PAC-Bayes Analysis of Transferred Sentence Vectors

Kento Nozawa
The University of Tokyo & RIKEN
nozawa@ms.k.u-tokyo.ac.jp
https://nzw0301.github.io

Issei Sato
The University of Tokyo & RIKEN
sato@k.u-tokyo.ac.jp

December 5, 2018

Abstract

Learning sentence vectors from an unlabeled corpus have attracted attention because such vectors are able to represent sentences flexibility. Simple heuristic methods using pre-trained word vectors are strong baselines for downstream machine learning tasks. However, they are not well understood from a theoretical perspective. We analyze sentence vector algorithms from a transfer learning perspective by using a PAC-Bayes bound, which enables us to understand existing heuristic methods and provides novel sentence vector algorithms. Note that this manuscript is an extended version of Nozawa and Sato (2018).

1 Preliminaries

1.1 Learning Word Vectors and Sentence Vectors

1.1.1 Neurl Word Vector Model: Continuous Skip-gram Model with Negative Sampling

Word vector models aim to learn a map from each word w to d-dimensional vector representation $\mathbf{h}_w$ given a sequence of words $[w_1, \dots, w_T]$. This word vector is also called *word embedding*. In this paper, we mainly treat the continuous skip-gram model (skip-gram) (Mikolov et al., 2013a), which is a widely used word vector model. Skip-gram learns two types of vectors $\mathbf{h}$ and $\mathbf{h}'$. We call $\mathbf{h}$ as an input word vector and $\mathbf{h}'$ as an output word vector. The terms $\hat{\mathbf{h}}$ and $\hat{\mathbf{h}}'$ define fixed input word vector and fixed output word vector trained on a corpus, respectively.

Intuitively, skip-gram estimates word vectors by predicting word w_c from its neighbor w_t. A fast skip-gram model training method is negative sampling proposed by Mikolov et al. (2013b), whose loss function is defined by

$$L_{SG} = -\sum_{t=1}^{T} \left[\sum_{w_c \in \mathcal{C}_t} \ln \sigma \left(\mathbf{h}_{w_t}^{\top} \mathbf{h}'_{w_c} \right) + \sum_{w_n \in \mathcal{NS}} \ln \sigma \left(-\mathbf{h}_{w_t}^{\top} \mathbf{h}'_{w_n} \right) \right], \tag{1}$$

where $\mathcal{C}_t$ represents a bag-of-words surrounding w_t in a sequence, σ is the logistic sigmoid function, and $\mathcal{NS}$ is a bag-of-words including negative words sampled from a noise distribution (Mikolov et al., 2013b).

1.1.2 Continuous Bag-of-Words with Negative Sampling

Continuous bag-of-word model (CBoW) (Mikolov et al., 2013a) is a similar model to skip-gram. CBoW estimates word vectors by predicting word w_t from its neighbors $\mathcal{C}_t$. Formally, CBoW with negative sampling minimizes the loss function below

$$L_{CBoW} = -\sum_{t=1}^{T} \left[\ln \sigma \left(\mathbf{h}_{\mathrm{avg}_t}^{\top} \mathbf{h}'_{w_t} \right) + \sum_{w_n \in \mathcal{NS}} \ln \sigma \left(-\mathbf{h}_{\mathrm{avg}_t}^{\top} \mathbf{h}'_{w_n} \right) \right], \tag{2}$$

where $\mathbf{h}_{\mathrm{avg}_t} = \frac{1}{|\mathcal{C}_t|} \sum_{w \in \mathcal{C}_t} \mathbf{h}_w$.

1.1.3 Sentence Vectors from Pre-trained Word Vectors

In the same way as word vector models, sentence vector models aim to learn a map from a sentence[1] from d-dimensional vector representation $\mathbf{h}_{\mathcal{S}}$ given sentences $\{\mathcal{S}_1, \dots, \mathcal{S}_N\}$, where $\mathcal{S}$ is a sequence of words. Since sentences consist of several words, sentence representation is affected by word information, such as sentiment polarity. Given pre-trained word vectors, the simplest way to obtain sentence vector $\mathbf{h}_{\mathcal{S}}$ is to average of pre-trained word vectors of words appearing in the sentence $\mathcal{S}$, e.g., $\mathbf{h}_{\mathcal{S}} = \frac{1}{|\mathcal{S}|}\sum_{w \in \mathcal{S}} \hat{\mathbf{h}}_w$, or $\mathbf{h}_{\mathcal{S}} = \frac{1}{|\mathcal{S}|}\sum_{w \in \mathcal{S}} \frac{\hat{\mathbf{h}}_w + \hat{\mathbf{h}}'_w}{2}$. Empirically, these simple methods can beat more complex methods in machine learning tasks for sentences (Hill et al., 2016; Wieting et al., 2016).

1.2 PAC-Bayes Bound

We introduce a PAC-Bayes bound used in our analysis. Let $\mathcal{D}$ be a data distribution over $\mathcal{X} \times \mathcal{Y}$, where $\mathbf{x} \in \mathcal{X}$ is the input sample and $y \in \mathcal{Y}$ is the output sample. Let $\mathcal{D}_S$ be a training dataset sampled i.i.d. from $\mathcal{D}$, and N be the number of training samples. Let $\mathcal{P}$ be a prior distribution over a hypotheses class $\mathcal{F}$, and $\mathcal{Q}$ be a posterior distribution over $\mathcal{F}$. Given the data distribution and stochastic hypotheses, the generalization risk is defined as $R_{\mathcal{D}}(\mathcal{Q}) = \mathbb{E}_{h \sim \mathcal{Q}} \mathbb{E}_{(\mathbf{x},y) \sim \mathcal{D}}\, l(\mathbf{x}, y, h)$, where l is a bounded loss function in $[0, l_{\max}]$. In the same way, the empirical risk is defined as $\hat{R}_{\mathcal{D}_{\mathcal{S}}}(\mathcal{Q}) = \mathbb{E}_{h \sim \mathcal{Q}} \frac{1}{N}\sum_{i=1}^{N} l(\mathbf{x}_i, y_i, h)$. Kullback-Leibler (KL) divergence is defined as $\mathrm{KL}(\mathcal{Q}||\mathcal{P}) = \mathbb{E}_{h \sim \mathcal{Q}} \log \frac{\mathcal{Q}(h)}{\mathcal{P}(h)}$.

Theorem 1 (PAC-Bayes Bound (Catoni, 2007, Theorem 1.2.6)). $\forall \lambda > 0$, *with probability at least* $1 - \delta$ *over training samples* $\mathcal{D}_{\mathcal{S}}$*,* $\forall h \sim \mathcal{Q}$*,*

$$R_D(\mathcal{Q}) \le \frac{1}{1 - \exp(-\frac{\lambda}{N})}\left(1 - \exp\left[-\frac{\lambda}{N}\hat{R}_{D_S}(\mathcal{Q}) - \frac{\mathrm{KL}[\mathcal{Q}||\mathcal{P}] + \log(1/\delta)}{N}\right]\right). \tag{3}$$

2 PAC-Bayesian Analysis of Transferred Sentence Vectors

We typically use pre-trained word vectors to learn sentence vectors because of the usefulness of their representations, which capture syntactic and semantics (Mikolov et al., 2013b). We formulate learning word vectors and sentence vectors in terms of transfer learning (Pan and Yang, 2010). In our transfer learning perspective, a source task minimizes the loss function of skip-gram with negative sampling on word sequences by updating input and output word vectors $\mathbf{h}$ and $\mathbf{h}'$. Then, a target task minimizes the loss function by updating sentence vector $\mathbf{h}_{\mathcal{S}}$ with the fixed pre-trained word vectors $\hat{\mathbf{h}}$ and $\hat{\mathbf{h}}'$ given a sentence.

This formulation enables us to analyze generalization errors in learning sentence vector with PAC-Bayesian theory, which can consider transferability to a target hypothesis from a learned source hypothesis through prior knowledge (McNamara and Balcan, 2017).

2.1 PAC-Bayes Bound

We define a PAC-Bayes bound to analyze sentence vector algorithms based on averaging of pre-trained word vectors. Given a sentence $\mathcal{S}$ and word vectors trained on the source task, let $\mathcal{D}$ be a data distribution over $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X} := \{\mathbf{x}_w | w \in \mathbb{N}_{\le|\mathcal{V}|}\}$ is input sample representing one-hot vector of word w, and $\mathcal{Y}$ is output space, e.g., $\{-1, 1\}$ for binary classification. $\mathcal{D}_S$ is a training dataset sampled i.i.d. from $\mathcal{D}$, that means sentence $\mathcal{S}$ assumes bag-of-words. Let $|\mathcal{D}_{\mathcal{S}}|$ be the number of training samples, and $|\mathcal{S}|$ be the number of words in $\mathcal{S}$. Our goal is to minimize generalization error $R_{\mathcal{D}}(\mathcal{Q}_{\mathcal{S}}) = \mathbb{E}_{h \sim \mathcal{Q}_{\mathcal{S}}} \mathbb{E}_{(\mathbf{x},y) \sim \mathcal{D}}\, l(\mathbf{x}, y, h)$, where l is a bounded loss function, and h is hypothesis parameterized by sentence vector $\mathbf{h}_{\mathcal{S}}$. We give a concrete form of y, l, and h in a later analysis. We define prior distribution over hypothesis class $\mathcal{F}$ as $\mathcal{P}_{\mathcal{S}} \propto \prod_{w \in \mathcal{S}} \mathcal{N}(;\hat{\mathbf{h}}'_w, \sigma^2 I)$, where σ^2 is variance parameter, and I is identity matrix. This prior distribution is known as the product of experts (Hinton, 2002) such that each expert is a Gaussian distribution parameterized by pre-trained $\hat{\mathbf{h}}'$ of word appearing in the given sentence $\mathcal{S}$. We also define posterior distribution over $\mathcal{F}$ as $\mathcal{Q}_{\mathcal{S}} = \mathcal{N}(;\mathbf{h}_{\mathcal{S}}, \sigma^2 I)$. Kullback-Leibler (KL) divergence is defined as $\mathrm{KL}(\mathcal{Q}||\mathcal{P}) = \mathbb{E}_{h \sim \mathcal{Q}} \log \frac{\mathcal{Q}(h)}{\mathcal{P}(h)}$. Theorem 2 shows the upper bound of generalization error of target task by empirical error of target task with the regularizer by pre-trained output word vectors.

[1] We name a sequence of words as a *sentence* including phrase, paragraph, and document.

Theorem 2 (PAC-Bayes Bound of Learning Sentence Vector with Pre-trained Word Vectors)**.** *Given a sentence $\mathcal{S}$, $\forall\lambda > 0$, with probability at least $1 - \delta$ over training samples $\mathcal{D}_{\mathcal{S}}$, $\forall h \in \mathcal{Q}_{\mathcal{S}}$,*

$$R_{\mathcal{D}}(\mathcal{Q}_{\mathcal{S}}) \leq \mathop{\mathbb{E}}_{h\sim\mathcal{Q}_{\mathcal{S}}} \frac{1}{|\mathcal{D}_{\mathcal{S}}|}\sum_{i=1}^{|\mathcal{D}_{\mathcal{S}}|} l(\mathbf{x}_i, y_i, h) + \frac{\lambda}{2\sigma^2}\left|\left|\mathbf{h}_{\mathcal{S}} - \frac{1}{|\mathcal{S}|}\sum_{w\in\mathcal{S}}\hat{\mathbf{h}}'_w\right|\right|_2^2 + C, \tag{4}$$

where C is a constant term that does not depends sentence vector $\mathbf{h}_{\mathcal{S}}$.

We prove Theorem 2 by using the PAC-Bayes bound (Catoni, 2007). The proof is shown in Appendix A.1.

2.2 Sentence Vectors via Regression

We use Theorem 2 to analyze a heuristic sentence vector models. Let $\mathcal{Y} := \{\hat{\mathbf{h}}_w \mid w \in \mathcal{V}\}$, hypothesis h be $h(\mathbf{x}) = \mathbf{h}_{\mathcal{S}}$. We introduce the loss function $l(\mathbf{x}, y, h) = \frac{1}{2}||y - \mathbf{h}_{\mathcal{S}}||_2^2$.[2] We can obtain the closed form of $\mathbf{h}_{\mathcal{S}}$ from Eq. (4),

$$\mathbf{h}_{\mathcal{S}} = \frac{1}{(1+\alpha)|\mathcal{S}|}\sum_{w\in\mathcal{S}}\left(\hat{\mathbf{h}}_w + \alpha\hat{\mathbf{h}}'_w\right), \tag{5}$$

where $\alpha = \frac{\lambda}{\sigma^2}$. The details of this solution are shown in Appendix A.2.

Corollary 2.1 (Sentence Vector by Averaging of Pre-trained Input Word Vectors)**.** *The sentence vector $\mathbf{h}_{\mathcal{S}}$, estimated by minimizing Eq.* (4) *with $\alpha = 0$, is equivalent to $\frac{1}{|\mathcal{S}|}\sum_{w\in\mathcal{S}}\hat{\mathbf{h}}_w$.*

Corollary 2.2 (Sentence Vector by Averaging of Pre-trained Input and Output Word Vectors)**.** *Empirically, the averaged vector of the input word vector and output word vector, $\frac{\hat{\mathbf{h}}+\hat{\mathbf{h}}'}{2}$, can improve the performance of downstream tasks (e.g., Levy et al. (2015)). This operation corresponds to the solution of Eq.* (4) *with the prior defined by the pre-trained output vectors and $\alpha = 1$.*

We also introduce another heuristic method with inverse document frequency (IDF) weighting (The details are in Appendix A.3). We change loss function to sample-dependent weighted loss function $l(\mathbf{x}, y, h) = \beta(\mathbf{x})h(\mathbf{x}) - y = \beta(\mathbf{x})h(\mathbf{x})$, where β is an weighing function from one-hot vector to $\mathbb{R}_+$. We also change the prior to $\mathcal{P}_{\mathcal{S}} \propto \prod_{w\in\mathcal{S}}\mathcal{N}(;\hat{\mathbf{h}}'_w, \sigma_w^2 I)$. When we set $\beta(\mathbf{x}) = |\mathcal{S}|\mathrm{IDF}(\mathbf{x})$ and $1/\sigma_w^2 = \mathrm{IDF}(w)$, we can obtain the closed form of $\mathbf{h}_{\mathcal{S}}$ from Eq. (4),

$$\mathbf{h}_{\mathcal{S}} = \frac{1}{(1+\lambda)\sum_{w\in\mathcal{S}}\mathrm{IDF}(w)}\sum_{w\in\mathcal{S}}\mathrm{IDF}(w)\left(\hat{\mathbf{h}}_w + \lambda\hat{\mathbf{h}}'_w\right). \tag{6}$$

2.3 Sentence Vectors via Binary Classification

From PAC-Bayes analysis, we found that the problems, i.e., loss function, differ between the source and target tasks in existing heuristics methods. From a transfer learning perspective, the target task's loss value tends to decrease easily if the source and target tasks are similar, i.e., loss function, data distribution, and task. We now consider that loss function for learning sentence representations is the same as that for learning word representations.[3] The new target task is to predict whether a sentence contains word w_t instead of predicting whether word w_c appears in the neighbor of w_t. We expect that sentence vectors are closed in the embedded space if sentences share similar meanings when generalization errors are sufficiently small.

We apply Theorem 2 to the task defined above. We change the output space to $\mathcal{Y} := \{-1, 1\}$, the loss function to zero-one loss $l(\mathbf{x}, y, h) = \mathbb{I}[h(\mathbf{x}) \neq y]$, and the hypothesis to $h(\mathbf{x}_w) = \mathrm{sign}((\hat{\mathbf{H}}\mathbf{x}_w)^\top\mathbf{h}_{\mathcal{S}})$, where the w-th column of $\hat{\mathbf{H}} \in \mathbb{R}^{d\times|\mathcal{V}|}$ corresponds to the w-th pre-trained input word vector $\hat{\mathbf{h}}_w$. In practice, we minimize the following loss function using negative sampling as surrogate loss function of zero-one loss.

$$L(\mathcal{D}_S, \mathbf{h}_{\mathcal{S}}) = -\frac{1}{|\mathcal{S}|}\left[\sum_{w\in\mathcal{S}}\ln\sigma\left(\hat{\mathbf{h}}_w^\top\mathbf{h}_{\mathcal{S}}\right) + \sum_{w_n\in\mathcal{NS}}\ln\sigma\left(-\mathbf{h}_{w_n}^\top\mathbf{h}_{\mathcal{S}}\right)\right] + \frac{\lambda}{2\sigma^2}\left|\left|\mathbf{h}_{\mathcal{S}} - \frac{1}{|\mathcal{S}|}\sum_{w\in\mathcal{S}}\hat{\mathbf{h}}'_w\right|\right|_2^2. \tag{7}$$

[2] When we want to evaluate the upper bound, bounded loss l is in $[0, 2]$ if we normalize each pre-trained word vectors and sentence vector by L2-norm.

[3] Negative sampling loss works as a surrogate loss of zero-one loss.

Corollary 2.3 (Relationship to Paragraph Vector Models). *PV-DBoW, a instance of paragraph vector models proposed by Le and Mikolov (2015), is the same to Eq.* (7) *with* $\lambda = 0$ *and without pre-trained word vectors.*

3 Other Transfer Learning Settings

3.1 Target Task with Pre-trained CBoW Vectors

In the same way as skip-gram model, we define hypothesis $h(\mathcal{S})$ as $\mathrm{sign}(\hat{\mathbf{h}}_{\mathrm{avg}_\mathcal{S}}^\top \mathbf{h}_\mathcal{S})$, where $\hat{\mathbf{h}}_{\mathrm{avg}_\mathcal{S}} = \frac{1}{|\mathcal{S}|}\sum_{w \in \mathcal{S}} \hat{\mathbf{h}}_w$. Given a sentence $\mathcal{S}$, we minimize the loss function defined by

$$L(\mathcal{D}_S, \mathbf{h}_\mathcal{S}) = -\frac{1}{|\mathcal{S}|}\left[\ln\sigma\left(\hat{\mathbf{h}}_{\mathrm{avg}_\mathcal{S}}^\top \mathbf{h}_\mathcal{S}\right) + \sum_{w_n \in \mathcal{NS}} \ln\sigma\left(-\hat{\mathbf{h}}_{w_n}\mathbf{h}_\mathcal{S}\right)\right] + \frac{\lambda}{2\sigma^2}\left\|\mathbf{h}_\mathcal{S} - \frac{1}{|\mathcal{S}|}\sum_{w \in \mathcal{S}} \hat{\mathbf{h}}'_w\right\|_2^2. \tag{8}$$

3.2 Target Task with Output Word Vectors

Table 1: Comparison of source tasks and target tasks in our transfer learning.

Model	Input $\mathbf{x}$	Output y	Hypothesis h	Loss l
Skip-gram	w_t, w_c	Binary	$\sigma(\mathbf{h}_{w_t}^\top \mathbf{h}'_{w_c})$	Negative log
CBoW	$w_t, \mathcal{C}_t$	Binary	$\sigma(\mathbf{h}_{\mathrm{avg}_t}^\top \mathbf{h}'_{w_t})$	Negative log
`Avg.`	$\mathbf{1}$	$\mathbb{R}^d$	$\mathbf{h}_\mathcal{S}$	Root square
`IDF-Avg.`	$\mathbf{1}$	$\mathbb{R}^d$	$\mathbf{h}_\mathcal{S}$	Weighted root square
`Input-Trans.` (Skip-gram)	$\mathbf{x}_w$	Binary	$\mathrm{sign}(\hat{\mathbf{h}}_w^\top \mathbf{h}_\mathcal{S})$	Zero-one
`Input-Trans.` (CBoW)	$\mathcal{S}$	Binary	$\mathrm{sign}(\hat{\mathbf{h}}_{\mathrm{avg}_\mathcal{S}}^\top \mathbf{h}_\mathcal{S})$	Zero-one
`Output-Trans.` (Skip-gram)	$\mathbf{x}_w$	Binary	$\mathrm{sign}(\mathbf{h}_\mathcal{S}^\top \hat{\mathbf{h}}'_w)$	Zero-one
`Output-Trans.` (CBoW)	$\mathbf{x}_w$	Binary	$\mathrm{sign}(\mathbf{h}_\mathcal{S}^\top \hat{\mathbf{h}}'_w)$	Zero-one
`Word-Input-Trans.` (Skip-gram)	$\mathcal{S}$	Binary	$\mathrm{sign}(\hat{\mathbf{h}}_w^\top \mathbf{h}_\mathcal{S})$	Zero-one
`Word-Input-Trans.` (CBoW)	$\mathcal{S}$	Binary	$\mathrm{sign}(\hat{\mathbf{h}}_{\mathrm{avg}_\mathcal{S}}^\top \mathbf{h}_\mathcal{S})$	Zero-one
`Word-Output-Trans.` (Skip-gram)	$\mathcal{S}$	Binary	$\mathrm{sign}(\mathbf{h}_\mathcal{S}^\top \hat{\mathbf{h}}'_w)$	Zero-one
`Word-Output-Trans.` (CBoW)	$\mathcal{S}$	Binary	$\mathrm{sign}(\mathbf{h}_\mathcal{S}^\top \hat{\mathbf{h}}'_w)$	Zero-one

We can consider another target task setting, switching the role of fixed input word vectors and fixed output word vectors. We use fixed pre-trained output word vector for binary classification, and use fixed pre-trained input word vectors in the prior distribution. Therefore, given a sentence $\mathcal{S}$, the hypothesis denotes $h(\mathbf{x}_w) = \mathrm{sign}(\mathbf{h}_\mathcal{S}^\top \hat{\mathbf{H}}'\mathbf{x}_w) = \mathrm{sign}(\mathbf{h}_\mathcal{S}^\top \hat{\mathbf{h}}'_w)$, the prior is defined by $\mathcal{P}_\mathcal{S} = \frac{\prod_{w \in \mathcal{S}} \mathcal{N}(;\hat{\mathbf{h}}_w, \sigma^2 I)}{\int \prod_{w \in \mathcal{S}} \mathcal{N}(\mathbf{h}; \hat{\mathbf{h}}_w, \sigma^2 I)\mathrm{d}\mathbf{h}}$. In this target task, we minimize the loss function defined by

$$L(\mathcal{D}_S, \mathbf{h}_\mathcal{S}) = -\frac{1}{|\mathcal{S}|}\left[\sum_{w \in \mathcal{S}} \ln\sigma\left(\mathbf{h}_\mathcal{S}^\top \hat{\mathbf{h}}'_w\right) + \sum_{w_n \in \mathcal{NS}} \ln\sigma\left(-\mathbf{h}_\mathcal{S}^\top \hat{\mathbf{h}}'_{w_n}\right)\right] + \frac{\lambda}{2\sigma^2}\left\|\mathbf{h}_\mathcal{S} - \frac{1}{|\mathcal{S}|}\sum_{w \in \mathcal{S}} \hat{\mathbf{h}}_w\right\|_2^2. \tag{9}$$

We note that we do not average over output word vectors in Eq. (9) with pre-trained word vectors by CBoW because the original CBoW's source task defined by Eq. (2) only averages over input word vectors. Table. 1 shows the all source task and target task treated in this paper.

4 Learning Sentence Vectors Consisting of Learnable Word Vectors

Aforementioned sentence vectors defined by Equations (7) to (9) are inefficient representation because they need to learn sentence vectors not only on train data, but also on test data. To avoid this inefficiency, we consider another sentence vector's modeling based on word vectors defined by

$$\mathbf{h}_{\mathcal{S}} = \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \mathbf{h}_w. \tag{10}$$

This formulation is similar to supervised `fastText` (Joulin et al., 2017) with uni-gram information and `fastSent` (Hill et al., 2016). Thanks to this word-based modeling, we do not need to learn sentence vector on test data. Moreover we can initialize each word vector $\mathbf{h}_w$ by prior's pre-trained word vector $\hat{\mathbf{h}}'$.

5 Experiments: Sentence Classification

We verified the proposed methods in sentence classification tasks because learned sentence vectors work as feature vectors for supervised machine learning tasks. We reported test accuracy averaged over three times with different random seeds.

We trained skip-gram and CBoW for source tasks. Using these pre-trained vectors, we compared models obtained from our analysis, `Avg.` (Eq. (5)) and `IDF-Avg.` (Eq. (6)), `Input-Trans.` (Eq. (7)), `Output-Trans.` (Eq. (9)), `Word-Input-Trans.` (Eq. (7) with Eq. (10)), and `Word-Output-Trans.` (Eq. (9) with Eq. (10)).

5.1 Settings of Training of Word Vectors

We used English Wikipedia articles[4] to train word vectors. We did pre-process this corpus with `wikifil.pl`[5], then we removed words appearing less than 5 times. In the pre-processed corpus, the size of vocabulary $|\mathcal{V}|$ is 2 472 211, and the number of tokens is 4 517 598 626.

We used the following parameters to train word vectors with forked `word2vec`[6]: the size of the vector dimension was 300, the window size was 5, the sub-sampling parameter was 0.0001, the number of iteration was 5, the number of negative samples was 5, the noise distribution parameter was $\frac{3}{4}$, the initial learning rates of skip-gram and CBoW were 0.25 and 0.5, respectively.

5.2 Classification Datasets

Table 2: Sentence Classification Datasets

Dataset	Task	#Train data	#Test data	#Classes
20news (Lang, 1995)	Topic classification	11 314	7 532	20
IMDb (Maas et al., 2011)	Sentiment analysis	25 000	25 000	2
SUBJ (Pang and Lee, 2004)	Subjectivity classification	8 000	2 000	2

We used three classification datasets: 1) 20 news topic classification (20news (Lang, 1995)), 2) movie review's sentiment analysis (IMDb (Maas et al., 2011)), and 3) movie review's subjectivity classification (SUBJ (Pang and Lee, 2004)). Table. 2 shows three classification datasets. We did pre-process all datasets with `wikifil.pl` modified for our experiments.[7] We split SUBJ dataset randomly into 80 % as training data and 20 % as test data for evaluation. For the others datasets, we tested on test datasets.

5.3 Settings of Proposed Algorithms

In the experiments, we used the number of negative samples is 15, noise samples are the uni-gram distribution of training sentences powered by $\frac{3}{4}$, the initial learning rates η were same to pre-trained word vector models. The number of iteration is 50. We used stochastic gradient descent to optimize sentence vectors with mini-batch as a

[4] We downloaded XML dump file created on Aug. 1, 2018, from `https://dumps.wikimedia.org/enwiki`.

[5] `https://github.com/facebookresearch/fastText/blob/master/wikifil.pl`

[6] `https://github.com/nzw0301/word2vec`

[7] `https://gist.github.com/nzw0301/4bc1e47172f8a8852e3ca5406dfa2015`

sentence. We linearly decreased the learning rate per 100 sentences. We fixed $\sigma^2 = 1$, and we used the same λ for all sentences. This hyper-parameter λ was searched in $\{10^{-2}, 10^{-1}, 1, 10\}$ by cross-validation of classification. All $\mathbf{h}_\mathcal{S}$ of `Input-Trans.` and `Output-Trans.` was initialized by $U(-\frac{1}{300}, \frac{1}{300})$. All word vector $\mathbf{h}_w$ of `Word-Input-Trans.` and `Word-Output-Trans.` are initialized by prior's pre-trained word vectors $\hat{\mathbf{h}}'_w$ and $\hat{\mathbf{h}}_w$, respectively. We implemented our proposed algorithms and training codes with `PyTorch`.

5.4 Settings of Sentence Classification

We used a logistic regression classifier with `scikit-learn` (Pedregosa et al., 2011) for sentence classification tasks. We choose the hyper-parameter C of logistic regression and pre-processing parameter as normalization by L2 norm by grid-search with five-fold cross-validation. All classification scores are averaged value over three times with different random seeds.

5.5 Results

Table 3: Test accuracy of sentence classification (averaged over three times).

Method	Source model	20news	IMDb	SUBJ
`Avg.` ($\alpha = 0$)	Skip-gram	0.749 ± 0.000	0.841 ± 0.000	0.907 ± 0.001
`Avg.` ($\alpha = 1$)	Skip-gram	0.747 ± 0.002	0.838 ± 0.000	0.905 ± 0.000
`Avg.` ($\alpha = 0$)	CBoW	0.733 ± 0.000	0.838 ± 0.000	0.905 ± 0.000
`Avg.` ($\alpha = 1$)	CBoW	0.737 ± 0.001	0.840 ± 0.000	0.904 ± 0.000
`IDF-Avg.` ($\lambda = 0$)	Skip-gram	0.735 ± 0.000	0.823 ± 0.001	**0.908 ± 0.001**
`IDF-Avg.` ($\lambda = 1$)	Skip-gram	0.732 ± 0.000	0.821 ± 0.000	0.905 ± 0.001
`IDF-Avg.` ($\lambda = 0$)	CBoW	0.726 ± 0.001	0.826 ± 0.000	0.903 ± 0.000
`IDF-Avg.` ($\lambda = 1$)	CBoW	0.723 ± 0.000	0.826 ± 0.001	0.904 ± 0.000
`Input-Trans.`	Skip-gram	**0.749 ± 0.002**	0.842 ± 0.000	**0.909 ± 0.000**
`Input-Trans.`	CBoW	0.718 ± 0.000	0.815 ± 0.000	0.908 ± 0.000
`Output-Trans.`	Skip-gram	0.749 ± 0.000	0.842 ± 0.000	0.908 ± 0.000
`Output-Trans.`	CBoW	0.736 ± 0.003	0.841 ± 0.001	**0.908 ± 0.003**
`Word-Input-Trans.`	Skip-gram	**0.751 ± 0.002**	0.844 ± 0.000	0.908 ± 0.000
`Word-Input-Trans.`	CBoW	0.746 ± 0.000	**0.854 ± 0.000**	0.905 ± 0.001
`Word-Output-Trans.`	Skip-gram	**0.751 ± 0.000**	0.844 ± 0.000	0.908 ± 0.000
`Word-Output-Trans.`	CBoW	0.733 ± 0.001	0.839 ± 0.001	0.904 ± 0.000

Table. 3 shows the sentence classification results. Test accuracies are slightly different among models.

6 Related Work

Representation learning (Bengio et al., 2013) is a class of the most fundamental machine learning tasks. Especially, unsupervised representation learning aims to learn re-usable representation extracting a useful feature from unlabeled data. The learned representation works as a feature vector for other machine learning tasks, such as classification and clustering.

6.1 Learning Sentence Vectors

Sentence representation models (e.g., Le and Mikolov (2015); Kiros et al. (2015)) aim to learn sentence vector from sentence collections. Even if we use a model with a lot of learning parameters to obtain sentence vectors such as recurrent neural network models (Kiros et al., 2015), more simple models based on pre-trained word vectors (Mikolov et al., 2013b; Pennington et al., 2014) beat these large models (Hill et al., 2016; Wieting et al., 2016). A simple operation such as post-processing (Arora et al., 2017) can improve the performance of sentence vectors based on pre-trained word vectors. Unlike word vectors' theoretical study (e.g., Levy and Goldberg (2014); Arora et al. (2016)), sentence vector algorithms are not understood well.

6.2 PAC-Bayes Bound and Transfer Learning

PAC-Bayes theory (McAllester, 1999) enables us to analyze generalization error over stochastic hypothesis class. Especially, in transfer learning setting, PAC-Bayes bound of neural transfer learning (McNamara and Balcan, 2017; Galanti et al., 2016) bound generalization error of target task by the empirical risk of source task. In similar machine learning problems, studies of meta-learning (Amit and Meir, 2018), lifelong learning (Pentina and Lampert, 2014, 2015), and domain adaptation (Germain et al., 2016a) also bound generalization error of target task by the empirical risk of source tasks.

7 Conclusion

We analyze several heuristic sentence vector models from pre-trained word vectors by using PAC-Bayes theory. In our experiments, the performance of all sentence vector models is almost same regarding of sentence classification tasks' feature vectors. In future work, we develop novel algorithms based on our PAC-Bayes analysis such that they improve the downstream tasks' performance.

Acknowledgments

We thank Ikko Yamane and the anonymous reviewers for useful discussions and their helpful comments. We also thank developers of `scikit-learn`, `gensim`, and `PyTorch`.

KN was supported by JSPS KAKENHI Grant Number 18J20470.

References

Kento Nozawa and Issei Sato. PAC-Bayes Analysis of Transferred Sentence Vectors. In *NeurIPS Workshop on Bayesian Deep Learning*, 2018.

Tomas Mikolov, Greg Corrado, Kai Chen, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. In *ICLR Workshop*, 2013a.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed Representations of Words and Phrases and their Compositionality. In *NeurIPS*, 2013b.

Felix Hill, Kyunghyun Cho, and Anna Korhonen. Learning Distributed Representations of Sentences from Unlabelled Data. In *NAACL*, pages 1367–1377, 2016.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. Towards Universal Paraphrastic Sentence Embeddings. In *ICLR*, 2016.

Olivier Catoni. *PAC-Bayesian Supervised Classification: The Thermodynamics of Statistical Learning*. Institute of Mathematical Statistics, 2007.

Sinno Jialin Pan and Qiang Yang. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010.

Daniel McNamara and Maria-Florina Balcan. Risk Bounds for Transferring Representations With and Without Fine-Tuning. In *ICML*, 2017.

Geoffrey E. Hinton. Training Products of Experts by Minimizing Contrastive Divergence. *Neural Computation*, 14(8): 1771–1800, 2002.

Omer Levy, Yoav Goldberg, and Ido Dagan. Improving Distributional Similarity with Lessons Learned from Word Embeddings. *TACL*, 3:211–225, 2015.

Quoc Le and Tomas Mikolov. Distributed Representations of Sentences and Documents. In *ICML*, 2015.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of Tricks for Efficient Text Classification. In *EACL*, volume 2, pages 427–431, 2017.

Ken Lang. NewsWeeder: Learning to Filter Netnews. In *ICML*, 1995.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning Word Vectors for Sentiment Analysis. In *ACL*, pages 142–150, 2011.

Bo Pang and Lillian Lee. A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts. In *ACL*, 2004.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine Learning in Python. *JMLR*, 12: 2825–2830, 2011.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation Learning: A Review and New Perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013.

Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. Skip-Thought Vectors. In *NeurIPS*, 2015.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. GloVe: Global Vectors for Word Representation. In *EMNLP*, 2014.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. A Simple but Tough to Beat Baseline for Sentence Embeddings. In *ICLR*, 2017.

Omer Levy and Yoav Goldberg. Neural Word Embedding as Implicit Matrix Factorization. In *NeurIPS*, 2014.

Sanjeev Arora, Yuanzhi Li, Yingyu Liang, Tengyu Ma, and Andrej Risteski. A Latent Variable Model Approach to PMI-based Word Embeddings. *TACL*, 4:385–399, 2016.

David A. McAllester. Some PAC-Bayesian Theorems. *Machine Learning*, 37:355–363, 1999.

Tomer Galanti, Lior Wolf, and Tamir Hazan. A Theoretical Framework for Deep Transfer Learning. *IMA*, pages 159–209, 2016.

Ron Amit and Ron Meir. Meta-Learning by Adjusting Priors Based on Extended PAC-Bayes Theory. In *ICML*, 2018.

Anastasia Pentina and Christoph H. Lampert. A PAC-Bayesian Bound for Lifelong Learning. In *ICML*, 2014.

Anastasia Pentina and Christoph H. Lampert. Lifelong Learning with Non-i.i.d. Tasks. In *NeurIPS*, 2015.

Pascal Germain, Amaury Habrard, François Laviolette, and Emilie Morvant. A New PAC-Bayesian Perspective on Domain Adaptation. In *ICML*, 2016a.

Pascal Germain, Francis Bach, Alexandre Lacoste, and Simon Lacoste-Julien. PAC-Bayesian Theory Meets Bayesian Inference. In *NeurIPS*, 2016b.

Joseph Lilleberg, Yun Zhu, and Yangqing Zhang. Support Vector Machines and Word2vec for Text Classification with Semantic Features. In *ICCI*CC*, pages 136–140, 2015.

Appendix A Proposed PAC-Bayes Bound and Concrete Loss Functions

A.1 Proof of Theorem 2

Given a sentence $\mathcal{S}$ and bounded loss l, and $\forall \lambda > 0$ with probability at least $1 - \delta$ over training samples $\mathcal{D}_{\mathcal{S}}$, minimization of Eq. (3) is equivalent to minimizing the loss function defined by

$$\hat{R}_{D_{\mathcal{S}}}(\mathcal{Q}_{\mathcal{S}}) + \lambda \mathrm{KL}[\mathcal{Q}_{\mathcal{S}} || \mathcal{P}_{\mathcal{S}}], \tag{11}$$

where λ is redefined by $^1/_\lambda$ for the simplification. We also modify Eq. (3) in the same way to Germain et al. (2016b) to obtain Eq. (11).

We focus on the $\mathrm{KL}[\mathcal{Q}_{\mathcal{S}} || \mathcal{P}_{\mathcal{S}}]$ term in Eq. (11). Recall that hypothesis h is defined by $h(\mathbf{x}_w) = \frac{1}{2} ||\hat{\mathbf{H}} \mathbf{x}_w - \mathbf{h}_{\mathcal{S}}||_2^2 = \frac{1}{2} ||\hat{\mathbf{h}}_w - \mathbf{h}_{\mathcal{S}}||_2^2$ in Eq. (11), and $\mathcal{P}_{\mathcal{S}} = \frac{\prod_{w \in \mathcal{S}} \mathcal{N}(;\hat{\mathbf{h}}'_w, \sigma^2 I)}{\int \prod_{w \in \mathcal{S}} \mathcal{N}(\mathbf{h};\hat{\mathbf{h}}'_w, \sigma^2 I) \mathrm{d}\mathbf{h}}$.

$$\begin{aligned}
\mathrm{KL}[\mathcal{Q}_{\mathcal{S}} || \mathcal{P}_{\mathcal{S}}] &= \int \mathcal{Q}_{\mathcal{S}} \log \frac{\mathcal{Q}_{\mathcal{S}}}{\mathcal{P}_{\mathcal{S}}} \mathrm{d}\mathbf{h} && (12) \\
&= \int \mathcal{Q}_{\mathcal{S}} \log \frac{\mathcal{Q}_{\mathcal{S}}}{\left[\frac{\prod_{w \in \mathcal{S}} \mathcal{N}(;\hat{\mathbf{h}}'_w, \sigma^2 I)}{\int \prod_{w \in \mathcal{S}} \mathcal{N}(\mathbf{h};\hat{\mathbf{h}}'_w, \sigma^2 I) \mathrm{d}\mathbf{h}} \right]} \mathrm{d}\mathbf{h} && (13) \\
&= \int \mathcal{Q}_{\mathcal{S}} \log \frac{\mathcal{Q}_{\mathcal{S}} Z}{\prod\limits_{w \in \mathcal{S}} \mathcal{N}(; \hat{\mathbf{h}}'_w, \sigma^2 I)} \mathrm{d}\mathbf{h} && (14) \\
&= \int \sum_{w \in \mathcal{S}} \mathcal{Q}_{\mathcal{S}} \log \frac{\mathcal{Q}_{\mathcal{S}} Z}{\mathcal{N}(; \hat{\mathbf{h}}'_w, \sigma^2 I)} \mathrm{d}\mathbf{h} && (15) \\
&= \int \sum_{w \in \mathcal{S}} \mathcal{Q}_{\mathcal{S}} \left(\log \mathcal{Q}_{\mathcal{S}} + \log Z - \log \mathcal{N}(; \hat{\mathbf{h}}'_w, \sigma^2 I) \right) \mathrm{d}\mathbf{h} && (16) \\
&= \int \sum_{w \in \mathcal{S}} \mathcal{Q}_{\mathcal{S}} \left(\log \frac{\mathcal{Q}_{\mathcal{S}}}{\mathcal{N}(; \hat{\mathbf{h}}'_w, \sigma^2 I)} + \log Z \right) \mathrm{d}\mathbf{h} && (17) \\
&= \sum_{w \in \mathcal{S}} \int \mathcal{Q}_{\mathcal{S}} \left(\log \frac{\mathcal{Q}_{\mathcal{S}}}{\mathcal{N}(; \hat{\mathbf{h}}'_w, \sigma^2 I)} \right) \mathrm{d}\mathbf{h} + \sum_{w \in \mathcal{S}} \int \mathcal{Q}_{\mathcal{S}} \log Z \mathrm{d}\mathbf{h} && (18) \\
&= \sum_{w \in \mathcal{S}} \int \mathcal{Q}_{\mathcal{S}} \left(\log \frac{\mathcal{Q}_{\mathcal{S}}}{\mathcal{N}(; \hat{\mathbf{h}}'_w, \sigma^2 I)} \right) \mathrm{d}\mathbf{h} + |\mathcal{S}| \log Z && (19) \\
&= \sum_{w \in \mathcal{S}} \mathrm{KL} \left[\mathcal{Q}_{\mathcal{S}} || \mathcal{N}(; \hat{\mathbf{h}}'_w, \sigma^2 I) \right] + |\mathcal{S}| \log Z, && (20)
\end{aligned}$$

where $Z = \int \prod\limits_{w \in \mathcal{S}} \mathcal{N}(\mathbf{h}; \hat{\mathbf{h}}'_w, \sigma^2 I) \mathrm{d}\mathbf{h}$. The only first term in Eq. (20) contributes the loss function because $|\mathcal{S}| \log Z$ is constant when $\mathcal{S}$ is fixed. Therefore,

$$\begin{aligned}
\sum_{w \in \mathcal{S}} \mathrm{KL} \left[\mathcal{Q}_{\mathcal{S}} || \mathcal{N}(; \hat{\mathbf{h}}'_w, \sigma^2 I) \right] &= \frac{1}{2\sigma^2} \sum_{w \in \mathcal{S}} \left(||\mathbf{h}_{\mathcal{S}}||_2^2 - 2\mathbf{h}_{\mathcal{S}}^\top \hat{\mathbf{h}}'_w + \left|\left| \hat{\mathbf{h}}'_w \right|\right|_2^2 \right) && (21) \\
&= \frac{1}{2\sigma^2} \left(|\mathcal{S}| \times ||\mathbf{h}_{\mathcal{S}}||_2^2 - 2\mathbf{h}_{\mathcal{S}}^\top \sum_{w \in \mathcal{S}} \hat{\mathbf{h}}'_w + \sum_{w \in \mathcal{S}} \left|\left| \hat{\mathbf{h}}'_w \right|\right|_2^2 \right) && (22) \\
&= \frac{|\mathcal{S}|}{2\sigma^2} ||\mathbf{h}_{\mathcal{S}}||_2^2 - 2\mathbf{h}_{\mathcal{S}}^\top \bar{\mathbf{h}}_{\mathcal{S}} + \frac{1}{2\sigma^2} \sum_{w \in \mathcal{S}} \left|\left| \hat{\mathbf{h}}'_w \right|\right|_2^2 && (23) \\
&= \frac{|\mathcal{S}|}{2\sigma^2} \left|\left| \mathbf{h}_{\mathcal{S}} - \bar{\mathbf{h}}_{\mathcal{S}} \right|\right|_2^2 - \frac{1}{2\sigma^2} \left(|\mathcal{S}| \left|\left| \bar{\mathbf{h}}_{\mathcal{S}} \right|\right|_2^2 + \sum_{w \in \mathcal{S}} \left|\left| \hat{\mathbf{h}}'_w \right|\right|_2^2 \right), && (24)
\end{aligned}$$

where $\bar{\mathbf{h}}_{\mathcal{S}} = \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \hat{\mathbf{h}}'_w$. In terms of minimization of loss function, we can ignore the second term in Eq. (24). Then, we obtain Eq. (4) by refining $\lambda = \lambda |\mathcal{S}|$.

A.2 Details of Eq. (5)

We minimize the upper bound defined by Eq. (4) as loss function defined by,

$$\begin{aligned} L(\mathcal{D}_{\mathcal{S}}, \mathcal{Q}_{\mathcal{S}}) &= \mathop{\mathbb{E}}_{h\sim\mathcal{Q}} \frac{1}{|\mathcal{D}_{\mathcal{S}}|} \sum_{i=1}^{|\mathcal{D}_{\mathcal{S}}|} l(\mathbf{x}_i, y_i, h) + \lambda \mathrm{KL}(\mathcal{Q}_{\mathcal{S}}||\mathcal{P}_{\mathcal{S}}) + C \qquad (25) \\ &= \mathop{\mathbb{E}}_{h\sim\mathcal{Q}} \frac{1}{|\mathcal{D}_{\mathcal{S}}|} \sum_{i=1}^{|\mathcal{D}_{\mathcal{S}}|} \left(\frac{1}{2} \left|\left| \hat{\mathbf{h}}_{w_i} - \mathbf{h}_{\mathcal{S}} \right|\right|_2^2 \right) + \lambda \left(\frac{1}{2\sigma^2} \left|\left| \mathbf{h}_{\mathcal{S}} - \frac{1}{|\mathcal{S}|} \sum_{w\in\mathcal{S}} \hat{\mathbf{h}}'_w \right|\right|_2^2 \right) + C. \qquad (26) \end{aligned}$$

Empirically, $\mathcal{X}_{\mathcal{S}}$ is the same to $\mathcal{S}$ in Eq. (26).

$$L(D_{\mathcal{S}}, \mathcal{Q}_{\mathcal{S}}) = \mathop{\mathbb{E}}_{h\sim\mathcal{Q}_{\mathcal{S}}} \frac{1}{|\mathcal{S}|} \sum_{w\in\mathcal{S}} \left(\frac{1}{2} \left|\left| \hat{\mathbf{h}}_w - \mathbf{h}_{\mathcal{S}} \right|\right|_2^2 \right) + \lambda \left(\frac{1}{2\sigma^2} \left|\left| \mathbf{h}_{\mathcal{S}} - \frac{1}{|\mathcal{S}|} \sum_{w\in\mathcal{S}} \hat{\mathbf{h}}'_w \right|\right|_2^2 \right) + C. \qquad (27)$$

We take derivative of Eq. (27) with respect to $\mathbf{h}_{\mathcal{S}}$, and set it to zero.

$$\begin{aligned} \nabla_{\mathbf{h}_{\mathcal{S}}} \hat{L}(\mathcal{D}_{\mathcal{S}}, \mathcal{Q}_{\mathcal{S}}) &= 0 \qquad (28) \\ \frac{1}{|\mathcal{S}|} \sum_{w\in\mathcal{S}} \left(\mathbf{h}_{\mathcal{S}} - \hat{\mathbf{h}}_w \right) + \frac{\lambda}{\sigma^2} \left(\mathbf{h}_{\mathcal{S}} - \frac{1}{|\mathcal{S}|} \sum_{w\in\mathcal{S}} \hat{\mathbf{h}}'_w \right) &= 0 \qquad (29) \\ \mathbf{h}_{\mathcal{S}} - \frac{1}{|\mathcal{S}|} \sum_{w\in\mathcal{S}} \hat{\mathbf{h}}_w + \alpha \mathbf{h}_{\mathcal{S}} - \frac{\alpha}{|\mathcal{S}|} \sum_{w\in\mathcal{S}} \hat{\mathbf{h}}'_w &= 0 \qquad (30) \\ (1+\alpha)\mathbf{h}_{\mathcal{S}} - \frac{1}{|\mathcal{S}|} \sum_{w\in\mathcal{S}} \hat{\mathbf{h}}_w - \frac{\alpha}{|\mathcal{S}|} \sum_{w\in\mathcal{S}} \hat{\mathbf{h}}'_w &= 0 \qquad (31) \\ (1+\alpha)\mathbf{h}_{\mathcal{S}} - \frac{1}{|\mathcal{S}|} \sum_{w\in\mathcal{S}} \left(\hat{\mathbf{h}}_w + \alpha \hat{\mathbf{h}}'_w \right) &= 0 \qquad (32) \\ (1+\alpha)\mathbf{h}_{\mathcal{S}} &= \frac{1}{|\mathcal{S}|} \sum_{w\in\mathcal{S}} \left(\hat{\mathbf{h}}_w + \alpha \hat{\mathbf{h}}'_w \right) \qquad (33) \\ \mathbf{h}_{\mathcal{S}} &= \frac{1}{(1+\alpha)|\mathcal{S}|} \sum_{w\in\mathcal{S}} \left(\hat{\mathbf{h}}_w + \alpha \hat{\mathbf{h}}'_w \right), \qquad (34) \end{aligned}$$

where $\alpha = \frac{\lambda}{\sigma^2}$.

A.3 Inverse Document Frequency Weighing from PAC-Bayes Bound

Inverse document frequency (IDF) is also a widely used heuristic method to weight a word in text collections. IDF is defined by

$$\mathrm{IDF}(w) = \log \left(\frac{N_{\mathcal{S}}}{\sum_{i=1}^{N_{\mathcal{S}}} \mathbb{I}\left[w \in \mathcal{S}_i \right]} \right) + 1, \qquad (35)$$

where $N_{\mathcal{S}}$ is the number of training sentences. Sentence vector by averaging of weighed word vectors by IDF is also a simple heuristic method (e.g., Lilleberg et al. (2015)). Our interest is to obtain this heuristic method from our PAC-Bayes bound.

We start from Eq. (20). Each variance σ^2 of priors depends on each word w.

$$\sum_{w\in\mathcal{S}} \mathrm{KL}\left[\mathcal{Q}_{\mathcal{S}}||\mathcal{N}(;\hat{\mathbf{h}}'_w,\sigma_w^2 I)\right] = \sum_{w\in\mathcal{S}}\frac{1}{2\sigma_w^2}\left(||\mathbf{h}_{\mathcal{S}}||_2^2 - 2\mathbf{h}_{\mathcal{S}}^\top\hat{\mathbf{h}}'_w + \left|\left|\hat{\mathbf{h}}'_w\right|\right|_2^2\right) + C \tag{36}$$

$$= ||\mathbf{h}_{\mathcal{S}}||_2^2 \sum_{w\in\mathcal{S}}\frac{1}{2\sigma_w^2} - 2\mathbf{h}_{\mathcal{S}}^\top \sum_{w\in\mathcal{S}}\frac{1}{2\sigma_w^2}\hat{\mathbf{h}}'_w + \sum_{w\in\mathcal{S}}\frac{1}{2\sigma_w^2}\left|\left|\hat{\mathbf{h}}'_w\right|\right|_2^2 + C \tag{37}$$

$$= \eta\,||\mathbf{h}_{\mathcal{S}}||_2^2 - 2\mathbf{h}_{\mathcal{S}}^\top \sum_{w\in\mathcal{S}}\frac{1}{2\sigma_w^2}\hat{\mathbf{h}}'_w + C' \tag{38}$$

$$= \eta\left(||\mathbf{h}_{\mathcal{S}}||_2^2 - \frac{2}{\eta}\mathbf{h}_{\mathcal{S}}^\top \sum_{w\in\mathcal{S}}\frac{1}{2\sigma_w^2}\hat{\mathbf{h}}'_w\right) + C' \tag{39}$$

$$= \eta\left|\left|\mathbf{h}_{\mathcal{S}} - \frac{1}{\eta}\sum_{w\in\mathcal{S}}\frac{1}{2\sigma_w^2}\hat{\mathbf{h}}'_w\right|\right|_2^2 + C'', \tag{40}$$

where $\eta = \sum_{w\in\mathcal{S}}\frac{1}{2\sigma_w^2}$, and C, C'', and C''' are the constant terms not depending on $\mathbf{h}_{\mathcal{S}}$.

We define weighted loss function as $l(\mathbf{x}_i, y_i, h) = \frac{\beta_i}{2}||\hat{\mathbf{h}}_{w_i} - \mathbf{h}_{\mathcal{S}}||_2^2$, where β is weighing function from $\mathbf{x}$ to $\mathbb{R}_+$. We follow the same way to Appendix A.2.

$$L(\mathcal{D}_{\mathcal{S}}, \mathcal{Q}_{\mathcal{S}}) = \mathop{\mathbb{E}}_{h\sim\mathcal{Q}_{\mathcal{S}}}\frac{1}{|\mathcal{D}_{\mathcal{S}}|}\sum_{i=1}^{|\mathcal{D}_{\mathcal{S}}|} l(\mathbf{x}_i, y_i, h) + \lambda\mathrm{KL}(\mathcal{Q}_{\mathcal{S}}||\mathcal{P}_{\mathcal{S}}) \tag{41}$$

$$= \mathop{\mathbb{E}}_{h\sim\mathcal{Q}_{\mathcal{S}}}\frac{1}{|\mathcal{D}_{\mathcal{S}}|}\sum_{i=1}^{|\mathcal{D}_{\mathcal{S}}|}\left(\frac{\beta_{w_i}}{2}\left|\left|\hat{\mathbf{h}}_{w_i} - \mathbf{h}_{\mathcal{S}}\right|\right|_2^2\right) + \lambda\eta\left|\left|\mathbf{h}_{\mathcal{S}} - \frac{1}{\eta}\sum_{w\in\mathcal{S}}\frac{1}{2\sigma_w^2}\hat{\mathbf{h}}'_w\right|\right|_2^2 + C. \tag{42}$$

Given $\mathcal{X}_{\mathcal{S}} = \mathcal{S}$, we take derivative of Eq. (42) with respect to $\mathbf{h}_{\mathcal{S}}$, and set it to zero.

$$\nabla_{\mathbf{h}_{\mathcal{S}}}\hat{L}(\mathcal{D}_{\mathcal{S}}, \mathbf{h}_{\mathcal{S}}) = 0 \tag{43}$$

$$\frac{1}{|\mathcal{S}|}\sum_{w\in\mathcal{S}}\beta_w\left(\mathbf{h}_{\mathcal{S}} - \hat{\mathbf{h}}_w\right) + 2\lambda\eta\left(\mathbf{h}_{\mathcal{S}} - \frac{1}{\eta}\sum_{w\in\mathcal{S}}\frac{1}{2\sigma_w^2}\hat{\mathbf{h}}'_w\right) = 0 \tag{44}$$

$$\frac{1}{|\mathcal{S}|}\sum_{w\in\mathcal{S}}\beta_w\mathbf{h}_{\mathcal{S}} - \frac{1}{|\mathcal{S}|}\sum_{w\in\mathcal{S}}\beta_w\hat{\mathbf{h}}_w + 2\lambda\eta\mathbf{h}_{\mathcal{S}} - 2\lambda\sum_{w\in\mathcal{S}}\frac{1}{2\sigma_w^2}\hat{\mathbf{h}}'_w = 0 \tag{45}$$

$$\left(\frac{\sum_{w\in\mathcal{S}}\beta_w}{|\mathcal{S}|} + 2\lambda\eta\right)\mathbf{h}_{\mathcal{S}} - \frac{1}{|\mathcal{S}|}\sum_{w\in\mathcal{S}}\beta_w\hat{\mathbf{h}}_w - 2\lambda\sum_{w\in\mathcal{S}}\frac{1}{2\sigma_w^2}\hat{\mathbf{h}}'_w = 0 \tag{46}$$

$$\left(\sum_{w\in\mathcal{S}}\beta_w + 2\eta\lambda|\mathcal{S}|\right)\mathbf{h}_{\mathcal{S}} - \sum_{w\in\mathcal{S}}\beta_w\hat{\mathbf{h}}_w - 2\lambda|\mathcal{S}|\sum_{w\in\mathcal{S}}\frac{1}{2\sigma_w^2}\hat{\mathbf{h}}'_w = 0 \tag{47}$$

$$\left(\sum_{w\in\mathcal{S}}\beta_w + 2\eta\lambda|\mathcal{S}|\right)\mathbf{h}_{\mathcal{S}} - \sum_{w\in\mathcal{S}}\left(\beta_w\hat{\mathbf{h}}_w + \lambda|\mathcal{S}|\frac{1}{\sigma_w^2}\hat{\mathbf{h}}'_w\right) = 0 \tag{48}$$

$$\left(\sum_{w\in\mathcal{S}}\beta_w + 2\eta\lambda|\mathcal{S}|\right)\mathbf{h}_{\mathcal{S}} = \sum_{w\in\mathcal{S}}\left(\beta_w\hat{\mathbf{h}}_w + \frac{\lambda|\mathcal{S}|}{\sigma_w^2}\hat{\mathbf{h}}'_w\right) \tag{49}$$

$$\mathbf{h}_{\mathcal{S}} = \frac{\sum_{w\in\mathcal{S}}\beta_w\hat{\mathbf{h}}_w + \frac{\lambda|\mathcal{S}|}{\sigma_w^2}\hat{\mathbf{h}}'_w}{\sum_{w\in\mathcal{S}}\beta_w + 2\eta\lambda|\mathcal{S}|}. \tag{50}$$

We set $\frac{1}{\sigma_w^2} = \mathrm{IDF}(w)$. Then, we evaluate $\eta = \frac{1}{2}\sum_{w \in \mathcal{S}} \frac{1}{\sigma^2} = \frac{1}{2}\sum_{w \in \mathcal{S}} \mathrm{IDF}(w)$, then

$$\mathbf{h}_{\mathcal{S}} = \frac{\sum_{w \in \mathcal{S}} \beta_w \hat{\mathbf{h}}_w + \lambda |\mathcal{S}| \mathrm{IDF}(w) \hat{\mathbf{h}}'_w}{\sum_{w \in \mathcal{S}} \beta_w + \lambda |\mathcal{S}| \sum_{w \in \mathcal{S}} \mathrm{IDF}(w)} \tag{51}$$

$$= \frac{\sum_{w \in \mathcal{S}} \beta_w \hat{\mathbf{h}}_w + \lambda |\mathcal{S}| \mathrm{IDF}(w) \hat{\mathbf{h}}'_w}{\sum_{w \in \mathcal{S}} \beta_w + \lambda |\mathcal{S}| \mathrm{IDF}(w)}. \tag{52}$$

We set $\beta_w = |\mathcal{S}|\mathrm{IDF}(w)$,

$$\mathbf{h}_{\mathcal{S}} = \frac{\sum_{w \in \mathcal{S}} |\mathcal{S}| \mathrm{IDF}(w) \hat{\mathbf{h}}_w + \lambda |\mathcal{S}| \mathrm{IDF}(w) \hat{\mathbf{h}}'_w}{\sum_{w \in \mathcal{S}} |\mathcal{S}| \mathrm{IDF}(w) + \lambda |\mathcal{S}| \mathrm{IDF}(w)} \tag{53}$$

$$= \frac{\sum_{w \in \mathcal{S}} \mathrm{IDF}(w) \hat{\mathbf{h}}_w + \lambda \mathrm{IDF}(w) \hat{\mathbf{h}}'_w}{\sum_{w \in \mathcal{S}} \mathrm{IDF}(w) + \lambda \mathrm{IDF}(w)} \tag{54}$$

$$= \frac{\sum_{w \in \mathcal{S}} \mathrm{IDF}(w) \left(\hat{\mathbf{h}}_w + \lambda \hat{\mathbf{h}}'_w \right)}{(1 + \lambda) \sum_{w \in \mathcal{S}} \mathrm{IDF}(w)}. \tag{55}$$