collectively so as to achieve semantically precise and compact representations for text classification/retrieval tasks.

[1] https://nlp.stanford.edu/projects/glove/
[2] https://code.google.com/archive/p/word2vec/

Inspired by the original W2V method, a paragraph feature (i.e. a sparse matrix) is taken into account in the prediction step with other surrounding words as if it is like another word ID. The loss function is minimized to estimate the next word truly under W2V assumption [19]. Also, [20] computes the representation from the entire text similar to the distribution characteristics used in LSA. In another study [21], both semantic and string distances features are adapted to model the text representation.

In particular, numerous studies concentrate on the encoding of word-level representations to go from word-level to text-level representations [22–24]. In [23, 24], it was shown that for short texts, term-frequency-based methods (TF-IDF) yield high performance by only incorporating word vectors without the need of extra sources/preprocessing step. This observation is quite important since the studies prove that word-level vectors can capture the semantic information sufficiently with no additional temporal reasoning. Fisher kernel is also used for effective text representation [22] with nonlinear kernel versions of the word vectors. However, as stated in [25], the high dimensionality limits its usage in various applications (i.e. with the distance-based metrics) and it lacks supervision to partition the latent space properly. Also, considering all word weights as equal can be tedious particularly for the text domain since a text can have lots of outliers (i.e. unnecessary words) that are not related with the content of the text. Ultimately, these methods transform the orderless word vectors to a compact representation at the end.

Similarly, convolution/recurrent neural networks can be utilized for encoding the semantic content from the sequence of the word vectors [26–29]. However, none of the text-level aggregation techniques achieves superiority to the others (i.e. convolution, recurrent neural networks, or encoding techniques) in the sense of accuracy by stemming the fact that the complexity of languages is a major factor and there is no distinct conclusion for effective text representation.

3. Text representation

In this section, we will formulate the text representation problem with the assumption of word vector encoding and a brief explanation about the baseline methods will be provided in the following section. Later, we will describe the proposed method in detail.

3.1. Preliminary

The main objective is to encode variable-sized word vectors $x_i, i = 1, ..., , M$ from a text document d such that an effective text-level representation z is computed. For this reason, the representation needs to capture the true content of a text in order to obtain high performance for different tasks.

As stated in [22, 30], since the studies validate that Fisher kernel outperforms the baseline methods, the main scope of this manuscript will focus on Fisher kernels and its variants for text representation. This statement can be justified with several complementary interpretations. First, since nonlinearity is reserved for the word vectors with a normalization step, it leads to an ideal solution by generalizing the feature space for different classification tasks. Moreover, the normalization step mitigates the influence of sparse responses so that more consistent representations can be derived for the outliers [12]. Second, the method encodes word vectors with the high-order statistics by calculating the deviation of samples to the overall distribution characteristics (i.e. cluster centers). As a result, the representation capacity is maximized compared to the other techniques (i.e. TF-IDF).

Briefly, the overall of Fisher kernel consists of the following steps:

- offline unsupervised clustering of a vector space with a set of training samples.
- aggregation of vectors with a probabilistic (i.e. Fisher vector [13]) or nonprobabilistic (i.e. VLAD [17]) way to compute the representation for each document.
- normalization steps (i.e. power and L2 normalization) to add additional robustness and nonlinearity to the representation.

However, the conventional kernel has several limitations particularly when supervised data is used. First, since the partitions of the word vector space are determined in an unsupervised manner (i.e. first step of the kernel), the distinctive power of the representation can be limited and the resultant partitions may not be separable for the given categories [16]. Second, the method is not able to propagate the error (e.g., classification error) to the initial words/fragments layers; thus, it becomes impossible to deploy/promote high-level representations for these words/fragments by exploiting the supervision of data. In other words, there is no way to enhance the representations of individual words. Third, considering word vectors equally for different problems can be misleading and decreases the selectivity of the representation since the relative importance for each word can be different. Lastly, due to the high dimensionality, imbalanced sample sizes can lead to overfitting with an ordinary multiclass learning algorithm.

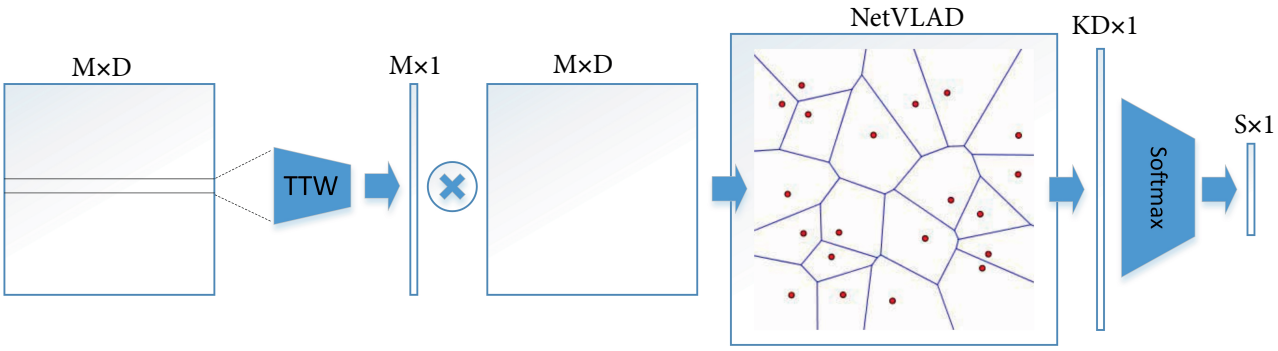

Figure 1. Flowchart of the proposed method. Here, TTW corresponds to the trainable term weighting scheme. M, D, K and S denote number of words in a document, dimension of word vector, number of cluster centers and number of categories respectively

3.2. Selective word-encoding for text representation

In this section, we will introduce our solutions for the drawbacks that we mentioned previously. The flow of the proposed method is illustrated in Figure 1.

3.2.1. Trainable word-encoding model

As stated earlier, unsupervised clustering used in the space partitioning remains weak to reveal hidden relations from data. Therefore, we adapted the NetVLAD algorithm [16] to NLP domain in order to encode precomputed word vectors $x_i \in \mathbb{R}^D$ and summarize the entire text as a single representation $z \in \mathbb{R}^{D \times K}$ by leveraging the supervised data. This adaptation is critical since this practically increases the supervision/control on the

latent partitions. Furthermore, the vectors can be aggregated with the trainable cluster centers $c_k \in \mathbb{R}^D$ and membership scores $a_k(.)$ can be learned from data as follows:

$$z_k = \sum_{i=1}^{M} a_k(\hat{x}_i)\,(\hat{x}_i - c_k), \tag{1}$$

where z_k indicates the aggregation results of word vectors for cluster k, and $\hat{x}_i - c_k$ denotes the vector difference. Note that the final representation z corresponds to the concatenation of the responses for all cluster results as $z = [z_1, z_2, ..., z_k, ..., z_K]$. Also, l2-normalization is applied at the end. Note that $\hat{x}_i$ is equal to $\hat{x}_i = \gamma(x_i)x_i$. Here, $a_k(.)$ is the function with a set of trainable parameters $w_k \in \mathbb{R}^D$ which computes the membership score of each word vector x_i for the corresponding cluster k:

$$a_k(x_i) = \frac{e^{w_k\, x_i}}{\sum_k e^{w_k\, x_i}}. \tag{2}$$

Here, $\gamma(x_i)$ determines the relative importance of word term x_i for the task. Note that this term does not exist in the original paper [16] and this is one of the contributions presented in the manuscript. To ease the understandability and clarity, we will provide further information about this term in the following section.

Alongside the high quality estimates, another contribution of the method is that the method is completely differentiable and it propagates the error to the initial layers (i.e. previous layers of the encoding stage) with the backpropagation algorithm. This allows us to extend the neural network models on word vectors and promote the existing ones by this classification error. This is highly critical since it enables us to define a novel term weight (i.e. $\gamma(.)$) that calculates the relative importance of words before the encoding step.

Table 2. Architecture of trainable term-weighting scheme. The notation of conv1D(W, C, H) indicates a 1D convolution operation whose input size, filter depth, and kernel size are denoted as W, C, and H, respectively.

Index	Inputs	Operation	Output shape
(1)	-	Word vectors	M × D
(2)	(1)	conv1D(D, 1, 128)	M × 128
(3)	(1)	conv1D(D, 3, 64)	M × 64
(4)	(1)	conv1D(D, 5, 32)	M × 32
(5)	(2),(3),(4)	concatenation	M × 224
(6)	(5)	Layer norm and ReLU	M × 224
(7)	(6)	conv1D(224, 1, 1)	M × 1
(8)	(7)	Batch norm and tanh	M × 1

3.2.2. Trainable term-weighting scheme (TTW)

Even though stop-words can be easily discarded from the corpus by considering their term-frequencies, there are still irrelevant/unimportant words which can perturb the performance of classification results. In conventional methods, remaining word vectors after the elimination of stop-words are encoded as if they all have equal importance to the problem and it makes the method sensitive to the outliers. However, the weight/distinctiveness/importance of words can be completely different for different classification problems.

For this purpose, we propose a novel scheme which determines the relative importance of terms for the given task in an end-to-end learning manner with all-trainable parameters and encodes the words by regarding the selectivity scores calculated from the function $\gamma(.)$.

As a foundation, the authors in [31, 32] particularly show that the usage of proper weights for the words can boost the performance significantly for the text representation. However, having such prior weight for each word so that it should be obtained from data automatically is impractical. Furthermore, the weight of each word can differ for each task, thus assigning a constant weight for each word can be misleading. In this manuscript, we attempted to obtain the term scores from the surrounding of words with a neural network model such that the relative importance can be used in the encoding process for state-of-the-art performance.

As formulated in Eq. 1, $\gamma(x_i)$ computes the relative importance scores of the word term x_i from the context and its surroundings. Intuitively, it composes of 1D convolutions as summarized in Table 2. To ease the understandability, for instance, if we concatenate all word vectors sequentially and denote it as $X \in \mathbb{R}^{M \times D}$, 1D-convolution filter whose input size, filter depth, and filter size are equal to W, C, and H, respectively are convolved and the shape of the estimated hidden response is equal to $V \in \mathbb{R}^{M \times H}$. Note that the filter depth intuitively computes the feature correlations between the sequential words.

For the architecture, the initial layer is inception module [33]. By this way, multiple filters with different filter depth (C) can be used. In particular, the sequential correlations with its surroundings can be captured by the filters whose C value is equal to 3 and 5. In addition, 1D-convolution whose filter depth is equal to 1 is also applied. This step is critical, since alongside the surrendering words, the current word vector should also be considered to exploit the correlation between elements.

Furthermore, in order to improve the sparsity of the responses, layer normalization [34] is utilized. Hence, instead of batch characteristics, only the responses of the filters are normalized separately. Lastly, the scores are mapped to $[0, 1]$ at the output of the function. To this end, each word vector x_i is weighted by $\gamma(.)$ function as $\hat{x}_i = \gamma(x_i)x_i$. By this way, a word vector whose the weight score is equal zero does not contribute to the text representation, on the contrary, the scores with one are fully represented in the text representation.

Note that the number of layers in $\gamma(.)$ can be further tuned by the users by accounting the trade-off for several factors such as complexity, parameter convergence, and accuracy.

3.2.3. Learning Scheme

Support vector machines (SVM) and softmax are the types of loss functions excessively used for multiclass classification tasks in NLP and each type has its own advantages and disadvantages as stated in the literature (i.e. computational complexity and generalization capacity). In this work, we used softmax loss as our baseline and proposed a novel loss function that accounts the class-imbalance problem between the categories in the learning step. To this end, the function reweights the loss based on this objective.

Formally, from a set of text-level representation and label pairs $\{z_n, y_n\}$, $n = 1, ..., , N$ and $y_n \in \mathbb{R}^S$, the novel softmax loss function minimizes the cross-entropy between the prediction and labels as follows:

$$\mathcal{L} = -\frac{1}{N} \sum_{l=1}^{L} \sum_{n=1}^{N} \alpha_l \, y_n \, \log(p_n), \tag{3}$$

where p_n is the prediction score calculated from the softmax activation. Moreover, α_l rebalances the loss

function and reduces the chances of overfitting by accounting the inverse of the sample sizes for each category:

$$\alpha_l = 1.0 - \mathbb{E}[y_l|y]. \tag{4}$$

Here, $\mathbb{E}[.]$ calculates the ratio of the samples that belong to the class l and the overall data for each mini-batch. Therefore, α yields a high score for the category with a smaller number in the mini-batch and vice versa.

To this end, the trainable parameters were optimized with the Adam solver [35] by minimizing the loss function in Eq. 3. The learning rate and number of training iteration were set to 0.001 and 10K, respectively. Mini-batch size N was fixed to 32 and the learning rate decayed by 0.1 at 8K.

4. Experiments

First, we describe the datasets that we used in the experiments. Later, the experimental results and the discussion related to the outcomes are provided.

4.1. Datasets and baselines

In the experiments, two datasets were used. First, we used Turkish parliament records[3] which contain the written speeches of the members of four major political parties. Alongside the classification for the membership based on their registered political parties as in [30], the distinction between the government and opposition parties were also surveyed. Moreover, we believe that the findings of this task can be beneficial for the political analysts in the future. Briefly, the dataset consists of four parties namely AKP (A), CHP (C), MHP (M), and HDP (H), and the sizes of train/test samples are 30731/7683, respectively. The detail characteristics for the dataset in numbers are provided in Table 3.

Table 3. The data sample distribution of the Turkish parliament records of four parties.

	A	C	M	H
Train	9687	11,305	6834	2905
Test	2433	2832	1673	745

Second, the daily articles of the columnists from two major Turkish newspaper websites, Hurriyet[4] and Sabah[5], were recorded. The first objective was to determine the type of articles as Politics (P), Sports (S), Economics (E), Magazine (M), and Culture (C) from the text. The second objective was to predict the columnist ID from their own articles. Even though each columnist has their own writing style, the shared opinions/typos between the columnists made the task quite challenging. The dataset comprised the articles from 160 columnists. For each columnist, 100 samples were reserved for the training step while 20 samples were used in the test step. Also, Table 4 illustrates the numbers for the dataset based on article types.

Note that for both tasks (see the Tables 3 and 4), there exists a noticeable class-imbalance problem that makes it harder to obtain consistent performance for all classes. Therefore, three contributions presented in the manuscript should be evaluated simultaneously. For instance, TTW should be used with the proposed loss function, otherwise, the importance of words will overfit to the dominant class and worse results can be obtained.

[3] https://www.tbmm.gov.tr/tutanak/tutanaklar.htm
[4] http://www.hurriyet.com.tr/
[5] https://www.sabah.com.tr/

Table 4. The data sample distribution of five article types.

	Politics (P)	Sports (S)	Economics (E)	Magazine (M)	Culture (C)
Train	3300	3300	1600	6400	1400
Test	660	660	320	1280	280

Lastly, we implemented two nontrainable aggregation algorithms, Fisher vector (FV) [13] and VLAD [17] to compare our method. Remark that the cluster size is set to 256 which yields the best performance for all methods. Furthermore, a two-layered feed-forward neural network was utilized with the softmax-cross entropy loss for the methods to make fair comparisons. For the evaluation of the performance, we report the mean average precision scores (mAP).

4.2. Experimental results

4.2.1. Turkish parliament records

We conducted experiments on the Turkish parliament records and the results are shown in Table 5. From the results, it can be observed that the proposed method obtained superior performance compared to the baseline methods. In particular, the performance for the categories (i.e. political parties) whose sample sizes are quite small was improved significantly due to the novel loss function. Furthermore, the trainable term-weighting scheme introduced noticeable performance improvements since the influences of words were reweighed based on their relative importance to the task. This is crucial since the results validate that the network is able to learn the importance of words from the content in an end-to-end manner.

In addition, we investigated the performance of true discrimination of the government (G) (i.e. AKP) and the opponent (O) (i.e. CHP, MHP, and HDP parties). As expected, the overall accuracy was significantly improved. Similarly, the proposed method obtained the best performance compared to the baselines.

To evaluate the impact of word-level representation to text representation, the CBOW configuration yielded the highest accuracies for the most of the cases compared to the other methods. The reason is that besides it properly conserves the true semantic of the words, it is also robust to syntactic affixes which are highly common in the agglutinative languages such as Turkish.

Lastly, we believe that the findings based on the manuscripts (i.e. particularly for this dataset) can be useful for political analysts. Therefore, we provide an affinity matrix in Table 6 for the best configuration (i.e. NetVLAD+SW+CBOW) which can be discussed in further studies. For example, the discussion on the parties which share the closest or more distinct opinions can be analyzed from the outcomes of the party membership. We believe that this can be even extend to the different time periods as well.

4.2.2. Newspaper articles

The experiments performed on the newspaper dataset are presented in Table 7. Even if the proposed method outperformed the nontrainable baselines, it achieved compatible results with the NetVLAD configurations. The reasonable explanation is that some of the words can be quite distinct for the article types and they can be highly separable from the others. This assumption practically weakens the necessity of term weighting for the problem and even the trainable models can directly obtain these words from the texts by clustering them based

Table 5. The performance results for the different methods on Turkish parliament records. The first part illustrates the true prediction scores of the membership of the party members. The second part shows the true prediction of the membership between government and opponent parties. For each part, the overall average performance is also illustrated in the last columns. The bold marks indicate the best overall performance for each column.

	A	C	M	H	Avg.	G	O	Avg.
FV+CBOW	0.836	0.800	0.720	0.738	0.773	0.822	0.924	0.873
FV+Skip-gram	0.841	0.814	0.742	0.728	0.781	0.838	0.927	0.883
FV+Glove	0.800	0.767	0.670	0.685	0.730	0.761	0.917	0.839
VLAD+CBOW	0.838	0.794	0.751	0.742	0.781	0.801	0.942	0.872
VLAD+Skip-gram	0.834	0.810	0.754	0.725	0.781	0.851	0.927	0.889
VLAD+Glove	0.811	0.769	0.742	0.728	0.763	0.754	0.931	0.843
NetVLAD+CBOW	0.816	0.855	0.764	0.794	0.807	0.747	0.973	0.860
NetVLAD+Skip-gram	0.796	0.818	0.789	0.748	0.788	0.790	0.955	0.873
NetVLAD+Glove	0.690	0.862	0.743	0.716	0.753	0.732	0.966	0.849
NetVLAD+TTW+CBOW (ours)	**0.848**	**0.868**	0.796	0.781	**0.823**	0.822	**0.963**	0.892
NetVLAD+TTW+Skip-gram (ours)	0.838	0.804	**0.829**	**0.795**	0.817	**0.867**	0.942	**0.905**
NetVLAD+TTW+Glove (ours)	0.834	0.846	0.789	0.736	0.801	0.778	0.957	0.867

Table 6. Affinity scores for NetVLAD+TTW+CBOW configuration on Turkish Parliament records.

	A	C	M	H
A	0.848	0.084	0.049	0.018
C	0.039	0.868	0.066	0.024
M	0.040	0.151	0.796	0.011
H	0.040	0.157	0.021	0.781

on their similarities. Hence, the additional use of weighting scheme can yield minor performance improvements. This observation can also be justified with the performance obtained in the word-level representation. Similarly, the Glove method, which it exploits word–word cooccurrence to compute word vectors, yields highly accurate results. For this reason, singly word encoding with less supervision might be sufficient to obtain highly accurate results for the problem.

Moreover, a similar performance improvement can be observed for the identification of columnists from their articles. As we stated, the identification of columnist IDs is challenging. However, the results validate that the use of distinct words in the articles can be determinant for the problem.

Lastly, even though W2V configurations yield superior performance, the Glove method can obtain compatible performance for the problem.

5. Conclusion

In this manuscript, we propose a selective word-encoding method for effective text representation which encodes word vectors from a text document by exploiting a trainable term-weighting scheme. Throughout the paper, three critical contributions are presented. First, we adapted a trainable aggregation model to encode the individual word vectors computed by different word-embedding-based methods. This adaptation allows us to propose a novel term-weighting model based on a neural network architecture which can estimate the relative importance of words from the context in an end-to-end learning manner. Lastly, we introduced a softmax loss function with a simple yet effective modification for robustness to the class-imbalance problem that might be

Table 7. The performance results for the different methods on Newspaper articles. First part shows the true prediction performance for the article types while the second part gives the performance for true prediction of columnist ids from their articles. The overall best results are indicated in bold.

	P	E	C	M	S	Avg.	Cid
FV+CBOW	0.899	0.806	0.650	0.910	0.943	0.842	0.822
FV+Skip-Gram	0.798	0.609	0.750	0.933	0.853	0.788	0.826
FV+Glove	0.799	0.578	0.631	0.835	0.684	0.705	0.818
VLAD+CBOW	0.895	0.815	0.650	0.911	0.934	0.841	0.823
VLAD+Skip-Gram	0.866	0.771	0.639	0.918	0.910	0.821	0.830
VLAD+Glove	0.863	0.790	0.703	0.914	0.919	0.838	0.822
NetVLAD+CBOW	**0.911**	0.787	0.767	**0.939**	0.960	0.873	0.840
NetVLAD+Skip-Gram	0.883	0.803	0.778	0.934	0.954	0.870	0.832
NetVLAD+Glove	0.916	0.831	0.760	0.932	0.928	0.874	0.830
NetVLAD+SW+CBOW (ours)	0.896	**0.856**	**0.789**	0.913	0.939	**0.878**	**0.844**
NetVLAD+SW+Skip-Gram (ours)	0.907	0.843	0.732	0.914	**0.965**	0.872	0.835
NetVLAD+SW+Glove (ours)	0.896	0.831	0.692	0.935	0.946	0.860	0.840

observed for the classification tasks. The effectiveness of the proposed contributions was evaluated on various classification tasks and the obtained results validate that the method improves the performance compared to the baselines. In the future, the proposed method can be easily extended to various text problems such poet/author recognition as well as measuring the influence of predecessors to the successors.

Acknowledgment

The authors are pleased to thank NVIDIA for supporting this research with the Tesla K40 graphics card.

References

[1] Mikolov T, Sutskever I, Chen K, Corrado GS, Dean J. Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems Conference, 2013. pp. 3111-3119.

[2] Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems Conference, 2012. pp. 1097-1105.

[3] Cho K, Merriënboer BV, Gulcehre C, Bahdanau D, Bougares F, Schwenk H, Bengio Y. Learning phrase representations using rnn encoder-decoder for statistical machine translation. arXiv 1406.1078, 2014.

[4] Pennington J, Socher R, Manning C. Glove: global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014. pp. 1532-1543.

[5] Yu LC, Wang J, Lai KR, Zhang X. Refining word embeddings for sentiment analysis. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, 2017. pp. 534-539.

[6] Mikolov T, Chen K, Corrado G, Dean J. Efficient estimation of word representations in vector space. arXiv 1301.3781, 2013.

[7] Bakarov A, Gureenkova O. Automated detection of non-relevant posts on the russian imageboard 2ch: importance of the choice of word representations. In: International Conference on Analysis of Images, Social Networks and Texts, 2017. pp. 16-21.

[8] Lin CC, Ammar W, Dyer C, Levi L. Unsupervised pos induction with word embeddings. arXiv preprint arXiv:1503.06760, 2015.

[9] Rossiello G, Basile P, Semeraro G. Centroid-based text summarization through compositionality of word embeddings. In: Proceedings of the MultiLing 2017 Workshop on Summarization and Summary Evaluation Across Source Types and Genres; 2017. pp. 12-21.

[10] Kalender M, Korkmaz EE. Turkish entity discovery with word embeddings. Turkish Journal of Electrical Engineering and Computer Sciences 2017; 25: 2388-2398.

[11] Akın AA, Akın MD. Zemberek, an open source NLP framework for Turkic languages. Structure 2007; 10: 1-5.

[12] Babenko A, Lempitsky V. Aggregating local deep features for image retrieval. In: Proceedings of the IEEE International Conference on Computer Vision; 2015. pp. 1269-1277.

[13] Perronnin F, Sánchez J, Mensink T. Improving the fisher kernel for large-scale image classification. In: European Conference on Computer Vision; Berlin, Heidelberg; 2010. pp. 143-156.

[14] Koç A, Utlu I, Senel LK, Ozaktas HM. Imparting interpretability to word embeddings. arXiv 1807.07279, 2018.

[15] Young T, Hazarika D, Poria S, Cambria E. Recent trends in deep learning based natural language processing. IEEE Computational Intelligence Magazine 2018; 13 (3): 55-75.

[16] Arandjelovic R, Gronat P, Torii A, Pajdla T, Sivic J. NetVLAD: CNN architecture for weakly supervised place recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016. pp. 5297-5307.

[17] Jégou H, Douze M, Schmid C, and Pérez P. Aggregating local descriptors into a compact image representation. In: Computer Vision and Pattern Recognition (CVPR), 2010. pp. 3304-3311.

[18] Deerwester S, Dumais ST, Furnas GW, Landauer TK, Harshman R. Indexing by latent semantic analysis. Journal of the American Society for Information Science 1990; 41(6): 391-407.

[19] Le Q, Mikolov T. Distributed representations of sentences and documents. In: International Conference on Machine Learning, 2014. pp. 1188-1196.

[20] Guo W, Diab M. Modeling sentences in the latent space. In: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics 2012. pp. 864-872.

[21] Islam A, Inkpen D. Semantic text similarity using corpus-based word similarity and string similarity. ACM Transactions on Knowledge Discovery from Data (TKDD) 2008; 2 (2): 10.

[22] Clinchant S, Perronnin F. Aggregating continuous word embeddings for information retrieval. In: Proceedings of the Workshop on Continuous Vector Space Models and Their Compositionality, 2013. pp. 100-109.

[23] Boom CD, Canneyt SV, Demeester T, Dhoedt B. Representation learning for very short texts using weighted word embedding aggregation. Pattern Recognition Letters 2016; 80: 150-156.

[24] Kenter T, Rijke MD. Short text similarity with word embeddings. In: Proceedings of the 24th ACM international on Conference on Information and Knowledge Management, 2015. pp. 1411-1420.

[25] Norouzi M, Blei DM. Minimal loss hashing for compact binary codes. In: Proceedings of the 28th International Conference on Machine Learning, 2011. pp. 353-360.

[26] He H, Gimpel K, Lin J. Multi-perspective sentence similarity modeling with convolutional neural networks. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, 2015. pp. 1576-1586.

[27] Hu B, Lu Z, Li H, Chen Q. Convolutional neural network architectures for matching natural language sentences. In: Advances in Neural Information Processing Systems Conference, 2014. pp. 2042-2050.

[28] Kim Y. Convolutional neural networks for sentence classification. arXiv 1408.5882, 2014.

[29] Kiros R, Zhu Y, Salakhutdinov RR, Zemel R, Urtasun R, Torralba A, Fidler S. Skip-thought vectors. In: Advances in Neural Information Processing Systems Conference, 2014. pp. 3294-3302.

[30] Esen E, Özkan S. Analysis of turkish parliament records in terms of party coherence. In: Signal Processing and Communications Applications Conference, 2017. pp. 1-4.

[31] Zheng G, Callan J. Learning to reweight terms with distributed representations. In: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2015. pp. 575-584.

[32] Ji Y, Eisenstein J. Discriminative improvements to distributional sentence similarity. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, 2013. pp. 891-896.

[33] Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z. Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016. pp. 2818-2826.

[34] Ba JR, Kiros JR, Hinton GE. Layer normalization. arXiv 1607.06450, 2016.

[35] Kingma DP and Ba J. Adam: a method for stochastic optimization. arXiv 1412.6980, 2014.