
PAC-Bayes Analysis of Transferred Sentence Vectors

Kento Nozawa

The University of Tokyo & RIKEN
nozawa@ms.k.u-tokyo.ac.jp

Issei Sato

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

Abstract

Learning sentence vectors from an unlabeled corpus have attracted attention because such vectors are able to represent sentences flexibly. Simple heuristic methods using pre-trained word vectors are a strong baseline for 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.

1 Learning Word Vectors and Sentence Vectors

Word Vectors: 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) [1], which is a widely used word vector model.¹ Skip-gram learns two types of word 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. [2], whose loss function is defined by

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

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 [2].

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 sentence² from d -dimensional vector representation \mathbf{h}_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}_S is to average of pre-trained word vectors of words appearing in the sentence \mathcal{S} , e.g., $\mathbf{h}_S = \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \hat{\mathbf{h}}_w$, or $\mathbf{h}_S = \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \frac{\hat{\mathbf{h}}_w + \hat{\mathbf{h}}'_w}{2}$. Empirically, these simple methods can beat more complex methods in machine learning tasks for sentences [3, 4].

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 [2]. We formulate learning word vectors and

¹Another popular model, Continuous Bag-of-Words [1], is described in Appendix A.1.

²We name a sequence of words as a *sentence* including phrase, paragraph, and document.

sentence vectors in terms of transfer learning [5]. 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}_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 [6] (See general PAC-Bayes bound in Appendix A.2, and related work of PAC-Bayesian transfer learning in Appendix B.2).

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 S and word vectors trained on the source task, let \mathcal{D} be a data distribution over $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X} := \{\mathbf{x}_w | w \in \mathbb{N}_{\leq |\mathcal{Y}|}\}$ 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 S assumes bag-of-words. Let $|\mathcal{D}_S|$ be the number of training samples, and $|\mathcal{S}|$ be the number of words in S . Our goal is to minimize generalization error $R_{\mathcal{D}}(\mathcal{Q}_S) = \mathbb{E}_{h \sim \mathcal{Q}_S} \mathbb{E}_{(\mathbf{x}, y) \sim \mathcal{D}} l(\mathbf{x}, y, h)$, where l is a bounded loss function, and h is hypothesis parameterized by sentence vector \mathbf{h}_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 $\mathcal{P}_S \propto \prod_{w \in \mathcal{S}} \mathcal{N}(\cdot; \hat{\mathbf{h}}'_w, \sigma^2 I)$, where σ^2 is variance parameter, and I is identity matrix. This prior distribution is known as the product of experts [7] such that each expert is a Gaussian distribution parameterized by pre-trained $\hat{\mathbf{h}}'$ of word appearing in the given sentence S . We also define posterior distribution over \mathcal{F} as $\mathcal{Q}_S = \mathcal{N}(\cdot; \mathbf{h}_S, \sigma^2 I)$. Theorem 1 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.

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

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

where C is a constant term that does not depend on sentence vector \mathbf{h}_S .

We prove Theorem 1 by using the PAC-Bayes bound [8]. The proof is shown in Appendix C.1.

Sentence Vectors via Regression: We use Theorem 1 to analyze a heuristic sentence vector models. Let $\mathcal{Y} := \{0\}$, hypothesis h be $h(\mathbf{x}_w) = \frac{1}{2} \|\hat{\mathbf{H}}\mathbf{x}_w - \mathbf{h}_S\|_2^2$, 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 $\hat{\mathbf{h}}_w$. We introduce the loss function $l(\mathbf{x}, y, h) = h(\mathbf{x}) - y = h(\mathbf{x})$, because $y = 0$.³ We can obtain the closed form of \mathbf{h}_S from Eq. (2),

$$\mathbf{h}_S = \frac{1}{(1 + \alpha)|\mathcal{S}|} \sum_{w \in \mathcal{S}} (\hat{\mathbf{h}}_w + \alpha \hat{\mathbf{h}}'_w), \quad (3)$$

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

Corollary 1.1 (Sentence Vector by Averaging of Pre-trained Input Word Vectors). *The sentence vector \mathbf{h}_S , estimated by minimizing Eq. (2) with $\alpha = 0$, is equivalent to $\frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \hat{\mathbf{h}}_w$.*

Corollary 1.2 (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. [9]). This operation corresponds to the solution of Eq. (2) 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 C.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 $\mathcal{P}_S \propto \prod_{w \in \mathcal{S}} \mathcal{N}(\cdot; \hat{\mathbf{h}}'_w, \sigma_w^2 I)$. When we set $\beta(\mathbf{x}) = |\mathcal{S}| \text{IDF}(\mathbf{x})$ and

³When 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.

$\frac{1}{\sigma_w^2} = \text{IDF}(w)$, we can obtain the closed form of \mathbf{h}_S from Eq. (2),

$$\mathbf{h}_S = \frac{1}{(1 + \lambda) \sum_{w \in \mathcal{S}} \text{IDF}(w)} \sum_{w \in \mathcal{S}} \text{IDF}(w) (\hat{\mathbf{h}}_w + \lambda \hat{\mathbf{h}}'_w). \quad (4)$$

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.⁴ 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 1 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 $h(\mathbf{x}_w) = \text{sign}((\hat{\mathbf{H}}\mathbf{x}_w)^\top \mathbf{h}_S)$. In practice, we minimize the following loss function using negative sampling as surrogate loss function of zero-one loss.

$$L = -\frac{1}{|\mathcal{S}|} \left[\sum_{w \in \mathcal{S}} \ln \sigma(\hat{\mathbf{h}}_w^\top \mathbf{h}_S) + \sum_{w_n \in \mathcal{N}_S} \ln \sigma(-\mathbf{h}_{w_n}^\top \mathbf{h}_S) \right] + \frac{\lambda}{2\sigma^2} \left\| \mathbf{h}_S - \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \hat{\mathbf{h}}'_w \right\|_2^2. \quad (5)$$

Corollary 1.3 (Relationship to Paragraph Vector Models), *PV-DBoW, a instance of paragraph vector models [10], is the same to Eq. (5) with $\lambda = 0$ and without pre-trained word vectors.*

We also consider other target tasks to learn sentence vectors in Appendix D.

3 Experiments: Sentence Classification

Table 1: 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	0.908 ± 0.002
Input-trans.	CBoW	0.717 ± 0.000	0.817 ± 0.001	0.907 ± 0.001
Output-trans.	Skip-gram	0.749 ± 0.000	0.842 ± 0.000	0.908 ± 0.000
Output-trans.	CBoW	0.734 ± 0.003	0.841 ± 0.002	0.910 ± 0.003

We verified the proposed methods in sentence classification tasks, because learned sentence vectors work as feature vectors for supervised machine learning tasks. We used three classification datasets: 1) 20 news topic classification (20news [11]), 2) movie review’s sentiment analysis (IMDb [12]), and 3) movie review’s subjectivity classification (SUBJ [13]). We trained a one-vs-rest logistic regression classifier with five-fold cross-validation. Table. 1 shows the sentence classification results. We reported test accuracy averaged over three times with different random seeds. Details of the experimental datasets and procedure of experiments are described in Appendix E.

We trained skip-gram and CBoW on English Wikipedia articles. Using these pre-trained vectors, we compared models obtained from our analysis, Avg. (Eq. (3)), IDF-Avg. (Eq. (4)), Input-trans. (Eq. (5)), and Output-trans. (Eq. (54)). For Input-trans. and Output-trans., we fixed $\sigma^2 = 1$ and, we used the same λ that was searched in $\{10^{-2}, 10^{-1}, 1, 10\}$.

⁴Negative sampling loss works as a surrogate loss of zero-one loss.

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

- [1] Tomas Mikolov, Greg Corrado, Kai Chen, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. In *ICLR Workshop*, 2013.
- [2] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed Representations of Words and Phrases and their Compositionality. In *NeurIPS*, 2013.
- [3] Felix Hill, Kyunghyun Cho, and Anna Korhonen. Learning Distributed Representations of Sentences from Unlabelled Data. In *NAACL*, pages 1367–1377, 2016.
- [4] John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. Towards Universal Paraphrastic Sentence Embeddings. In *ICLR*, 2016.
- [5] Sinno Jialin Pan and Qiang Yang. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010.
- [6] Daniel McNamara and Maria-Florina Balcan. Risk Bounds for Transferring Representations With and Without Fine-Tuning. In *ICML*, 2017.
- [7] Geoffrey E. Hinton. Training Products of Experts by Minimizing Contrastive Divergence. *Neural Computation*, 14(8):1771–1800, 2002.
- [8] Olivier Catoni. *PAC-Bayesian Supervised Classification: The Thermodynamics of Statistical Learning*. Institute of Mathematical Statistics, 2007.
- [9] Omer Levy, Yoav Goldberg, and Ido Dagan. Improving Distributional Similarity with Lessons Learned from Word Embeddings. *TACL*, 3:211–225, 2015.
- [10] Quoc Le and Tomas Mikolov. Distributed Representations of Sentences and Documents. In *ICML*, 2015.
- [11] Ken Lang. NewsWeeder: Learning to Filter Netnews. In *ICML*, 1995.
- [12] 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.
- [13] Bo Pang and Lillian Lee. A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts. In *ACL*, 2004.
- [14] Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. Skip-Thought Vectors. In *NeurIPS*, 2015.
- [15] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. GloVe: Global Vectors for Word Representation. In *EMNLP*, 2014.
- [16] Sanjeev Arora, Yingyu Liang, and Tengyu Ma. A Simple but Tough to Beat Baseline for Sentence Embeddings. In *ICLR*, 2017.
- [17] Omer Levy and Yoav Goldberg. Neural Word Embedding as Implicit Matrix Factorization. In *NeurIPS*, 2014.
- [18] 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.
- [19] David A. McAllester. Some PAC-Bayesian Theorems. *Machine Learning*, 37:355–363, 1999.
- [20] Tomer Galanti, Lior Wolf, and Tamir Hazan. A Theoretical Framework for Deep Transfer Learning. *IMA*, pages 159–209, 2016.

- [21] Ron Amit and Ron Meir. Meta-Learning by Adjusting Priors Based on Extended PAC-Bayes Theory. In *ICML*, 2018.
- [22] Anastasia Pentina and Christoph H. Lampert. A PAC-Bayesian Bound for Lifelong Learning. In *ICML*, 2014.
- [23] Anastasia Pentina and Christoph H. Lampert. Lifelong Learning with Non-i.i.d. Tasks. In *NeurIPS*, 2015.
- [24] Pascal Germain, Amaury Habrard, François Laviolette, and Emilie Morvant. A New PAC-Bayesian Perspective on Domain Adaptation. In *ICML*, 2016.
- [25] Pascal Germain, Francis Bach, Alexandre Lacoste, and Simon Lacoste-Julien. PAC-Bayesian Theory Meets Bayesian Inference. In *NeurIPS*, 2016.
- [26] 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.
- [27] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic Differentiation in PyTorch. In *NeurIPS Workshop*, 2017.
- [28] 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.

Appendix A Preliminaries

A.1 Continuous Bag-of-Words with Negative Sampling

Continuous bag-of-words model (CBoW) [1] 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

$$L_{CBoW} = - \sum_{t=1}^T \left[\ln \sigma \left(\mathbf{h}_{\text{avg}_t}^\top \mathbf{h}'_{w_t} \right) + \sum_{w_n \in \mathcal{N}_t} \ln \sigma \left(-\mathbf{h}_{\text{avg}_t}^\top \mathbf{h}'_{w_n} \right) \right], \quad (6)$$

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

A.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 $\hat{R}_{\mathcal{D}_S}(\mathcal{Q}) = \mathbb{E}_{h \sim \mathcal{Q}} \frac{1}{N} \sum_{i=1}^N l(\mathbf{x}_i, y_i, h)$. Kullback-Leibler (KL) divergence is defined as $\text{KL}(\mathcal{Q} \parallel \mathcal{P}) = \mathbb{E}_{h \sim \mathcal{Q}} \log \frac{\mathcal{Q}(h)}{\mathcal{P}(h)}$.

Theorem 2 (PAC-Bayes Bound [8, Theorem 1.2.6]). $\forall \lambda > 0$, with probability at least $1 - \delta$ over training samples \mathcal{D}_S , $\forall h \sim \mathcal{Q}$,

$$R_{\mathcal{D}}(\mathcal{Q}) \leq \frac{1}{1 - \exp(-\frac{\lambda}{N})} \left(1 - \exp \left[-\frac{\lambda}{N} \hat{R}_{\mathcal{D}_S}(\mathcal{Q}) - \frac{\text{KL}[\mathcal{Q} \parallel \mathcal{P}] + \log(1/\delta)}{N} \right] \right). \quad (7)$$

Appendix B Related Work

B.1 Learning Sentence Vectors

Sentence representation models (e.g., Le and Mikolov [10], Kiros et al. [14]) 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 [14], more simple models based on pre-trained word vectors [2, 15] beat these large models [3, 4]. A simple operation such as post-processing [16] can improve the performance of sentence vectors based on pre-trained word vectors. Unlike word vectors' theoretical study (e.g., Levy and Goldberg [17], Arora et al. [18]), sentence vector algorithms are not understood well.

B.2 PAC-Bayes Bound and Transfer Learning

PAC-Bayes theory [19] enables us to analyze generalization error over stochastic hypothesis class. Especially, in a transfer learning setting, PAC-Bayes bound of neural transfer learning [6, 20] bound generalization error of target task by the empirical risk of source task. In similar machine learning problems, studies of meta-learning [21], lifelong learning [22, 23], and domain adaptation [24] also bound generalization error of target task by the empirical risk of source tasks.

Appendix C Proposed PAC-Bayes Bound and Concrete Loss Functions

C.1 Proof of Theorem 1

Given a sentence \mathcal{S} and bounded loss l , and $\forall \lambda > 0$ with probability at least $1 - \delta$ over training samples \mathcal{D}_S , minimization of Eq. (7) is equivalent to minimizing the loss function defined by

$$\hat{R}_{\mathcal{D}_S}(\mathcal{Q}_S) + \lambda \text{KL}[\mathcal{Q}_S \parallel \mathcal{P}_S], \quad (8)$$

where λ is redefined by $1/\lambda$ for the simplification. We also modify Eq. (7) in the same way to Germain et al. [25] to obtain Eq. (8).

We focus on the $\text{KL}[\mathcal{Q}_S|\mathcal{P}_S]$ term in Eq. (8). Recall that hypothesis h is defined by $h(\mathbf{x}_w) = \frac{1}{2}\|\hat{\mathbf{H}}\mathbf{x}_w - \mathbf{h}_S\|_2^2 = \frac{1}{2}\|\hat{\mathbf{h}}_w - \mathbf{h}_S\|_2^2$ in Eq. (8), $\mathcal{P}_S = \frac{\prod_{w \in \mathcal{S}} \mathcal{N}(\cdot; \hat{\mathbf{h}}'_w, \sigma^2 I)}{\int \prod_{w \in \mathcal{S}} \mathcal{N}(\mathbf{h}; \hat{\mathbf{h}}'_w, \sigma^2 I) d\mathbf{h}}$.

$$\text{KL}[\mathcal{Q}_S|\mathcal{P}_S] = \int \mathcal{Q}_S \log \frac{\mathcal{Q}_S}{\mathcal{P}_S} d\mathbf{h} \quad (9)$$

$$= \int \mathcal{Q}_S \log \left[\frac{\mathcal{Q}_S}{\frac{\prod_{w \in \mathcal{S}} \mathcal{N}(\cdot; \hat{\mathbf{h}}'_w, \sigma^2 I)}{\int \prod_{w \in \mathcal{S}} \mathcal{N}(\mathbf{h}; \hat{\mathbf{h}}'_w, \sigma^2 I) d\mathbf{h}}} \right] d\mathbf{h} \quad (10)$$

$$= \int \mathcal{Q}_S \log \frac{\mathcal{Q}_S Z}{\prod_{w \in \mathcal{S}} \mathcal{N}(\cdot; \hat{\mathbf{h}}'_w, \sigma^2 I)} d\mathbf{h} \quad (11)$$

$$= \int \sum_{w \in \mathcal{S}} \mathcal{Q}_S \log \frac{\mathcal{Q}_S Z}{\mathcal{N}(\cdot; \hat{\mathbf{h}}'_w, \sigma^2 I)} d\mathbf{h} \quad (12)$$

$$= \int \sum_{w \in \mathcal{S}} \mathcal{Q}_S \left(\log \mathcal{Q}_S + \log Z - \log \mathcal{N}(\cdot; \hat{\mathbf{h}}'_w, \sigma^2 I) \right) d\mathbf{h} \quad (13)$$

$$= \int \sum_{w \in \mathcal{S}} \mathcal{Q}_S \left(\log \frac{\mathcal{Q}_S}{\mathcal{N}(\cdot; \hat{\mathbf{h}}'_w, \sigma^2 I)} + \log Z \right) d\mathbf{h} \quad (14)$$

$$= \sum_{w \in \mathcal{S}} \int \mathcal{Q}_S \left(\log \frac{\mathcal{Q}_S}{\mathcal{N}(\cdot; \hat{\mathbf{h}}'_w, \sigma^2 I)} \right) d\mathbf{h} + \sum_{w \in \mathcal{S}} \int \mathcal{Q}_S \log Z d\mathbf{h} \quad (15)$$

$$= \sum_{w \in \mathcal{S}} \int \mathcal{Q}_S \left(\log \frac{\mathcal{Q}_S}{\mathcal{N}(\cdot; \hat{\mathbf{h}}'_w, \sigma^2 I)} \right) d\mathbf{h} + |\mathcal{S}| \log Z \quad (16)$$

$$= \sum_{w \in \mathcal{S}} \text{KL} \left[\mathcal{Q}_S \|\mathcal{N}(\cdot; \hat{\mathbf{h}}'_w, \sigma^2 I) \right] + |\mathcal{S}| \log Z, \quad (17)$$

where $Z = \int \prod_{w \in \mathcal{S}} \mathcal{N}(\mathbf{h}; \hat{\mathbf{h}}'_w, \sigma^2 I) d\mathbf{h}$. The only first term in Eq. (17) contributes the loss function because $|\mathcal{S}| \log Z$ is constant when \mathcal{S} is fixed. Therefore,

$$\sum_{w \in \mathcal{S}} \text{KL} \left[\mathcal{Q}_S \|\mathcal{N}(\cdot; \hat{\mathbf{h}}'_w, \sigma^2 I) \right] = \frac{1}{2\sigma^2} \sum_{w \in \mathcal{S}} \left(\|\mathbf{h}_S\|_2^2 - 2\mathbf{h}_S^\top \hat{\mathbf{h}}'_w + \|\hat{\mathbf{h}}'_w\|_2^2 \right) \quad (18)$$

$$= \frac{1}{2\sigma^2} \left(|\mathcal{S}| \times \|\mathbf{h}_S\|_2^2 - 2\mathbf{h}_S^\top \sum_{w \in \mathcal{S}} \hat{\mathbf{h}}'_w + \sum_{w \in \mathcal{S}} \|\hat{\mathbf{h}}'_w\|_2^2 \right) \quad (19)$$

$$= \frac{|\mathcal{S}|}{2\sigma^2} \|\mathbf{h}_S\|_2^2 - 2\mathbf{h}_S^\top \bar{\mathbf{h}}_S + \frac{1}{2\sigma^2} \sum_{w \in \mathcal{S}} \|\hat{\mathbf{h}}'_w\|_2^2 \quad (20)$$

$$= \frac{|\mathcal{S}|}{2\sigma^2} \|\mathbf{h}_S - \bar{\mathbf{h}}_S\|_2^2 - \frac{1}{2\sigma^2} \left(|\mathcal{S}| \|\bar{\mathbf{h}}_S\|_2^2 + \sum_{w \in \mathcal{S}} \|\hat{\mathbf{h}}'_w\|_2^2 \right), \quad (21)$$

where $\bar{\mathbf{h}}_S = \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \hat{\mathbf{h}}'_w$. In terms of minimization of loss function, we can ignore the second term in Eq. (21). Then, we obtain Eq. (2) by refining $\lambda = \lambda|\mathcal{S}|$.

C.2 Details of Eq. (3)

We minimize the upper bound defined by Eq. (2) as loss function defined by,

$$\begin{aligned}
 L(\mathcal{D}_S, \mathcal{Q}_S) &= \mathbb{E}_{h \sim \mathcal{Q}} \frac{1}{|\mathcal{D}_S|} \sum_{i=1}^{|\mathcal{D}_S|} l(\mathbf{x}_i, y_i, h) + \lambda \text{KL}(\mathcal{Q}_S \| \mathcal{P}_S) + C \quad (22) \\
 &= \mathbb{E}_{h \sim \mathcal{Q}} \frac{1}{|\mathcal{D}_S|} \sum_{i=1}^{|\mathcal{D}_S|} \left(\frac{1}{2} \left\| \hat{\mathbf{h}}_{w_i} - \mathbf{h}_S \right\|_2^2 \right) + \lambda \left(\frac{1}{2\sigma^2} \left\| \mathbf{h}_S - \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \hat{\mathbf{h}}'_w \right\|_2^2 \right) + C. \quad (23)
 \end{aligned}$$

Empirically, \mathcal{X}_S is the same to \mathcal{S} in Eq. (23).

$$L(D_S, \mathcal{Q}_S) = \mathbb{E}_{h \sim \mathcal{Q}_S} \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \left(\frac{1}{2} \left\| \hat{\mathbf{h}}_w - \mathbf{h}_S \right\|_2^2 \right) + \lambda \left(\frac{1}{2\sigma^2} \left\| \mathbf{h}_S - \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \hat{\mathbf{h}}'_w \right\|_2^2 \right) + C. \quad (24)$$

We take derivative of Eq. (24) with respect to \mathbf{h}_S , and set it to zero.

$$\nabla_{\mathbf{h}_S} \hat{L}(\mathcal{D}_S, \mathcal{Q}_S) = 0 \quad (25)$$

$$\frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} (\mathbf{h}_S - \hat{\mathbf{h}}_w) + \frac{\lambda}{\sigma^2} \left(\mathbf{h}_S - \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \hat{\mathbf{h}}'_w \right) = 0 \quad (26)$$

$$\mathbf{h}_S - \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \hat{\mathbf{h}}_w + \alpha \mathbf{h}_S - \frac{\alpha}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \hat{\mathbf{h}}'_w = 0 \quad (27)$$

$$(1 + \alpha) \mathbf{h}_S - \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \hat{\mathbf{h}}_w - \frac{\alpha}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \hat{\mathbf{h}}'_w = 0 \quad (28)$$

$$(1 + \alpha) \mathbf{h}_S - \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} (\hat{\mathbf{h}}_w + \alpha \hat{\mathbf{h}}'_w) = 0 \quad (29)$$

$$(1 + \alpha) \mathbf{h}_S = \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} (\hat{\mathbf{h}}_w + \alpha \hat{\mathbf{h}}'_w) \quad (30)$$

$$\mathbf{h}_S = \frac{1}{(1 + \alpha) |\mathcal{S}|} \sum_{w \in \mathcal{S}} (\hat{\mathbf{h}}_w + \alpha \hat{\mathbf{h}}'_w), \quad (31)$$

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

C.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

$$\text{IDF}(w) = \log \left(\frac{N_S}{\sum_{i=1}^{N_S} \mathbb{I}[w \in \mathcal{S}_i]} \right) + 1, \quad (32)$$

where N_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. [26]). Our interest is to obtain this heuristic method from our PAC-Bayes bound.

We start from Eq. (17). Each variance σ^2 of priors depends on each word w .

$$\sum_{w \in \mathcal{S}} \text{KL} \left[\mathcal{Q}_S | \mathcal{N}(\cdot; \hat{\mathbf{h}}'_w, \sigma_w^2 I) \right] = \sum_{w \in \mathcal{S}} \frac{1}{2\sigma_w^2} \left(\|\mathbf{h}_S\|_2^2 - 2\mathbf{h}_S^\top \hat{\mathbf{h}}'_w + \|\hat{\mathbf{h}}'_w\|_2^2 \right) + C \quad (33)$$

$$= \|\mathbf{h}_S\|_2^2 \sum_{w \in \mathcal{S}} \frac{1}{2\sigma_w^2} - 2\mathbf{h}_S^\top \sum_{w \in \mathcal{S}} \frac{1}{2\sigma_w^2} \hat{\mathbf{h}}'_w + \sum_{w \in \mathcal{S}} \frac{1}{2\sigma_w^2} \|\hat{\mathbf{h}}'_w\|_2^2 + C \quad (34)$$

$$= \eta \|\mathbf{h}_S\|_2^2 - 2\mathbf{h}_S^\top \sum_{w \in \mathcal{S}} \frac{1}{2\sigma_w^2} \hat{\mathbf{h}}'_w + C' \quad (35)$$

$$= \eta \left(\|\mathbf{h}_S\|_2^2 - \frac{2}{\eta} \mathbf{h}_S^\top \sum_{w \in \mathcal{S}} \frac{1}{2\sigma_w^2} \hat{\mathbf{h}}'_w \right) + C' \quad (36)$$

$$= \eta \left\| \mathbf{h}_S - \frac{1}{\eta} \sum_{w \in \mathcal{S}} \frac{1}{2\sigma_w^2} \hat{\mathbf{h}}'_w \right\|_2^2 + C'', \quad (37)$$

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}_S .

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

$$L(\mathcal{D}_S, \mathcal{Q}_S) = \mathbb{E}_{h \sim \mathcal{Q}_S} \frac{1}{|\mathcal{D}_S|} \sum_{i=1}^{|\mathcal{D}_S|} l(\mathbf{x}_i, y_i, h) + \lambda \text{KL}(\mathcal{Q}_S | \mathcal{P}_S) \quad (38)$$

$$= \mathbb{E}_{h \sim \mathcal{Q}_S} \frac{1}{|\mathcal{D}_S|} \sum_{i=1}^{|\mathcal{D}_S|} \left(\frac{\beta_{w_i}}{2} \|\hat{\mathbf{h}}_{w_i} - \mathbf{h}_S\|_2^2 \right) + \lambda \eta \left\| \mathbf{h}_S - \frac{1}{\eta} \sum_{w \in \mathcal{S}} \frac{1}{2\sigma_w^2} \hat{\mathbf{h}}'_w \right\|_2^2 + C. \quad (39)$$

Given $\mathcal{X}_S = \mathcal{S}$, we take derivative of Eq. (39) with respect to \mathbf{h}_S , and set it to zero.

$$\nabla_{\mathbf{h}_S} \hat{L}(\mathcal{D}_S, \mathbf{h}_S) = 0 \quad (40)$$

$$\frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \beta_w (\mathbf{h}_S - \hat{\mathbf{h}}_w) + 2\lambda \eta \left(\mathbf{h}_S - \frac{1}{\eta} \sum_{w \in \mathcal{S}} \frac{1}{2\sigma_w^2} \hat{\mathbf{h}}'_w \right) = 0 \quad (41)$$

$$\frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \beta_w \mathbf{h}_S - \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \beta_w \hat{\mathbf{h}}_w + 2\lambda \eta \mathbf{h}_S - 2\lambda \sum_{w \in \mathcal{S}} \frac{1}{2\sigma_w^2} \hat{\mathbf{h}}'_w = 0 \quad (42)$$

$$\left(\frac{\sum_{w \in \mathcal{S}} \beta_w}{|\mathcal{S}|} + 2\lambda \eta \right) \mathbf{h}_S - \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \beta_w \hat{\mathbf{h}}_w - 2\lambda \sum_{w \in \mathcal{S}} \frac{1}{2\sigma_w^2} \hat{\mathbf{h}}'_w = 0 \quad (43)$$

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

$$\left(\sum_{w \in \mathcal{S}} \beta_w + 2\eta \lambda |\mathcal{S}| \right) \mathbf{h}_S - \sum_{w \in \mathcal{S}} \left(\beta_w \hat{\mathbf{h}}_w + \lambda |\mathcal{S}| \frac{1}{\sigma_w^2} \hat{\mathbf{h}}'_w \right) = 0 \quad (45)$$

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

$$\mathbf{h}_S = \frac{\sum_{w \in \mathcal{S}} \beta_w \hat{\mathbf{h}}_w + \frac{\lambda |\mathcal{S}|}{\sigma_w^2} \hat{\mathbf{h}}'_w}{\sum_{w \in \mathcal{S}} \beta_w + 2\eta \lambda |\mathcal{S}|}. \quad (47)$$

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

$$\mathbf{h}_{\mathcal{S}} = \frac{\sum_{w \in \mathcal{S}} \beta_w \hat{\mathbf{h}}_w + \lambda |\mathcal{S}| \text{IDF}(w) \hat{\mathbf{h}}'_w}{\sum_{w \in \mathcal{S}} \beta_w + \lambda |\mathcal{S}| \sum_{w \in \mathcal{S}} \text{IDF}(w)} \quad (48)$$

$$= \frac{\sum_{w \in \mathcal{S}} \beta_w \hat{\mathbf{h}}_w + \lambda |\mathcal{S}| \text{IDF}(w) \hat{\mathbf{h}}'_w}{\sum_{w \in \mathcal{S}} \beta_w + \lambda |\mathcal{S}| \text{IDF}(w)}. \quad (49)$$

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

$$\mathbf{h}_{\mathcal{S}} = \frac{\sum_{w \in \mathcal{S}} |\mathcal{S}| \text{IDF}(w) \hat{\mathbf{h}}_w + \lambda |\mathcal{S}| \text{IDF}(w) \hat{\mathbf{h}}'_w}{\sum_{w \in \mathcal{S}} |\mathcal{S}| \text{IDF}(w) + \lambda |\mathcal{S}| \text{IDF}(w)} \quad (50)$$

$$= \frac{\sum_{w \in \mathcal{S}} \text{IDF}(w) \hat{\mathbf{h}}_w + \lambda \text{IDF}(w) \hat{\mathbf{h}}'_w}{\sum_{w \in \mathcal{S}} \text{IDF}(w) + \lambda \text{IDF}(w)} \quad (51)$$

$$= \frac{\sum_{w \in \mathcal{S}} \text{IDF}(w) (\hat{\mathbf{h}}_w + \lambda \hat{\mathbf{h}}'_w)}{(1 + \lambda) \sum_{w \in \mathcal{S}} \text{IDF}(w)}. \quad (52)$$

Appendix D Other Transfer Learning Settings

D.1 Target Task with Pre-trained CBoW Vectors

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

$$L(\mathcal{D}_{\mathcal{S}}, \mathbf{h}_{\mathcal{S}}) = -\frac{1}{|\mathcal{S}|} \left[\ln \sigma \left(\hat{\mathbf{h}}_{\text{avg}_{\mathcal{S}}}^{\top} \mathbf{h}_{\mathcal{S}} \right) + \sum_{w_n \in \mathcal{N}_{\mathcal{S}}} \ln \sigma \left(-\hat{\mathbf{h}}_{w_n} \mathbf{h}_{\mathcal{S}} \right) \right] + \frac{\lambda}{2\sigma^2} \left\| \mathbf{h}_{\mathcal{S}} - \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \hat{\mathbf{h}}'_w \right\|_2^2. \quad (53)$$

D.2 Target Task with Output Word Vectors

Table 2: 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	$\sigma(\mathbf{h}_{\text{avg}_t}^{\top} \mathbf{h}'_{w_t})$	Negative log
Avg.	\mathbf{x}_w	\mathbb{R}	$\frac{1}{2} \ \mathbf{h}_{\mathcal{S}} - \hat{\mathbf{h}}_w\ _2^2$	Root square
IDF-Avg.	\mathbf{x}_w	\mathbb{R}	$\frac{1}{2} \ \mathbf{h}_{\mathcal{S}} - \hat{\mathbf{h}}_w\ _2^2$	Weighted root square
Input-trans. (Skip-gram)	\mathbf{x}_w	Binary	$\text{sign}(\hat{\mathbf{h}}_w^{\top} \mathbf{h}_{\mathcal{S}})$	Zero-one
Input-trans. (CBoW)	\mathcal{S}	Binary	$\text{sign}(\hat{\mathbf{h}}_{\text{avg}_{\mathcal{S}}}^{\top} \mathbf{h}_{\mathcal{S}})$	Zero-one
Output-trans. (Skip-gram)	\mathbf{x}_w	Binary	$\text{sign}(\mathbf{h}_{\mathcal{S}}^{\top} \hat{\mathbf{h}}'_w)$	Zero-one
Output-trans. (CBoW)	\mathbf{x}_w	Binary	$\text{sign}(\mathbf{h}_{\mathcal{S}}^{\top} \hat{\mathbf{h}}'_w)$	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 $h(\mathbf{x}_w) = \text{sign}(\mathbf{h}_{\mathcal{S}}^{\top} \hat{\mathbf{H}}' \mathbf{x}_w) = \text{sign}(\mathbf{h}_{\mathcal{S}}^{\top} \hat{\mathbf{h}}'_w)$, the prior is defined by $\mathcal{P}_{\mathcal{S}} = \frac{\prod_{w \in \mathcal{S}} \mathcal{N}(\hat{\mathbf{h}}_w, \sigma^2 I)}{\int \prod_{w \in \mathcal{S}} \mathcal{N}(\mathbf{h}; \hat{\mathbf{h}}_w, \sigma^2 I) d\mathbf{h}}$. In this target task, we minimize the loss function defined by

$$L(\mathcal{D}_{\mathcal{S}}, \mathbf{h}_{\mathcal{S}}) = -\frac{1}{|\mathcal{S}|} \left[\sum_{w \in \mathcal{S}} \ln \sigma \left(\mathbf{h}_{\mathcal{S}}^{\top} \hat{\mathbf{h}}'_w \right) + \sum_{w_n \in \mathcal{N}_{\mathcal{S}}} \ln \sigma \left(-\mathbf{h}_{\mathcal{S}}^{\top} \hat{\mathbf{h}}'_{w_n} \right) \right] + \frac{\lambda}{2\sigma^2} \left\| \mathbf{h}_{\mathcal{S}} - \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \hat{\mathbf{h}}_w \right\|_2^2. \quad (54)$$

We note that we do not average over output word vectors in Eq. (54) with pre-trained word vectors by CBoW because the original CBoW’s source task defined by Eq. (6) only averages over input word vectors.

Table. 2 shows the all source task and target task treated in this paper.

Appendix E Details of Experiments

E.1 Settings of Training of Word Vectors

We used English Wikipedia articles⁵ to train word vectors. We did pre-process this corpus with `wikifil.pl`⁶, 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`⁷: 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.

E.2 Classification Datasets

Table 3: Sentence Classification Datasets

Dataset	Task	#Train data	#Test data	#Classes
20news [11]	Topic classification	11 314	7 532	20
IMDb [12]	Sentiment analysis	25 000	25 000	2
SUBJ [13]	Subjectivity classification	8 000	2 000	2

Table. 3 shows three classification datasets. We did pre-process all datasets with `wikifil.pl` modified for our experiments.⁸ 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.

E.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 sentence. We linearly decreased the learning rate per 100 sentences. 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. We implemented our proposed algorithms with PyTorch [27].

E.4 Settings of Sentence Classification

We used a logistic regression classifier with `scikit-learn` [28] 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.

⁵We downloaded XML dump file created on Aug. 1, 2018 from <https://dumps.wikimedia.org/enwiki>.

⁶<https://github.com/facebookresearch/fastText/blob/master/wikifil.pl>

⁷<https://github.com/nzw0301/word2vec>

⁸<https://gist.github.com/nzw0301/4bc1e47172f8a8852e3ca5406dfa2015>