P-SIF: Document Embeddings Using Partition Averaging

Vivek Gupta[1,3], **Ankit Saw**[2], **Pegah Nokhiz**[1], **Praneeth Netrapalli**[3], **Piyush Rai**[4], **Partha Talukdar**[5]
[1]School of Computing, University of Utah, [2]Info Edge (India) Limited, [3]Microsoft Research Lab, Bangalore,
[4]Computer Science Department, IIT Kanpur, [5]Indian Institute of Science, Bangalore
vgupta@cs.utah.edu, ankit.kgpian@gmail.com, pnokhiz@cs.utah.edu,
praneeth@microsoft.com, piyush@cse.iitk.ac.in, ppt@iisc.ac.in

Abstract

Simple weighted averaging of word vectors often yields effective representations for sentences which outperform sophisticated seq2seq neural models in many tasks. While it is desirable to use the same method to represent documents as well, unfortunately, the effectiveness is lost when representing long documents involving multiple sentences. One of the key reasons is that a longer document is likely to contain words from many different topics; hence, creating a single vector while ignoring all the topical structure is unlikely to yield an effective document representation. This problem is less acute in single sentences and other short text fragments where the presence of a single topic is most likely. To alleviate this problem, we present P-SIF, a partitioned word averaging model to represent long documents. P-SIF retains the simplicity of simple weighted word averaging while taking a document's topical structure into account. In particular, P-SIF learns topic-specific vectors from a document and finally concatenates them all to represent the overall document. We provide theoretical justifications on the correctness of P-SIF. Through a comprehensive set of experiments, we demonstrate P-SIF's effectiveness compared to simple weighted averaging and many other baselines.

Introduction

Many approaches such as (Socher et al. 2013; Liu, Qiu, and Huang 2015; Le and Mikolov 2014; Ling et al. 2015) are proposed which go beyond words to capture the semantic meaning of sentences. These techniques either use the simple composition of the word-vectors or sophisticated neural network architectures for sentence representation. Recently, (Arora, Liang, and Ma 2017) proposed a smooth inverse frequency (SIF) based word vector averaging model to embed a sentence. They further improved their embedding by removing the first principal component of the weighted average vectors. However, all these approaches are limited to capturing the meaning of a single sentence and representing the sentence in the same space as words, thus reducing their expressive power. Generally, longer texts contain words from multiple topics, so creating a single vector from simple averaging of word-vectors will disregard all the topical structure. [1] Hence, these techniques are largely unable to capture the semantic meanings of larger pieces of text, e.g., multi-sentence documents.

To address these limitations, we present a novel document embedding method called *partition SIF* weighted averaging (P-SIF) to embed documents which usually contain multiple sentences efficiently. P-SIF learns topic-specific vectors from a document and finally concatenates them all to represent the overall document. Thus, P-SIF retains the simplicity of simple weighted word averaging while taking a document's topical structure into account. We also provide theoretical justifications for the proposed approach and demonstrate its efficacy via a comprehensive set of experiments. P-SIF achieves significant improvements over several embedding techniques on several tasks despite being simple. We have released the source code for P-SIF embeddings. [2] The novel characteristics of P-SIF are described below:

- P-SIF can embed larger multi-sentence documents, as it pays attention to the topical structure of the document.
- P-SIF is based on simple weighted word vectors averaging rather than considerably more sophisticated tensor factorization or neural network-based methods.
- P-SIF is unsupervised since it only uses pre-trained word embeddings without using any label information.
- P-SIF outperforms many existing methods on text similarity, text classification, and other supervised tasks.

Related Work

Most of the prior work has computed sentence embeddings by coordinate wise vector and matrix-based compositional operations over word vectors, e.g., (Levy and Goldberg 2014) use unweighted averaging of word vectors(Le and Mikolov 2014) for representing short phrases, (Singh and Mukerjee 2015) propose tfidf-weighted averaging of word vectors to form document vectors, (Socher et al. 2013) propose a recursive neural network defined over a parse tree, and trained with supervision.

[1] Topical structure denotes word distribution across topics.
[2] https://github.com/vgupta123/P-SIF

Next, (Le and Mikolov 2014) propose *PV-DM* and *PV-DBOW* models which treat each sentence as a shared global latent vector. Other approaches use seq2seq models such as Recurrent Neural Networks (Mikolov et al. 2010) and Long Short Term Memory (Gers, Schraudolph, and Schmidhuber 2002) which can handle long term dependency, hence capturing the syntax structure. Other neural network models include the use of hierarchy and convolutional neural networks such as (Kim 2014). (Wieting et al. 2015) learns paraphrastic sentence embeddings by modifying word embeddings via supervision from the Paraphrase pairs dataset (PPDB) (Ganitkevitch, Van Durme, and Callison-Burch 2013).

Recently, a lot of work is harnessing topic modeling (Blei et al. 2003) along with word vectors to learn better word and sentence representations, e.g., LDA (Chen and Liu 2014), weight-BoC (Kim, Kim, and Cho 2017), TWE (Liu et al. 2015) , NTSG (Liu, Qiu, and Huang 2015), WTM (Fu et al. 2016), w2v-LDA (Nguyen et al. 2015), TV+MeanWV (Li et al. 2016a), LTSG (Law et al. 2017), Gaussian-LDA (Das, Zaheer, and Dyer 2015), Topic2Vec (Niu et al. 2015), TM (Dieng, Ruiz, and Blei 2019b), LDA2vec (Moody 2016), D-ETM (Dieng, Ruiz, and Blei 2019a) and MvTM (Li et al. 2016b). (Kiros et al. 2015) propose skip-thought document embedding vectors which transformed the idea of abstracting the distributional hypothesis from word to sentence level. (Wieting et al. 2016) propose a neural network model which optimizes the word embeddings based on the cosine similarity of the sentence embeddings. Moreover, several recent deep contextual word embeddings such as ELMo (Peters et al. 2018), USE (Cer et al. 2018) and BERT (Devlin et al. 2019) are proposed. These contextual embeddings are state-of-the-art on multiple tasks as they effectively capture the surrounding contexts.

(Gupta et al. 2016) propose methods which employ a clustering-based technique and tf-idf values to form a composite document vector extending the Bag-of-Words (BoW) model (Harris 1954). They represent documents in higher dimensions by using hard clustering over word embeddings. (Mekala et al. 2017) extend this by proposing SCDV using an overlapping clustering technique and direct idf weighting of word vectors. The learned representations try to capture a global context of a sentence, similar to an n-gram model. Our method is the same in essence, but is based on topic-based partitioning; moreover, unlike (Mekala et al. 2017)'s approach, our method is supported by theoretical guarantees.

Averaging vs Partition Averaging

Figure 1, represents the word-vector space, where similar meaning words occur closer to each other. We can apply sparse coding to partition the word-vector space to a five topic vector space. These five topic vector spaces represent the five topics present in corpus. Some words are multi-sense and belong to multiple topics with some proportion. In Figure 1 we represent words' topic number in subscript and proportion in brackets. Let's consider a document d_n: *"Data journalists deliver data science news to general public. They often take part in interpreting the data models. In addition, they create graphical designs and interview the directors and CEOs."*

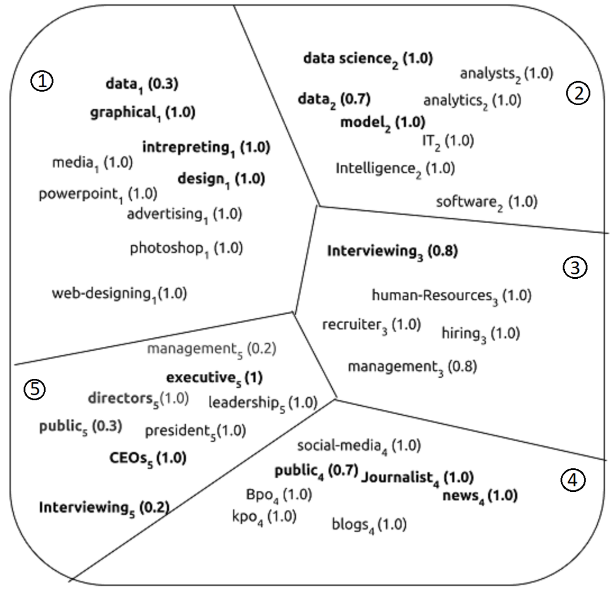

Figure 1: Words in different partitions are represented by different subscripts and separated by hyper-planes. Bold fonts represent words' presence in document d_n.

If we directly average words to represent document ($\vec{v}_{d_n}$), as is done in SIF (Arora, Liang, and Ma 2017), then different semantic meaning words, e.g., words in partition 1 such as *'graphical'*, *'design'*, and *'data'* are averaged with words of different semantic meaning of partition 2 such as *'data science'*, *'model'*, and *'data'*. In addition, the document is represented in the same d dimensional space as word vectors. Overall, averaging represents the documents as a single point in the vector space and does not consider the 5 different semantic topics. However, we can weight (topic proportion) average of words within a partition and concatenate ($\oplus$) the average word vectors across partitions to represent document ($\vec{v}_{d_n}$), as is done in our proposed method P-SIF. By doing this, words belonging to different semantic topics are separated by concatenation ($\oplus$) as they represent separate meanings, whereas words in similar topics are simply averaged since they represent the same meaning. For example, average of words belonging to partition 1 such as *'graphical'*, *'design'*, and *'data'* are concatenated to average of words in partition 2 such as *'data science'*, *'model'*, and *'data'*. The final document vector $\vec{v}_{d_n}$ is represented in a higher $5 \times d$ dimension vector space, thus having more representational power (d is the dimension of word vectors). Overall, the 5 different semantic topics are taken into account for representation. Additionally, this representation also takes the weight according to which each word belongs to various topics into account, meaning it handles words' multi-sense natures. For example, *'data'* belongs to partition 1 with probability 0.3 and partition 2 with probability 0.7. Hence, partitioned averaging with topic weighting is essential for representing longer text documents.

The Proposed Algorithm: P-SIF

In this section, we present the new proposed document embedding learning method in algorithm 1. The feature formation algorithm can be divided into three major steps:

Sparse Dictionary Learning for Word Vectors (Algo 1: Lines 1 - 3): Given word vectors $v_w \in R^d$, a sparsity parameter k, and an upper bound K , we find a set of unit norm vectors $\vec{A}_1, \vec{A}_2, \ldots, \vec{A}_K$, such that $\vec{v}_w = \sum_{j=1}^{K} \alpha_{(w,j)}\vec{A}_j + \vec{\eta}_w$ where at most k out of K of the coefficients $\alpha_{(w,1)}, \ldots, \alpha_{(w,K)}$ are nonzero (so-called sparsity constraint), and $\vec{\eta}_w$ is a noise vector. Sparse coding is usually solved for a given K and k by using alternating minimization such as k-svd (Aharon et al. 2006) to find the $\vec{A}_i'$s that minimize the following L_2-reconstruction error: $\|\vec{v}_w - \sum_{j=1}^{K} \alpha_{(w,j)}\vec{A}_j\|$. (Arora et al. 2016b) show that multiple senses of a word reside as a linear superposition within the word embedding and can be recovered by simple sparse coding. Therefore, one can use the sparse coding of word vectors to detect multiple senses of words. Additionally, the atoms of sparse coding $(\vec{A}_1, \ldots, \vec{A}_K)$ over word-vectors $(\vec{v}_w)$ represent all prominent topics in the corpus. For a given word w, the k non-zero coefficient of α_w essentially represents the distribution of words over topics. Furthermore, restricting K to be much smaller than the number of the words ensures that the same topic needs to be used for multiple words. The learned $\vec{A}_j$ is a significant topic because the sparse coding ensures that each basis element is softly chosen by many words.

Sparse Dictionary Learning vs. Overlapping Clustering: Sparse coding can also be treated as a linear algebraic analogue of overlapping clustering, where the $\vec{A}_i$'s act as cluster centers and each $\vec{v}_w$ is assigned to each cluster in a soft way (using the coefficients $\alpha_{(w,j)}$, of which only k out of K are non-zero) to a linear combination of at most k clusters. In practice, sparse coding optimization produces coefficients $\alpha_{(w,j)}$ which are almost all positive, even though *unconstrained*. One can use overlapping clustering where each word belongs to every cluster with some probability $P(c_k|w_i)$ which can be thought of as a substitute for $\alpha_{(w,k)}$, similar to the approach in SCDV (Mekala et al. 2017). Instead of GMM, we use a dictionary learning-based approach which imposes a sparsity constraint implicitly during optimization through regularization. Additionally, such high dimensional data structure regularizers, e.g., sparse encodings, help in overcoming the curse of high dimensionality. For single-sentence documents with a small number of topics, it is better to use overlapping clustering because of an easier unconstrained optimization. However, in case of multi-sentence documents where the number of topics is large, dictionary learning performs better than overlapping clustering due to 1) Sparse constraint optimization forces non-redundant clusters (minimally sufficient #clusters) and 2) The sparse constraint diminishes the noise from the long tail of word-cluster assignments $P(c_k|w_i)$ (Olshausen and Field 1997; Gao et al. 2010; Yang et al. 2009).

Word Topics Vector Formation (Algo 1: Lines 4 - 9): For single sentence documents all words of a document belong to a single topic. However, for multi-sentence documents, words of a document generally originate from multiple topics. To capture this, topic modeling algorithms such as LDA (Blei et al. 2003) are used to represent the documents. These representations essentially represent the global contexts of the documents as a distribution over topics. However, these representations do not take the local context initiating from the distributional semantics such as word vectors into account. Since our multi-sentence documents have words from multiple topics, a simple averaging technique will not work. Hence, we concatenate the word embeddings over words' topic distributions. This helps to represent semantically similar words in the same topic, while words which are semantically different are represented in different topics. Concatenation of word embeddings over topics also helps in the expression of words' multi-sense nature. For each word $\vec{w}$, we create K different word-cluster vectors of d dimensions $\vec{cv}_{wk}$ by weighting the word embedding with its learned dictionary coefficient $\alpha_{w,k}$ of the k^{th} context. [3] We then concatenate all the K word-cluster vectors $\vec{cv}_{wk}$ into a $K \times d$ dimensional embedding to form a word-topic vector $\vec{tv}_w \in R^{K\times d}$. We weigh word-vectors by coefficients of the learned dictionary to capture the cross correlation $(\alpha_i\alpha_j)$ between topics. The word-topic-vector $\vec{tv_w}$, which we average to represent documents, captures both local and global semantics.

SIF Weight Averaging and Common Component Removal (Algo 1: Lines 10 - 16): Finally, for all words appearing in document D_n, we weight the word-topics vectors $\vec{tv}_i$ by smooth inverse frequency $\left(\frac{a}{a+p(w)}\right)$. Next, we remove the common contexts from the weighted average document vectors by removing the first principal component from the weighted average vectors. [4] Common component removal reduces the noise and redundancy from the document vectors which makes the representations more discriminating. (Arora, Liang, and Ma 2017) empirically shows that SIF weighting outperforms the tf-idf weighting. However, they only use simple averaging to represent a sentence. Detailed code architecture of P-SIF is in the supplementary material. [5]

Derivation of P-SIF Embeddings : We provide theoretical justifications by showing connections of P-SIF with random-walk based latent variable models (Arora, Liang, and Ma 2017; Arora et al. 2016a; 2016b). Full derivations are provided in the supplementary material [5].

Kernels meet Embeddings

In this section, we present one of the novelties of this work where we show that many common sentence embeddings can be represented as similarity kernels over word and topic vectors. Let D_A and D_B represent two documents containing n and m words respectively. $w_1^A, w_2^A \ldots w_n^A$ denotes D_A's words and $w_1^B, w_2^B \ldots w_m^B$ denotes D_B's words.

[3] Empirically, we observed that this weighting generally improves the performance. [4] We did not remove the common component from final vectors when we used Doc2VecC-initialized (Chen 2017) word vectors with P-SIF. Because frequent words' word-vectors become close to $\vec{0}$.

[5] https://vgupta123.github.io/docs/appendix_aaai2020.pdf

Algorithm 1: P-SIF Embedding

Data: d dimensional Word embeddings $\{\vec{v}_w : w \in V\}$ where word w is in vocabulary V. Documents $\{d_n : d_n \in D\}$, a set of sentences D in corpus C, parameter a and estimated unigram probability $\{p(w) : w \in V\}$ of word w in C, a sparsity parameter k, and an upper bound K.

Result: Document vectors $\{\vec{v}_{d_n} : d_n \in D\}$

/* Dictionary learning on word-vectors */

for *each word* w *in* V **do**

  $\vec{v}_w = \sum_{j=1}^{K} \alpha_{w,j} \vec{A}_j + \vec{\eta}_w$;

end

/* Word topic-vector formation */

for *each word* w *in* V **do**

  for *each coefficient,* $\alpha_{w,k}$ *of word* w **do**

    $\vec{cv}_{w,k} \leftarrow \vec{v}_w \times \alpha_{w,k}$;

  end

  $\vec{tv}_w \leftarrow \bigoplus_{k=1}^{K} \vec{cv}_{wk}$;

  /* $\bigoplus$ is concatenation, $\times$ is scalar vector multiplication */

end

/* SIF reweighed embedding */

for *each document* d_n *in* D **do**

  $\vec{v}_{d_n} \leftarrow \frac{1}{|d_n|} \sum_{w \in d_n} \frac{a}{a+p(w)} \vec{tv}_w$;

end

Form a matrix X whose columns are $\{\vec{v}_{d_n} : d_n \in D\}$, and let $\vec{u}$ be the first singular vector;

for *each document* $d_n \in D$ **do**

  $\vec{v}_{d_n} \leftarrow \vec{v}_{d_n}$ - $\vec{u}\vec{u}^T \vec{v}_{d_n}$;

end

1. Simple Word Vector Averaging : $K^1(D_A, D_B) = \frac{1}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} \langle \vec{v}_{w_i^A} \cdot \vec{v}_{w_j^B} \rangle$
2. TWE: Topical Word Embeddings (Liu et al. 2015) : $K^2(D_A, D_B) = \frac{1}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} \langle \vec{v}_{w_i^A} \cdot \vec{v}_{w_j^B} \rangle + \langle \vec{tv}_{w_i^A} \cdot \vec{t}_{w_j^B} \rangle$
3. P-SIF: Partition Word Vector Averaging (Our approach) : $K^3(D_A, D_B) = \frac{1}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} \langle \vec{v}_{w_i^A} \cdot \vec{v}_{w_j^B} \rangle \times \langle \vec{t}_{w_i^A} \cdot \vec{t}_{w_j^B} \rangle$
4. Relaxed Word Mover Distance (Kusner et al. 2015) : $K^4(D_A, D_B) = \frac{1}{n} \sum_{i=1}^{n} \max_j \langle \vec{v}_{w_i^A} \cdot \vec{v}_{w_j^B} \rangle$

Here, $\vec{v}_w$ represents the word vector of word w and $\vec{t}_w = \alpha_w \in R^K$ represents the topic vector of word w, where K is the number of topics. $\langle \vec{a} \cdot \vec{b} \rangle$ represents the dot product of two vectors $\vec{a}$ and $\vec{b}$. $c \times d$ represents the scalar product of c and d. $\bigoplus$ represents the row-wise concatenation of the vectors. Refer to the Supplementary material [5] for the detailed proof.

Experimental Results

We perform a comprehensive set of experiments on several text similarity and multiclass or multilabel text classification tasks. Due to limited space, some details on experiments are in the Supplementary material [5].

Textual Similarity Task

Datasets and Baselines: We perform our experiments on the SemEval dataset (2012 - 2017). These experiments involve 27 semantic textual similarity (STS) tasks (2012 - 2016) (Agirre et al. 2012; 2016) the SemEval 2015 Twitter task (Xu, Callison-Burch, and Dolan 2015), and the SemEval 2014 Semantic relatedness task (Marelli et al. 2014). The objectives of these tasks are to predict the similarity between two sentences. We compare our approach with several unsupervised, semi-supervised and supervised embedding baselines mostly taken from (Arora, Liang, and Ma 2017; Wu et al. 2018; Ethayarajh 2018). Details on the baselines are listed below:

Unsupervised: We used ST, avg-GloVe, tfidf-GloVe, and GloVe + WR as a baseline. ST denotes the skip-thought vectors by (Kiros et al. 2015), avg-GloVe denotes the unweighted average of the GloVe Vectors by (Pennington, Socher, and Manning 2014) [6], and tfidf-Glove denotes the tf-idf weighted average of GloVe vectors. We also compared our method with the SIF weighting (W) common component removal (R) GloVe vectors (GloVe + WR) by (Arora, Liang, and Ma 2017). For STS 16, we also compared our embedding with Skip-Thoughts (Kiros et al. 2015), BERT pretrained embedding average (Devlin et al. 2019) , Universal Sentence Encoder (Cer et al. 2018) and Sent2Vec (Pagliardini, Gupta, and Jaggi 2018) embeddings.

Semi-Supervised: We used avg-PSL, PSL + WR, and the avg-PSL used the unweighted average of the PARAGRAM-SL999 (PSL) word vectors by (Wieting et al. 2015) as a baseline, obtained by training on PPDB dataset(Ganitkevitch, Van Durme, and Callison-Burch 2013). The word vectors are trained using unlabeled data. Furthermore, sentence embeddings are obtained from unweighted word vectors averaging. We also compared our method with the SIF weighting (W) common component removal (R) PSL word vectors (PSL + WR) by (Arora, Liang, and Ma 2017).

Supervised: We compared our method with PP, PP-proj., DAN, RNN, iRNN, LSTM (o.g), LSTM (no) and GRAN. All these methods are initialized with PSL word vectors and then trained on the PPDB dataset (Ganitkevitch, Van Durme, and Callison-Burch 2013). PP (Wieting et al. 2016) is the average of word vectors, while PP-proj is the average of word vectors followed by a linear projection. The word vectors are updated during the training. DAN denotes the deep averaging network (Iyyer et al. 2015). RNN is a Recurrent neural network, iRNN is the identity activated Recurrent Neural Network based on identity-initialized weight matrices. The LSTM is the version from (Gers, Schraudolph, and Schmidhuber 2002), either with output gates (denoted as LSTM (o.g.)) or without (denoted as LSTM (no)). GRAN denotes state of the art supervised averaging based Gated Recurrent Averaging Network from (Wieting and Gimpel 2017). For STS 16 we also compared our embedding with Tree-LSTM

[6] We used the 300-dimensional word vectors that are publicly available at http://nlp.stanford.edu/projects/glove/.

(Tai, Socher, and Manning 2015) embedding.

Experimental Settings: We use the Pearson's coefficient between the predicted and the ground-truth scores for the evaluation. We use the PARAGRAM-SL999 (PSL) from (Wieting et al. 2015) as word embeddings, obtained by training on the PPDB (Ganitkevitch, Van Durme, and Callison-Burch 2013) dataset [7]. We use the fixed weighting parameter a value of 10^{-3}, and the word frequencies $p(w)$ are estimated from the common-crawl dataset. We tune the number of contexts (K) to minimize the reconstruction loss over the word-vectors. We fix the non-zero coefficient $k = K/2$, for the SIF experiments. For the GMM-based partitioning of the vocabulary, we tune the number of clusters' parameter K through a 5-fold cross validation.

Results and Analysis: The average results for each year are reported in Tables 1 and 2. We denote our embeddings by P-SIF + PSL (+ PSL denotes using the PSL word vectors). We report the average results for the STS tasks. The detailed results on each sub-dataset are in the Supplementary material [5]. We observe that P-SIF + PSL outperforms PSL + WR on all datasets, thus supporting the usefulness of our partitioned averaging. Despite being simple, our method outperforms many complicated methods such as seq2seq, Tree-LSTM(Tai, Socher, and Manning 2015), and Skip-Thoughts(Kiros et al. 2015). We observe that partitioning through overlapping clustering algorithms such as GMM generates a better performance compared to partitioning through sparse dictionary algorithms such as k-svd for some Semantic Textual Similarity (STS) task datasets. The main reason for this peculiar observation is related to the fact that some STS datasets contain documents which are single sentences of a maximum length of 40 words. As discussed earlier (sparse dictionary learning vs. overlapping clustering), for single sentence documents with a small number of topics, overlapping clustering optimizes better than sparse dictionary learning. Therefore, we use GMM for partitioning words into suitable clusters for some STS tasks. But both k-svd and GMM outperform simple averaging (SIF) by significant margins on most STS tasks. [8] We also report qualitative results with real examples in the Supplementary [5].

Text Classification Task

The document embeddings obtained by our method can be used as direct features for many classification tasks.

Datasets and Baselines: We run multi-class experiments on 20NewsGroup dataset, [9] and multi-label classification experiments on Reuters-21578 dataset. [10] We use *script* for preprocessing the dataset. [11] We consider several embedding baselines mostly taken from (Mekala et al. 2017; Wu et al. 2018; Arora et al. 2016b). More details on experimental settings and hyper-parameters' values are described in the Supplementary material [5]. We considered the following baselines: The Bag-of-Words (BoW) model (Harris 1954), the Bag of Word Vector (BoWV) (Gupta et al. 2016) model, Sparse Composite Document Vector (SCDV) (Mekala et al. 2017) [12] paragraph vector models (Le and Mikolov 2014), Topical word embeddings (TWE-1) (Liu et al. 2015), Neural Tensor Skip-Gram Model (NTSG-1 to NTSG-3) (Liu, Qiu, and Huang 2015), tf-idf weighted average word-vector model(Singh and Mukerjee 2015) and weighted Bag of Concepts (weight-BoC) (Kim, Kim, and Cho 2017) where we built document-topic vectors by counting the member words in each topic, and Doc2VecC (Chen 2017) where averaging and training of word vectors are done jointly. Moreover, we used SIF (Arora, Liang, and Ma 2017) smooth inverse frequency weight with common component removal from weighted average vectors as a baseline. We also compared our results with other topic modeling based document embedding methods such as WTM (Fu et al. 2016), w2v-LDA (Nguyen et al. 2015), LDA (Chen and Liu 2014), TV+MeanWV (Li et al. 2016a)), LTSG (Law et al. 2017), Gaussian-LDA (Das, Zaheer, and Dyer 2015), Topic2Vec (Niu et al. 2015), Lda2Vec (Moody 2016), MvTM (Li et al. 2016b) and BERT (Devlin et al. 2019). For BERT, we reported the results on the unsupervised pretrained (pr) model because of a fair comparison to our approach which is also unsupervised.

Experimental Settings: We fix the document embeddings and only learn the classifier. We learn word vector embeddings using Skip-Gram with a window size of 10, Negative Sampling (SGNS) of 10, and minimum word frequency of 20. We use 5-fold cross-validation on the $F1$ score to tune hyperparameters. We use LinearSVM for multi-class classification and Logistic regression with the OneVsRest setting for multi-label classification. We fix the number of dictionary elements to either 40 or 20 (with Doc2vecC initialize word vectors) and non-zero coefficient to $k = K/2$ during dictionary learning for all experiments. We use the best parameter settings, as reported in all our baselines to generate their results. We use 200 dimensions for tf-idf weighted word-vector model, 400 for paragraph vector model, 80 topics and 400 dimensional vectors for TWE, NTSG, LTSG and 60 topics and 200 dimensional word vectors for SCDV (Mekala et al. 2017). We evaluate the classifiers' performance using standard metrics such as accuracy, macro-averaging precision, recall and F-score for multiclass classification tasks.We evaluate multi-label classifications' performance using Precision@K, nDCG@k, Coverage error, Label ranking average precision(LRAPS) and F1 score.[13]

Results and Analysis: We observe that P-SIF outperforms all other methods by a significant margin on both 20NewsGroup (Table 4) and Reuters (Table 5). We observe that the dictionary learns more diverse and non-redundant topics compared to overlapping clustering (SCDV) since we require only 40 partitions rather than 60 partitions in SCDV to obtain the best performance. Simple tf-idf weighted averaging-based document representations do not show significant improvement in performance by increasing word vector dimensions. We achieve a $< 0.4\%$ improvement in the accuracy when the word-vector dimensions increase

[7] For a fair comparison with SIF we use PSL vectors instead of unsupervised GloVe and Word2Vec vectors. [8] k-svd always outperforms GMM on both classification datasets since the documents are multi-sentence with #words $>>$ 40. [9] https://bit.ly/2pqLCaN [10] https://goo.gl/NrOfu

[11] https://gist.github.com/herrfz/7967781

[12] https://github.com/dheeraj7596/SCDV

[13] https://goo.gl/4GrR3M

Table 1: Experimental results (Pearson's r $\times$ 100) on textual similarity tasks. Many results are collected from (Wieting et al. 2016), DAN (Iyyer et al. 2015) and (Wieting and Gimpel 2017) (GRAN) except for tfidf-GloVe.

Tasks	PP	PP proj	DAN	RNN	iRNN	LSTM (no)	LSTM (o.g.)	GRAN	ST	Avg Glove	tfidf Glove	Avg PSL	Glove +WR	PSL +WR	PSIF +PSL
STS'12	58.7	60.0	56.0	48.1	58.4	51.0	46.4	62.5	30.8	52.5	58.7	52.8	56.2	59.5	**65.7**
STS'13	55.8	56.8	54.2	44.7	56.7	45.2	41.5	63.4	24.8	42.3	52.1	46.4	56.6	61.8	**64.0**
STS'14	70.9	71.3	69.5	57.7	70.9	59.8	51.5	**75.9**	31.4	54.2	63.8	59.5	68.5	73.5	74.8
STS'15	75.8	74.8	72.7	57.2	75.6	63.9	56.0	**77.7**	31.0	52.7	60.6	60.0	71.7	76.3	77.3
Sick'14	71.6	71.6	70.7	61.2	71.2	63.9	59.0	72.9	49.8	65.9	69.4	66.4	72.2	72.9	**73.4**
Twit15	52.9	52.8	53.7	45.1	52.9	47.6	36.1	50.2	24.7	30.3	33.8	36.3	48.0	49.0	**54.9**

Table 2: P-SIF comparison with the recent embedding techniques on various STS tasks. Baselines taken from (Conneau and Kiela 2018), (Perone, Silveira, and Paula 2018), (Cer et al. 2018), (Devlin et al. 2019), (Wu et al. 2018) and (Ethayarajh 2018)

Task	ELMo orig+all	ELMo orig+top	Bert(pr) Avg.	USE	p-mean	Fast Text	Skip Thoughts	Infer Sent	Char pharse	WME +PSL	PSIF +PSL	u-SIF +PSL
STS 12	55	54	53	65	54	58	41	61	66	62.8	65.7	**65.8**
STS 13	51	49	67	68	52	58	29	56	57	56.3	63.98	**65.2**
STS 14	63	62	62	64	63	65	40	68	74.7	68.0	74.8	**75.9**
STS 15	69	67	73	77	66	68	46	71	76.1	64.2	77.3	**77.6**
STS 16	64	63	67	73	67	64	52	77	-	-	**73.7**	72.3
Average	60.4	59	64.4	69.4	60.4	62.6	41.6	66.6	68.5	62.9	71.1	**71.4**

Table 3: Comparison of P-SIF (SGNS) with the recently proposed word mover distance and word mover embedding (Wu et al. 2018) based on accuracy. In $\pm x$, x is the variance across several runs.

Dataset	Bbcsport	Twitter	Ohsumed	Classic	Reuters	Amazon	20NG	Recipe-L
SIF(GloVe)	97.3 $\pm$ 1.2	57.8 $\pm$ 2.5	**67.1**	92.7 $\pm$ 0.9	87.6	94.1 $\pm$ 0.2	72.3	71.1 $\pm$ 0.5
Word2Vec Avg	97.3 $\pm$ 0.9	72.0 $\pm$ 1.5	63	95.2 $\pm$ 0.4	96.9	94.0 $\pm$ 0.5	71.7	74.9 $\pm$ 0.5
PV-DBOW	97.2 $\pm$ 0.7	67.8 $\pm$ 0.4	55.9	97.0 $\pm$ 0.3	96.3	89.2 $\pm$ 0.3	71	73.1 $\pm$ 0.5
PV-DM	97.9 $\pm$ 1.3	67.3 $\pm$ 0.3	59.8	96.5 $\pm$ 0.7	94.9	88.6 $\pm$ 0.4	74	71.1 $\pm$ 0.4
Doc2VecC	90.5 $\pm$ 1.7	71.0 $\pm$ 0.4	63.4	96.6 $\pm$ 0.4	96.5	91.2 $\pm$ 0.5	78.2	76.1 $\pm$ 0.4
KNN-WMD	95.4 $\pm$ 1.2	71.3 $\pm$ 0.6	55.5	97.2 $\pm$ 0.1	96.5	92.6 $\pm$ 0.3	73.2	71.4 $\pm$ 0.5
SCDV	98.1 $\pm$ 0.6	74.2 $\pm$ 0.4	53.5	96.9 $\pm$ 0.1	97.3	93.9 $\pm$ 0.4	78.8	78.5$\pm$ 0.5
WME	98.2 $\pm$ 0.6	**74.5 $\pm$ 0.5**	64.5	97.1 $\pm$ 0.4	97.2	94.3 $\pm$ 0.4	78.3	**79.2 $\pm$ 0.3**
P-SIF	99.05 $\pm$ 0.9	73.39 $\pm$ 0.9	**67.1**	96.95 $\pm$ 0.5	**97.67**	94.17 $\pm$ 0.3	79.15	78.24 $\pm$ 0.3
P-SIF (Doc2VecC)	**99.68 $\pm$ 0.9**	72.39 $\pm$ 0.9	**67.1**	**97.7 $\pm$ 0.5**	97.62	**94.83 $\pm$ 0.3**	**86.31**	77.61 $\pm$ 0.3

from 200 to 500 on 20NewsGroup. We observe that increasing the word-vectors' dimensions beyond 500 does not improve SIF and P-SIF's performances. We further improve the performance on both datasets using Doc2vecC-initialized (Chen 2017) word-vectors which reduce word level noise in the P-SIF representations. We represent this approach by P-SIF (Doc2VecC) in Table 4 and Table 5. On 20NewsGroup, we require only 20 partitions instead of 40 with Doc2VecC-initialized word vectors. This shows that better word vector representations help in learning more diverse and non-redundant partitions. We also report our results (micro-F1) on each of the 20 classes of 20NewsGroup in the Supplementary material [5]. Additionally, we empirically show that our proposed embedding P-SIF outperforms the word mover distance (Kusner et al. 2015) and performs comparable with the word mover embedding (Wu et al. 2018) in Table 3. [14] Overall, P-SIF outperforms most methods on several datasets by a significant margin.

Comparison with Contextual Embeddings: Despite its simplicity, P-SIF is able to outperform unsupervised contextual embeddings such as BERT (pr) and ELMo. We assume the reason behind this is P-SIF's focused ability to effectively capture both global and local semantics in sparse higher dimension representations. On other hand, BERT tries to capture both syntax and semantics in single lower dimensional continuous representations. In both classification and similarity tasks, understanding syntax is not as prominent as understanding semantics.

[14] For datasets and baseline details refer to (Wu et al. 2018).

Analysis and Discussion

Effect of Document-Length: We conduct a small experiment to show that our model performs better compared to SIF when we have large size documents. We have divided 26 STS datasets by average document length, i.e., the number of words in documents in bins of $(10-20, 20-30, 30-40, 40-50)$ words. Next, we average the relative performance improvement by P-SIF and SCDV by accuracy with respect to SIF $\left(\frac{\text{Method}-\text{SIF}}{\text{SIF}}\%\right)$ for datasets in each bin. In Figure 2, we observe that for complex multi-sentence documents with more words, P-SIF performs relatively better than SCDV. We also note that short texts require fewer number of partitions to achieve their best performance which is quite intuitive since short text documents will map into fewer topics.

Table 4: Multi-class classification performance on 20News-Groups.

Model	Acc	Prec	Rec	Fmes
P-SIF (Doc2VecC)	**86.0**	**86.1**	**86.1**	**86.0**
P-SIF	**85.4**	**85.5**	**85.4**	**85.2**
BERT (pr)	84.9	84.9	85.0	85.0
SCDV	84.6	84.6	84.5	84.6
Doc2VecC	84.0	84.1	84.1	84.0
RandHash	83.9	83.99	83.9	83.76
BoE	83.1	83.1	83.1	83.1
NTSG	82.5	83.7	82.8	82.4
SIF	82.3	82.6	82.9	82.2
BoWV	81.6	81.1	81.1	80.9
LTSG	82.8	82.4	81.8	81.8
p-means	82.0	81.9	82.0	81.6
WTM	80.9	80.3	80.3	80.0
w2v-LDA	77.7	77.4	77.2	76.9
ELMo	74.1	74.0	74.1	73.9
TV+MeanWV	72.2	71.8	71.5	71.6
MvTM	72.2	71.8	71.5	71.6
TWE-1	81.5	81.2	80.6	80.6
Lda2Vec	81.3	81.4	80.4	80.5
LDA	72.2	70.8	70.7	70.0
weight-AvgVec	81.9	81.7	81.9	81.7
BoW	79.7	79.5	79.0	79.0
weight-BOC	71.8	71.3	71.8	71.4
PV-DBoW	75.4	74.9	74.3	74.3
PV-DM	72.4	72.1	71.5	71.5

Table 5: Performance on multi-label classification on Reuters.

Model	Prec @1	Prec @5	nDCG @5	Cover. Error	LRAPS Score	F1
P-SIF (Doc2VecC)	**94.92**	**37.98**	**50.40**	**6.03**	**93.95**	**82.87**
P-SIF	**94.77**	**37.33**	**49.97**	**6.24**	**93.72**	**82.41**
BERT (pr)	93.8	37	49.6	6.3	93.1	81.9
SCDV	94.20	36.98	49.55	6.48	93.30	81.75
Doc2VecC	93.45	36.86	49.28	6.83	92.66	81.29
p-means	93.29	36.65	48.95	10.8	91.72	77.81
BoWV	92.90	36.14	48.55	8.16	91.46	79.16
TWE	90.91	35.49	47.54	8.16	91.46	79.16
SIF	90.40	35.09	47.32	8.98	88.10	76.78
PV-DM	87.54	33.24	44.21	13.2	86.21	70.24
PV-DBoW	88.78	34.51	46.42	11.3	87.43	73.68
AvgVec	89.09	34.73	46.48	9.67	87.28	71.91
tfidf AvgVec	89.33	35.04	46.83	9.42	87.90	71.97

Effect of Sparse Partitioning: Partitioning and concatenation of word embeddings over topics also helps in the representation of multi-sense words, which would have been left-out by simple averaging of the word embeddings in document representation otherwise. Empirically, on both datasets, we observe that the dictionary learns more diverse and non-redundant topics compared to overlapping clustering because of sparsity constraints. We require only 20 partitions rather than 60 in SCDV to obtain the best performance, meaning we just need $(20 * 300)$ dimensions of embeddings (mostly sparse) compared to $(60 * 300)$ dimensions of embeddings (mostly non-sparse). Thus, we obtain a performance gain (F1-Score) of 1.5% with less than 0.33 of the

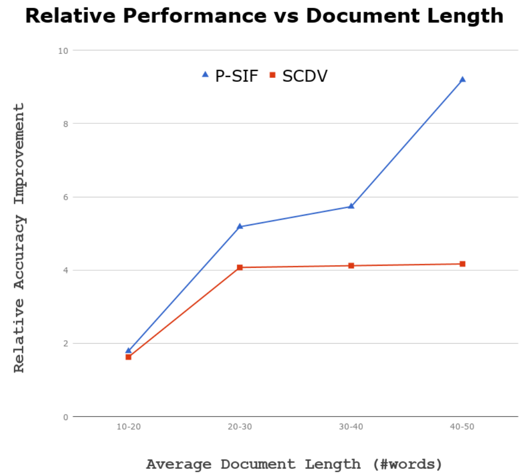

Figure 2: Relative performance improvement of P-SIF and SCDV over SIF w.r.t the average document length.

size of the SCDV embeddings. Lastly, due to fewer dimensions, the feature formation time is less in P-SIF.

Conclusions and Future Work

We propose a novel unsupervised document feature formation technique based on partitioned word vector averaging. Our embedding retains the simplicity of simple weighted word averaging while taking documents' topical structure into account. Our simple and efficient approach achieves significantly better performance on several textual similarity and textual classification tasks, e.g., we outperform contextual embeddings such as BERT (pr) and ELMo. One limitation of our work is its ignorance of words' order and syntax. In the future, we plan to address this problem and model partitioning, averaging, and learning as a joint process.

References

Agirre, E.; Diab, M.; Cer, D.; and Gonzalez-Agirre, A. 2012. Semeval-2012 task 6. In *Semeval*, 385–393. ACL.

Agirre, E.; Banea, C.; Cer, D.; Diab, M.; Gonzalez-Agirre, A.; Mihalcea, R.; Rigau, G.; and Wiebe, J. 2016. Semeval-2016. In *SemEval-2016*, 497–511.

Aharon, M.; Elad, M.; Bruckstein, A.; et al. 2006. K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on signal processing* 54(11):4311.

Arora, S.; Li, Y.; Liang, Y.; Ma, T.; and Risteski, A. 2016a. A latent variable model approach to pmi-based word embeddings. *TACL*.

Arora, S.; Li, Y.; Liang, Y.; Ma, T.; and Risteski, A. 2016b. Linear algebraic structure of word senses, with applications to polysemy. *arXiv preprint arXiv:1601.03764*.

Arora, S.; Liang, Y.; and Ma, T. 2017. A simple but tough-to-beat baseline for sentence embeddings. *ICLR*.

Blei, D. M.; Ng, D. M.; Jordan, M. I.; and Lafferty, J. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*.

Cer, D.; Yang, Y.; Kong, S.-y.; Hua, N.; Limtiaco, N.; John, R. S.; Constant, N.; Guajardo-Cespedes, M.; Yuan, S.; Tar, C.; et al. 2018. Universal sentence encoder. *arXiv preprint arXiv:1803.11175*.

Chen, Z., and Liu, B. 2014. Topic modeling using topics from many domains, lifelong learning and big data. In *ICML*, 703–711.

Chen, M. 2017. Efficient vector representation for documents through corruption. *ICLR*.

Conneau, A., and Kiela, D. 2018. Senteval: An evaluation toolkit for universal sentence representations. *arXiv preprint arXiv:1803.05449*.

Das, R.; Zaheer, M.; and Dyer, C. 2015. Gaussian lda for topic models with word embeddings. In *ACL*.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *NAACL*.

Dieng, A. B.; Ruiz, F. J.; and Blei, D. M. 2019a. The dynamic embedded topic model. *arXiv preprint arXiv:1907.05545*.

Dieng, A. B.; Ruiz, F. J.; and Blei, D. M. 2019b. Topic modeling in embedding spaces. *arXiv preprint arXiv:1907.04907*.

Ethayarajh, K. 2018. Unsupervised random walk sentence embeddings: A strong but simple baseline. *ACL 2018* 91.

Fu, X.; Wang, T.; Li, J.; Yu, C.; and Liu, W. 2016. Improving distributed word representation and topic model by word-topic mixture model. In *ACL*, 190–205.

Ganitkevitch, J.; Van Durme, B.; and Callison-Burch, C. 2013. Ppdb: The paraphrase database. In *NAACL*, 758–764.

Gao, S.; Tsang, I. W.-H.; Chia, L.-T.; and Zhao, P. 2010. Local features are not lonely–laplacian sparse coding for image classification. In *CVPR*, 3555–3561. IEEE.

Gers, F. A.; Schraudolph, N. N.; and Schmidhuber, J. 2002. Learning precise timing with lstm recurrent networks. *Journal of machine learning research* 3(Aug):115–143.

Gupta, V.; Karnick, H.; Bansal, A.; and Jhala, P. 2016. Product classification in e-commerce using distributional semantics. In *COLING*, 536–546.

Harris, Z. 1954. Distributional structure. *Word* 10:146–162.

Iyyer, M.; Manjunatha, V.; Boyd-Graber, J.; and Daumé III, H. 2015. Deep unordered composition rivals syntactic methods for text classification. In *ACL*, volume 1, 1681–1691.

Kim, H. K.; Kim, H.; and Cho, S. 2017. Bag-of-concepts: Comprehending document representation through clustering words in distributed representation. *Neurocomputing*.

Kim, Y. 2014. Convolutional neural networks for sentence classification. In *EMNLP*, 1746–1751.

Kiros, R.; Zhu, Y.; Salakhutdinov, R. R.; Zemel, R.; Urtasun, R.; Torralba, A.; and Fidler, S. 2015. Skip thought vectors. *NeurIPS*.

Kusner, M.; Sun, Y.; Kolkin, N.; and Weinberger, K. 2015. From word embeddings to document distances. In *International Conference on Machine Learning*, 957–966.

Law, J.; Zhuo, H. H.; He, J.; and Rong, E. 2017. Ltsg: Latent topical skip-gram for mutually learning topic model and vector representations. *arXiv preprint arXiv:1702.07117*.

Le, Q. V., and Mikolov, T. 2014. Distributed representations of sentences and documents. In *ICML*, volume 14, 1188–1196.

Levy, O., and Goldberg, Y. 2014. Neural word embedding as implicit matrix factorization. In *NeurIPS*, 2177–2185.

Li, S.; Chua, T.-S.; Zhu, J.; and Miao, C. 2016a. Generative topic embedding: a continuous representation of documents. In *ACL*.

Li, X.; Chi, J.; Li, C.; Ouyang, J.; and Fu, B. 2016b. Integrating topic modeling with word embeddings by mixtures of vmfs. *COLING* 151—-160.

Ling, W.; Dyer, C.; Black, A.; and Trancoso, I. 2015. Two/too simple adaptations of wordvec for syntax problems. In *NAACL*.

Liu, Y.; Liu, Z.; Chua, T.-S.; and Sun, M. 2015. Topical word embeddings. In *AAAI*, 2418–2424.

Liu, P.; Qiu, X.; and Huang, X. 2015. Learning context-sensitive word embeddings with neural tensor skip-gram model. In *IJCAI*.

Marelli, M.; Bentivogli, L.; Baroni, M.; Bernardi, R.; Menini, S.; and Zamparelli, R. 2014. Semeval-2014 task 1. In *SemEval 2014*.

Mekala, D.; Gupta, V.; Paranjape, B.; and Karnick, H. 2017. Scdv: Sparse composite document vectors using soft clustering over distributional representations. In *EMNLP*, 659–669.

Mikolov, T.; Karafiát, M.; Burget, L.; Černockỳ, J.; and Khudanpur, S. 2010. Recurrent neural network based language model. In *ISCA*.

Moody, C. E. 2016. Mixing dirichlet topic models and word embeddings to make lda2vec. *arXiv preprint arXiv:1605.02019*.

Nguyen, D. Q.; Billingsley, R.; Du, L.; and Johnson, M. 2015. Improving topic models with latent feature word representations. *ACL* 3:299–313.

Niu, L.; Dai, X.; Zhang, J.; and Chen, J. 2015. Topic2vec: learning distributed representations of topics. In *IALP*, 193–196. IEEE.

Olshausen, B. A., and Field, D. J. 1997. Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision research* 37(23):3311–3325.

Pagliardini, M.; Gupta, P.; and Jaggi, M. 2018. Unsupervised Learning of Sentence Embeddings using Compositional n-Gram Features. In *NAACL 2018*.

Pennington, J.; Socher, R.; and Manning, C. 2014. Glove: Global vectors for word representation. In *EMNLP*, 1532–1543.

Perone, C. S.; Silveira, R.; and Paula, T. S. 2018. Evaluation of sentence embeddings in downstream and linguistic probing tasks. *arXiv preprint arXiv:1806.06259*.

Peters, M. E.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; and Zettlemoyer, L. 2018. Deep contextualized word representations. In *NAACL*.

Singh, P., and Mukerjee, A. 2015. Words are not equal: Graded weighting model for building composite document vectors. In *ICON*.

Socher, R.; Perelygin, A.; Wu, J. Y.; Chuang, J.; Manning, C. D.; Ng, A. Y.; Potts, C.; et al. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*.

Tai, K. S.; Socher, R.; and Manning, C. D. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *ACL*, volume 1, 1556–1566.

Wieting, J., and Gimpel, K. 2017. Revisiting recurrent networks for paraphrastic sentence embeddings. *ACL*.

Wieting, J.; Bansal, M.; Gimpel, K.; Livescu, K.; and Roth, D. 2015. From paraphrase database to compositional paraphrase model and back. *TACL* 3:345–358.

Wieting, J.; Bansal, M.; Gimpel, K.; and Livescu, K. 2016. Towards universal paraphrastic sentence embeddings. *ICLR*.

Wu, L.; Yen, I. E.-H.; Xu, K.; Xu, F.; Balakrishnan, A.; Chen, P.-Y.; Ravikumar, P.; and Witbrock, M. J. 2018. Word mover's embedding: From word2vec to document embedding. In *EMNLP*.

Xu, W.; Callison-Burch, C.; and Dolan, B. 2015. Semeval-2015 task 1. In *SemEval 2015*, 1–11.

Yang, J.; Yu, K.; Gong, Y.; and Huang, T. 2009. Linear spatial pyramid matching using sparse coding for image classification. In *CVPR*, 1794–1801. IEEE.