24th European Conference on Artificial Intelligence - ECAI 2020
Santiago de Compostela, Spain

Improving Document Classification with Multi-Sense Embeddings

Vivek Gupta[1] and **Ankit Kumar**[2] and **Pegah Nokhiz**[3] and **Harshit Gupta**[4] and **Partha Talukdar**[5]

Abstract. Efficient representation of text documents is an important building block in many NLP tasks. Research on long text categorization has shown that simple weighted averaging of word vectors for sentence representation often outperforms more sophisticated neural models. Recently proposed Sparse Composite Document Vector (SCDV) [32] extends this approach from sentences to documents using soft clustering over word vectors. However, SCDV disregards the multi-sense nature of words, and it also suffers from the curse of higher dimensionality. In this work, we address these shortcomings and propose SCDV-MS. SCDV-MS utilizes multi-sense word embeddings and learns a lower dimensional manifold. Through extensive experiments on multiple real-world datasets, we show that SCDV-MS embeddings outperform previous state-of-the-art embeddings on multi-class and multi-label text categorization tasks. Furthermore, SCDV-MS embeddings are more efficient than SCDV in terms of time and space complexity on textual classification tasks. We have released SCDV-MS source code with the paper. [6]

1 Introduction

Distributed word embeddings such as word2vec [35] are effective in capturing the semantic meanings of the words by representing them in a lower-dimensional continuous space. A smooth inverse frequency-based word vector averaging technique for sentence embeddings was proposed by [4]. However, because the final representation is in the same space as the word vectors, these methods are only capable of capturing the meaning of a single sentence. Thus, embedding a large text document in a dense, low-dimensional space is a challenging task. [32] attempted to resolve this problem by proposing a clustering-based word vector averaging approach (SCDV) for embedding larger text documents. SCDV embeds each document by cluster-based averaging, thus representing each document in a more representative space than the original vectors. This model combines the word embedding models with a latent topic model where the topic space is learned efficiently using a soft clustering technique over embeddings. The final document vectors are also made sparse to reduce time and space complexities in several downstream tasks.

SCDV has many shortcomings: applying thresholding-based sparsity on the document representations can be unreliable since it is highly sensitive to the number of documents in the corpus. SCDV does not utilize the word contexts to disambiguate word sense for learning sense-aware document representations. Ignoring the multi-sense nature of the words during the representation leads to cluster ambiguity. SCDV neglects the negative additive effect of common words such as *'and', 'the',* etc. during final document representations. Lastly, the documents represented by SCDV suffer from the curse of high dimensionality and cannot be utilized for deep learning applications which require low-dimensional continuous representations. To overcome these challenges, we proposed a novel document representation technique, namely SCDV-MS. Our proposed SCDV-MS mitigated the above shortcomings by the following contributions.

1. To overcome the problem of cluster ambiguity SCDV-MS replaced the single sense word vector representations with multi-sense context-sensitive word representations to resolve word sense disambiguation.
2. SCDV-MS removed the noise in the final representation by applying a threshold-based sparsity directly on fuzzy word cluster assignments instead of the document representations. Sparser representations result in better performance, lower time and space complexities.
3. To overcome the noisy negative additive effect of common words such as *'and', 'the',* etc. SCDV-MS learned and used Doc2VecC-initialized [9] robust word vectors to zero out common and high frequent words.
4. Lastly, we showed that the sparse word-topic vectors can be projected into a non-linear local neighborhood preserving a manifold to learn continuous distributed representations much more effectively and efficiently than SCDV which proves to be useful for deep learning application.

Overall, we show that: disambiguating the multiple senses of words based on their context words (adjacent words) can lead to better document representations. Sparsity in representations is helpful for effective and efficient lower-dimensional manifold representation learning. Representation noise at words' level has a significant impact on the final downstream tasks.

In section 1, we provided a brief introduction to the problem statement. In section 2 we discuss the related work in document representations. In section 3, we describe the proposed algorithm and then discuss the proposed modifications compared to SCDV in section 4. We move on to experiments in section 5, followed by conclusions in section 6.

2 Related Work

For document representation, averaging of word-vectors with an unweighted scheme was proposed by [46, 36, 37, 34]. [44] extended the

[1] University of Utah, email: vgupta@cs.utah.edu
[2] JeevanSathi, email: ankit.kgpian@gmail.com
[3] University of Utah, email: pnokhiz@cs.utah.edu
[4] IIT Guwahati, email: harshitgupta@iitg.ac.in
[5] IISC Bangalore, email: ppt@iisc.ac.in
[6] https://github.com/vgupta123/SCDV-MS

previous simple averaging model by tf-idf weighting of word vectors to form document vectors. [26] proposed paragraph models (PV-DM and PV-DBOW) similar to word vector models (CBoW and SGNS) by treating each paragraph as a pseudoword. [47] used a parse tree to train a Recursive Neural Network (RNN) with supervision. In addition, several neural network models such as seq2seq models: Recurrent Neural Networks (RNN) [33], Long Short Term Memory Networks (LSTM) [16] and a hierarchical model: Convolutional Neural Networks (CNN) [23, 21] were proposed to capture syntax while representing documents. [50] used supervised learning over the Paraphrase Dataset (PPDB) to learn Paraphrastic Sentence embeddings (PSL). Later, [49] also propose an optimization of word embeddings based on a neural network and a cosine similarity measure.

Several models such as WTM [15], TWE [30], NTSG [30], LTSG [25], w2v-LDA [39], TV+MeanWV [28], Topic2Vec [40], Gaussian-LDA [11], Lda2vec [38], ETM [14], KEDR [45], D-ETM [13] and MvTM [29] combine topic modeling [8] with word vectors to learn better word and sentence representations. [24] cast the distributional hypothesis to a sentence level by proposing skip-thought document vectors. Recently, two deep contextual word embeddings namely ELMo [41] and BERT [12] were proposed. These contextual embeddings perform as state of the art on multiple NLP tasks since they are very effective in capturing the surrounding context. Interestingly, [27] checks the effect of using multi-sense embeddings on various NLP tasks. However, our goal is different and aim at effectively using multi-sense words embeddings to learn better document representations. A hard clustering-based averaging of word vectors was proposed by [17, 3] to form document vectors. [18] extended the approach with a better partitioning technique and tried it on other natural language tasks. [32] further improved the state-of-the-art SCDV by using fuzzy clustering and tf-idf weighted word averaging. Their method outperformed earlier models on several NLP tasks.

3 Proposed Algorithm SCDV-MS

In this section, we will describe our new proposed algorithm 1 in details. The algorithm is similar to SCDV [32], but with important modifications and has three main components as described below:

3.1 Word Sense Disambiguation (Algo 1: 1 - 6):

We employed the widely-used AdaGram [6] algorithm to disambiguate the multi-sense words in our corpora. We chose AdaGram because it's a nonparametric Bayesian extension of Skip-gram which automatically learns the counts of the senses of the multi-sense words and their sense representations. [7] We first trained the AdaGram algorithm on the training corpora. [8] We used the trained model to annotate the words with the corresponding word senses in all train-test examples. We then trained the Doc2vecC algorithm on an annotated corpus to obtain the final multi-sense word vectors i.e. learning one sense vector for each word sense. Lastly, we obtained the idf values of words of the vocabulary which we will use as a means for weighting the rare words (Lines 4-6 Algo 1).

3.2 Word Vector Clustering (Algo 1: 6 - 9)

Similar to the SCDV approach, we clustered the word embeddings using Gaussian Mixture Models (GMM), which is a soft clustering technique, and obtained the word-cluster assignments probabilities

Algorithm 1: SCDV-MS

Data: Documents D_n, $n = 1, \ldots, N$
Result: Document vectors $\vec{dv}_{D_n}$, $n = 1, \ldots, N$

`/* Word Sense Disambiguation */`

1 Use adagram for word sense disambiguation;
2 Annotate each word with a sense according to the neighboring context words;
3 Obtain word vectors ($\vec{wv}_i$) on annotated corpus using Doc2VecC;
4 **for** *each word* $w_i \in V$ **do**
5 obtain idf values, idf(w_i), $i = 1..|V|$;
`/* |V| is the vocabulary */`
6 **end**

`/* Word Vector Clustering */`

7 Fuzzy clusters $\vec{wv}$ in K clusters;
8 Each word w_i and cluster c_k, obtain $\mathrm{P}(c_k|w_i)$;
9 $\mathrm{SP}(\vec{c}|w_i)$ = make-sparse($\mathrm{P}(\vec{c}|w_i)$);

`/* Word Topic Vectors */`

10 **for** *each word* $w_i \in V$ **do**
11 **for** *each cluster* c_k **do**
12 $\vec{wcv}_{ik} = \vec{wv}_i \times \mathrm{SP}(c_k|w_i)$;
13 **end**
14 $\vec{wtv}_i = \mathrm{idf}(\mathrm{w}_i) \times \bigoplus_{k=1}^{K} \vec{wcv}_{ik}$;
`/* ⊕ is concatenation */`
`/* Optional: Manifold Learning */`
15 $\vec{rwtv}_w$ = manifold-proj($\vec{wtv}_w$);
16 **end**

`/* SCDV-MS Representation */`

17 **for** $n \in (1..N)$ **do**
`/* initialize vectors */`
18 $\vec{dv}_{D_n} = \vec{0}$;
19 **for** *word* w_i *in* D_n **do**
20 $\vec{dv}_{D_n}$ += $\vec{wtv}_{w_i}$;
21 **end**
22 **end**

$\mathrm{P}(c_k|w_i)$. Additionally, we made use of the fact that GMMs have an irrelevant noisy tail and made the cluster probability assignment $\mathrm{P}(\vec{c}|w_i)$ manually sparse by zeroing the values of $\mathrm{P}(c_k|w_i)$. Retaining only the top l maximum $\mathrm{P}(c_k|w_i)$ and zeroing the rest $K - l$ values results in a sparse word-cluster assignment vector $\mathrm{SP}(\vec{c}|w_i)$ for each word. Here, K represents the total number of clusters and l is the sparsity constant ($l << K$). One can use different values of l for each word (w_i) depending on the values of $\mathrm{P}(c_k|w_i)$. However, in our experiments we did not observe significant performance difference when l is varied with respect to the words.

$$\mathrm{SP}(c_k|w_i) = \begin{cases} \mathrm{P}(c_k|w_i) & \text{if } k \in \{k| \arg\max_k^l \mathrm{P}(c_k|w_i)\} \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

$\arg\max_k^l \mathrm{P}(c_k|w_i)$ outputs indices of top l maximum assignments.

3.3 Word Topic Vectors (Algo 1: 9 - 16)

Similar to SCDV, for each word $w_i \in V$, we created K different word-cluster vectors of d dimensions ($\vec{wcv}_{ik}$) by weighting the

[7] One could also replace AdaGram with [10] and [5]
[8] https://github.com/sbos/AdaGram.jl

word's embedding with sparse probability distribution for the k^{th} cluster, i.e., $SP(c_k|w_i)$. Next, we concatenated the K word-cluster vectors ($\vec{wcv}_{ik}$) into a word-topic vector ($\vec{wtv}_i$), which is a $K \times d$ embedding vector. We then weighted it with inverse document frequency (idf(w_i)) of w_i to obtain word-topic vectors ($\vec{wtv}_i$). For all words appearing in a given document D_n, we computed the average of the corresponding projected lower dimensional word-topic vectors $\vec{wtv}_i$ to obtain the document vector $\vec{dv}_{D_n}$. Furthermore, one can optionally project the ($\vec{wtv}_i$) into a lower dimensional continuous representation called reduced word topic vectors, ($\vec{rwtv}_i$), using manifold learning algorithms, namely Random Projection [2], PCA [1] and Denoising Autoencoders [48]. We can then use them instead of ($\vec{wtv}_i$) for document representation. We call this reduction-based representation method as R-SCDV-MS. Refer to section 4.1 on manifold learning for more details.

$$\vec{wcv}_{ik} = \vec{wv}_i \times \text{SP}(c_k|w_i) \tag{2}$$

$$\vec{wtv}_i = \text{idf}(w_i) \times \oplus_{k=1}^{K} \vec{wcv}_{ik} \tag{3}$$

$$\vec{rwtv}_i = \text{manifold-proj}(\vec{wtv}_i) \tag{4}$$

$\bigoplus$ is the concatenation and manifold-proj is the manifold learning algorithm we utilized. Figures 2 and 3 show the flow-chart of high level flow of our proposed SCDV-MS embedding.

4 Discussion on Proposed Modifications

In this section, we will describe the modifications applied to the SCDV embeddings in details.

4.1 Word Representation: Single Sense vs Multi Sense

SCDV-MS used a multi-sense approach instead of single sense word embeddings because SCDV does not disambiguate the senses of the words based on the context words used in the documents. SCDV-MS performed an automatic word sense disambiguation using multi-sense word embeddings according to the context determined by the neighboring words to resolve cluster ambiguity for polysemous words. Table 1 shows examples of multi-sense words along with their fitting context and the prominent words of the assigned clusters.

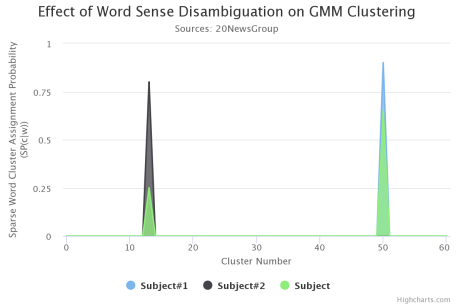

Figure 1: Effect of sense disambiguation on word cluster assignment probabilities

Figure 1 shows the effect of sense disambiguation on fuzzy GMM clustering on the 20NewsGroup dataset. The same word is assigned to different clusters depending on its context which helps in resolving the word cluster ambiguities, e.g., without sense disambiguation, the word *'Subject'* belongs to cluster 13 with probability 0.25 and cluster 50 with probability 0.65. But after sense disambiguation we acquire two word embeddings of the word *'Subject'*, i.e., *'Subject#1'* and *'Subject#2'*. *'Subject#1'* belongs to cluster 50 with probability of 0.9 and *'Subject#2'* belongs to cluster 13 with probability 0.8. So depending on the context in which word *'Subject'* is used, the algorithm assigns *'Subject'* to a single cluster based on its sense; thus word cluster ambiguity is resolved. We observe similar disambiguation effects for other polysemous words in the corpus.

4.2 Thresholding Word Cluster Assignments

In SCDV, the thresholding is applied to the final document vectors. However, applying the hard thresholding in an earlier stage in word cluster assignments ($P(c_i|w)$) results in better heavy tail noise removal and yields more robust representations. Thus, SCDV-MS applied the hard thresholding directly on the word cluster assignments instead of the final document representations. Furthermore, applying sparsity over vocabulary words with fewer dimensions ($O(VK)$) instead of millions of documents ($O(NKd)$) results in higher efficiency ($Nd >> V$). Here, N is the number documents, V is vocabulary, K is number of clusters, and d is word vector dimensions. Empirically, on 20NewGroups we observe that about 98% of entries in word-cluster assignments ($P(c_i|w)$) for all words are close to 0 (< 0.01). For each word on average, the probability of cluster assignment ($P(c_i|w)$) for 58 cluster assignments out of 60 (variance of 1.56) is less than 0.01. Thus, applying thresholding at word-cluster assignment level is reasonable.

4.3 Doc2VecC vs SGNS Representations

In the SCDV approach, the SGNS word vectors represent the common words (mainly stop words) as non-zero vectors. This makes the clusters redundant and generates a heavy tail noise. SCDV-MS addressed this issue by using Doc2VecC [9] which introduces corruption while doing context addition in word embeddings to help in learning robust word vector representations. In this approach, the common words of the corpus are forcefully learned as zeroed vectors. We observed that using Doc2VecC trained word vectors results in non-redundant diverse clusters. Thus, using Doc2VecC trained word vectors not only improves the performance but also reduces the feature size. There is no running time overhead for Doc2VecC compared to the SGNS.

4.4 Low Dimensional Manifold Learning

SCDV [32] represents documents as high dimensional sparse vectors. The SCDV approach showed that such vectors are useful for linear classification models. However, being useful for many downstream applications (especially the ones using deep learning models) requires a continuous low dimensional document representation similar to word vectors. To overcome this issue, R-SCDV-MS projected the sparse word-topic vectors into a lower-dimensional manifold which preserves the local neighborhood using simple techniques such as random projection. Furthermore, the manifold learning is applied over word vocabularies instead of millions of documents, which is more efficient. Dimensionality reduction for SCDV-MS is roughly ($O(\frac{N}{V})$) (where N is the number of documents and V is the size of vocabulary) faster than SCDV.

5 Experimental Results

Document embeddings obtained using SCDV-MS can be used as direct features for downstream supervised tasks. We experimented with text classification tasks (see Table 2) to show the effectiveness of our

Table 1: Examples of multi-sense words along with their context words and the corresponding prominent cluster words

Multi-Sense Words	Sentence (Context Words)	Prominent Cluster Words
Subject	The math **subject** is a nightmare for many students	physics, chemistry, math, science
	In anxiety, he sent an email without a **subject**	mail, letter, email, gmail
Break	After promotion, he went to Maldives for spring **break**	vacation, holiday, trip, spring
	Breaking government websites is common for hackers	encryption, cipher, security, privacy
	Use **break** to stop growing recursion loops	if, elseif, endif, loop, continue
Unit	The S.I. **unit** of distance is meter	calculation, distance, mass, length
	Multimeter shows a **unit** of 5V	electronics, KWH, digital, signal
Interest	His **interest** lies in astrophysics	information, enthusiasm, question
	Bank's **interest** rates are controlled by RBI	bank, market, finance, investment

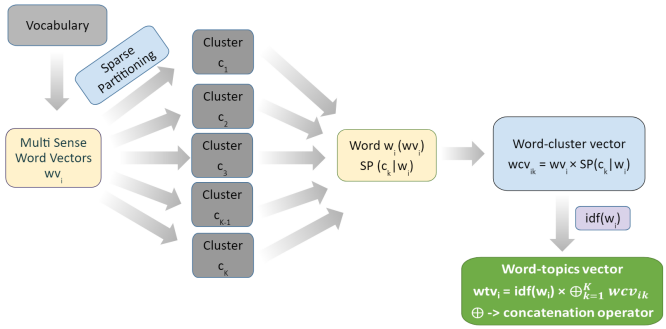

Figure 2: Flowchart representing modified $\vec{wtv}$ computation.

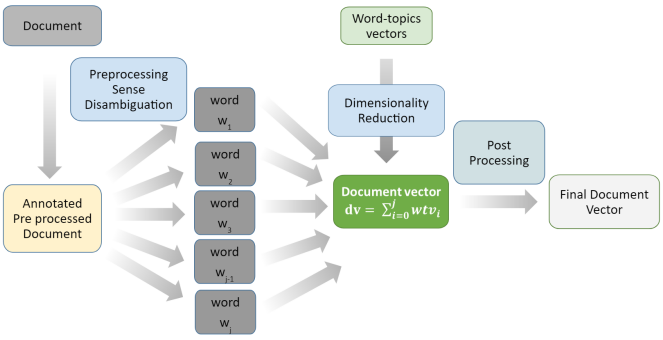

Figure 3: Flowchart representing final document vector computation.

embeddings. We evaluated the following questions through our experiments.

Table 2: Text classification datasets overview.

Task	Dataset	#Classes	#train / #test	#words/doc
Multi-class	20NewsGroup	20	11K / 8K	40-80
Multi-label	Reuters-21578	444	13K / 6K	200-400

Q1. Does disambiguating word-cluster assignments using multi-sense embedding improve classification performance?

Q2. Does hard thresholding over word-cluster assignments improve performance, space and time complexities?

Q3. Is representational noise reduction using Doc2Vec initialization effective?

Q4. Can effective lower dimensional manifold be learned from the sparse high dimensional word topic vectors?

Baselines: We considered the following baselines: Bag-of-Words (BoW) [19], Bag of Word Vector (BoWV) [17], [9] Sparse Composite Document Vectors (SCDV) [32], [10] paragraph vectors [26], pmeans [43], ELMo [42], Topical word embeddings (TWE-1) [31], Neural Tensor Skip-Gram Model (NTSG-1 to NTSG-3) [30], tf-idf weighted average word-vector [44] and weighted Bag of Concepts (weight-BoC) [22], and BERT [12]. In BoC we built topic-document vectors by counting the member words in each topic. For BERT, we reported the results on the unsupervised pre-trained (pr) model because of a fair comparison to our approach which is also unsupervised. In Doc2VecC [9] averaging and training the vectors was done jointly with corruption. Also, in SIF [4] we used the inverse frequency weights for weighting while averaging word vectors, and finally removed the common components from the average. The results of our proposed embeddings is represented by SCDV-MS in Tables 5 and 3.

[9] https://bit.ly/2X0XfBH
[10] https://bit.ly/36NxGZh

We also compared our representation with othe methods, described in the related work.

The Experimental Setting: We learned the word embeddings with Doc2VecC using commonly used parameters, e.g., negative sample size of 10, minimum word frequency of 20, and the window size of 10. We ensure for usual data cleansing like stop word removal, lemmetization and stemming for all the baselines. In addition, we used simple models such as LinearSVM for multi-class classification and Logistic regression with a OneVsRest setting for the multi-label classification tasks so that we can directly compare our results with the previous approaches which uses the same classifiers. Similar to SCDV, to tune the hyperparameters, we employed a 5-fold cross-validation on the F1 score. We also used the Doc2VecC model [9] to initialize the word embeddings on the annotated corpora for performance improvement. To ensure a fair comparison with SCDV, we fixed the same number of clusters to 60 and used full covariance for GMM clustering for all experiments based on our best empirical results with cross-validation. We tuned the hard threshold sparsity constant l from range $\{3, 5, 7\}$ with cross-validation to select the best hyper-parameter for making the word cluster assignments sparse. Moreover, we used AdaGram [6] for disambiguating the sense of multi-sense words using a neighborhood of 5 context words on both sides, so that the window size is 10. We first ranked our words based on their tf-idf scores; we then selected a practicable number (top 5000 words) as candidates for the polysemic words. Next, we selected the true polysemic words by applying AdaGram on the candidates. [11] The best parameter settings were used to generate baselines results. We used 200 dimensions for the tf-idf weighted word-vector model, 400 for the paragraph vector model, 80 topics and 400 dimensional vectors for TWE/NTSG/LTSG, and 60 topics and 200 dimensional word vectors for SCDV and BOWV. We reported the average of 5 runs. Our results were robust across multiple

[11] https://bit.ly/2Jv6wxX

runs with a variance of O(10^{-6}).

5.1 Text Classification Results

We evaluated the classifier's performance on multi-class classification using several metrics such as accuracy, macro-averaging precision, recall, and macro F1-score. Table 3 shows a comparison with multiple state-of-the-art document representations (the first 7 except BERT/ELMo are clustering-based, the next 11 topic-word embeddings based, the next 6 are simple averaging or topic modeling methods) on the 20NewsGroup dataset. We also reported the results (micro F1) on the 20 classes of 20NewsGroup in Table 4. Furthermore, we evaluated the multi-label classification performance using several metrics such as Precision@K, nDCG@k [7], Coverage error, Label ranking average precision score (LRAPS)[12] and macro F1-score. Table 5 shows the results on the Reuters dataset.

Table 3: Performance on multi-class classification.

Model	Accuracy	Precision	Recall	F-measure
SCDV-MS	**86.19**	**86.20**	**86.18**	**86.16**
R-SCDV-MS	84.9	84.9	84.9	84.9
BERT (pr)[12]	84.9	84.9	85.0	85.0
SCDV [32]	84.6	84.6	84.5	84.6
RandBin	83.9	83.99	83 .9	83.76
BoWV [17]	81.6	81.1	81.1	80.9
pmeans [43]	81.9	81.9	81.9	81.5
Doc2VecC [9]	84.0	84.1	84.1	84.0
BoE [20]	83.1	83.1	83.1	83.1
NTSG-2 [30]	82.5	83.7	82.8	82.4
LTSG [25]	82.8	82.4	81.8	81.8
WTM [15]	80.9	80.3	80.3	80.0
ELMo [42]	74.1	74.0	74.1	73.9
w2v-LDA [39]	77.7	77.4	77.2	76.9
TV+MeanWV [28]	72.2	71.8	71.5	71.6
MvTM [29]	72.2	71.8	71.5	71.6
TWE-1 [31]	81.5	81.2	80.6	80.6
lda2Vec [38]	81.3	81.4	80.4	80.5
lda [8]	72.2	70.8	70.7	70.0
weight-AvgVec [44]	81.9	81.7	81.9	81.7
BoW [44]	79.7	79.5	79.0	79.0
weight-BOC [44]	71.8	71.3	71.8	71.4
PV-DBoW [26]	75.4	74.9	74.3	74.3
PV-DM [26]	72.4	72.1	71.5	71.5

Datasets: We evaluated our approach by running multi-class classification experiments on the 20NewsGroup dataset, [13] and multi-label classification experiments on the Reuters-21578 dataset. [14] For more details on dataset statistics refer to Table 2. We used *script* for datasets preprocessing. [15]

Results and Analysis. We observed that our modified embeddings (SCDV-MS) with Doc2VecC-initialized word vectors, direct thresholding on word cluster assignments, and multi-sense disambiguation using AdaGram outperforms all earlier embeddings on both the 20NewsGroup and the Reuters datasets. From class-wise results in Table 4, we notice a consistent performance improvement where we are outperforming SCDV in 18 out of 20 classes. It should be noted that the improvement on Reuters is not as great as the 20NewsGroup dataset due to the fact that the number of unique polysemic words

[12] Section 3.3.3.2 of `https://goo.gl/4GrR3M`
[13] `http://qwone.com/~jason/20Newsgroups/`
[14] `https://goo.gl/NrOfu`
[15] `https://bit.ly/2PXDdXj`

Table 4: Class-wise F1-Score on the 20newsgroup dataset with different document representations.

Class Name	SCDV	SCDV-MS	R-SCDV-MS
alt.atheism	80.14	**81.35**	80.39
comp.graphics	**78.99**	76.84	76.95
comp.os.ms-windows.misc	75.65	77.65	**78.28**
comp.sys.ibm.pc.hardware	72.08	**73.43**	68.38
comp.sys.mac.hardware	82.15	**86.82**	80.16
comp.windows.x	81.8	82.97	**83.27**
misc.forsale	82.8	**85.13**	84.99
rec.autos	89.06	**92.53**	91.77
rec.motorcycles	94.27	**96.11**	94.27
rec.sport.baseball	93.57	**96.47**	93.68
rec.sport.hockey	**97.27**	96.78	96.41
sci.crypt	93.1	92.82	**93.5**
sci.electronics	77.38	**77.45**	74.25
sci.med	88.58	**92.30**	91.57
sci.space	90.33	**91.40**	90.71
soc.religion.christian	89.56	**89.97**	89.76
talk.politics.guns	80.69	**84.18**	83.05
talk.politics.mideast	95.96	95.95	**96.1**
talk.politics.misc	69.33	73.49	**73.67**
talk.religion.misc	65.53	**65.54**	60.48

Table 5: Performance on various metrics for multi-label classification on the Reuters dataset.

Model	Prec @1	Prec @5	nDCG @5	Cover. Error	LRAPS	F1 Score
SCDV-MS	**95.06**	**37.56**	**50.20**	**5.87**	**94.21**	**82.71**
R-SCDV-MS	93.56	37.00	49.47	6.74	92.96	81.94
BERT (pr)	93.8	37	49.6	6.3	93.1	81.9
SCDV	94.00	37.05	49.6	6.65	93.34	81.77
Doc2VecC	93.45	36.86	49.28	6.83	92.66	81.29
pmeans	93.29	36.65	48.95	7.66	91.72	77.81
BoWV	92.90	36.14	48.55	8.16	91.46	79.16
TWE-1	90.91	35.49	47.54	8.16	91.46	79.16
PV-DM	87.54	33.24	44.21	13.15	86.21	70.24
PV-DBoW	88.78	34.51	46.42	11.28	87.43	73.68
tfidf AvgVec	89.33	35.04	46.83	9.42	87.90	71.97

Table 6: Ablation Study reporting F1 scores. In $\pm x$, x is the variance across several runs.

Ablation (w/o)	20NewsGroup	Reuters
Sparsity	85.28 ± 0.002	82.17 ± 0.001
Doc2VecC	85.41 ± 0.001	82.08 ± 0.002
MultiSense	85.16 ± 0.001	82.43 ± 0.001
All	84.61 ± 0.004	81.77 ± 0.003
None	$\mathbf{86.16 \pm 0.002}$	$\mathbf{82.71 \pm 0.002}$

Table 7: Performance in terms of accuracy with various dimensionality reduction methods on the 20NewsGroup dataset. Similar results were acquired for precision, recall, and F1 score.

Dimension	Random Projection	PCA (SubSpace)	Autoencoder
200	78.97	80.62	81.44
500	82.19	83.14	83.83
1000	83.75	83.80	84.31
2000	84.47	84.34	84.80
3000	**84.94**	**84.86**	**84.90**

Figure 4: Percentage loss in F1-Score (%RL) after Random Projection-based $\vec{wtv}$ dimensionality reduction on 20NewsGroup.

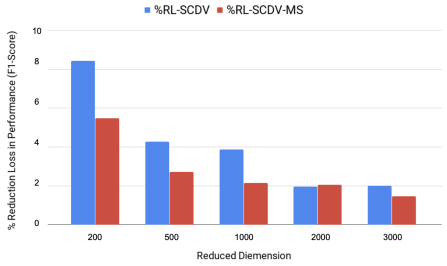

Figure 5: Percentage loss in F1-Score (%RL) after Autoencoder-based $\vec{wtv}$ dimensionality reduction on 20NewsGroup.

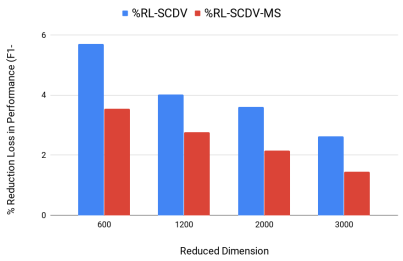

Figure 6: Percentage loss in F1-Score (%RL) after PCA (Subspace)-based $\vec{wtv}$ dimensionality reduction on 20NewsGroup.

Table 8: Performance of Convolutional Neural Network (CNN) for multi-class text classification on 20NewsGroup with the original word embeddings (200 dimensions) and the reduced word topic vectors (2000 dimensions). In $\pm x$, x is the variance across several runs.

Embedding	Dimen	Accuracy	Precision	Recall	F1-Score
Word2Vec	200	82.07 $\pm$ 0.003	82.22 $\pm$ 0.005	81.9 $\pm$ 0.004	81.9 $\pm$ 0.005
Reduce Word Topic Vector	200	**82.13 $\pm$ 0.002**	**82.36 $\pm$ 0.003**	**82.06 + $\pm$ 0.003**	**82.05 $\pm$ 0.003**
Word2Vec	2000	82.31 $\pm$ 0.004	82.87 $\pm$ 0.005	82.31 $\pm$ 0.004	82.38 $\pm$ 0.005
Reduce Word Topic Vector	2000	**82.85 $\pm$ 0.001**	**83.18 $\pm$ 0.004**	**82.64 $\pm$ 0.002**	**82.68 $\pm$ 0.003**

Table 9: Time and Space complexity analysis of embedding methods. Bold values represent the best results.

Method	Vocab	$\vec{wtv}$ Dim	$\vec{wtv}$ Sparsity(%)	$\vec{dv}$ Sparsity(%)	Cluster (sec)	Feature (μ sec)	Predict (μ sec)	Training (min)	Model Size (KB)	$\vec{wtv}$ Space(MB)
SCDV	15591	12000	1	**81**	**242**	2.56	119	82	1900	748
SCDV-MS	25466	12000	**98**	74	569	**0.06**	111	79	1900	**71**
R-SCDV-MS	25466	2000	0	0	576	0.86	**14**	**66**	**333**	203

Table 10: PCA based subspace rank reduction criteria.

Red-Dim	Subspace Rank Reduction Criteria
500	rank $>$ 10 reduce to 10 else the original rank
1000	rank $>$ 20 reduce to 15 else the original rank
2000	rank $>$ 100 reduce to 30 else the original rank
3000	rank $>$ 100 reduce to 50 else the original rank

Table 11: Performance on reduced word topic vectors using several reduction techniques on various dimensions for multi-label classification – the Reuters dataset.

Method	Dimen	Prec@1 nDCG	Prec @5	nDCG @5	Cover Error	LRAP	F1 Score
Auto Encoder	500	92.14	36.53	48.74	8.02	91.34	79.96
	1000	92.95	36.82	49.17	7.14	92.32	81.05
	2000	93.39	36.95	49.39	6.87	92.75	81.65
	3000	**93.56**	**37**	**49.47**	**6.74**	**92.96**	**81.94**
Random Projection	500	91.98	36.26	48.47	7.41	91	79.03
	1000	92.59	36.62	48.91	6.98	91.84	80.39
	2000	**93.39**	36.83	49.26	6.95	92.59	81.12
	3000	93.32	**36.91**	**49.33**	**6.78**	**92.75**	**81.39**
PCA (SubSpace)	500	90.73	36.03	48.06	8.40	90.11	78.06
	1000	92.15	36.55	48.78	7.44	91.48	80
	2000	92.95	36.87	49.22	6.86	92.30	81.2
	3000	**93.38**	**36.97**	**49.4**	**6.69**	**92.8**	**81.48**

Table 12: Time complexity for dimensionality reduction of word topic vectors to a 2000-dimension dense representation using various reduction techniques on 20NewsGroup

Method	Random Projection	PCA (SubSpace)	Autoencoder
Time (sec)	35	66	608

in Reuters (250) is significantly fewer than 20NewsGroup (1000); thus each word is assigned to only one cluster. Therefore, the use of AdaGram for sense disambiguation and the sparsity operation over the word-cluster assignments does not improve the performance by a large margin. We verified this claim in the ablation study below. We can conclude that our modifications yield notable improvements if the dataset has more multi-sense words.

Ablation Study: To understand the contributions of each of the three modifications, we compared five different versions of our embeddings. In the first version, we ablated the sparsity of the word-cluster assignments and applied sparsity directly on the document vectors similar to SCDV while keeping the Doc2VecC multi-sense word embeddings and the sense annotated corpus intact. In the second version, we ablated the Doc2VecC embeddings with normal SGNS embedding while keeping the word topic vector sparsity, and the sense annotated corpus intact. In the third version, we ablated multi-sense embeddings and the annotation of the corpus while keeping the Doc2VecC word training and the word topic vector sparsity intact. We also compared our results with an all ablation approach, i.e., the SCDV baseline and a none ablation approach, i.e., our new embeddings in SCDV-MS. Table 6 shows the results obtained with ablation on 20NewsGroup and Reuters datasets. We obtained the best performance with the none ablation approach, i.e., SCDV-MS. Thus, we can conclude that all three modifications is needed to yield the best performance. Multi-sense is the most pivotal improvement for 20newsgroup since by ablating it we observed the lowest performance out of ablating each of the three modifications. Whereas on the Reuters dataset, multi-sense was the least important because of fewer multi-sense words. On Reuters, the noise removal at word level representations was the most important.

Comparison with Contextual Embeddings: SCDV-MS is a lot

simpler than unsupervised contextual embeddings like BERT (pr) and ELMo, but it outperformed them. We presume that SCDV-MS's concentration on capturing semantics (local and global) in sparse high dimension representations instead of capturing both semantics and syntax in single lower dimensional continuous representations (what BERT does) is the reason behind our method's superior performance. Because understanding syntax is not as influential as semantics in our classification and similarity tasks.

5.2 Lower Dimensional Manifold Learning

We tried three popular lower-dimensional manifold learning algorithms to reduce high dimensional sparse word topic vectors to lower-dimensional word embeddings, namely Random Projection [2], PCA (Subspace) [1], and Autoencoders [48]. For PCA (Subspace), we observed that not all subspaces (200 dimensions) of word topic vectors have a complete full rank (200). Most of the subspaces were of ranks much smaller than 200 (rank $\leq$ 200). Therefore, we applied PCA on each of 200 dimension subspaces separately and concatenated the subspace reduced vectors. We refer to Table 10 as the criteria for PCA-based subspace rank reduction. For Auto-encoders, we used the standard architecture for reducing the word topic vectors from 12000 to 2000 dimensions, through a intermediate layer of 4000 dimensions (see Figure 7). We used the mean squared error minimization, $\tanh$ non-linear activation, and Adam optimization routine for training the autoencoders. We used the initial learning rate of 0.001, $\beta_1 = 0.9$ and $\beta_2 = 0.999$ for Adam optimization routine.

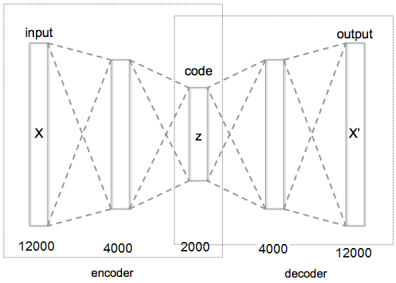

Figure 7: The AutoEncoder architecture ($12000 - 4000 - 2000 - 4000 - 12000$) used to reduce the word topic vectors from 12000 dimensions to 2000 dimensions.

Results: Table 7 shows the performance of the dimensionality reduction techniques such as Random Projection, PCA (Subspace), and Autoencoders on the 20NewsGroup dataset. We observed that autoencoders outperform other reduction methods because they easily fit any non-linear function. Also, we observed only a small decrease of 1% in the performance after a reduction to 2000 dimensions with most methods. This decrease is associated to loss due to data compression and can be explained by information theory. However, word topic vectors can be efficiently projected into a lower-dimensional manifold without a significant loss in the performance. We compared the percentage loss in performance (F1-Score) $\%\text{RL} = \frac{\text{Orignal} - \text{Reduced}}{\text{Orignal}}$ on text classification with a dimension-reduced $\vec{wtv}$ through random projection for both SCDV and SCDV-MS on 20newsgroup. In Figures 4, 5 and 6 we observe that the $\%RL$ loss in SCDV-MS's F1-Score is distinctively less compared to SCDV for all reduction methods. Furthermore, we observed that the reduction time for SCDV-MS was shorter than SCDV, particularly for random projection, because SCDV-MS $\vec{wtv}$ are sparser than SCDV $\vec{wtv}$. We also tried a direct reduction of final document representations which yielded a poor performance and took a longer reduction time for both embeddings. Overall, reducing SCDV-MS $\vec{wtv}$ is much more effective than reducing SCDV $\vec{wtv}$ or document vectors. Similar observations for a multi-label classification task on the Reuters dataset, see Table 11.

Application to Deep Learning: One significance of our reduction of $\vec{wtv}$ is that they can be used as direct word embeddings in popular deep learning architectures such as CNNs on downstream classification tasks. We used the same architecture (see the supplementary material, Figure 8 [16]) for both embeddings. Employing CNN, our results outperformed the original word embeddings of the same dimension for the 20NewsGroup classification task, shown in Table 8.

5.3 Time and Space Complexity

Table 9 illustrates empirical results for time and space complexities on SCDV (12000 dimensions), SCDV-MS (12000 dimensions) and reduced R-SCDV-MS (2000 dimensions).

Feature Formation Time: Due to the direct thresholding of word cluster assignments in SCDV-MS, the word topic vectors ($\vec{wtv}$) are extremely sparse. SCDV-MS $\vec{wtv}$ has only 2% of active attributes (98% sparse), whereas SCDV $\vec{wtv}$ has 99% of active attributes (only 1% sparse). Therefore, we can use an efficient sparse operation (sparse addition and multiplication) over sparse vectors to speedup feature formation. We observed that by adding sparsity we can reduce the feature formation time by a significant factor of 43.

Overall Prediction Time: Overall, due to enhanced feature formation and reduced $\vec{wtv}$ loading time, SCDV-MS will predict faster compared to the original SCDV. However, we observed an insignificant difference in the prediction time and model size as SCDV sparsifies the final document vectors (both equally sparse). Furthermore, after reducing the SCDV-MS $\vec{wtv}$ to 2000-dense dimension features using auto-encoders, we observed a distinctive reduction of 8.5 times in the prediction time. One can directly store the reduced $\vec{wtv}$ for the complete vocabulary instead of the reduction model. Refer to Table 12 for SCDV-MS $\vec{wtv}$ reduction timing. Furthermore, one can directly also reduce the words appearing in the documents i.e. use a real-time reduction model during prediction.

Vector Space Complexity, Training Time, and Model Size: We only require 0.1 of the original space to store sparse $\vec{wtv}$. We achieved these improvements despite having 1.63 times of the size of the original vocabulary due to multi-sense word embeddings. The projected $\vec{wtv}$ is 3 times larger than the $\vec{wtv}$ of SCDV-MS; however, it is 3.7 times smaller than the $\vec{wtv}$ in SCDV. SCDV is marginally (1.1 times) sparser than SCDV-MS due to manual thresholding of document vectors. SCDV-MS also has a slightly faster training time compared to the original SCDV. We also observed that our training model on reduced vectors (R-SCDV-MS) is 6 times smaller than SCDV, and the training process is 1.25 times faster than SCDV. In comparision to pre-trained BERT which uses 12-layer-768-hidden - 12-heads in total 110M parameters which amount to 16GB RAM size, our model, including embeddings, uses 80MB of total RAM.

6 Conclusion

In this paper, we proposed several novel modifications to overcome the shortcomings of SCDV, a state-of-the-art document embedding method. Our proposed SCDV-MS, outperformed the previous embedding (including SCDV and BERT (pr)) on the downstream tasks

[16] https://bit.ly/33473wk

of text classification. Overall, we have shown that disambiguating multi-sense words based on context words (adjacent words) can lead to better document representations. Sparsity in representation is helpful for effective and efficient lower-dimensional manifold representation learning. Representation noise in words level can have a significant impact on the downstream tasks.

REFERENCES

[1] Hervé Abdi and Lynne J Williams, 'Principal component analysis', *Wiley interdisciplinary reviews: computational statistics*, (2010).
[2] Dimitris Achlioptas, 'Database-friendly random projections: Johnson-lindenstrauss with binary coins', *Journal of computer and System Sciences*, (2003).
[3] Hadi Amiri and Mitra Mohtarami, 'Vector of locally aggregated embeddings for text representation', *NAACL 2019*, (2019).
[4] Sanjeev Arora, Yingyu Liang, and Tengyu Ma, 'A simple but tough-to-beat baseline for sentence embeddings', *ICLR*, (2017).
[5] Ben Athiwaratkun, Andrew Gordon Wilson, and Anima Anandkumar, 'Probabilistic fasttext for multi-sense word embeddings', *arXiv preprint arXiv:1806.02901*, (2018).
[6] Sergey Bartunov, Dmitry Kondrashkin, Anton Osokin, and Dmitry Vetrov, 'Breaking sticks and ambiguities with adaptive skip-gram', in *Artificial Intelligence and Statistics*, (2016).
[7] Kush Bhatia, Himanshu Jain, Purushottam Kar, Manik Varma, and Prateek Jain, 'Sparse local embeddings for extreme multi-label classification', in *NeurIPS*, (2015).
[8] David M. Blei, David M. Ng, Michael I. Jordan, and John Lafferty, 'Latent dirichlet allocation', *JMLR*, **3**, 2003, (2003).
[9] Minmin Chen, 'Efficient vector representation for documents through corruption', *5th ICLR*, (2017).
[10] Dat Quoc Nguyen Dai Quoc Nguyen, Ashutosh Modi, Stefan Thater, and Manfred Pinkal, 'A mixture model for learning multi-sense word embeddings', *ACL 2017*, (2017).
[11] Rajarshi Das, Manzil Zaheer, and Chris Dyer, 'Gaussian lda for topic models with word embeddings', in *ACL*. ACL, (2015).
[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, 'Bert: Pre-training of deep bidirectional transformers for language understanding', *arXiv preprint arXiv:1810.04805*, (2018).
[13] Adji B Dieng, Francisco JR Ruiz, and David M Blei, 'The dynamic embedded topic model', *arXiv preprint arXiv:1907.05545*, (2019).
[14] Adji B Dieng, Francisco JR Ruiz, and David M Blei, 'Topic modeling in embedding spaces', *arXiv preprint arXiv:1907.04907*, (2019).
[15] Xianghua Fu, Ting Wang, Jing Li, Chong Yu, and Wangwang Liu, 'Improving distributed word representation and topic model by word-topic mixture model', in *Proceedings of The 8th Asian Conference on Machine Learning*, (2016).
[16] Felix A Gers, Nicol N Schraudolph, and Jürgen Schmidhuber, 'Learning precise timing with lstm recurrent networks', *JMLR*, **3**(Aug), (2002).
[17] Vivek Gupta, Harish Karnick, Ashendra Bansal, and Pradhuman Jhala, 'Product classification in e-commerce using distributional semantics', in *COLING 2016*, (2016).
[18] Vivek Gupta, Ankit Kumar Saw, Partha Pratim Talukdar, and Praneeth Netrapalli. Unsupervised document representation using partition word-vectors averaging, 2019.
[19] Zellig Harris, 'Distributional structure', *Word*, **10**, 146–162, (1954).
[20] Peng Jin, Yue Zhang, Xingyuan Chen, and Yunqing Xia, 'Bag-of-embeddings for text classification', in *International Joint Conference on Artificial Intelligence*, number 25, (2016).
[21] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom, 'A convolutional neural network for modelling sentences', in *Proceedings of the 52nd Annual Meeting of the ACL*, volume 1, (2014).
[22] Han Kyul Kim, Hyunjoong Kim, and Sungzoon Cho, 'Bag-of-concepts: Comprehending document representation through clustering words in distributed representation', *Neurocomputing*, (2017).
[23] Yoon Kim, 'Convolutional neural networks for sentence classification', in *EMNLP*, (2014).
[24] Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler, 'Skip thought vectors', *NeurIPS*, (2015).
[25] Jarvan Law, Hankz Hankui Zhuo, Junhua He, and Erhu Rong, 'Ltsg: Latent topical skip-gram for mutually learning topic model and vector representations', *arXiv preprint arXiv:1702.07117*, (2017).
[26] Quoc V Le and Tomas Mikolov, 'Distributed representations of sentences and documents.', in *ICML*, volume 14, (2014).
[27] Jiwei Li and Dan Jurafsky, 'Do multi-sense embeddings improve natural language understanding?', *arXiv preprint arXiv:1506.01070*, (2015).
[28] Shaohua Li, Tat-Seng Chua, Jun Zhu, and Chunyan Miao, 'Generative topic embedding: a continuous representation of documents', in *ACL*. ACL, (2016).
[29] Ximing Li, Jinjin Chi, Changchun Li, Jihong Ouyang, and Bo Fu, 'Integrating topic modeling with word embeddings by mixtures of vmfs', *COLING 2016*, (2016).
[30] Pengfei Liu, Xipeng Qiu, and Xuanjing Huang, 'Learning context-sensitive word embeddings with neural tensor skip-gram model.', in *IJCAI*, (2015).
[31] Yang Liu, Zhiyuan Liu, Tat-Seng Chua, and Maosong Sun, 'Topical word embeddings.', in *Proceedings of AAAI 2015*, (2015).
[32] Dheeraj Mekala, Vivek Gupta, Bhargavi Paranjape, and Harish Karnick, 'Scdv: Sparse composite document vectors using soft clustering over distributional representations', in *EMNLP*, (2017).
[33] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur, 'Recurrent neural network based language model', in *Eleventh Annual Conference of the International Speech Communication Association*, (2010).
[34] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean, 'Distributed representations of words and phrases and their compositionality', in *NeurIPS*, (2013).
[35] Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig, 'Linguistic regularities in continuous space word representations.', in *NAACL*, (2013).
[36] Jeff Mitchell and Mirella Lapata, 'Vector-based models of semantic composition', in *Proceedings of ACL-08: HLT*, Columbus, Ohio, (June 2008). ACL.
[37] Jeff Mitchell and Mirella Lapata, 'Composition in distributional models of semantics', *Cognitive science*, (2010).
[38] Christopher E Moody, 'Mixing dirichlet topic models and word embeddings to make lda2vec', *arXiv preprint arXiv:1605.02019*, (2016).
[39] Dat Quoc Nguyen, Richard Billingsley, Lan Du, and Mark Johnson, 'Improving topic models with latent feature word representations', *TACL*, **3**, (2015).
[40] Liqiang Niu, Xinyu Dai, Jianbing Zhang, and Jiajun Chen, 'Topic2vec: learning distributed representations of topics', in *IALP*. IEEE, (2015).
[41] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer, 'Deep contextualized word representations', in *Proc. of NAACL*, (2018).
[42] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer, 'Deep contextualized word representations', in *Proc. of NAACL*, (2018).
[43] Andreas Rücklé, Steffen Eger, Maxime Peyrard, and Iryna Gurevych, 'Concatenated p-mean word embeddings as universal cross-lingual sentence representations', *arXiv preprint arXiv:1803.01400*, (2018).
[44] Pranjal Singh and Amitabha Mukerjee, 'Words are not equal: Graded weighting model for building composite document vectors', in *ICON-2015*, (2015).
[45] Roberta A Sinoara, Jose Camacho-Collados, Rafael G Rossi, Roberto Navigli, and Solange O Rezende, 'Knowledge-enhanced document embeddings for text classification', *Knowledge-Based Systems*, **163**, 955–971, (2019).
[46] Paul Smolensky, 'Tensor product variable binding and the representation of symbolic structures in connectionist systems', *Artificial intelligence*, (1990).
[47] Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, Christopher Potts, et al., 'Recursive deep models for semantic compositionality over a sentiment treebank', in *EMNLP*, (2013).
[48] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol, 'Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion', *JMLR*, (Dec), (2010).
[49] John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu, 'Towards universal paraphrastic sentence embeddings', *ICLR*, (2016).
[50] John Wieting, Mohit Bansal, Kevin Gimpel, Karen Livescu, and Dan Roth, 'From paraphrase database to compositional paraphrase model and back', *Transactions of the ACL*, **3**, (2015).