P-SIF: Document Embeddings using Partition Averaging

Derivation of P-SIF Embeddings

To derive the P-SIF embedding, we propose a generative model which treats corpus generation as a dynamic process where the t^{th} word is produced at step t. This process is driven by random walk over a unit norm sphere with the center at the origin. Let $\vec{v}_{c_t}$ be the d dimensional vector from the origin to the current walk point at time t. We call this vector the context vector $\vec{v}_{c_t}$ as it represents the context in the discussion. Let Z_c represent the partition function for the random context vector $\vec{v}_{c_t}$, given by $Z_c = \sum_w \exp(\langle \vec{v}_{c_t}, \vec{v}_w \rangle)$. c_0 and $\vec{v}_{c_0}$ represent a common context and its corresponding d dimensional context vector based on syntax.

Using log linear model of (Mnih and Hinton 2007), we define the probability of observing a word w from the random walk with current context c_t at time t as

$$Pr[w|c_t] \propto \exp(\langle \vec{c}_t, \vec{v}_w \rangle) \tag{1}$$

It is easy to show that such random walk under some reasonable assumptions (Arora et al. 2016a) can give word-word co-occurrence probabilities similar to empirical works like word2vec (Mikolov et al. 2013) and GloVe (Pennington, Socher, and Manning 2014). To account for frequent stop-words which occur more often regardless of context and the common context related to document syntax, two correction terms need be added: one based on $p(w)$ and the other on the common context vector $\vec{v}_{c_0}$ in Equation (1). These terms allow words with a low inner product with $\vec{c}_t$ a chance to appear either from $p(w)$ if they are frequent or by the common context $\vec{c}_0$ if they have a large dot product with $\vec{c}_0$. Given a context vector c_t, the probability of a word w in document d being generated by context c_t is given by,

$$Pr[w|c_t] = \lambda p(w) + (1-\lambda)\frac{\exp(\langle \vec{c}'_t, \vec{v}_w \rangle)}{Z_{c'_t}} \tag{2}$$

where $\vec{c}'_t = \beta\vec{c}_0 + (1-\beta)\ \vec{c}_t$, $\langle \vec{c}_0, \vec{c}_t \rangle = 0$, λ and β are scalar hyper-parameters.

For generating a document from the above random walk-based latent variable model, we consider the following assumptions:

1. Context vector ($\vec{v}_{c_0}$) does not change significantly while words are generated from the random walk, as shown by (Arora, Liang, and Ma 2017), except the jumps due to topic change.
2. Total number of topics in the entire corpus is K, which can be determined by sparse dictionary learning as shown by (Arora et al. 2016b) over word vectors $\vec{v}_w$.
3. Word vectors $\vec{v}_w$ are uniformly distributed, thus making the partition function Z_c roughly the same in all directions for a given context c emerging from each of the K topics.

For a document d, the likelihood of document is being generated by the K contexts is given by:

$$p(d|\{c_1, c_2 \dots c_K\}) \propto \prod_{j=1}^{K} \prod_{\{w \in d\}} p(w|c_j) \tag{3}$$

$$= \prod_{j=1}^{K} \prod_{w \in d} \Big[\lambda p(w) + (1-\lambda)\frac{\exp(\langle \vec{v}_w, \vec{v}_{c_j} \rangle)}{Z_j}\Big] \tag{4}$$

$$\text{Let, } f_w(c_j) = \log\Big[\lambda p(w) + (1-\lambda)\frac{\exp(\langle \vec{v}_w, \vec{v}_{c_j} \rangle)}{Z_j}\Big] \tag{5}$$

Here, $p(w|c_j)$ is the probability that word w is generated by context c_j, the value of which is determined by 1) The overall frequency of word w in the corpus, i.e., prior probability ($p(w)$) and 2) The relative frequency of w appearing with context j with respect to other contexts (determined by $\alpha_{(w,j)}$).

Using simple algebra and treating $p(w)$ as a constant, we can show that $\nabla(f_w(c_j))$ equals,

$$\frac{1}{\lambda p(w) + (1-\lambda)\exp(\langle \vec{v}_w, \vec{v}_{c_j} \rangle)/Z_j} * \frac{1-\lambda}{Z_j}\exp(\langle \vec{v}_w, \vec{v}_{c_j} \rangle)\vec{v}_w \tag{6}$$

Then, by using the Taylor expansion, we can show

$$f_w(c_j) \approx f_w(c_j = 0) + \nabla(f_w(c_j = 0))^T \vec{v}_{c_j} \tag{7}$$

$$f_w(c_j) \approx constant + \nabla(f_w(c_j = 0))^T \vec{v}_{c_j} \tag{8}$$

Therefore, the maximum likelihood estimator (MLE) for $\vec{v}_{c_j}$ on the unit sphere (ignoring normalization) given $a = \frac{1-\lambda}{\lambda Z_j}$, is approximately [14]

$$\arg\max \sum_{w \in d} f_w(c_j) \propto \sum_{w \in d} \frac{a}{p(w)+a}\vec{v}_w \tag{9}$$

Thus, the MLE estimate is approximately a weighted average of the word-vectors generated from context j in document d from the random walk. We can get the overall context representation $\vec{v}_{c_d}$ of the document by simple concatenation over all K topics i.e. $\vec{v}_{c_d} = \bigoplus_{j=1}^{K} \vec{v}_{c_j}$. Here, $\bigoplus$ represents the concatenation operation. For a document if no word is generated from the context c_j then we can substitute the context vector $\vec{v}_{c_j}$ by a $\vec{0}$ vector to represent $\vec{v}_{c_d}$ in $K \times d$ dimensions. The embedding of a sentence can be obtained by $\vec{v}_{c_s} = \sum_{\{w \in s\}} \frac{a}{p(w)+a}\vec{v}_w$ where $a = \frac{1-\lambda}{\lambda Z_s}$.

Relation to SIF model: (Arora, Liang, and Ma 2017) show sentences can be represented as averaging of word vectors, under the two assumptions:

- uniform distribution of word vectors $\vec{v}_w$ which implies that the partition function Z_t is roughly the same in all directions for a sentence.
- context vector $\vec{v}_{c_h}$ remains constant while the words in the sentence are emitted, implying the replacement of $\vec{v}_{c_h}$ in the sentences by $\vec{v}_{c_s}$ and partition function Z_t by Z_s.

However, the above assumptions do not hold true for a document with multiple sentences where one can expect to have more frequent jumps during a random walk due to topic change. [15] Instead of assuming a single context for the whole document c_h, we assume that the total number of contexts over a given corpus is bounded by the number of topics K (as shown by (Arora et al. 2016b)), and the random walk can perform jumps to switch context from one context to the rest of $K-1$ contexts. The partition function remains the same in all directions only for the words emerging from the same context c_j instead of the words coming from all the K contexts. Thus, our approach is a strict generalization of the sentence embedding approach by (Arora, Liang, and Ma 2017) which is a special case of $K = 1$.

Details of Textual Similarity Task

In this supplementary section, we present the details of the STS tasks for each year. Each year, there are 4 to 6 STS tasks, as shown in Table 6. Note that tasks with the same name in different years are

[14] Note that $argmax_{c:\|\vec{c}\|=1} C + \langle \vec{c}, \vec{g} \rangle = \frac{\vec{g}}{\|\vec{g}\|}$ for any constant C

[15] It is trivial to assume that these jumps occur more frequently in multi-sentence documents because of more number of topics.

different tasks in reality. We provide detailed results for each tasks in STS 12 - 15 in Table 7. Our method outperforms all other methods from (Arora, Liang, and Ma 2017) and (Wieting et al. 2016) on all 16 out of 22 tasks. Our method performs significantly better in comparison to all unsupervised embedding methods. In addition, P-SIF is very close to the best performance by supervised methods on the rest of the datasets. Our method was also able to outperform state of the art supervised averaging based Gated Recurrent Averaging Network (GRAN) (Wieting and Gimpel 2017) on 11 datasets shown in Table 7. Our results also outperform state of the art methods on many recent supervised embedding methods on the STS 16 task (See Table 8).

Experimental Details

Textual Similarity Task:

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 [16]. 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.

1. **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) [17], 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.
2. **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).
3. **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 of (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 (Tai, Socher, and Manning 2015) embedding.

[16] For a fair comparison with SIF we use PSL vectors instead of unsupervised GloVe and Word2Vec vectors.

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

Textual Classification Task:

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).

Baseline Details: 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) [18] 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 pre-trained (pr) model because of a fair comparison to our approach which is also unsupervised.

Class wise Performance on 20NewsGroup

We also reported the precision, recall, and micro-F1 results of separate 20 classes of the 20 NewsGroup dataset. We compared our embedding (P-SIF) with Bag of Words, and SCDV embeddings. In Table 9, P-SIF (Doc2VecC) (20 partitions) embeddings outperforms SCDV (60 partitions) on 18 out of the 20 classes.

Other Supervised Tasks

We also considered three out of domain supervised tasks: the SICK similarity task, the SICK entailment task, and the Stanford Sentiment Treebank (SST) binary classification task by (Socher et al. 2013). We used the setup similar to (Wieting et al. 2016) and (Arora, Liang, and Ma 2017) for a fair comparison, including the linear projection maps which take the embedding into 2400 dimensions (same as skip-thought vectors), and is learned during the

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

Table 6: The STS tasks by year. Tasks with the same name in different years are different tasks

STS12	STS13	STS14	STS15	STS16
MSRpar	headline	deft forum	anwsers-forums	headlines
MSRvid	OnWN	deft news	answers-students	plagiarism
SMT-eur	FNWN	headline	belief	posteditng
OnWN	SMT	images	headline	answer-answer
SMT-news		OnWN	images	question-question
		tweet news		

Table 7: Experimental results (Pearson's r $\times$ 100) on textual similarity tasks. The highest score in each row is in bold. The methods can be supervised (denoted as Su.), semi-supervised (Se.), or unsupervised (Un.). See the main text for description of the methods. Many results are collected from (Wieting et al. 2016) and (Wieting and Gimpel 2017) (GRAN) except the tfidf-GloVe and our representation.

TaskType	Supervised								UnSupervised			Semi Supervised			P-SIF
Tasks	PP	PP proj	DAN	RNN	iRNN	LSTM (no)	LSTM (o.g.)	GRAN	ST	avg Glove	tfidf Glove	avg PSL	Glove +WR	PSL +WR	P-SIF +PSL
MSRpar	42.6	43.7	40.3	18.6	43.4	16.1	9.3	47.7	16.8	47.7	50.3	41.6	35.6	43.3	**52.4**
MSRvid	74.5	74.0	70.0	66.5	73.4	71.3	71.3	85.2	41.7	63.9	77.9	60.0	83.8	84.1	**85.6**
SMT-eur	47.3	49.4	43.8	40.9	47.1	41.8	44.3	49.3	35.2	46.0	54.7	42.4	49.9	44.8	**58.7**
OnWN	70.6	70.1	65.9	63.1	70.1	65.2	56.4	71.5	29.7	55.1	64.7	63.0	66.2	71.8	**72.2**
SMT-news	58.4	**62.8**	60.0	51.3	58.1	60.8	51.0	58.7	30.8	49.6	45.7	57.0	45.6	53.6	59.5
STS12	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**
headline	72.4	72.6	71.2	59.5	72.8	57.4	48.5	**76.1**	34.6	63.8	69.2	68.8	69.2	74.1	75.7
OnWN	67.7	68.0	64.1	54.6	69.4	68.5	50.4	81.4	10.0	49.0	72.9	48.0	82.8	82.0	**84.4**
FNWN	43.9	46.8	43.1	30.9	45.3	24.7	38.4	**55.6**	30.4	34.2	36.6	37.9	39.4	52.4	54.8
SMT	39.2	39.8	38.3	33.8	39.4	30.1	28.8	40.3	24.3	22.3	29.6	31.0	37.9	38.5	**41.0**
STS13	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**
deft forum	48.7	51.1	49.0	41.5	49.0	44.2	46.1	**55.7**	12.9	27.1	37.5	37.2	41.2	51.4	53.2
deft news	73.1	72.2	71.7	53.7	72.4	52.8	39.1	**77.1**	23.5	68.0	68.7	67.0	69.4	72.6	75.2
headline	69.7	70.8	69.2	57.5	70.2	57.5	50.9	**72.8**	37.8	59.5	63.7	65.3	64.7	70.1	70.2
images	78.5	78.1	76.9	67.6	78.2	68.5	62.9	**85.8**	51.2	61.0	72.5	62.0	82.6	84.8	84.8
OnWN	78.8	79.5	75.7	67.7	78.8	76.9	61.7	85.1	23.3	58.4	75.2	61.1	82.8	84.5	**88.1**
tweet news	76.4	75.8	74.2	58.0	76.9	58.7	48.2	**78.7**	39.9	51.2	65.1	64.7	70.1	77.5	77.5
STS14	70.9	71.3	69.5	57.7	70.9	59.8	51.5	**75.8**	31.4	54.2	63.8	59.5	68.5	73.5	74.8
ans-forum	68.3	65.1	62.6	32.8	67.4	51.9	50.7	**73.1**	36.1	30.5	45.6	38.8	63.9	70.1	70.7
ans-student	78.2	77.8	78.1	64.7	78.2	71.5	55.7	72.9	33.0	63.0	63.9	69.2	70.4	75.9	**79.6**
belief	76.2	75.4	72.0	51.9	75.9	61.7	52.6	**78**	24.6	40.5	49.5	53.2	71.8	75.3	75.3
headline	74.8	75.2	73.5	65.3	75.1	64.0	56.6	**78.6**	43.6	61.8	70.9	69.0	70.7	75.9	76.8
images	81.4	80.3	77.5	71.4	81.1	70.4	64.2	**85.8**	17.7	67.5	72.9	69.9	81.5	84.1	84.1
STS15	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**
SICK14	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**
Twitter15	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 8: Experimental results (Pearson's r $\times$ 100) on textual similarity tasks on STS 16. The highest score in each row is in bold.

Tasks	Skip thoughts	LSTM	Tree LSTM	Sent2Vec	Doc2Vec	GloVe Avg	GloVe tf-idf	PSL Avg	PSL tf-idf	GloVe +WR	PSL +WR	P-SIF +PSL
headlines	51.02	75.7	74.08	75.06	69.16	49.66	52.76	70.86	72.24	72.86	74.48	**75.6**
plagiarism	66.71	71.73	67.62	80.06	80.6	59.84	61.48	77.96	80.06	79.46	79.74	**81.6**
post editing	69.95	72.31	70.65	82.85	82.85	59.89	62.34	80.41	81.45	82.03	82.05	**83.7**
ans-ans	28.63	44.17	52.27	57.73	41.12	19.8	22.47	38.5	41.56	58.15	59.98	**60.2**
ques-ques	40.46	60.69	55.26	73.03	**73.03**	46.84	56.58	48.69	59.1	69.36	66.41	67.2
STS16	51.4	64.9	64.0	**73.7**	69.4	47.2	51.1	63.3	66.9	72.4	72.5	**73.7**

Table 9: Class performnce on the 20newsgroup dataset. P-SIF represents our embedding with 40 partitions. P-SIF (Doc2VecC) represents our embeddings initialized with Doc2VecC trained word-vectors with 20 partitions.

	BoW			SCDV			P-SIF			P-SIF (Doc2VecC)		
Class Name	**Pre.**	**Rec.**	**F-mes**	**Pre.**	**Rec.**	**F-mes**	**Pre.**	**Rec.**	**F-mes**	**Pre.**	**Rec.**	**F-mes**
alt.atheism	67.8	72.1	69.8	80.2	79.5	79.8	83.3	80.2	**81.72**	83	79.9	81.4
comp.graphics	67.1	73.5	70.1	75.3	77.4	76.3	76.6	78.1	77.3	76.8	79.2	**77.9**
comp.os.ms-windows.misc	77.1	66.5	71.4	78.6	77.2	**77.8**	76.3	77.7	76.9	77.2	78.2	77.7
comp.sys.ibm.pc.hardware	62.8	72.4	67.2	75.6	73.5	**74.5**	73.4	74.5	73.9	71.1	74.2	72.6
comp.sys.mac.hardware	77.4	78.2	77.8	83.4	85.5	84.4	87.1	84.4	85.7	87.5	87.5	**87.5**
comp.windows.x	83.2	73.2	77.8	87.6	78.6	82.8	89.3	78	83.2	88.8	78.5	**83.3**
misc.forsale	81.3	88.2	84.6	81.4	85.9	83.5	82.7	88	**85.2**	82.4	86.4	84.3
rec.autos	80.7	82.8	81.7	91.2	90.6	90.9	93	90.1	91.5	92.8	90.7	**91.7**
rec.motorcycles	92.3	87.9	90.0	95.4	95.7	95.5	93.6	95.5	94.5	97	96.5	**96.7**
rec.sport.baseball	89.8	89.2	89.5	93.2	94.7	93.9	93.3	95.2	94.2	95.2	95.7	**95.4**
rec.sport.hockey	93.3	93.7	93.5	96.3	99.2	97.7	95.6	98.5	97.0	96.8	98.8	**97.7**
sci.crypt	92.2	86.1	89.0	92.5	94.7	93.5	89.8	93.2	91.47	93.4	96.7	**95.0**
sci.electronics	70.9	73.3	72.08	74.6	74.9	74.7	79.6	78.6	**79.1**	78	79.3	78.6
sci.med	79.3	81.3	80.2	91.3	88.4	89.8	91.9	88.6	90.2	92.7	89.9	**91.2**
sci.space	90.2	88.3	89.2	88.5	93.8	91.07	89.4	94	91.6	90.7	94.4	**92.5**
soc.religion.christian	77.3	87.9	82.2	83.3	92.3	87.5	84	94.3	88.8	86	92.5	**89.1**
talk.politics.guns	71.7	85.7	78.0	72.7	90.6	80.6	73.1	91.2	81.1	77.3	89.8	**83.1**
talk.politics.mideast	91.7	76.9	83.6	96.2	95.4	95.8	97	94.5	95.7	97.5	94.2	**95.8**
talk.politics.misc	71.7	56.5	63.2	80.9	59.7	68.7	81	59	68.2	82	62	**70.6**
talk.religion.misc	63.2	55.4	59.04	73.5	57.2	64.3	72.2	59	**64.9**	67.4	62.4	64.8

Table 10: Results on similarity, entailment, and sentiment tasks. The row for similarity (SICK) shows Pearson's r $\times$ 100 and the last two rows show accuracy. The highest score in each row is in bold. Results in Column 2 to 6 are collected from (Wieting et al. 2016), and those in Column 7 for skip-thought are from (Kiros et al. 2015), Column 8 for PSL + WR are from (Arora, Liang, and Ma 2017).

Tasks	**PP**	**DAN**	**RNN**	**LSTM (no)**	**LSTM (o.g.)**	**skip thought**	**PSL +WR**	**P-SIF +PSL**
similarity (SICK)	84.9	85.96	73.13	85.45	83.4	85.8	86.3	**87.6**
entailment (SICK)	83.1	84.5	76.4	83.2	82.0	-	84.6	**85.5**
sentiment (SST)	79.4	83.4	86.5	86.6	**89.2**	-	82.2	86.4

training. We compared our method to PP, DAN, RNN, LSTM, skip-thoughts and other baselines. Detailed results are in Table 10.

Results and Analysis. Our method (P-SIF) obtains a better performance compared to PSL + WR on all the three tasks similarity, entailment, and sentiment. We obtained the best results for two of the supervised tasks, although many of these methods (DAN, RNN, LSTM) are trained with supervision. Furthermore, the skip thought vectors use a higher dimension of 2400 instead of 300 dimensions (which we projected to 2400 for a fair comparison). Our method wasn't able to outperform the sentiment task compared to supervised tasks because a) due to the antonym problem word-vectors capture the sentimental meaning of words and b) in our weighted average scheme, we didn't assign more weights to sentiment words such as *'not'*, *'good'*, *'bad'*, there may be some important sentiment words which are down-weighted by the SIF weighting scheme. However, we outperform PSL + WR by a significant margin and have a less performance gap with the best supervised approach.

Proof: Kernels meet Embeddings

1. $K^1(D_A, D_B)$ represents document similarity between the documents represented by average word vectors $d_x = \sum_i \vec{v}_i^x$

Proof:

$$\vec{dv}^{\,A} = \frac{1}{n}\sum_{i=1}^{n} \vec{v}_{w_i^A} \tag{10}$$

$$\vec{dv}^{\,A} = \frac{1}{m}\sum_{j=1}^{m} \vec{v}_{w_j^B} \tag{11}$$

$$K^1(D_A, D_B) = \langle \vec{dv}^{\,A} \cdot \vec{dv}^{\,B} \rangle \tag{12}$$

By substituting values from 10 and 11 to 12, we will get

$$K^1(D_A, D_B) = \langle \frac{1}{n}\sum_{i=1}^{n} \vec{v}_{w_i^A} \cdot \frac{1}{m}\sum_{j=1}^{m} \vec{v}_{w_j^B} \rangle \tag{13}$$

$$K^1(D_A, D_B) = \frac{1}{nm}\langle \sum_{i=1}^{n} \vec{v}_{w_i^A} \cdot \sum_{j=1}^{m} \vec{v}_{w_j^B} \rangle \tag{14}$$

Using the distributive property of dot product

$$\langle \sum_{i=1}^{n} \vec{v}_{w_i^A} \cdot \sum_{j=1}^{m} \vec{v}_{w_j^B} \rangle = \sum_{i=1}^{n}\sum_{j=1}^{m} \langle \vec{v}_{w_i^A} \cdot \vec{v}_{w_j^B} \rangle \tag{15}$$

By substituting 15 in 14, we will get

$$K^1(D_A, D_B) = \frac{1}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m}\langle \vec{v}_{w_i^A} \cdot \vec{v}_{w_j^B} \rangle \tag{16}$$

2. $K^2(D_A, D_B)$ represents the document similarity between the documents represented by topical word vectors (Liu, Qiu, and Huang 2015)

Proof:

$$\vec{dv}^{\,A} = \frac{1}{n}\sum_{i=1}^{n}\vec{tv}_{w_i^A} = \sum_{i=1}^{n}\alpha_{w_i^A} \oplus \vec{v}_{w_i^A} \tag{17}$$

$$\vec{dv}^{\,B} = \frac{1}{m}\sum_{j=1}^{m}\vec{tv}_{w_i^B} = \sum_{j=1}^{m}\alpha_{w_i^B} \oplus \vec{v}_{w_i^B} \tag{18}$$

$$K^2(D_A, D_B) = \langle \vec{dv}^{\,A} \cdot \vec{dv}^{\,B} \rangle \tag{19}$$

By substituting values from 17 and 18 to 19, we will get

$$K^2(D_A, D_B) = \frac{1}{nm}\langle \sum_{i=1}^{n}\vec{tv}_{w_i^A} \cdot \sum_{j=1}^{m}\vec{tv}_{w_j^B} \rangle \tag{20}$$

Using the distributive property of dot product

$$K^2(D_A, D_B) = \frac{1}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m}\langle \vec{tv}_{w_i^A} \cdot \vec{tv}_{w_j^B} \rangle \tag{21}$$

$$K^2(D_A, D_B) = \frac{1}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m}\left\langle \left(\alpha_{w_i^A} \oplus \vec{v}_{w_i^A}\right) \cdot \left(\alpha_{w_i^B} \oplus \vec{v}_{w_i^B}\right) \right\rangle \tag{22}$$

Using the distributive and scalar multiplication property of dot product

$$\left\langle \left(\alpha_{w_i^A} \oplus \vec{v}_{w_i^A}\right) \cdot \left(\alpha_{w_i^B} \oplus \vec{v}_{w_i^B}\right) \right\rangle = \langle \alpha_{w^A} \cdot \alpha_{w^B} \rangle + \langle \vec{v}_{w_i^A} \cdot \vec{v}_{w_i^B} \rangle \tag{23}$$

Since, $t_{w^A} = \alpha_{w^A}$ and $t_{w^B} = \alpha_{w^B}$

$$\left\langle \left(\alpha_{w_i^A} \oplus \vec{v}_{w_i^A}\right) \cdot \left(\alpha_{w_i^B} \oplus \vec{v}_{w_i^B}\right) \right\rangle = \langle t_{w^A} \cdot t_{w^B} \rangle + \langle \vec{v}_{w_i^A} \cdot \vec{v}_{w_i^B} \rangle \tag{24}$$

By substituting 24 in 22, we will get

$$K^2(D_A, D_B) = \frac{1}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m}\langle \vec{v}_{w_i^A} \cdot \vec{v}_{w_j^B} \rangle + \langle \vec{t}_{w_i^A} \cdot \vec{t}_{w_j^B} \rangle \tag{25}$$

3. $K^3(D_A, D_B)$ represents the document similarity between the documents represented by partition average word vectors (P-SIF)

Proof:

$$\vec{dv}^{\,A} = \frac{1}{n}\sum_{i=1}^{n}\vec{tv}_{w_i^A} = \sum_{i=1}^{n}\bigoplus_{k=1}^{K}\alpha_{w_i^A,k}\vec{v}_{w_i^A} \tag{26}$$

$$\vec{dv}^{\,B} = \frac{1}{m}\sum_{j=1}^{m}\vec{tv}_{w_i^B} = \sum_{j=1}^{m}\bigoplus_{k=1}^{K}\alpha_{w_j^B,k}\vec{v}_{w_i^B} \tag{27}$$

$$K^3(D_A, D_B) = \langle \vec{dv}^{\,A} \cdot \vec{dv}^{\,B} \rangle \tag{28}$$

By substituting values from 26 and 27 in 28, we will get

$$K^3(D_A, D_B) = \frac{1}{nm}\langle \sum_{i=1}^{n}\vec{tv}_{w_i^A} \cdot \sum_{j=1}^{m}\vec{tv}_{w_j^B} \rangle \tag{29}$$

Using the distributive property of dot product

$$K^3(D_A, D_B) = \frac{1}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m}\langle \vec{tv}_{w_i^A} \cdot \vec{tv}_{w_j^B} \rangle \tag{30}$$

$$= \frac{1}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m}\left\langle \left(\bigoplus_{k=1}^{K}\alpha_{w_i^A,k}\vec{v}_{w_i^A}\right) \cdot \left(\bigoplus_{k=1}^{K}\alpha_{w_j^B,k}\vec{v}_{w_i^B}\right) \right\rangle \tag{31}$$

Using distributive and scalar multiplication property of dot product

$$\begin{aligned}&\left\langle \left(\bigoplus_{k=1}^{K}\alpha_{w_i^A,k}\vec{v}_{w_i^A}\right) \cdot \left(\bigoplus_{k=1}^{K}\alpha_{w_j^B,k}\vec{v}_{w_j^B}\right) \right\rangle \\ &= \left(\sum_{k=1}^{K}(\alpha_{w_i^A,k} \cdot \alpha_{w_j^B,k})\right) \times \langle \vec{v}_{w_i^A} \cdot \vec{v}_{w_i^B} \rangle\end{aligned} \tag{32}$$

Since, $t_{w^A,k} = \alpha_{w^A,k}$ and $t_{w^B,k} = \alpha_{w^B,k}$

$$\begin{aligned}&\left\langle \left(\bigoplus_{k=1}^{K}\alpha_{w^A,k}\vec{v}_{w_i^A}\right) \cdot \left(\bigoplus_{k=1}^{K}\alpha_{w^B,k}\vec{v}_{w_i^B}\right) \right\rangle \\ &= \left(\sum_{k=1}^{K}(t_{w_i^A,k} \cdot t_{w_j^B,k})\right) \times \langle \vec{v}_{w_i^A} \cdot \vec{v}_{w_i^B} \rangle\end{aligned} \tag{33}$$

$$\begin{aligned}&\left\langle \left(\bigoplus_{k=1}^{K}\alpha_{w^A,k}\vec{v}_{w_i^A}\right) \cdot \left(\bigoplus_{k=1}^{K}\alpha_{w^B,k}\vec{v}_{w_i^B}\right) \right\rangle \\ &= \left(\sum_{k=1}^{K}t_{w_i^A,k} \cdot t_{w_j^B,k}\right) \times \langle \vec{v}_{w_i^A} \cdot \vec{v}_{w_i^B} \rangle\end{aligned} \tag{34}$$

From the definition of the dot product of vectors,

$$\begin{aligned}&\left\langle \left(\bigoplus_{k=1}^{K}\alpha_{w^A,k}\vec{v}_{w_i^A}\right) \cdot \left(\bigoplus_{k=1}^{K}\alpha_{w^B,k}\vec{v}_{w_i^B}\right) \right\rangle \\ &= \langle \vec{t}_{w_i^A} \cdot \vec{t}_{w_j^B} \rangle \times \langle \vec{v}_{w_i^A} \cdot \vec{v}_{w_j^B} \rangle\end{aligned} \tag{35}$$

By subatituting 35 in 31, we will get

$$K^3(D_A, D_B) = \frac{1}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m}\langle \vec{v}_{w_i^A} \cdot \vec{v}_{w_j^B} \rangle \times \langle \vec{t}_{w_i^A} \cdot \vec{t}_{w_j^B} \rangle \tag{36}$$

4. $K^4(D_A, D_B)$ represents the document similarity between the documents represented by the relaxed word mover distance (Kusner et al. 2015) when words of D_A are matched to D_B
Proof: From the definition of the relaxed word mover distance in (Kusner et al. 2015). Relaxed word mover maps each word

w_i^A of document D_A to the closest word w_j^B of document D_B.

Since, word w_j^B of document D_B is closer to word w_i^A of document D_A, we will have

$$w_j^B = \arg\max_{w_j^B} \langle \vec{v}_{w_i^A} \cdot \vec{v}_{w_j^B} \rangle \tag{37}$$

Therefore, similarity contribution $K_{(i,j)}$ from word w_i^A of document D_A is given by:

$$K_{(i,j)} = \frac{1}{n} \max_{w_j^B} \langle \vec{v}_{w_i^A} \cdot \vec{v}_{w_j^B} \rangle \tag{38}$$

Total similarity contribution from all the n words of the document d_A:

$$K^4(D_A, D_B) = \frac{1}{n} \sum_{i=1}^{n} K_{(i,j)} = \frac{1}{n} \sum_{i=1}^{n} \max_{w_j^B} \langle \vec{v}_{w_i^A} \cdot \vec{v}_{w_j^B} \rangle \tag{39}$$

We can write $\max_{w_j^B}$ as $\max_j$, thus finally

$$K^4(D_A, D_B) = \frac{1}{n} \sum_{i=1}^{n} \max_{j} \langle \vec{v}_{w_i^A} \cdot \vec{v}_{w_j^B} \rangle \tag{40}$$

Qualitative Example: Document Similarity

Let's consider a corpus (C) with N documents with the corresponding most frequent vocabulary (V). Figure 3 represents the word-vectors space V, where similar meaning words are closer. We can apply sparse coding and partition the words-vector space into five (total topics $K = 5$) topic vector spaces. Some words are polysemic and belong to multiple topics with some proportion, as shown in Figure 3. For example, words such as *baby*, *person*, *dog* and *kangaroo*, belong to multiple topics with a significant proportion. Words and corresponding vectors in these topic vector spaces are represented by topic numbers in the subscript. Table 11 shows an example pair from the STS Task 2012 MSRVid dataset and the corresponding SIF (averaging) and P-SIF (partition averaging) representation vectors. We can see that in the SIF representation, we are averaging words vectors which semantically have different meanings. The document is represented in the same d dimensional word-vectors space. Overall, SIF represents the document as a single point in the vector space and does not take account of different semantic meanings of the topics. Whereas, in the P-SIF representation, we treat the five different semantic topics distinctly. Words belonging to different semantic topics are separated by concatenation ($\oplus$) as they represent different meanings, whereas words coming from the same topic are averaged as they represent the same meaning. The final document vector $\vec{v}_{d_n}$ has more representational power as it is represented in a higher $5 \times d$ dimensional vector space. Thus, partitioned averaging with topic weighting is important for representing documents. Empirically, P-SIF assigned a lower score of 0.16 (rescaled to a 0-1 scale) for sentences (d_n^1,d_n^2) where the ground truth is 0.15 (rescaled to a 0-1 scale), whereas SIF gave similarity score of 0.57 (0-1 scale), farther than the ground score. Thus, we obtain a relative improvement of 98% in the error difference from the ground truth. Here, the simple averaging-based embedding of d_n^1 and d_n^2, brings the document representations closer. But partitioned based averaging, P-SIF, projects the documents farther in a higher-dimensional space.

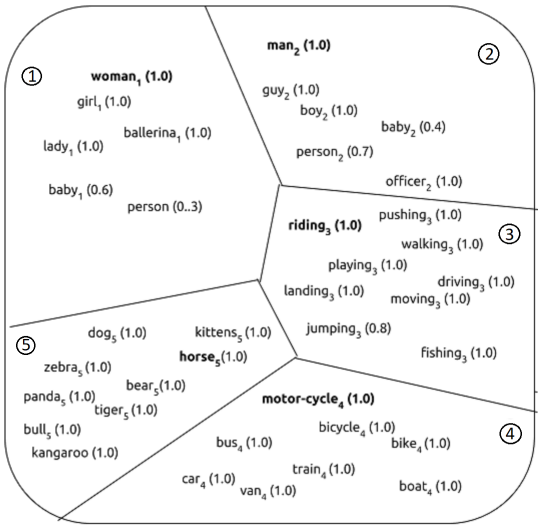

Figure 3: Words in different topics are represented by different subscripts and separated by hyperplanes. Bold represents words from example documents.

Qualitative Results: Similarity task

Table 12 represents successful example pair from STS 2012 MSRvid dataset where P-SIF assigns similarity scores closer to ground truth than SIF. Table 13 represents the failed example pairs from STS 2012 MSRvid dataset where SIF assigns a similarity score closer to the ground truth than P-SIF. We now introduce the header notations used in the Table 12 and 13 in details.

- GT: represents the given ground truth similarity score in a range of 0-5.
- NGT: represents the normalized ground truth similarity score. NGT is obtained by dividing the GT score by 5 so that it is in a range of 0-1.
- SIF$_{sc}$: represents the SIF embedding similarity score in a range of 0-1.
- P-SIF$_{sc}$: represents the P-SIF embedding similarity score in a range of 0-5.
- SIF$_{err}$: represents absolute error $\|SIF_{sc} - NGT\|$ between normalized ground truth similarity score and the SIF embedding similarity score.
- P-SIF$_{err}$: represents the absolute error $\|\text{P-SIF}_{sc} - NGT\|$ between the ground truth similarity score and the P-SIF embedding similarity score.
- Diff$_{err}$: represents absolute difference between SIF$_{err}$ and P-SIF$_{err}$. Examples where P-SIF performs better Diff$_{err}$ = P-SIF$_{err}$ - SIF$_{err}$ (used in Table 12). Examples where SIF performs better Diff$_{err}$ = SIF$_{err}$ - P-SIF$_{err}$ (used in Table 13)
- Rel$_{err}$: represents relative difference between SIF$_{err}$ and P-SIF$_{err}$. Examples where P-SIF performs better Rel$_{err}$ = $\frac{Diff_{err}}{SIF_{err}}$ (used in Table 12). Examples where SIF performs better Rel$_{err}$ = $\frac{Diff_{err}}{\text{P-SIF}_{err}}$ (used in Table 13)

Figure 4 shows the code flow architecture of our proposed P-SIF embeddings.

Table 11: STS Task 2012 MSRVid dataset similarity example pair. Here, P-SIF assigns a score of 0.16 (rescaled to a 0-1 scale) to sentences (d_n^1,d_n^1), where the ground truth of 0.15 (0-1 scale), whereas SIF assigns a similarity score of 0.57 (rescaled to a 0-1 scale). Thus, we obtain a relative improvement of 98% in the error difference. Here, $\oplus$ represents concatenation. $\vec{v}_{\text{zero}}$ is the zero padding vector.

	Document 1 (d_n^1)	Document 2 (d_n^2)	Score
Doc	A man is riding a motorcycle	A woman is riding a horse	0.15
SIF	$\vec{v}_{\text{man}_2} + \vec{v}_{\text{riding}_3} + \vec{v}_{\text{motorcycle}_4}$	$\vec{v}_{\text{woman}_1} + \vec{v}_{\text{riding}_3} + \vec{v}_{\text{horse}_5}$	0.57
P-SIF	$\vec{v}_{\text{zero}_1} \oplus \vec{v}_{\text{man}_2} \oplus \vec{v}_{\text{riding}_3} \oplus \vec{v}_{\text{motorcycle}_4} \oplus \vec{v}_{\text{zero}_5}$	$\vec{v}_{\text{women}_1} \oplus \vec{v}_{\text{zero}_2} \oplus \vec{v}_{\text{riding}_3} \oplus \vec{v}_{\text{zero}_4} \oplus \vec{v}_{\text{horse}_5}$	0.16

Table 12: STS 2012 MSRVid example where the P-SIF scores were closer to the ground truth, whereas SIF scores were more away from the ground truth

sentence1	sentence2	GT	NGT	SIF$_{sc}$	P-SIF$_{sc}$	SIF$_{err}$	P-SIF$_{err}$	Diff$_{err}$	Rel$_{err}$
People are playing baseball .	The cricket player hit the ball .	0.5	0.1	0.2928	0.0973	0.1928	0.0027	0.1901	0.986
A woman is carrying a boy .	A woman is carrying her baby .	2.333	0.4666	0.5743	0.4683	0.1077	0.0017	0.106	0.9843
A man is riding a motorcycle .	A woman is riding a horse .	0.75	0.15	0.5655	0.157	0.4155	0.007	0.4085	0.9833
A woman slices a lemon .	A man is talking into a microphone .	0	0	-0.1101	-0.0027	0.1101	0.0027	0.1074	0.9754
A man is hugging someone .	A man is taking a picture .	0.4	0.08	0.2021	0.0767	0.1221	0.0033	0.1188	0.9731
A woman is dancing .	A woman plays the clarinet .	0.8	0.16	0.3539	0.1653	0.1939	0.0053	0.1886	0.9727
A train is moving .	A man is doing yoga .	0	0	0.1674	-0.0051	0.1674	0.0051	0.1623	0.9695
Runners race around a track .	Runners compete in a race .	3.2	0.64	0.7653	0.6438	0.1253	0.0038	0.1214	0.9694
A man is driving a car .	A man is riding a horse .	1.2	0.24	0.3584	0.2443	0.1184	0.0043	0.114	0.9636
A man is playing a guitar .	A woman is riding a horse .	0.5	0.1	-0.0208	0.0955	0.1208	0.0045	0.1163	0.9629
A man is riding on a horse .	A girl is riding a horse .	2.6	0.52	0.6933	0.5082	0.1733	0.0118	0.1615	0.9319
A woman is deboning a fish .	A man catches a fish .	1.25	0.25	0.4538	0.2336	0.2038	0.0164	0.1875	0.9196
A man is playing a guitar .	A man is eating pasta .	0.533	0.1066	-0.0158	0.0962	0.1224	0.0104	0.112	0.915
A woman is dancing .	A man is eating .	0.143	0.0286	-0.1001	0.0412	0.1287	0.0126	0.1161	0.9023
The ballerina is dancing .	A man is dancing .	1.75	0.35	0.512	0.3317	0.162	0.0183	0.1437	0.8871
A woman plays the guitar .	A man sings and plays the guitar .	1.75	0.35	0.5036	0.3683	0.1536	0.0183	0.1353	0.8807
A girl is styling her hair .	A girl is brushing her hair .	2.5	0.5	0.7192	0.5303	0.2192	0.0303	0.1889	0.8618
A guy is playing hackysack	A man is playing a key-board .	1	0.2	0.3718	0.2268	0.1718	0.0268	0.145	0.8441
A man is riding a bicycle .	A monkey is riding a bike .	2	0.4	0.6891	0.4614	0.2891	0.0614	0.2277	0.7876
A woman is swimming underwater .	A man is slicing some carrots .	0	0	-0.2158	-0.0562	0.2158	0.0562	0.1596	0.7397
A plane is landing .	A animated airplane is landing .	2.8	0.56	0.801	0.6338	0.241	0.0738	0.1672	0.6937
The missile exploded .	A rocket exploded .	3.2	0.64	0.8157	0.6961	0.1757	0.0561	0.1196	0.6806
A woman is peeling a potato .	A woman is peeling an apple .	2	0.4	0.6938	0.5482	0.2938	0.1482	0.1456	0.4956
A woman is writing .	A woman is swimming .	0.5	0.1	0.3595	0.2334	0.2595	0.1334	0.1261	0.4859
A man is riding a bike .	A man is riding on a horse .	2	0.4	0.6781	0.564	0.2781	0.164	0.1142	0.4105
A panda is climbing .	A man is climbing a rope .	1.6	0.32	0.4274	0.3131	0.1074	0.0069	0.1005	0.9361
A man is shooting a gun .	A man is spitting .	0	0	0.2348	0.1305	0.2348	0.1305	0.1043	0.444

Table 13: STS 2012 MSRVid examples where the P-SIF score were far away from the ground truth, whereas the SIF scores were closer to the actual ground truth

sentence1	sentence2	GT	NGT	SIF$_{sc}$	P-SIF$_{sc}$	SIF$_{err}$	P-SIF$_{err}$	Diff$_{err}$	Rel$_{err}$
takes off his sunglasses .	A boy is screaming .	0.5	0.1	0.1971	0.3944	0.0971	0.2944	0.1973	0.6703
The rhino grazed on the grass .	A rhino is grazing in a field .	4	0.8	0.7275	0.538	0.0725	0.262	0.1895	0.7234
An animal is biting a persons finger .	A slow loris is biting a persons finger .	3	0.6	0.6018	0.7702	0.0018	0.1702	0.1684	0.9892
Animals are playing in water .	Two men are playing ping pong .	0	0	0.0706	0.2238	0.0706	0.2238	0.1532	0.6846
Someone is feeding a animal .	Someone is playing a piano .	0	0	-0.0037	0.1546	0.0037	0.1546	0.1509	0.976
The lady sliced a tomatoe .	Someone is cutting a tomato .	4	0.8	0.693	0.5591	0.107	0.2409	0.1339	0.5559
The lady peeled the potatoe .	A woman is peeling a potato .	4.75	0.95	0.7167	0.5925	0.2333	0.3575	0.1242	0.3474
A man is slicing something .	A man is slicing a bun .	3	0.6	0.5976	0.4814	0.0024	0.1186	0.1162	0.9802
A boy is crawling into a dog house .	A boy is playing a wooden flute .	0.75	0.15	0.1481	0.2674	0.0019	0.1174	0.1155	0.9839
A man and woman are talking .	A man and woman is eating .	1.6	0.32	0.3574	0.4711	0.0374	0.1511	0.1137	0.7527
A man is cutting a potato .	A woman plays an electric guitar .	0.083	0.0166	-0.1007	-0.2128	0.1173	0.2294	0.112	0.4884
A person is cutting a meat .	A person riding a mechanical bull	0	0	0.0152	0.1242	0.0152	0.1242	0.1091	0.8778
A woman is playing the flute .	A man is playing the guitar .	1	0.2	0.1942	0.0876	0.0058	0.1124	0.1065	0.948

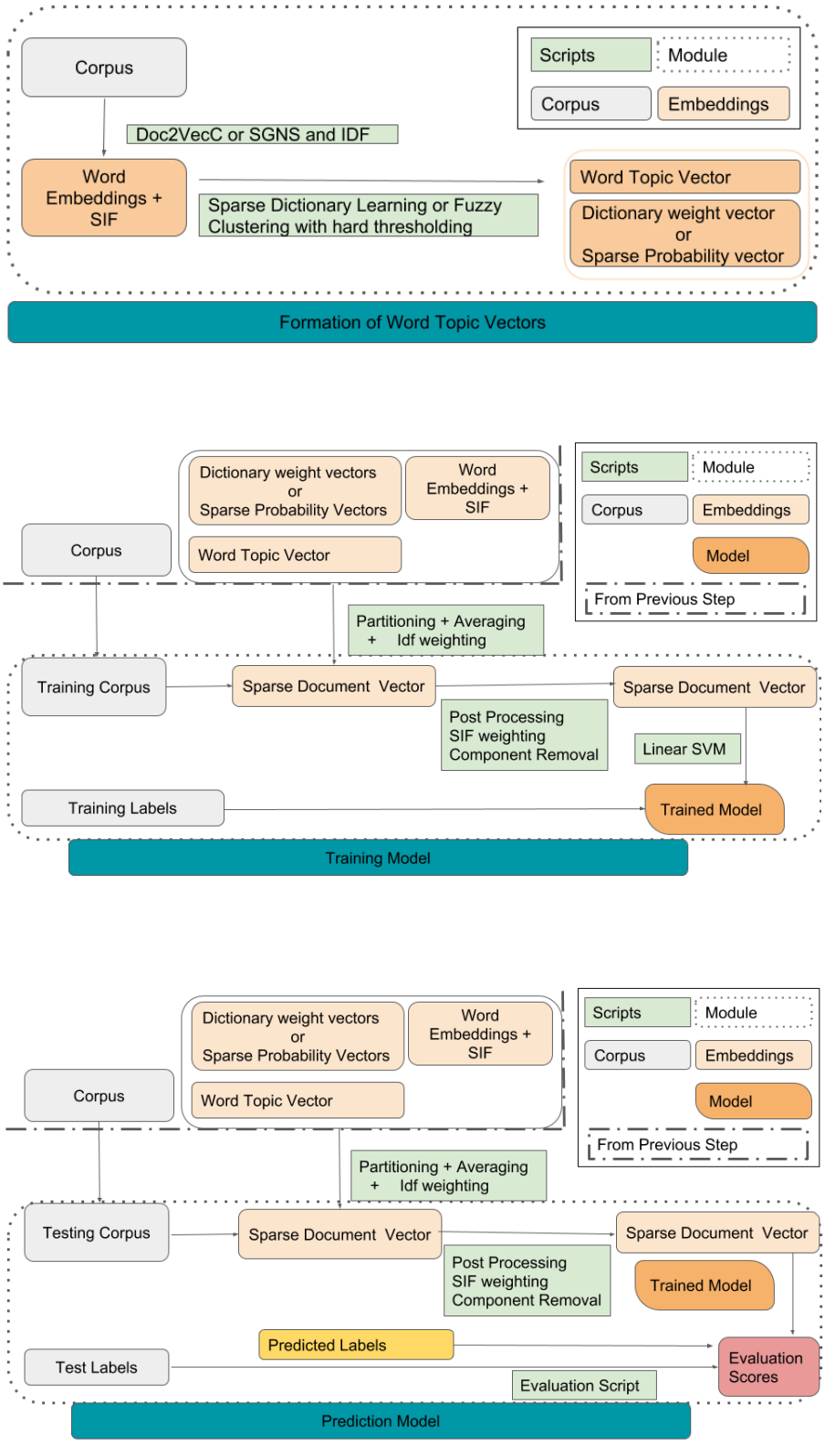

Figure 4: Overview of code flow architecture of P-SIF.