PAC-Bayes Analysis of Sentence Representation

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

February 13, 2019

Abstract

Learning sentence vectors from an unlabeled corpus has attracted attention because such vectors can represent sentences in a lower dimensional and continuous space. Simple heuristics using pre-trained word vectors are widely applied to machine learning tasks. However, they are not well understood from a theoretical perspective. We analyze learning sentence vectors from a transfer learning perspective by using a PAC-Bayes bound that enables us to understand existing heuristics. We show that simple heuristics such as averaging and inverse document frequency weighted averaging are derived by our formulation. Moreover, we propose novel sentence vector learning algorithms on the basis of our PAC-Bayes analysis.

1 Introduction

Representation learning [Bengio et al., 2013] is a class of the most fundamental machine learning tasks. It aims to obtain re-usable representations that capture useful features from a massive amount of data. Usually, a simple representation for a data sample is a fixed length d-dimensional real-valued vector. Therefore, it is used as the feature vector of several machine learning tasks, such as classification, clustering, and visualization. Representation learning by solving supervised tasks requires a lot of labeled data, especially, when using deep neural network models. For example, DeCAF [Donahue et al., 2014] image feature representations are based on AlexNet [Krizhevsky et al., 2012] trained on ImageNet [Deng et al., 2009], and InferSent [Conneau et al., 2017] sentence feature representations based on deep neural networks trained on the SNLI dataset [Bowman et al., 2015], which consists of 570k labeled sentence pairs. Kornblith et al. [2018] report a strong correlation between a source task's performance and a target task's performance with representations trained on the source task regarding image classification tasks. Therefore, a large amount of labeled data is one of the most critical factors of supervised representation learning. In contrast, unsupervised representation learning does not require label information. In addition, we easily apply these algorithms not only to labeled datasets, e.g., ImageNet and SNLI, but also large unlabeled datasets, e.g., images and text collected by a web crawler.

Unsupervised word representation learning [Collobert et al., 2011, Mnih and Kavukcuoglu, 2013, Mikolov et al., 2013a, Pennington et al., 2014, Bojanowski et al., 2017] is one of the most successful representation learning tasks. These word representations often improve the performance of other natural language processing tasks, such as text classification [Mikolov et al., 2018], neural machine translation [Qi et al., 2018], word similarity and analogy task [Bojanowski et al., 2017], and several benchmark tasks [Peters et al., 2018a,b]. In addition, we can easily use publicly available word vectors trained on unlabeled text datasets, e.g., SENNA[1] [Collobert et al., 2011], word2vec[2] [Mikolov et al., 2013a], GloVe[3] [Pennington et al., 2014], and fastText[4] [Grave et al., 2018].

Unsupervised sentence representation learning [Le and Mikolov, 2014, Kiros et al., 2015] is a more complex task than word representation learning because a sentence's meaning is determined by several factors, such as words, the word order, and grammar. The simplest algorithm is to average pre-trained word vectors over words appearing in a sentence. Even if we use models based on recurrent neural networks with many learning parameters, simpler models

---

[1] https://ronan.collobert.com/senna/
[2] https://code.google.com/archive/p/word2vec/
[3] https://nlp.stanford.edu/projects/glove/
[4] https://fasttext.cc/docs/en/crawl-vectors

based on pre-trained word vectors beat these large models [Hill et al., 2016, Wieting et al., 2016, Arora et al., 2017, Shen et al., 2018, Wieting and Kiela, 2019]. Unlike theoretical perspectives on word representation learning [Levy and Goldberg, 2014, Arora et al., 2016, Melamud and Goldberger, 2017], sentence representation learning is not understood well theoretically. As a first step toward theoretical understanding of sentence representation learning, we analyze simple heuristics from a transfer learning perspective by using PAC-Bayes theory.

PAC-Bayes theory [McAllester, 1999] enables us to analyze the generalization risk over a stochastic hypothesis class. PAC-Bayes bounds of neural transfer learning [Galanti et al., 2016, McNamara and Balcan, 2017] bound the generalization risk of a target task by the empirical risk of a source task. In similar machine learning problems, studies on meta-learning [Amit and Meir, 2018], lifelong learning [Pentina and Lampert, 2014, 2015], and domain adaptation [Germain et al., 2016a] also bound the generalization risk of a target task by the empirical risks of source tasks. However, these bounds cannot directly be used for the simple heuristic sentence representation learning based on the publicly available pre-trained word vectors without their empirical risks. In contrast, our transfer learning formulation and PAC-Bayes bound do not require the empirical risk of the source task.

Our contributions are as follows.

1. We introduce the concept of generalization to sentence representation learning from pre-trained word vectors.

2. We derive heuristic sentence vector methods from pre-trained word vectors by using PAC-Bayes theory.

3. We propose novel unsupervised sentence representation learning on the basis of our PAC-Bayes analysis.

2   Preliminaries

2.1   Learning Word Vectors

Word vector models aim to learn a map from each word to a d-dimensional real valued vector given a sequence of words $[w_1, \ldots, w_T]$. This word vector is also called a *word embedding* or a *distributed word representation*. In this paper, we use a popular neural word vector model, continuous skip-gram model, simply call Skip-gram, proposed by Mikolov et al. [2013b]. Skip-gram learns two types of vectors $\mathbf{i}$ and $\mathbf{o}$ associated with each word type. We call $\mathbf{i}$ an input word vector and $\mathbf{o}$ an output word vector. The terms $\hat{\mathbf{i}}$ and $\hat{\mathbf{o}}$ define a fixed input word vector and fixed output word vector trained on a sequence of words, respectively.

2.1.1   Continuous Skip-gram with Negative Sampling

Intuitively, Skip-gram learns word vectors by predicting context word w_c from given target word w_t in a training sequence. To do so, Skip-gram minimizes the negative log-likelihood of $p(w_c \mid w_t)$ modeled by a softmax function over training vocabulary $\mathcal{V}$. This straightforward training is intractable because evaluating the softmax layer takes $\mathcal{O}(|\mathcal{V}|)$. A fast method for training Skip-gram model is negative sampling proposed by Mikolov et al. [2013a], whose loss function is defined by

$$L_{SG} = \sum_{t=1}^{T} \sum_{w_c \in \mathcal{C}_t} \ell_{neg}(\mathbf{i}_{w_t}, \mathbf{o}_{w_c}), \tag{1}$$

$$\ell_{neg}(\mathbf{i}, \mathbf{o}) = - \left[\ln \sigma \left(\mathbf{i}^\top \mathbf{o} \right) + \sum_{w_n \in \mathcal{NS}} \ln \sigma \left(-\mathbf{i}^\top \mathbf{o}_{w_n} \right) \right], \tag{2}$$

where $\mathcal{C}_t$ represents a bag-of-words surrounding w_t in a sequence, σ is a logistic sigmoid function, and $\mathcal{NS}$ is a bag-of-words including k negative words sampled from pre-defined noise distribution p_n. Negative sampling loss (2) can be considered as a new predictive task such that word vector $\mathbf{i}_{w_t}$ predicts whether word w comes from context $\mathcal{C}_t$ or noise distribution p_n given w_t.

2.2   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 sentence[5] to d-dimensional vector $\mathbf{h}_\mathcal{S}$ given sentences $\{\mathcal{S}_1, \ldots, \mathcal{S}_N\}$, where $\mathcal{S}$ is a sequence of words. Since sentences consist of words, sentence

---

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

representation is affected by word information, such as sentiment polarity. We typically use pre-trained word vectors to learn sentence vectors because of the usefulness of their representations, which capture syntax and semantics [Mikolov et al., 2013a]. Given pre-trained word vectors $\{\hat{\mathbf{i}}_w, \hat{\mathbf{o}}_w : w \in \mathcal{V}\}$, the simplest way to obtain sentence vector $\mathbf{h}_\mathcal{S}$ is to average the pre-trained word vectors of words appearing in sentence $\mathcal{S}$, for example,

$$\mathbf{h}_\mathcal{S} = \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \hat{\mathbf{i}}_w, \tag{3}$$

$$\mathbf{h}_\mathcal{S} = \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \frac{\hat{\mathbf{i}}_w + \hat{\mathbf{o}}_w}{2}. \tag{4}$$

Although these heuristics ignore the word order in a sentence, empirically, they can outperform more complex deep neural network models in natural language processing tasks [Hill et al., 2016, Wieting et al., 2016, Arora et al., 2017].

2.3 PAC-Bayes Bound

We introduce a PAC-Bayes bound used in our analysis. Let $\mathcal{D}$ be an unknown test data distribution over $\mathcal{X} \times \mathcal{Y}$, where $\mathbf{x} \in \mathcal{X}$ is an input data sample and $y \in \mathcal{Y}$ is an output data 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 hypotheses class $\mathcal{H}$ and $\mathcal{Q}$ be a posterior distribution over $\mathcal{H}$. Given the test data distribution and stochastic hypotheses, the generalization risk is defined as $R(\mathcal{Q}) = \mathbb{E}_{h \sim \mathcal{Q}} \mathbb{E}_{(\mathbf{x},y) \sim \mathcal{D}} \ell(\mathbf{x}, y, h)$, where ℓ is a bounded loss function in $[0, 1]$. [6] In the same way, the empirical risk is defined as $\hat{R}(\mathcal{Q}) = \mathbb{E}_{h \sim \mathcal{Q}} \frac{1}{N} \sum_{i=1}^{N} \ell(\mathbf{x}_i, y_i, h)$. 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 $\mathcal{D}_\mathcal{S}$, $\forall \mathcal{Q}$,*

$$R(\mathcal{Q}) \leq \frac{1 - \exp\left[-\frac{\lambda}{N}\hat{R}(\mathcal{Q}) - \frac{\mathrm{KL}(\mathcal{Q}||\mathcal{P}) - \log \delta}{N}\right]}{1 - \exp(-\frac{\lambda}{N})}. \tag{5}$$

Intuitively, Eq. (5) means generalization risk $R(\mathcal{Q})$ is bounded by empirical risk $\hat{R}(\mathcal{Q})$ and hypothesis's complexity term $\mathrm{KL}(\mathcal{Q}||\mathcal{P})$. Hyperparameter λ adjusts the trade-off between the empirical risk and the complexity term, for example, the complexity term does not tend to contribute to the upper bound when $\lambda \to \infty$.

3 PAC-Bayesian Analysis of Sentence Vectors

We formulate learning word vectors and sentence vectors regarding transfer learning [Pan and Yang, 2010]. In this section, we assume that the source task is Skip-gram with negative sampling. Roughly, from our transfer learning perspective, the source task minimizes the loss function of Skip-gram (1) on word sequences by updating input and output word vectors $\mathbf{i}$ and $\mathbf{o}$. Then, a target task minimizes a loss function by updating sentence vector $\mathbf{h}_\mathcal{S}$ with the fixed pre-trained word vectors $\hat{\mathbf{i}}$ and $\hat{\mathbf{o}}$ given sentences. This transfer learning formulation enables us to analyze the generalization risk in learning sentence vectors with PAC-Bayes theory, which can consider transferability to a target hypothesis from a learned source hypothesis through prior knowledge. In Section 3.1, we explain in more details our transfer learning setting and the concept of the generalization of learning sentence vectors, and we also describe why the generalization becomes the matter to learn sentence vectors.

3.1 Generalization and Predictive Sentence Vectors with Pre-trained Word Vectors

Empirically, predictive word vector models, such as Skip-gram and CBoW [Mikolov et al., 2013b,a], stably outperform count models, such as a word co-occurrence matrix or its low-rank approximation with singular value decomposition in the natural language processing tasks [Baroni et al., 2014, Levy et al., 2015]. They learn word representations by

---

[6]When the maximum of $\ell > 1$, we can use the same bound by using rescaling loss $\ell/\ell_{\max}$ in the same way as Germain et al. [2016b], where $\ell_{\max}$ is the maximum value of ℓ.

solving a predictive task that makes learned word representations informative, for example, Skip-gram with negative sampling (1) predicts whether word w is from context $\mathcal{C}_t$ or noise distribution p_n by using target word's vector $\mathbf{i}_{w_t}$ described in Section 2.1.1. Predictive word vector models affect sentence vector models, for example, the PV-DBOW model [Le and Mikolov, 2014] predicts words in a paragraph given its paragraph id, and the Skip-Thoughts model [Kiros et al., 2015] predicts the previous sentence and the next sentence given the current sentence. Thus, it is essential for learning better sentence representations to solve predictive tasks.

We define test data distribution $\mathcal{D}$ and training dataset $\mathcal{D}_S$ to consider the generalization of predictive sentence vectors based on negative sampling loss for sentence $\mathcal{S}$, for example, "Paris is one of the most beautiful cities in the world." The predictive task is to classify whether w in $\mathcal{S}$ comes from generative distribution or noise distribution p_n. We assume that sentence $\mathcal{S}$ is a bag-of-words, and we also assume that each word w in $\mathcal{S}$ is sampled from a generative model based on a unigram language model with a Dirichlet prior:

$$\boldsymbol{\phi} \sim \text{Dirchlet}(\boldsymbol{\phi}; \boldsymbol{\gamma}), \tag{6}$$

$$w \sim \text{Multinomial}(w; \boldsymbol{\phi}), \tag{7}$$

where $\boldsymbol{\gamma} \in \mathbb{R}_+^{|\mathcal{V}|}$ is a hyperparameter of the Dirichlet distribution. We assume noise distribution p_n is defined over all vocabulary, i.e., the unigram distribution, and unknown test dataset $\mathcal{D}$ generates word w and label y in the following ways:

$$y \sim \text{Bernoulli}(y; \pi) = p^{\frac{y+1}{2}}(1-p)^{\frac{1-y}{2}} \text{ for } y \in \{-1, 1\}, \tag{8}$$

$$w \sim p(w \mid y; \boldsymbol{\phi}), \tag{9}$$

$$p(w \mid y = 1; \boldsymbol{\phi}) = \text{Multinomial}(w; \boldsymbol{\phi}), \tag{10}$$

$$p(w \mid y = -1; \boldsymbol{\phi}) = p_n(w), \tag{11}$$

where π is a hyperparameter of the Bernoulli distribution. Let an input data sample be $\mathbf{x} \in \mathcal{X} = \{\mathbf{x}_w \mid w \in \mathcal{V}\}$ and an output data sample be $y \in \mathcal{Y} = \{-1, 1\}$, where $\mathbf{x}_w$ is a one-hot vector of word w, and y is a binary label. The observed training dataset denotes $\mathcal{D}_S = \{(\mathbf{x}_w, 1) \mid w \in \mathcal{S}\} \bigcup \{(\mathbf{x}_{w_i}, -1) \mid w_i \in p_n(w)\}_{i=1}^{k|\mathcal{S}|}$, where k is the number of negative samples per each word in $\mathcal{S}$. This training dataset is sampled from $\mathcal{D}$. Test dataset $\mathcal{D}$ contains positive data related to other words to describe *Paris*, i.e., "France," "capital," and "incredible," in addition to the training dataset for the example sentence.

We formulate the binary classification problem as the problem mentioned earlier. Let $h : \mathcal{X} \rightarrow \mathcal{Y}$ be a hypothesis $h(\mathbf{x}; \mathbf{h}_\mathcal{S}, \mathbf{W}) = \text{sign}[\mathbf{h}_\mathcal{S}^\top(\mathbf{W}\mathbf{x})]$ parameterized by sentence vector $\mathbf{h}_\mathcal{S}$ and weight matrix $\mathbf{W} \in \mathbb{R}^{d \times |\mathcal{V}|}$. Note that $\mathbf{h}_\mathcal{S}$ and $\mathbf{W}$ depend only on the sentence. We expect $\mathbf{h}_\mathcal{S}$ to capture a representation related to *Paris* in the vector space $\mathbb{R}^d$ for correct classification. Let loss function ℓ be zero-one loss $\ell(\mathbf{x}, y, h) = \mathbb{I}[h(\mathbf{x}; \mathbf{h}_\mathcal{S}, \mathbf{W}) \neq y]$. When we estimate $\mathbf{h}_\mathcal{S}$ by minimizing usual empirical risk $L(\mathcal{D}_S) = \frac{1}{|\mathcal{D}_S|} \sum_{(x,y) \in \mathcal{D}_S} \ell(\mathbf{x}, y, h)$, $\mathbf{h}_\mathcal{S}$ can capture empirical $\hat{\boldsymbol{\phi}}$ on the basis of the training data; however, it does not become a feature representation such that the hypothesis predicts $y = 1$ for a word excluding $\mathcal{D}_S$ but it is likely to be sampled from Eq. (7), i.e., "France." As the result of minimizing the empirical risk, this sentence vector is poor to represent the sentence's meaning to describe *Paris*, in other words, the sentence vector overfits on the training dataset. This is why we need to consider the generalization of learning sentence vectors. Unfortunately, we cannot minimize the generalization risk, $L(\mathcal{D}) = \mathbb{E}_{(x,y) \sim \mathcal{D}} \ell(\mathbf{x}, y, h)$, since this risk depends on unknown $\mathcal{D}$. Therefore we need to minimize an upper bound of the generalization risk.

We use the PAC-Bayes framework to bound generalization risk R because it is empirically tighter than Vapnik-Chervonenkis dimension based generalization bounds [Dziugaite and Roy, 2017, McNamara and Balcan, 2017]. We apply the PAC-Bayes framework to the previous predictive task. PAC-Bayes theory assumes h is a stochastic hypothesis. That means sentence vector $\mathbf{h}_\mathcal{S}$ is sampled from posterior $\mathcal{Q}$ such as multivariate Gaussian distribution $\mathcal{N}(; \boldsymbol{\mu}_\mathcal{S}, \sigma_\mathcal{Q}^2 I)$, where $\boldsymbol{\mu}_\mathcal{S}$ is a d dimensional mean vector, $\sigma_\mathcal{Q}^2$ is a variance parameter, and I is the identity matrix. We can also introduce a posterior distribution related to $\mathbf{W}$. However, on the basis of empirical studies [Baroni et al., 2014, Levy et al., 2015], we use fixed Skip-gram's pre-trained vectors $\{\hat{\mathbf{i}}_w \mid w \in \mathcal{V}\}$ as $\mathbf{W}$ where the w-th column of $\mathbf{W}$ corresponds to $\hat{\mathbf{i}}_w$, because these vectors already capture considerable word syntax and semantics on the source task's sequence. That means $\mathbf{W}$ is not sampled from a posterior distribution over the hypothesis class in the PAC-Bayesian framework.

By using pre-trained word vectors in the hypothesis, we can formulate transfer learning where the source task is learning word vectors by Skip-gram with negative sampling on a large unlabeled corpus, and the target task is learning posterior parameters $\boldsymbol{\mu}_\mathcal{S}$ and $\sigma_\mathcal{Q}^2$ with fixed pre-trained word vectors by minimizing Eq. (5) including the predictive task.

4

In this transfer learning formulation and PAC-Bayes framework, it is natural to use pre-trained output word vectors $\{\hat{\mathbf{o}}_w \mid w \in \mathcal{V}\}$ as the parameters of prior distribution $\mathcal{P}$ of $\mathbf{h}_\mathcal{S}$ because words in the sentence probably co-occur in the source task's sequence as well, i.e., "`Paris`" and "`beautiful`." Similar to the posterior, we use a multivariate Gaussian distribution parameterized by output word vectors, for example, the summing over vectors of words appearning in $\mathcal{S}$, $\mathcal{P} = \mathcal{N}(; \sum_{w \in \mathcal{S}} \hat{\mathbf{o}}_w, \sigma_\mathcal{P}^2 I)$, where $\sigma_\mathcal{P}^2$ is a variance parameter. The prior prevents the posterior from overfitting on the target's training data by penalizing the posterior far from the prior via KL divergence in Eq. (5).

3.2 Loss Function for Sentence Vector Training Based on PAC-Bayes Bound

We define a PAC-Bayes bound to analyze sentence representation learning defined by Eq. (3) and Eq. (4) on the basis of the setting described in Section 3.1. Given sentence $\mathcal{S}$ and pre-trained input and output word vectors in the source task, let $|\mathcal{S}|$ be the number of words in $\mathcal{S}$ and $\mathcal{D}_S$ be a training dataset sampled i.i.d. from $\mathcal{D}$. We define prior distribution over hypothesis class $\mathcal{H}$ as $\mathcal{P}_\mathcal{S} \propto \prod_{w \in \mathcal{S}} \mathcal{N}(; \hat{\mathbf{o}}_w, \sigma_\mathcal{P}^2 I)$. This prior distribution is known as the product of experts [Hinton, 2002] such that each expert is a multivariate Gaussian distribution parameterized by $\hat{\mathbf{o}}$ of the words appearing in $\mathcal{S}$. We also define a posterior distribution as $\mathcal{Q}_\mathcal{S} := \mathcal{N}(; \boldsymbol{\mu}_\mathcal{S}, \sigma_\mathcal{Q}^2 I)$. Theorem 2 shows the loss function based on PAC-Bayes bound.

Theorem 2 (Loss Function of Learning Sentence Vector from Pre-trained Word Vectors Based on PAC-Bayes Bound). *Let $\mathcal{Q}_\mathcal{S} = \mathcal{N}(; \boldsymbol{\mu}_\mathcal{S}, \sigma_\mathcal{Q}^2 I)$ be the posterior of sentence vector $\mathbf{h}_\mathcal{S}$. Given $\lambda > 0$, $\mathcal{S}$, $\mathcal{D}_S$, $\mathcal{P}_\mathcal{S} \propto \prod_{w \in \mathcal{S}} \mathcal{N}(; \hat{\mathbf{o}}_w, \sigma_\mathcal{P}^2 I)$, and bounded loss ℓ, minimizing Eq. (5) is equivalent to minimizing the loss function defined by*

$$L(\mathcal{D}_S, \mathcal{Q}_\mathcal{S}) = \mathop{\mathbb{E}}_{h \sim \mathcal{Q}_\mathcal{S}} \frac{1}{N} \sum_{i=1}^N \ell(\mathbf{x}_i, y_i, h) + \frac{|\mathcal{S}|}{2\sigma_\mathcal{P}^2 \lambda} \left\| \boldsymbol{\mu}_\mathcal{S} - \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \hat{\mathbf{o}}_w \right\|_2^2 + \frac{|\mathcal{S}|d}{2\lambda} \left(\ln \frac{\sigma_\mathcal{P}^2}{\sigma_\mathcal{Q}^2} + \frac{\sigma_\mathcal{Q}^2}{\sigma_\mathcal{P}^2} \right) + C \qquad (12)$$

where C is a constant term that does not depend on $\mathcal{Q}$.

We prove Theorem 2 by using Theorem 1. The proof is shown in appendix A.1. In Eq. (12), the first term is the empirical risk of a target task, and the second and the third terms penalize posterior $\mathcal{Q}_\mathcal{S}$ far from prior $\mathcal{P}_\mathcal{S}$.

3.3 Sentence Vectors with Squared L2 Loss

We use Theorem 2 to analyze heuristic sentence vector models. Intuitively, this predictive task is to predict a pre-trained word vector from a hypothesis directly. Let $\mathcal{X} := \{\mathbf{x}_w \mid w \in \mathcal{V}\}$, where $\mathbf{x}_w$ represents a one-hot vector of word w. Let $\mathcal{Y} := \{\hat{\mathbf{i}}_w \mid w \in \mathcal{V}\}$ and hypothesis h be $h(\mathbf{x}) := \mathbf{h}_\mathcal{S}$. Let training samples $\mathcal{D}_S := \{(\mathbf{x}_w, \hat{\mathbf{i}}_w) \mid w \in \mathcal{S}\}$. Each sentence does not depend on other sentences at all, so hypothesis h always returns sentence vector $\mathbf{h}_\mathcal{S}$. We use squared L2 loss function $\ell(\mathbf{x}, y, h) := \frac{1}{2} \|y - h\|_2^2$. We assume that ℓ is bounded in $[0, \ell_{\max}]$. In this setting, we can obtain the closed form of parameters of $\mathcal{Q}_\mathcal{S}$:

$$\boldsymbol{\mu}_\mathcal{S} = \frac{1}{(1+\alpha)|\mathcal{S}|} \sum_{w \in \mathcal{S}} \left(\hat{\mathbf{i}}_w + \alpha \hat{\mathbf{o}}_w \right), \qquad (13)$$

$$\sigma_\mathcal{Q}^2 = \sigma_\mathcal{P}^2, \qquad (14)$$

where $\alpha = \frac{|\mathcal{S}|}{\sigma_\mathcal{P}^2 \lambda}$. Details on solutions are given in appendix A.2. From Eq. (13), we can obtain two heuristic sentence representation learning algorithms.

Corollary 2.1 (Sentence Vector by Averaging Pre-trained Input Word Vectors). *Posterior's mean vector $\boldsymbol{\mu}_\mathcal{S}$ that is estimated by minimizing Eq. (12) with $\alpha = 0$, in other words, $\lambda \to \infty$, is equivalent to Eq. (3), $\frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \hat{\mathbf{i}}_w$.*

Corollary 2.2 (Sentence Vector by Averaging Pre-trained Input and Output Word Vectors). *Posterior's mean vector $\boldsymbol{\mu}_\mathcal{S}$ that is estimated by minimizing Eq. (12) with $\alpha = 1$, is equivalent to Eq. (4), $\frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \frac{\hat{\mathbf{i}}_w + \hat{\mathbf{o}}_w}{2}$.*

Empirically, the averaged vector of an input word vector and an output word vector, $\frac{\hat{\mathbf{i}} + \hat{\mathbf{o}}}{2}$, can improve the performance of downstream tasks, e.g., Levy et al. [2015]. Corollary 2.2 offers a novel perspective on the operation in the heuristic sentence representation learning algorithm.

3.3.1 IDF Weighting

Sentence vectors by averaging weighed word vectors with inverse document frequency (IDF) is another simple heuristic [Lilleberg et al., 2015]. IDF is also a widely used heuristic to weight a word in text collections. The IDF of word w is defined by

$$\text{IDF}(w) = \log\left(\frac{N_{\mathcal{S}}}{\sum_{i=1}^{N_{\mathcal{S}}} \mathbb{I}\left[w \in \mathcal{S}_i\right]}\right) + 1, \tag{15}$$

where $N_{\mathcal{S}}$ is the number of training sentences. Our interest is to derive this heuristic from our PAC-Bayes bound. We change the loss function to sample-dependent weighted loss function $\ell(\mathbf{x}, y, h) = \beta(\mathbf{x})\|y - h(\mathbf{x})\|_2^2$, where β is a weighting function from a one-hot vector to $\mathbb{R}_+$. We also change the prior to $\mathcal{P}_{\mathcal{S}} \propto \prod_{w \in \mathcal{S}} \mathcal{N}(;\hat{\mathbf{o}}_w, \sigma_{\mathcal{P}_w}^2 I)$. When we set $\beta(\mathbf{x}) = |\mathcal{S}|\text{IDF}(\mathbf{x})$ and $\sigma_{\mathcal{P}_w}^2 = \text{IDF}(w)^{-1}$, we can obtain the closed form of $\boldsymbol{\mu}_{\mathcal{S}}$ and $\sigma_{\mathcal{Q}}^2$:

$$\boldsymbol{\mu}_{\mathcal{S}} = \frac{1}{\left(1 + \frac{1}{\lambda}\right)\sum_{w \in \mathcal{S}} \text{IDF}(w)} \sum_{w \in \mathcal{S}} \text{IDF}(w)\left(\hat{\mathbf{i}}_w + \frac{1}{\lambda}\hat{\mathbf{o}}_w\right), \tag{16}$$

$$\sigma_{\mathcal{Q}}^2 = \left(\frac{1}{|\mathcal{S}|}\sum_{w \in \mathcal{S}} \frac{1}{\sigma_{\mathcal{P}_w}^2}\right)^{-1}. \tag{17}$$

The details are in appendix A.3. This sentence vector differs slightly from IDF weighting of a sentence vector proposed by Lilleberg et al. [2015] since word vectors are weighted relatively by IDF in a sentence.

3.4 Sentence Vectors with Zero-one Loss

From our PAC-Bayes analysis of simple heuristics, we found that the problem, i.e., the loss function, differs between the source and target tasks in existing heuristics. From a transfer learning perspective, a 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. The new target task is based on the problem setting in Section 3.1: predicting whether a word in a sentence comes from a generative distribution or a noise distribution. We expect that sentence vectors are similar in embedded space if sentences share similar meanings when the generalization risk is sufficiently small.

We apply Theorem 2 to the predictive task defined in Section 3.1. Let the output space be $\mathcal{Y} := \{-1, 1\}$, loss function l be zero-one loss $\ell(\mathbf{x}, y, h) := \mathbb{I}[h(\mathbf{x}) \neq y]$, and the hypothesis be $h(\mathbf{x}_w) := \text{sign}[\mathbf{h}_{\mathcal{S}}^{\top}(\hat{\mathbf{I}}\mathbf{x}_w)]$, where the w-th column of $\hat{\mathbf{I}} \in \mathbb{R}^{d \times |\mathcal{V}|}$ corresponds to the w-th pre-trained input word vector $\hat{\mathbf{i}}_w$. Hypothesis distributions $\mathcal{P}$ and $\mathcal{Q}$ are the same as in the squared L2 loss case. In practice, we minimize the following loss function by using negative sampling as a surrogate loss function of zero-one loss:

$$L(\mathcal{D}_S, \mathcal{Q}_S) = \underset{h \sim \mathcal{Q}_S}{\mathbb{E}} \frac{1}{|\mathcal{S}|}\sum_{w \in \mathcal{S}} \ell_{neg}(\mathbf{h}_{\mathcal{S}}, \hat{\mathbf{i}}_w) + \frac{|\mathcal{S}|}{2\sigma_{\mathcal{P}}^2\lambda}\left\|\boldsymbol{\mu}_{\mathcal{S}} - \frac{1}{|\mathcal{S}|}\sum_{w \in \mathcal{S}} \hat{\mathbf{o}}_w\right\|_2^2 + \frac{|\mathcal{S}|d}{2\lambda}\left(\ln\frac{\sigma_{\mathcal{P}}^2}{\sigma_{\mathcal{Q}}^2} + \frac{\sigma_{\mathcal{Q}}^2}{\sigma_{\mathcal{P}}^2}\right). \tag{18}$$

Corollary 2.3 (Relationship to Paragraph Vector Models). *PV-DBOW, an instance of paragraph vector models proposed by Le and Mikolov [2014], with negative sampling loss is the same as Eq. (18) with $\lambda \to \infty$ and with trainable word vector $\mathbf{i}$ from scratch and do not take the expectation over the posterior.*

4 Other Transfer Learning Settings

4.1 Another Sentence Vector Modeling for Target Task: Learnable Word Vectors

Minimizing Eq. (18) is inefficient because each sentence vector's posterior $\mathcal{Q}$ is completely independent of other sentences. We consider another sentence vector's modeling based on trainable word vectors defined by

$$\mathbf{h}_{\mathcal{S}} = \frac{1}{|\mathcal{S}|}\sum_{w \in \mathcal{S}} \mathbf{h}_w. \tag{19}$$

We estimate posterior's parameters of word vector $\mathbf{h}_w$ on the target task instead of estimating posterior's parameters of $\mathbf{h}_S$ directly. This formulation is similar to supervised `fastText` [Joulin et al., 2017] and `fastSent` [Hill et al., 2016]. Thanks to this word-based modeling, we do not need to learn the vectors of new sentences. Moreover, we can initialize each word vector $\boldsymbol{\mu}_w$ by the prior's pre-trained word vector $\hat{\mathbf{o}}$. Let $\mathcal{X} := \{(\mathcal{S}_i, \mathbf{x}_w) \mid i \in \mathbb{N}_{\leq N_S}, w \in \mathcal{V}\}$, the prior of $\mathbf{h}_w$ be $\mathcal{P}_w = \mathcal{N}(\hat{\mathbf{o}}_w, \sigma_{\mathcal{P}_w}^2 I)$, and let the posterior of hypothesis be $\mathcal{Q}_w = \mathcal{N}(; \boldsymbol{\mu}_w, \sigma_{\mathcal{Q}_w}^2 I)$ for all w.

Theorem 3 (Loss Function of Word Based Learning Sentence Vector from Pre-trained Word Vectors Based on PAC-Bayes Bound)**.** *Let* $\mathcal{Q} = \{\mathcal{N}(; \boldsymbol{\mu}_w, \sigma_{\mathcal{Q}_w}^2 I) \mid w \in \mathcal{V}\}$ *be the posterior of word vector* $\mathbf{h}_w$*. Given* $\lambda > 0$*,* $\mathcal{D}_S$*,* $\mathcal{P}_w = \{\mathcal{N}(; \hat{\mathbf{o}}_w, \sigma_{\mathcal{P}_w}^2 I) \mid w \in \mathcal{V}\}$*, and bounded loss* ℓ*, minimizing Eq. (5) is equivalent to minimizing the loss function defined by*

$$L(\mathcal{D}_S, \mathcal{Q}) = \mathop{\mathbb{E}}_{h \sim \mathcal{Q}} \frac{1}{N} \sum_{i=1}^{N} \ell(\mathbf{x}_i, y_i, h) + \frac{1}{2\lambda} \sum_{w \in \mathcal{V}} \left[\frac{1}{\sigma_{\mathcal{P}_w}^2} ||\boldsymbol{\mu}_w - \hat{\mathbf{o}}_w||_2^2 + d \left(\ln \frac{\sigma_{\mathcal{P}_w}^2}{\sigma_{\mathcal{Q}_w}^2} + \frac{\sigma_{\mathcal{Q}_w}^2}{\sigma_{\mathcal{P}_w}^2} \right) \right] + C, \quad (20)$$

where C *is a constant term that does not depend on posterior parameters.*

The proof is almost the same as Theorem 2. The difference is replacing the KL term with $\mathrm{KL}(\mathcal{Q}||\mathcal{P}) = \sum_{w \in \mathcal{V}} \mathrm{KL}(\mathcal{Q}_w||\mathcal{P}_w)$.

The practical surrogate loss function using negative sampling loss defined by

$$L(\mathcal{D}_S, \mathcal{Q}) = \mathop{\mathbb{E}}_{h \sim \mathcal{Q}} \frac{1}{N} \sum_{(\mathcal{S}, w) \sim \mathcal{D}_S} \ell_{neg} \left(\frac{1}{|\mathcal{S}|} \sum_{v \in \mathcal{S}} \mathbf{h}_v, \hat{\mathbf{i}}_w \right) + \frac{1}{2\lambda} \sum_{w \in \mathcal{V}} \left[\frac{1}{\sigma_{\mathcal{P}_w}^2} ||\boldsymbol{\mu}_w - \hat{\mathbf{o}}_w||_2^2 + d \left(\ln \frac{\sigma_{\mathcal{P}_w}^2}{\sigma_{\mathcal{Q}_w}^2} + \frac{\sigma_{\mathcal{Q}_w}^2}{\sigma_{\mathcal{P}_w}^2} \right) \right]. \quad (21)$$

Recall that our goal is to obtain general sentence representations. However, Eq. (21) means that lower frequency words do not relatively change from the prior because negative sampling loss depends on the word frequency when $\frac{1}{\sigma_{\mathcal{P}_w}^2}$ is the same for all words. In natural language data, high-frequency words are meaningless to represent sentence meaning because they appear among almost all sentences, for example, "`the`" and "`to`." To avoid learning such poor sentence representations, we define the variance parameters of each prior distribution as

$$\sigma_{\mathcal{P}_w}^2 = \frac{\sum_{v \in \mathcal{V}} freq(v)}{freq(w)}, \quad (22)$$

where $freq(w)$ is the frequency of word w in a training dataset. Intuitively, this prior penalizes higher frequency words rather than lower frequency words.

4.2 Switching Pre-trained Word Vectors' Role

We consider another target task setting, that is switching the role of fixed input word vectors and fixed output word vectors. For squared L2 loss and zero-one loss, we use fixed pre-trained output word vectors in output samples and the hypothesis respectively, and we use fixed pre-trained input word vectors as the prior distribution's mean vector.

Table. 1 summarizes all source and target tasks discussed in this paper. We call sentence vector (13) as `PB-L2` and (16) as `PB-IDF-L2`. We also call sentence vectors minimizing Eq. (18) as `PB-neg`, and we call sentence vectors minimizing Eq. (21) with Eq. (22) as `w-PB-neg`. We add the prefix "`i-`" to their names when we switch the roles of input words vectors and output word vectors.

5 Experiments

We verified our analysis in sentence classification tasks because learned sentence vectors work as feature vectors for supervised machine learning tasks. We compared simple heuristic methods derived by our analysis and our sentence vector learning methods. We trained Skip-gram on a large text corpus as a source task. Using pre-trained vectors of Skip-gram, we estimated posterior's parameters, $\boldsymbol{\mu}$ and $\sigma_{\mathcal{Q}}^2$, obtained from our analysis as target tasks, which are shown in Table. 1. For `w-PB-neg` and `i-w-PB-neg`, we calculated a sentence vector of $\mathcal{S}$ by averaging over posterior's $\boldsymbol{\mu}_w$ of w appearing in S. We used posterior's $\boldsymbol{\mu}_S$ as a sentence vector of $\mathcal{S}$ for other models.

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

Model	Input $\mathbf{x}$	Output y	Hypothesis h	Loss ℓ	Prior's word vectors
Skip-gram	w_t, w_c	Binary	$\sigma(\mathbf{i}_{w_t}^\top \mathbf{o}_{w_c})$	Negative sampling	Negative sampling
PB-L2	$\mathbf{x}_w$	$\hat{\mathbf{i}}_w$	$\mathbf{h}_\mathcal{S}$	Squared L2	$\hat{\mathbf{O}}$
i-PB-L2	$\mathbf{x}_w$	$\hat{\mathbf{o}}_w$	$\mathbf{h}_\mathcal{S}$	Squared L2	$\hat{\mathbf{I}}$
PB-IDF-L2	$\mathbf{x}_w$	$\hat{\mathbf{i}}_w$	$\mathbf{h}_\mathcal{S}$	Weighted squared L2	$\hat{\mathbf{O}}$
i-PB-IDF-L2	$\mathbf{x}_w$	$\hat{\mathbf{o}}_w$	$\mathbf{h}_\mathcal{S}$	Weighted squared L2	$\hat{\mathbf{I}}$
PB-neg	$(\mathcal{S}, \mathbf{x}_w)$	Binary	$\sigma[\mathbf{h}_\mathcal{S}^\top (\hat{\mathbf{I}}\mathbf{x}_w)]$	Negative sampling	$\hat{\mathbf{O}}$
i-PB-neg	$(\mathcal{S}, \mathbf{x}_w)$	Binary	$\sigma[\mathbf{h}_\mathcal{S}^\top (\hat{\mathbf{O}}\mathbf{x}_w)]$	Negative sampling	$\hat{\mathbf{I}}$
w-PB-neg	$(\mathcal{S}, \mathbf{x}_w)$	Binary	$\sigma[(\frac{1}{|\mathcal{S}|}\sum_{w\in\mathcal{S}}\mathbf{h}_w)^\top (\hat{\mathbf{I}}\mathbf{x}_w)]$	Negative sampling	$\hat{\mathbf{O}}$
i-w-PB-neg	$(\mathcal{S}, \mathbf{x}_w)$	Binary	$\sigma[(\frac{1}{|\mathcal{S}|}\sum_{w\in\mathcal{S}}\mathbf{h}_w)^\top (\hat{\mathbf{O}}\mathbf{x}_w)]$	Negative sampling	$\hat{\mathbf{I}}$

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

5.1 Settings of the Source Task

We used English Wikipedia articles[7] to train word vectors. We pre-processed this corpus with `wikifil.pl`[8], and then removed words appearing less than 5 times. In the pre-processed corpus, the size of vocabulary $|\mathcal{V}|$ was 2 472 211, and the number of tokens was 4 517 598 626.

We used the following hyperparameters to train word vectors with `word2vec`[9]: word vector dimensionality d was 300, the window size was 5, the sub-sampling parameter was 0.0001, the number of iterations was 5, the number of negative samples k was 15, the noise distribution parameter was $\frac{3}{4}$, and the initial learning rate was 0.025.

5.2 Settings of Target Tasks

5.2.1 Classification Datasets

We used three classification datasets for the target tasks to learn sentence vectors: 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 the three classification datasets. We pre-processed both datasets in the same manner as pre-processing of the Wikipedia corpus by using `wikifil.pl`. We split the SUBJ dataset randomly into 80 % as training data and 20 % as test data for evaluation because this dataset was not split into training data and test data. Note that we *did not use* label information at all while training target tasks.

5.2.2 Settings of Sentence Vector Methods

We trained simple heuristic algorithms derived by our analysis. We called Eq. (13) with $\alpha \in \{0, 1\}$ as `Average`, and we call Eq. (16) with $\frac{1}{\lambda} \in \{0, 1\}$ as `IDF-Average`. In the same way as the notation on Table. 1, the prefix "i-" means that we switched the roles of input words vectors and output word vectors. Note that sentence vectors of `Average` with $\alpha = 1$ and `i-Average` with $\alpha = 1$ are the same, and also sentence vectors of `IDF-Average` with $\frac{1}{\lambda} = 1$ and `i-IDF-Average` with $\frac{1}{\lambda} = 1$ are the same. So we omitted the results of `i-Average` with $\alpha = 1$ and `i-IDF-Average` with $\frac{1}{\lambda} = 1$.

---

[7] We downloaded XML dump file created on Aug. 1, 2018.
[8] https://github.com/facebookresearch/fastText/blob/master/wikifil.pl
[9] https://github.com/nzw0301/word2vec

Table 3: Test accuracy of sentence classification (averaged over three times). Note that rows of `i-Average` with $\alpha = 1$ and `i-IDF-Average` with $\frac{1}{\lambda} = 1$ were omitted because their results were the same as the result without `i-` models.

Model		20news	IMDb	SUBJ
`Average`	$\alpha = 0$	0.748 ± 0.000	0.842 ± 0.001	$\mathbf{0.908 \pm 0.001}$
`Average`	$\alpha = 1$	0.745 ± 0.001	0.838 ± 0.000	0.904 ± 0.000
`IDF-Average`	$\frac{1}{\lambda} = 0$	0.737 ± 0.001	0.823 ± 0.001	0.907 ± 0.000
`IDF-Average`	$\frac{1}{\lambda} = 1$	0.735 ± 0.000	0.821 ± 0.000	0.906 ± 0.001
`i-Average`	$\alpha = 0$	0.752 ± 0.000	0.842 ± 0.000	0.906 ± 0.001
`i-IDF-Average`	$\frac{1}{\lambda} = 0$	0.737 ± 0.000	0.822 ± 0.000	0.902 ± 0.000
`PB-L2`		$\mathbf{0.753 \pm 0.000}$	0.841 ± 0.000	0.905 ± 0.000
`PB-IDF-L2`		0.735 ± 0.002	0.823 ± 0.000	0.907 ± 0.000
`i-PB-L2`		0.748 ± 0.001	0.841 ± 0.000	0.908 ± 0.000
`i-PB-IDF-L2`		0.737 ± 0.001	0.823 ± 0.000	0.906 ± 0.000
`PB-neg`		0.750 ± 0.000	0.842 ± 0.000	0.907 ± 0.000
`w-PB-neg`		0.752 ± 0.000	0.843 ± 0.000	0.902 ± 0.001
`i-PB-neg`		0.748 ± 0.001	0.842 ± 0.000	$\mathbf{0.910 \pm 0.001}$
`i-w-PB-neg`		0.750 ± 0.001	$\mathbf{0.844 \pm 0.000}$	$\mathbf{0.910 \pm 0.001}$

For the squared L2 loss based models, `PB-L2`, `PB-IDF-L2`, `i-PB-L2`, and `i-PB-IDF-L2`, we trained the sentence vectors with $\lambda \in \{0.25, 0.5, 1, 2, 4, 8\}$ and fixed the prior's variance parameter $\sigma_{\mathcal{P}}^2 = \sigma_{\mathcal{P}_w}^2 = 1$. Hyperparameters λ was searched by grid-search in the supervised tasks.

For the zero-one loss based models, `PB-neg`, `w-PB-neg`, `i-PB-neg`, and `i-w-PB-neg`, we used the following parameters to train sentence vectors: the number of iterations was 40, the number of negative samples k was 15, noise samples were the uni-gram distribution of target task's training sentences powered by $\frac{3}{4}$, initial learning rate η was the same as Skip-gram, 0.025. We used stochastic gradient descent to optimize sentence vectors with mini-batch as a sentence with linearly decreased the learning rate per epoch. We fixed $\sigma_{\mathcal{P}}^2 = 1$ for `PB-neg` and `i-PB-neg`. Hyperparameter λ was searched in $\{0.25, 0.5, 1, 2, 4, 8\}$ by cross-validation of the classification phase. Estimated parameters of `PB-neg` and `i-PB-neg`, $\boldsymbol{\mu}_{\mathcal{S}}$ and $\sigma_{\mathcal{S}}^2$, were initialized by $U(-\frac{0.5}{300}, \frac{0.5}{300})$, and estimated parameters of `w-PB-neg` and `i-w-PB-neg` were initialized by prior's parameters. We implemented all algorithms with `PyTorch` [Paszke et al., 2017]. We optimized the loss functions by using reparameterization trick [Kingma and Welling, 2014] for multivariate Gaussian distribution to take the expectation of posterior $\mathcal{Q}$. As an implementation technique to accelerate training, we applied lazy stochastic gradient descent [Carpenter, 2008] to optimize Eq. (21). We directly updated parameters not using automatic differentiation because we needed to evaluate trainable word vectors for each update.

5.3 Evaluation of Sentence Representations

We evaluated sentence vectors trained on the target tasks as feature vectors of classification tasks. We implemented a one-vs-rest logistic regression classifier with `scikit-learn` [Pedregosa et al., 2011] and `GNU Parallel` Tange [2018]. We chose hyperparameter C of logistic regression and the pre-processing parameters as L2 normalization of sentence vectors on the basis of grid-search with five-fold cross-validation. All classification scores were values averaged over three times with different random seeds.

5.4 Classification Results

Table. 3 shows the test accuracies and their standard deviation of the sentence classification results. Test accuracies were slightly different among models. Zero-one loss based models performed more stably than squared L2 loss based models and heuristics did on 20news and IMDb datasets.

6 Conclusion

We formulated learning sentence vectors from neural word vector models as transfer learning. We derived heuristic sentence vector models by applying PAC-Bayes theory to target tasks. Our analysis of sentence vectors is a first step towards understanding of practical sentence vector representation learning. We also proposed novel sentence representation learning on the basis of our PAC-Bayes analysis by replacing the loss function and hypothesis class. In our experiments, the performance of all sentence vector models was almost the same in sentence classification.

Recently, sentence representations transferred by bi-directional recurrent neural networks language modeling have archived the state-of-the-art performance in natural language processing tasks [Howard and Ruder, 2018, Peters et al., 2017]. Analyzing these complex deep predictive representation models by PAC-Bayes theory is left to future work. We also would like to consider a technique of Skip-gram training, sub-sampling proposed by Mikolov et al. [2013a], to make the target task more similar to the source task.

Acknowledgements

References

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.

Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition. In *ICML*, pages 647–655, 2014.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *NeurIPS*, 2012.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR*, pages 248–255, 2009.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. Supervised Learning of Universal Sentence Representations from Natural Language Inference Data. In *EMNLP*, pages 670–680, 2017.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. A Large Annotated Corpus for Learning Natural Language Inference. In *EMNLP*, pages 632–642, 2015.

Simon Kornblith, Jonathon Shlens, and Quoc V. Le. Do Better ImageNet Models Transfer Better?, 2018. URL `https://arxiv.org/pdf/1805.08974.pdf`. arXiv:1805.08974v2 [cs.CV].

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural Language Processing (Almost) from Scratch. *Journal of Machine Learning Research*, 12:2493–2537, 2011.

Andriy Mnih and Koray Kavukcuoglu. Learning Word Embeddings Efficiently with Noise-contrastive Estimation. In *NeurIPS*, 2013.

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

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. GloVe: Global Vectors for Word Representation. In *EMNLP*, pages 1532–1543, 2014.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching Word Vectors with Subword Information. *TACL*, 5(1):135–146, 2017.

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. Advances in Pre-Training Distributed Word Representations. In *LREC*, pages 52–55, 2018.

Ye Qi, Devendra Singh Sachan, Matthieu Felix, Sarguna Janani Padmanabhan, and Graham Neubig. When and Why are Pre-trained Word Embeddings Useful for Neural Machine Translation? In *NAACL-HLT*, pages 529–535, 2018.

Matthew E. Peters, Mark Neumann, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep Contextualized Word Representations. In *NAACL-HLT*, pages 2227–2237, 2018a.

Matthew E. Peters, Mark Neumann, Luke Zettlemoyer, and Wen-tau Yih. Dissecting Contextual Word Embeddings: Architecture and Representation. In *EMNLP*, pages 1499–1509, 2018b.

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. Learning Word Vectors for 157 Languages. In *LREC*, pages 3483–3487, 2018.

Quoc Le and Tomas Mikolov. Distributed Representations of Sentences and Documents. In *ICML*, pages 1188–1196, 2014.

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

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

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

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

Dinghan Shen, Guoyin Wang, Wenlin Wang, Martin Renqiang Min, Qinliang Su, Yizhe Zhang, Chunyuan Li, Ricardo Henao, and Lawrence Carin. Baseline Needs More Love: On Simple Word-Embedding-Based Models and Associated Pooling Mechanisms. In *ACL*, pages 440–450, 2018.

John Wieting and Douwe Kiela. No Training Required: Exploring Random Encoders for Sentence Classification. In *ICLR*, 2019.

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.

Oren Melamud and Jacob Goldberger. Information-Theory Interpretation of the Skip-Gram Negative-Sampling Objective Function. In *ACL*, pages 167–171, 2017.

David A. McAllester. PAC-Bayesian Model Averaging. In *COLT*, pages 164–170, 1999.

Tomer Galanti, Lior Wolf, and Tamir Hazan. A Theoretical Framework for Deep Transfer Learning. *IMA*, 5(2):159–209, 2016.

Daniel McNamara and Maria-Florina Balcan. Risk Bounds for Transferring Representations With and Without Fine-Tuning. In *ICML*, pages 2373–2381, 2017.

Ron Amit and Ron Meir. Meta-Learning by Adjusting Priors based on Extended PAC-Bayes Theory. In *ICML*, pages 205–214, 2018.

Anastasia Pentina and Christoph H. Lampert. A PAC-Bayesian Bound for Lifelong Learning. In *ICML*, pages 991–999, 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*, pages 859–868, 2016a.

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

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

Olivier Catoni. *PAC-Bayesian Supervised Classification: The Thermodynamics of Statistical Learning*, volume 56 of *IMS Lecture Notes - Monograph Series*. 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.

Marco Baroni, Georgiana Dinu, and Germán Kruszewski. *Don't Count, Predict!* A Systematic Comparison of Context-counting vs. Context-predicting Semantic Vectors. In *ACL*, pages 238–247, 2014.

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

Gintare Karolina Dziugaite and Daniel M. Roy. Computing Nonvacuous Generalization Bounds for Deep (Stochastic) Neural Networks with Many More Parameters than Training Data. In *UAI*, 2017.

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

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.

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*, pages 331–339, 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.

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.

Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. In *ICLR*, 2014.

Bob Carpenter. Lazy Sparse Stochastic Gradient Descent for Regularized Mutlinomial Logistic Regression. Technical report, 2008.

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.

Ole Tange. *GNU Parallel 2018*. Tange, Ole, March 2018. ISBN 9781387509881. doi: 10.5281/zenodo.1146014. URL `https://doi.org/10.5281/zenodo.1146014`.

Jeremy Howard and Sebastian Ruder. Universal Language Model Fine-tuning for Text Classification. In *ACL*, pages 328–339, 2018.

Matthew E. Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. Semi-supervised Sequence Tagging with Bidirectional Language Models. In *ACL*, pages 1756–1765, 2017.

A   Proposed PAC-Bayes Bound and Concrete Loss Functions

A.1   Proof of Theorem 2

Proof. Given sentence $\mathcal{S}$ and bounded loss ℓ, $\mathcal{P}_{\mathcal{S}} = \frac{\prod_{w \in \mathcal{S}} \mathcal{N}(;\hat{\mathbf{o}}_w, \sigma_{\mathcal{P}}^2 I)}{\int \prod_{w \in \mathcal{S}} \mathcal{N}(\mathbf{h};\hat{\mathbf{o}}_w, \sigma_{\mathcal{P}}^2 I) \mathrm{d}\mathbf{h}}$, and $\forall \lambda > 0$ with probability at least $1 - \delta$ over training samples $\mathcal{D}_S$, minimization of Eq. (5) is equivalent to minimizing the loss function defined by

$$L(\mathcal{D}_S, \mathcal{Q}) = \hat{R}(\mathcal{Q}_{\mathcal{S}}) + \frac{1}{\lambda} \mathrm{KL}(\mathcal{Q}_{\mathcal{S}} || \mathcal{P}_{\mathcal{S}}). \tag{23}$$

We focus on $\mathrm{KL}(\mathcal{Q}_{\mathcal{S}} || \mathcal{P}_{\mathcal{S}})$ term in Eq. (23). Recall that posterior $\mathcal{Q}_{\mathcal{S}} = \mathcal{N}(; \boldsymbol{\mu}_{\mathcal{S}}, \sigma_{\mathcal{Q}}^2 I)$.

$$\mathrm{KL}(\mathcal{Q}_{\mathcal{S}} || \mathcal{P}_{\mathcal{S}}) = \int \mathcal{Q}_{\mathcal{S}} \log \frac{\mathcal{Q}_{\mathcal{S}}}{\mathcal{P}_{\mathcal{S}}} \mathrm{d}\mathbf{h} \tag{24}$$

$$= \int \mathcal{Q}_{\mathcal{S}} \log \frac{\mathcal{Q}_{\mathcal{S}}}{\left[\frac{\prod_{w \in \mathcal{S}} \mathcal{N}(;\hat{\mathbf{o}}_w, \sigma_{\mathcal{P}}^2 I)}{\int \prod_{w \in \mathcal{S}} \mathcal{N}(\mathbf{h};\hat{\mathbf{o}}_w, \sigma_{\mathcal{P}}^2 I) \mathrm{d}\mathbf{h}} \right]} \mathrm{d}\mathbf{h} \tag{25}$$

$$= \int \mathcal{Q}_{\mathcal{S}} \log \frac{\mathcal{Q}_{\mathcal{S}} Z}{\prod_{w \in \mathcal{S}} \mathcal{N}(;\hat{\mathbf{o}}_w, \sigma_{\mathcal{P}}^2 I)} \mathrm{d}\mathbf{h} \tag{26}$$

$$= \int \sum_{w \in \mathcal{S}} \mathcal{Q}_{\mathcal{S}} \log \frac{\mathcal{Q}_{\mathcal{S}} Z}{\mathcal{N}(;\hat{\mathbf{o}}_w, \sigma_{\mathcal{P}}^2 I)} \mathrm{d}\mathbf{h} \tag{27}$$

$$= \int \sum_{w \in \mathcal{S}} \mathcal{Q}_{\mathcal{S}} \left(\log \mathcal{Q}_{\mathcal{S}} + \log Z - \log \mathcal{N}(;\hat{\mathbf{o}}_w, \sigma_{\mathcal{P}}^2 I) \right) \mathrm{d}\mathbf{h} \tag{28}$$

$$= \int \sum_{w \in \mathcal{S}} \mathcal{Q}_{\mathcal{S}} \left(\log \frac{\mathcal{Q}_{\mathcal{S}}}{\mathcal{N}(;\hat{\mathbf{o}}_w, \sigma_{\mathcal{P}}^2 I)} + \log Z \right) \mathrm{d}\mathbf{h} \tag{29}$$

$$= \sum_{w \in \mathcal{S}} \int \mathcal{Q}_{\mathcal{S}} \left(\log \frac{\mathcal{Q}_{\mathcal{S}}}{\mathcal{N}(;\hat{\mathbf{o}}_w, \sigma_{\mathcal{P}}^2 I)} \right) \mathrm{d}\mathbf{h} + \sum_{w \in \mathcal{S}} \int \mathcal{Q}_{\mathcal{S}} \log Z \mathrm{d}\mathbf{h} \tag{30}$$

$$= \sum_{w \in \mathcal{S}} \int \mathcal{Q}_{\mathcal{S}} \left(\log \frac{\mathcal{Q}_{\mathcal{S}}}{\mathcal{N}(;\hat{\mathbf{o}}_w, \sigma_{\mathcal{P}}^2 I)} \right) \mathrm{d}\mathbf{h} + |\mathcal{S}| \log Z \tag{31}$$

$$= \sum_{w \in \mathcal{S}} \mathrm{KL} \left(\mathcal{Q}_{\mathcal{S}} || \mathcal{N}(;\hat{\mathbf{o}}_w, \sigma_{\mathcal{P}}^2 I) \right) + |\mathcal{S}| \log Z, \tag{32}$$

where $Z = \int \prod_{w \in \mathcal{S}} \mathcal{N}(\mathbf{h}; \hat{\mathbf{o}}_w, \sigma_\mathcal{P}^2 I) d\mathbf{h}$. The only first term in Eq. (32) contributes the loss function because $|\mathcal{S}| \log Z$ is constant when $\mathcal{S}$ is fixed. Therefore,

$$\sum_{w \in \mathcal{S}} \mathrm{KL}\left(\mathcal{Q}_\mathcal{S} || \mathcal{N}(; \hat{\mathbf{o}}_w, \sigma_\mathcal{P}^2 I)\right) = \frac{1}{2\sigma_\mathcal{P}^2} \sum_{w \in \mathcal{S}} \left(||\boldsymbol{\mu}_\mathcal{S}||_2^2 - 2\boldsymbol{\mu}_\mathcal{S}^\top \hat{\mathbf{o}}_w + ||\hat{\mathbf{o}}_w||_2^2 \right)$$
$$+ \frac{|\mathcal{S}|}{2} \left\{ \ln \frac{|\sigma_\mathcal{P}^2 I|}{|\sigma_\mathcal{Q}^2 I|} - d + \mathrm{Tr}\left[(\sigma_\mathcal{P}^2 I)^{-1} \sigma_\mathcal{Q}^2 I \right] \right\} \tag{33}$$

$$= \frac{1}{2\sigma_\mathcal{P}^2} \left(|\mathcal{S}| \times ||\boldsymbol{\mu}_\mathcal{S}||_2^2 - 2\boldsymbol{\mu}_\mathcal{S}^\top \sum_{w \in \mathcal{S}} \hat{\mathbf{o}}_w + \sum_{w \in \mathcal{S}} ||\hat{\mathbf{o}}_w||_2^2 \right)$$
$$+ \frac{|\mathcal{S}|}{2} \left(d \ln \frac{\sigma_\mathcal{P}^2}{\sigma_\mathcal{Q}^2} - d + d \frac{\sigma_\mathcal{Q}^2}{\sigma_\mathcal{P}^2} \right) \tag{34}$$

$$= \frac{|\mathcal{S}|}{2\sigma_\mathcal{P}^2} \left(||\boldsymbol{\mu}_\mathcal{S}||_2^2 - 2\boldsymbol{\mu}_\mathcal{S}^\top \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \hat{\mathbf{o}}_w \right) + \frac{1}{2\sigma_\mathcal{P}^2} \sum_{w \in \mathcal{S}} ||\hat{\mathbf{o}}_w||_2^2$$
$$+ \frac{|\mathcal{S}|d}{2} \left(\ln \frac{\sigma_\mathcal{P}^2}{\sigma_\mathcal{Q}^2} - 1 + \frac{\sigma_\mathcal{Q}^2}{\sigma_\mathcal{P}^2} \right) \tag{35}$$

$$= \frac{|\mathcal{S}|}{2\sigma_\mathcal{P}^2} \left|\left| \boldsymbol{\mu}_\mathcal{S} - \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \hat{\mathbf{o}}_w \right|\right|_2^2 + \frac{|\mathcal{S}|d}{2} \left(\ln \frac{\sigma_\mathcal{P}^2}{\sigma_\mathcal{Q}^2} + \frac{\sigma_\mathcal{Q}^2}{\sigma_\mathcal{P}^2} \right)$$
$$- \frac{|\mathcal{S}|d}{2} - \frac{1}{2\sigma_\mathcal{P}^2} \left(|\mathcal{S}| \left|\left| \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \hat{\mathbf{o}}_w \right|\right|_2^2 - \sum_{w \in \mathcal{S}} ||\hat{\mathbf{o}}_w||_2^2 \right). \tag{36}$$

Regarding minimization for the loss function by updating $\mathcal{Q}$, we can ignore the third and the firth terms in Eq. (36), so we replace them with C. $\qquad\square$

A.2 Details of Eq. (13) and Eq. (14)

Given bounded squared loss function $\ell(x, y, h) = \frac{1}{2} ||y - \mathbf{h}_\mathcal{S}||$, we minimize the upper bound based on Eq. (12) defined by

$$L(\mathcal{D}_S, \mathcal{Q}_\mathcal{S}) = \mathbb{E}_{h \sim \mathcal{Q}_\mathcal{S}} \frac{1}{N} \sum_{i=1}^N \left(\frac{1}{2} \left|\left| \hat{\mathbf{i}}_{w_i} - \mathbf{h}_\mathcal{S} \right|\right|_2^2 \right) + \frac{|\mathcal{S}|}{2\sigma_\mathcal{P}^2 \lambda} \left|\left| \boldsymbol{\mu}_\mathcal{S} - \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \hat{\mathbf{o}}_w \right|\right|_2^2 + \frac{|\mathcal{S}|d}{2\lambda} \left(\ln \frac{\sigma_\mathcal{P}^2}{\sigma_\mathcal{Q}^2} + \frac{\sigma_\mathcal{Q}^2}{\sigma_\mathcal{P}^2} \right) + C. \tag{37}$$

In our setting, $\mathcal{D}_S = \{(\mathbf{x}_w, \hat{\mathbf{h}}_w) \mid w \in \mathcal{S}\}$ in Eq. (37), and we use reparameterization trick for $\mathbf{h}_\mathcal{S}$ in the first term,

$$L(D_\mathcal{S}, \mathcal{Q}_\mathcal{S}) = \mathbb{E}_{h \sim \mathcal{Q}_\mathcal{S}} \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \left(\frac{1}{2} \left|\left| \hat{\mathbf{i}}_w - \mathbf{h}_\mathcal{S} \right|\right|_2^2 \right) + \frac{|\mathcal{S}|}{2\sigma_\mathcal{P}^2 \lambda} \left|\left| \boldsymbol{\mu}_\mathcal{S} - \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \hat{\mathbf{o}}_w \right|\right|_2^2 + \frac{|\mathcal{S}|d}{2\lambda} \left(\ln \frac{\sigma_\mathcal{P}^2}{\sigma_\mathcal{Q}^2} + \frac{\sigma_\mathcal{Q}^2}{\sigma_\mathcal{P}^2} \right) + C \tag{38}$$

$$= \mathbb{E}_{\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\epsilon}; \mathbf{0}, I)} \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \left(\frac{1}{2} \left|\left| \hat{\mathbf{i}}_w - \boldsymbol{\mu}_\mathcal{S} - \sqrt{\sigma_\mathcal{Q}^2} \boldsymbol{\epsilon} \right|\right|_2^2 \right)$$
$$+ \frac{|\mathcal{S}|}{2\sigma_\mathcal{P}^2 \lambda} \left|\left| \boldsymbol{\mu}_\mathcal{S} - \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \hat{\mathbf{o}}_w \right|\right|_2^2 + \frac{|\mathcal{S}|d}{2\lambda} \left(\ln \frac{\sigma_\mathcal{P}^2}{\sigma_\mathcal{Q}^2} + \frac{\sigma_\mathcal{Q}^2}{\sigma_\mathcal{P}^2} \right) + C. \tag{39}$$

We take derivative of Eq. (39) with respect to $\boldsymbol{\mu}_S$, and then set it to zero.

$$\nabla_{\boldsymbol{\mu}_S} L(\mathcal{D}_S, \mathcal{Q}_S) = 0 \tag{40}$$

$$\underset{\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\epsilon}; \mathbf{0}, I)}{\mathbb{E}} \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \left(\sqrt{\sigma_{\mathcal{Q}}^2} \boldsymbol{\epsilon} + \boldsymbol{\mu}_S - \hat{\mathbf{i}}_w \right) + \frac{|\mathcal{S}|}{\sigma_{\mathcal{P}}^2 \lambda} \left(\boldsymbol{\mu}_S - \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \hat{\mathbf{o}}_w \right) = 0 \tag{41}$$

$$\sqrt{\sigma_{\mathcal{Q}}^2} \underset{\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\epsilon}; \mathbf{0}, I)}{\mathbb{E}} \boldsymbol{\epsilon} + \sum_{w \in \mathcal{S}} \frac{1}{|\mathcal{S}|} \left(\boldsymbol{\mu}_S - \hat{\mathbf{i}}_w \right) + \frac{|\mathcal{S}|}{\sigma_{\mathcal{P}}^2 \lambda} \left(\boldsymbol{\mu}_S - \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \hat{\mathbf{o}}_w \right) = 0 \tag{42}$$

$$\boldsymbol{\mu}_S - \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \hat{\mathbf{i}}_w + \alpha \boldsymbol{\mu}_S - \frac{\alpha}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \hat{\mathbf{o}}_w = 0 \tag{43}$$

$$(1 + \alpha) \boldsymbol{\mu}_S - \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \hat{\mathbf{i}}_w - \frac{\alpha}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \hat{\mathbf{o}}_w = 0 \tag{44}$$

$$(1 + \alpha) \boldsymbol{\mu}_S - \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \left(\hat{\mathbf{i}}_w + \alpha \hat{\mathbf{o}}_w \right) = 0 \tag{45}$$

$$(1 + \alpha) \boldsymbol{\mu}_S = \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \left(\hat{\mathbf{i}}_w + \alpha \hat{\mathbf{o}}_w \right) \tag{46}$$

$$\boldsymbol{\mu}_S = \frac{1}{(1 + \alpha)|\mathcal{S}|} \sum_{w \in \mathcal{S}} \left(\hat{\mathbf{i}}_w + \alpha \hat{\mathbf{o}}_w \right), \tag{47}$$

where $\alpha = \frac{|\mathcal{S}|}{\sigma_{\mathcal{P}}^2 \lambda}$.

We also take derivative of Eq. (39) with respect to $\sigma_{\mathcal{Q}}^2$, and set it to zero.

$$\nabla_{\sigma_{\mathcal{Q}}^2} L(\mathcal{D}_S, \mathcal{Q}_S) = 0 \tag{48}$$

$$\sqrt{\sigma_{\mathcal{Q}}^2} \underset{\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\epsilon}; \mathbf{0}, I)}{\mathbb{E}} \boldsymbol{\epsilon} + \frac{|\mathcal{S}| d}{2\lambda} \left(-\frac{1}{\sigma_{\mathcal{Q}}^2} + \frac{1}{\sigma_{\mathcal{P}}^2} \right) = 0 \tag{49}$$

$$-\frac{1}{\sigma_{\mathcal{Q}}^2} + \frac{1}{\sigma_{\mathcal{P}}^2} = 0 \tag{50}$$

$$\sigma_{\mathcal{Q}}^2 = \sigma_{\mathcal{P}}^2. \tag{51}$$

16

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

We start from Eq. (32). Each variance $\sigma^2_{\mathcal{P}_w}$ of priors depends on each word w.

$$\text{KL}(\mathcal{Q}_\mathcal{S}||\mathcal{P}_\mathcal{S}) = \sum_{w\in\mathcal{S}} \text{KL}\left(\mathcal{Q}_\mathcal{S}||\mathcal{N}(;\hat{\mathbf{o}}_w, \sigma^2_{\mathcal{P}_w}I)\right) \tag{52}$$

$$= \sum_{w\in\mathcal{S}}\left[\frac{1}{2\sigma^2_{\mathcal{P}_w}}\left(||\mathbf{h}_\mathcal{S}||_2^2 - 2\mathbf{h}_\mathcal{S}^\top\hat{\mathbf{o}}_w + ||\hat{\mathbf{o}}_w||_2^2\right) + \frac{d}{2}\left(\ln\frac{\sigma^2_{\mathcal{P}_w}}{\sigma^2_\mathcal{Q}} - 1 + \frac{\sigma^2_\mathcal{Q}}{\sigma^2_{\mathcal{P}_w}}\right)\right] \tag{53}$$

$$= ||\mathbf{h}_\mathcal{S}||_2^2\sum_{w\in\mathcal{S}}\frac{1}{2\sigma^2_{\mathcal{P}_w}} - 2\mathbf{h}_\mathcal{S}^\top\sum_{w\in\mathcal{S}}\frac{1}{2\sigma^2_{\mathcal{P}_w}}\hat{\mathbf{o}}_w + \sum_{w\in\mathcal{S}}\frac{1}{2\sigma^2_{\mathcal{P}_w}}||\hat{\mathbf{o}}_w||_2^2 + \sum_{w\in\mathcal{S}}\frac{d}{2}\left(\ln\frac{\sigma^2_{\mathcal{P}_w}}{\sigma^2_\mathcal{Q}} - 1 + \frac{\sigma^2_\mathcal{Q}}{\sigma^2_{\mathcal{P}_w}}\right) \tag{54}$$

$$= \eta||\mathbf{h}_\mathcal{S}||_2^2 - 2\mathbf{h}_\mathcal{S}^\top\sum_{w\in\mathcal{S}}\frac{1}{2\sigma^2_{\mathcal{P}_w}}\hat{\mathbf{o}}_w + \sum_{w\in\mathcal{S}}\frac{d}{2}\left(-\ln\sigma^2_\mathcal{Q} + \frac{\sigma^2_\mathcal{Q}}{\sigma^2_{\mathcal{P}_w}}\right) - |\mathcal{S}|\frac{d}{2} \tag{55}$$

$$= \eta\left(||\mathbf{h}_\mathcal{S}||_2^2 - \frac{2}{\eta}\mathbf{h}_\mathcal{S}^\top\sum_{w\in\mathcal{S}}\frac{1}{2\sigma^2_{\mathcal{P}_w}}\hat{\mathbf{o}}_w\right) + \sum_{w\in\mathcal{S}}\frac{d}{2}\left(-\ln\sigma^2_\mathcal{Q} + \frac{\sigma^2_\mathcal{Q}}{\sigma^2_{\mathcal{P}_w}}\right) - |\mathcal{S}|\frac{d}{2} \tag{56}$$

$$= \eta\left\|\mathbf{h}_\mathcal{S} - \frac{1}{\eta}\sum_{w\in\mathcal{S}}\frac{1}{2\sigma^2_{\mathcal{P}_w}}\hat{\mathbf{o}}_w\right\|_2^2 + \sum_{w\in\mathcal{S}}\frac{d}{2}\left(-\ln\sigma^2_\mathcal{Q} + \frac{\sigma^2_\mathcal{Q}}{\sigma^2_{\mathcal{P}_w}}\right) - \left(|\mathcal{S}|\frac{d}{2} + \eta\left\|\frac{1}{\eta}\sum_{w\in\mathcal{S}}\frac{1}{2\sigma^2_{\mathcal{P}_w}}\hat{\mathbf{o}}_w\right\|_2^2\right), \tag{57}$$

where $\eta = \sum_{w\in\mathcal{S}}\frac{1}{2\sigma^2_{\mathcal{P}_w}}$. The last term can be ignored because it does not depend on posterior parameters $\boldsymbol{\mu}_\mathcal{S}$ and $\sigma^2_\mathcal{Q}$, so we replace it with C.

The weighted loss function is defined by $\ell(\mathbf{x}_i, y_i, h) = \frac{\beta(\mathbf{x}_i)}{2}||\hat{\mathbf{i}}_{w_i} - \mathbf{h}_\mathcal{S}||_2^2$, where β is a weighing function from $\mathbf{x}$ to $\mathbb{R}_+$. We abbreviate $\beta(\mathbf{x}_w)$ to β_w. We follow the same way as appendix A.2.

$$L(\mathcal{D}_S, \mathcal{Q}_\mathcal{S}) = \mathop{\mathbb{E}}_{h\sim\mathcal{Q}_\mathcal{S}}\frac{1}{N}\sum_{i=1}^N\ell(\mathbf{x}_i, y_i, h) + \frac{1}{\lambda}\text{KL}(\mathcal{Q}_\mathcal{S}||\mathcal{P}_\mathcal{S}) \tag{58}$$

$$= \mathop{\mathbb{E}}_{h\sim\mathcal{Q}_\mathcal{S}}\frac{1}{|\mathcal{S}|}\sum_{w\in\mathcal{S}}\left(\frac{\beta_w}{2}\left\|\hat{\mathbf{i}}_w - \mathbf{h}_\mathcal{S}\right\|_2^2\right)$$

$$+ \frac{1}{\lambda}\eta\left\|\boldsymbol{\mu}_\mathcal{S} - \frac{1}{\eta}\sum_{w\in\mathcal{S}}\frac{1}{2\sigma^2_{\mathcal{P}_w}}\hat{\mathbf{o}}_w\right\|_2^2 + \sum_{w\in\mathcal{S}}\frac{d}{2\lambda}\left(-\ln\sigma^2_\mathcal{Q} + \frac{\sigma^2_\mathcal{Q}}{\sigma^2_{\mathcal{P}_w}}\right) + C \tag{59}$$

$$= \mathop{\mathbb{E}}_{\boldsymbol{\epsilon}\sim\mathcal{N}(\boldsymbol{\epsilon};\mathbf{0},I)}\frac{1}{|\mathcal{S}|}\sum_{w\in\mathcal{S}}\left(\frac{\beta_w}{2}\left\|\hat{\mathbf{i}}_w - \boldsymbol{\mu}_\mathcal{S} - \sqrt{\sigma^2_\mathcal{Q}}\boldsymbol{\epsilon}\right\|_2^2\right)$$

$$+ \frac{1}{\lambda}\eta\left\|\boldsymbol{\mu}_\mathcal{S} - \frac{1}{\eta}\sum_{w\in\mathcal{S}}\frac{1}{2\sigma^2_{\mathcal{P}_w}}\hat{\mathbf{o}}_w\right\|_2^2 + \sum_{w\in\mathcal{S}}\frac{d}{2\lambda}\left(-\ln\sigma^2_\mathcal{Q} + \frac{\sigma^2_\mathcal{Q}}{\sigma^2_{\mathcal{P}_w}}\right) + C. \tag{60}$$

We take derivative of Eq. (60) with respect to $\boldsymbol{\mu}_{\mathcal{S}}$, and then set it to zero.

$$\nabla_{\boldsymbol{\mu}_{\mathcal{S}}} L(\mathcal{D}_{\mathcal{S}}, \mathcal{Q}_{\mathcal{S}}) = 0 \tag{61}$$

$$\mathop{\mathbb{E}}_{\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\epsilon}; \mathbf{0}, I)} \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \beta_w \left(\sqrt{\sigma_{\mathcal{Q}}^2} \boldsymbol{\epsilon} + \boldsymbol{\mu}_{\mathcal{S}} - \hat{\mathbf{i}}_w \right) + 2 \frac{1}{\lambda} \eta \left(\boldsymbol{\mu}_{\mathcal{S}} - \frac{1}{\eta} \sum_{w \in \mathcal{S}} \frac{1}{2 \sigma_{\mathcal{P}_w}^2} \hat{\mathbf{o}}_w \right) = 0 \tag{62}$$

$$\frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \beta_w \boldsymbol{\mu}_{\mathcal{S}} - \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \beta_w \hat{\mathbf{i}}_w + 2 \frac{1}{\lambda} \eta \boldsymbol{\mu}_{\mathcal{S}} - 2 \frac{1}{\lambda} \sum_{w \in \mathcal{S}} \frac{1}{2 \sigma_{\mathcal{P}_w}^2} \hat{\mathbf{o}}_w = 0 \tag{63}$$

$$\left(\frac{\sum_{w \in \mathcal{S}} \beta_w}{|\mathcal{S}|} + 2 \frac{1}{\lambda} \eta \right) \boldsymbol{\mu}_{\mathcal{S}} - \frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \beta_w \hat{\mathbf{i}}_w - 2 \frac{1}{\lambda} \sum_{w \in \mathcal{S}} \frac{1}{2 \sigma_{\mathcal{P}_w}^2} \hat{\mathbf{o}}_w = 0 \tag{64}$$

$$\left(\sum_{w \in \mathcal{S}} \beta_w + 2 \eta \frac{1}{\lambda} |\mathcal{S}| \right) \boldsymbol{\mu}_{\mathcal{S}} - \sum_{w \in \mathcal{S}} \left(\beta_w \hat{\mathbf{i}}_w + \frac{1}{\lambda} |\mathcal{S}| \frac{1}{\sigma_{\mathcal{P}_w}^2} \hat{\mathbf{o}}_w \right) = 0 \tag{65}$$

$$\left(\sum_{w \in \mathcal{S}} \beta_w + 2 \eta \frac{1}{\lambda} |\mathcal{S}| \right) \boldsymbol{\mu}_{\mathcal{S}} = \sum_{w \in \mathcal{S}} \left(\beta_w \hat{\mathbf{i}}_w + \frac{1}{\lambda} |\mathcal{S}| \frac{1}{\sigma_{\mathcal{P}_w}^2} \hat{\mathbf{o}}_w \right) \tag{66}$$

$$\boldsymbol{\mu}_{\mathcal{S}} = \frac{\sum_{w \in \mathcal{S}} \beta_w \hat{\mathbf{i}}_w + \frac{1}{\lambda} |\mathcal{S}| \frac{1}{\sigma_{\mathcal{P}_w}^2} \hat{\mathbf{o}}_w}{\sum_{w \in \mathcal{S}} \beta_w + 2 \eta \frac{1}{\lambda} |\mathcal{S}|}. \tag{67}$$

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

$$\boldsymbol{\mu}_{\mathcal{S}} = \frac{\sum_{w \in \mathcal{S}} \beta_w \hat{\mathbf{i}}_w + \frac{1}{\lambda} |\mathcal{S}| \text{IDF}(w) \hat{\mathbf{o}}_w}{\sum_{w \in \mathcal{S}} \beta_w + \frac{1}{\lambda} |\mathcal{S}| \sum_{w \in \mathcal{S}} \text{IDF}(w)} \tag{68}$$

$$= \frac{\sum_{w \in \mathcal{S}} \beta_w \hat{\mathbf{i}}_w + \frac{1}{\lambda} |\mathcal{S}| \text{IDF}(w) \hat{\mathbf{o}}_w}{\sum_{w \in \mathcal{S}} \beta_w + \frac{1}{\lambda} |\mathcal{S}| \text{IDF}(w)}. \tag{69}$$

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

$$\boldsymbol{\mu}_{\mathcal{S}} = \frac{\sum_{w \in \mathcal{S}} |\mathcal{S}| \text{IDF}(w) \hat{\mathbf{i}}_w + \frac{1}{\lambda} |\mathcal{S}| \text{IDF}(w) \hat{\mathbf{o}}_w}{\sum_{w \in \mathcal{S}} |\mathcal{S}| \text{IDF}(w) + \frac{1}{\lambda} |\mathcal{S}| \text{IDF}(w)} \tag{70}$$

$$= \frac{\sum_{w \in \mathcal{S}} \text{IDF}(w) \hat{\mathbf{i}}_w + \frac{1}{\lambda} \text{IDF}(w) \hat{\mathbf{o}}_w}{\sum_{w \in \mathcal{S}} \text{IDF}(w) + \frac{1}{\lambda} \text{IDF}(w)} \tag{71}$$

$$= \frac{\sum_{w \in \mathcal{S}} \text{IDF}(w) \left(\hat{\mathbf{i}}_w + \frac{1}{\lambda} \hat{\mathbf{o}}_w \right)}{\left(1 + \frac{1}{\lambda} \right) \sum_{w \in \mathcal{S}} \text{IDF}(w)}. \tag{72}$$

We also take derivative of Eq. (60) with respect to $\sigma_{\mathcal{Q}}^2$, and then set it to zero.

$$\nabla_{\sigma_{\mathcal{Q}}^2} L(\mathcal{D}_{\mathcal{S}}, \mathcal{Q}_{\mathcal{S}}) = 0 \tag{73}$$

$$\sqrt{\sigma_{\mathcal{Q}}^2} \mathop{\mathbb{E}}_{\boldsymbol{\epsilon} \sim \mathcal{N}(; \mathbf{0}, I)} \boldsymbol{\epsilon} + \sum_{w \in \mathcal{S}} \frac{d}{2\lambda} \left(-\frac{1}{\sigma_{\mathcal{Q}}^2} + \frac{1}{\sigma_{\mathcal{P}_w}^2} \right) = 0 \tag{74}$$

$$- \sum_{w \in \mathcal{S}} \left(\frac{1}{\sigma_{\mathcal{Q}}^2} - \frac{1}{\sigma_{\mathcal{P}_w}^2} \right) = 0 \tag{75}$$

$$\sigma_{\mathcal{Q}}^2 = \left(\frac{1}{|\mathcal{S}|} \sum_{w \in \mathcal{S}} \frac{1}{\sigma_{\mathcal{P}_w}^2} \right)^{-1}. \tag{76}$$