How Large a Vocabulary Does Text Classification Need? A Variational Approach to Vocabulary Selection

Wenhu Chen[†], Yu Su[¶], Yilin Shen[‡], Zhiyu Chen[†], Xifeng Yan[†], William Wang[†]
University of California, Santa Barbara[†]
Ohio State University, Columbus[¶]
Samsung Research, Moutain View[‡]
{wenhuchen,zhiyuchen,xyan,william}@cs.ucsb.edu su.809@osu.edu yilin.shen@samsung.com

Abstract

With the rapid development in deep learning, deep neural networks have been widely adopted in many real-life natural language applications. Under deep neural networks, a pre-defined vocabulary is required to vectorize text inputs. The canonical approach to select pre-defined vocabulary is based on the word frequency, where a threshold is selected to cut off the long tail distribution. However, we observed that such a simple approach could easily lead to under-sized vocabulary or over-sized vocabulary issues. Therefore, we are interested in understanding how the end-task classification accuracy is related to the vocabulary size and what is the minimum required vocabulary size to achieve a specific performance. In this paper, we provide a more sophisticated variational vocabulary dropout (VVD) based on variational dropout to perform vocabulary selection, which can intelligently select the subset of the vocabulary to achieve the required performance. To evaluate different algorithms on the newly proposed vocabulary selection problem, we propose two new metrics: Area Under Accuracy-Vocab Curve and Vocab Size under X% Accuracy Drop. Through extensive experiments on various NLP classification tasks, our variational framework is shown to significantly outperform the frequency-based and other selection baselines on these metrics.

1 Introduction

Over the past decade, deep neural networks have become arguably the most popular model choice for a vast number of natural language processing (NLP) tasks and have constantly been delivering state-of-the-art results. Because neural network models assume continuous data, to apply a neural network on any text data, the first step is to vectorize the discrete text input with a word embedding matrix through look-up operation, which in turn assumes a pre-defined vocabulary set. For many NLP tasks, the vocabulary size can easily go up to the order of tens of thousands, which potentially makes the word embedding the largest portion of the trainable parameters. For example, a document classification task like AG-news (Zhang et al., 2015) can include up to 60K unique words, with the embedding matrix accounting for 97.6% of the trainable parameters (Table 1), which leads to under-representation of the neural networks' own parameters.

Cutoff Freq.	Vocab	Remain Vocab	#Emb	#CNN	#Emb Ratio
1	60K	100%	15M	0.36M	97.6%
5	40K	21.7%	10M	0.36M	95.6%
10	24K	13%	6M	0.36M	94.3%
20	14K	9.4%	3.5M	0.36M	90%
100	4K	2.7%	1M	0.36M	73%

Table 1: Illustration of the frequency-based vocabulary selection heuristic on a typical CNN-based document classification model (Section 4.1). #Emb is the number of parameters in the word embedding matrix (256 dimensions), and #CNN is that in the CNN model.

Intuitively, using the full or very large vocabulary are neither economical, as it limits model applicability on computation- or memory-constrained scenarios (Yogatama et al., 2015; Faruqui et al., 2015), nor necessary, as many words may contribute little to the end task and could have been safely removed from the vocabulary. Therefore, how to select the best vocabulary is a problem of both theoretical and practical interests. Somewhat surprisingly, this *vocabulary selection* problem is largely under-addressed in the literature: The *de facto* standard practice is to do

frequency-based cutoff (Luong et al., 2015; Kim, 2014), and only retain the words more frequent than a certain threshold (Table 1). Although this simple heuristic has demonstrated strong empirical performance, its task-agnostic nature implies that likely it is not the optimal strategy for many tasks (or any task). Task-aware vocabulary selection strategies and a systematic comparison of different strategies are still lacking.

In this work, we present the first systematic study of the vocabulary selection problem. Our study will be based on text classification tasks, a broad family of NLP tasks including document classification (DC), natural language inference (NLI), natural language understanding in dialog systems (NLU), etc. Specifically, we aim to answer the following questions:

1. *How important a role does the vocabulary selection algorithm play in text classification?*
2. *How to dramatically reduce the vocabulary size while retaining the accuracy?*

The rest of the paper is organized as follows: We first formally define the vocabulary selection problem (subsection 2.1) and present a quantitative study on classification accuracy with different vocabulary selections to showcase its importance in the end task (subsection 2.2). We also propose two new metrics for evaluating the performance of vocabulary selection in text classification tasks (subsection 2.3). We then propose a novel, task-aware vocabulary selection algorithm called *Varitional Vocabulary Dropout (VVD)* (section 3) which draws on the idea of variational dropout (Kingma et al., 2015): If we learn a dropout probability p_w for each given word w in the vocabulary $\mathbb{V}$ during the model training on a given task, the learned dropout probabilities p_w will imply the importance of word w to the end task and can, therefore, be leveraged for vocabulary selection. We propose to infer the latent dropout probabilities under a Bayesian inference framework. During test time, we select the sub vocabulary $\hat{\mathbb{V}}$ by only retaining words with dropout probability lower than a certain threshold. For any words deselected using VVD, we will simply regard them as a special token with null vector representation $[0, 0, \cdots, 0]$. Please note that our proposed algorithm needs to re-train a word embedding matrix, thus it is tangential to the research of pre-trained word embedding like Word2Vec (Mikolov et al., 2013) or Glove (Pennington et al., 2014) though we can use them to initialize our embedding.

We conduct comprehensive experiments to evaluate the performance of VVD (section 4) on different end classification tasks. Specifically, we compare against an array of strong baseline selection algorithms, including the frequency-based algorithm (Luong et al., 2015), TF-IDF algorithm (Ramos et al., 2003), and structure lasso algorithm (Friedman et al., 2010), and demonstrate that it can consistently outperform these competing algorithms by a remarkable margin. To show that the conclusions are widely held, our evaluation is based on a wide range of text classification tasks and datasets with different neural networks including Convolutional Neural Network (CNN) (Kim, 2014), Bi-directional Long-Short Term Memory (BiLSTM) (Bahdanau et al., 2014) and Enhanced LSTM (ESIM) (Chen et al., 2017). In summary, our contributions are three-fold:

1. We formally define the vocabulary selection problem, demonstrate its importance, and propose new evaluation metrics for vocabulary selection in text classification tasks.
2. We propose a novel vocabulary selection algorithm based on variational dropout by re-formulating text classification under the Bayesian inference framework. The code will be released in Github[1].
3. We conduct comprehensive experiments to demonstrate the superiority of the proposed vocabulary selection algorithm over a number of strong baselines.

2 Vocabulary Selection

2.1 Problem Definition

We now formally define the problem setting and introduce the notations for our problem. Conventionally, we assume the neural classification model vectorizes the discrete language input into a vector representation via an embedding matrix $W \in \mathbb{R}^{V*D}$, where V denotes the size of the vocabulary, and D denotes the vector dimension. The embedding is associated with a pre-defined word-to-index dictionary $\mathbb{V} = \{w_i : i | 1 \leq i \leq V\}$ where w_i denotes a literal word corresponding to i_{th} row in the embedding matrix. The embedding matrix W covers the subset of a vocabulary of interests for a particular NLP task, note that the value of V

[1] https://github.com/wenhuchen/Variational-Vocabulary-Selection.git

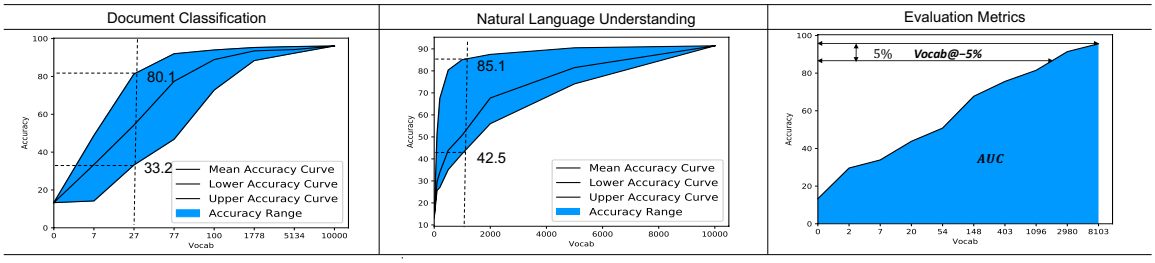

Figure 1: Monte-Carlo simulation on vocabulary selection. Left: CNN-based document classification on AG-news dataset. Middle: Natural language understanding on Snips dataset. Right: Metrics for vocabulary selection.

is known to be very large due to the rich variations in human languages. Here we showcase the embedding matrix size of a popular text classification model[2] on AG-news dataset (Zhang et al., 2015) in Table 1. From which we can easily observe that the embedding matrix is commonly occupying most of the parameter capacity, which could be the bottleneck in many real-world applications with limited computation resources.

In order to alleviate such redundancy problem and make embedding matrix as efficient as possible, we are particularly interested in discovering the minimum row-sized embedding $\hat{W}$ to achieve nearly promising performance as using the full row-sized embedding W. More formally, we define the our problem as follows:

$$\begin{aligned} &\underset{\hat{W},\hat{\theta}}{\operatorname{argmin}} \#Row(\hat{W}) \\ s.t. \quad &Acc(f_{\hat{\theta}}(x;\hat{W}),y) - Acc(f_{\theta}(x;W),y) \leq \epsilon \end{aligned} \tag{1}$$

where #Row is a the number of rows in the matrix $\hat{W}$, f_θ is the learned neural model with parameter θ to predict the class given the inputs x, Acc is the function which measure accuracy between model prediction and y (reference output), and ϵ is the tolerable performance drop after vocabulary selection. It is worth noting that here θ includes all the parameter set of the neural network except embedding matrix W. For each vocabulary selection algorithm $\mathcal{A}$, we propose to draw its characteristic curve $Acc(f_{\hat{\theta}}(x;\hat{W}),y) = g_{\mathcal{A}}(\#Row(\hat{W}))$ to understand the relationship between the vocabulary capacity and classification accuracy, which we call as (characteristic) accuracy-vocab curve throughout our paper.

2.2 Importance of Vocabulary Selection

In order to investigate the importance of the role played by the vocabulary selection algorithm, we design a Monte-Carlo simulation strategy to approximate accuracy's lower bound and upper bound of a given vocabulary size reached by a possible selection algorithm $\mathcal{A}$. More specifically, for a given vocabulary size of $\hat{V}$, there exist $\binom{V}{\hat{V}}$ algorithms which can select distinct vocabulary subset $\hat{\mathbb{V}}$ from the full vocabulary $\mathbb{V}$. Directly enumerating these possibilities are impossible, we instead propose to use a Monte-Carlo vocabulary selection strategy which can randomly pick vocabulary subset $\hat{\mathbb{V}}$ to simulate the possible selection algorithms by running it N times. After simulation, we obtain various point estimations $(Acc_1, \cdots, Acc_N|\hat{V})$ at each given $\hat{V}$ and depict the point estimates in Figure 1 to approximately visualize the upper and lower bound of the accuracy-vocab curve. From Figure 1, we can easily observe that the accuracy range under a limited-vocabulary is extremely large, when the budget $\hat{V}$ increases, the gap gradually shrinks. For example, for document classification with a budget of 1000, a selection algorithm $\mathcal{A}$ can yield a potential accuracy ranging from 42.5 to 85.1, while for natural language understanding task with a budget of 27, a selection algorithm $\mathcal{A}$ can yield a potential accuracy ranging from 33.2 to 80.1. Such a Monte-Carlo simulation study has demonstrated the significance of vocabulary selection strategy in NLP tasks and also implicate the enormous potential of an optimal vocabulary selection algorithm.

2.3 Evaluation Metrics

In order to evaluate how well a given selection algorithm $\mathcal{A}$ performs, we propose evaluation metrics as depicted in Figure 1 by quantitatively studying its characteristic accuracy-vocab curve. These metrics namely Area Under Curve (AUC) and Vocab@-X% separately measure the vocabulary selection performance globally and locally.

[2] https://github.com/dennybritz/cnn-text-classification-tf

Specifically, AUC computes enclosed area by the curve, which gives an overview of how well the vocabulary selection algorithm performs. In comparison, Vocab@-X% computes the minimum vocabulary size required if X% performance drop is allowed, which straightforwardly represents how large vocabulary is required to achieve a given accuracy. For the local evaluation metric, we mainly consider Vocab@-3% and Vocab@-5%. However, we observe that directly computing AUC lays too much emphasis on the large-vocabulary region, thus unable to represent an algorithm's selection capability under the low-vocabulary conditions. Therefore, we propose to take the logarithm of the vocabulary size and then compute the normalized enclosed area by:

$$AUC = \frac{\int_{\hat{V}} Acc(\log(\hat{V}))d\log(\hat{V})}{\int_{\hat{V}} Acc(V)d\log(\hat{V})} \quad (2)$$

It is worth noting that Vocab@-X% takes value from range $[0, V]$ with smaller values indicate better performance. Since AUC is normalized by Acc(V), it takes value from range $[0, 1]$ regardless of the classification error.

3 Our Method

Inspired by DNN dropout (Srivastava et al., 2014; Wang and Manning, 2013), we propose to tackle the vocabulary selection problem from word-level dropout perspective, where we assume each word w_i (an integer index) is associated with its characteristic dropout rate p_i, which represents the probability of being replaced with an empty placeholder, specifically, higher dropout probability indicates less loss suffered from removing it from the vocabulary. Hence, the original optimization problem in Equation 1 can be thought of as inferring the latent dropout probability vector $\mathbf{p} = [p_1, \cdots, p_V]$. The overview of our philosophy is depicted in Figure 2, where we associate with each row of the embedding matrix a dropout probability and then re-train the complete system, which grasps how much contribution each word from the vocabulary makes to the end NLP task and remove those "less contributory" words from the vocabulary without hurting the performance.

3.1 Bernouli Dropout

Here we first assume that the neural network vectorizes the discrete inputs with an embedding matrix W to project given words x into vector space

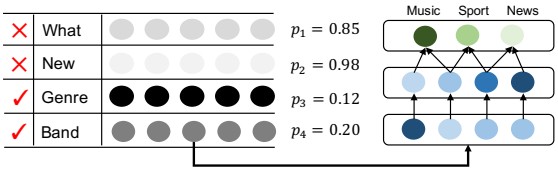

Figure 2: Variational dropout in classification models, "New" and "What" can be safely removed without harming performance due to large dropout probability.

$\mathbb{R}^D$, and then propose to add random dropout noise into the embedding input to simulate the dropout process as follows:

$$E(x|\mathbf{b}) = (\mathbf{b} \odot \mathit{OneHot}(x)) \cdot W \quad (3)$$

where *OneHot* is a function to transform a word x into its one-hot form $\mathit{OneHot}(x) \in \mathbb{R}^V$, and $\mathbf{b} \in \mathbb{R}^V$ is the Bernouli dropout noise with $b_i \sim Bern(1 - p_i)$. The embedding output vector $E(x|\mathbf{b})$ is computed with a given embedding matrix W under a sampled Bernouli vector $\mathbf{b}$. In order to infer the latent Bernouli distribution with parameters $\mathbf{p}$ under the Bayesian framework where training pairs $(\mathbf{x} = x_1 \cdots x_n, y)$ are given as the evidence, we first define an objective function as $\mathcal{L}(f_\theta(\mathbf{x}), y)$ and then derive its lower bound as follows (with $\bar{\mathbf{p}} = 1 - \mathbf{p}$):

$$\begin{aligned}
&\log \mathcal{L}(f_\theta(\mathbf{x}), y) = \log \int_{\mathbf{b}} \mathcal{L}(f_\theta(E(\mathbf{x}|\mathbf{b})), y)\mathcal{P}(\mathbf{b})d\mathbf{b} \\
&\geq \mathop{\mathbb{E}}_{\mathbf{b}\sim Bern(\bar{\mathbf{p}})}[\log \mathcal{L}(f_\theta(E(\mathbf{x}|\mathbf{b})), y)] - KL(Bern(\bar{\mathbf{p}})||\mathcal{P}(\mathbf{b})) \\
&= \mathcal{L}(W; \theta)
\end{aligned}$$

where $\mathcal{P}(\mathbf{b})$ is the prior distribution, and $Bern(\bar{\mathbf{p}})$ denotes the Bernouli approximate posterior with parameter $\mathbf{p}$. Here we use $E(\mathbf{x})$ as the simplied form of $\{E(x_1), \cdots, E(x_n)\}$, we separate the text classification model's parameters θ with the embedding parameters W and assume the classification model f_θ directly takes embedding E as input.

3.2 Gaussian Relaxation

However, the Bernouli distribution is hard to reparameterize, where we need to enumerate 2^V different values to compute the expectation over the stochastic dropout vector $\mathbf{b}$. Therefore, we follow Wang and Manning (2013) to use a continuous Gaussian approximation, where the Bernouli noise $\mathbf{b}$ is replaced by a Gaussian noise $\mathbf{z}$:

$$E(x|z) = (\mathbf{z} \odot \mathit{OneHot}(x)) \cdot W \quad (4)$$

where $\mathbf{z} \in \mathbb{R}^V$ follows Gaussian distribution $z_i \sim \mathcal{N}(1, \alpha_i = \frac{p_i}{1-p_i})$. It is worth noting that

α and p are one-to-one corresponded, and α is a monotonously increasing function of p. For more details, please refer to Wang and Manning (2013). Based on such approximation, we can use α as dropout criteria, e.g. throw away words with α above a certain given threshold α_T. We further follow Louizos et al. (2017); Kingma et al. (2015); Molchanov et al. (2017) to re-interpret the input noise as the intrinsic stochasticity in the embedding weights B itself as follows:

$$E(x|z) = \textit{OneHot}(x) \cdot B \tag{5}$$

where $B \in \mathbb{R}^{V*D}$ follows a multi-variate Gaussian distribution $B_{ij} \sim \mathcal{N}(\mu_{ij} = W_{ij}, \sigma_{ij}^2 = \alpha_i W_{ij}^2)$, where the random weights in each row has a tied variance/mean ratio α_i. Thus, we rewrite the evidence lower bound as follows:

$$\begin{aligned}
&\log \mathcal{L}(f_\theta(\mathbf{x}), y)) = \log \int_B \mathcal{L}(f_\theta(E(\mathbf{x}|z)), y))\mathcal{P}(B)dB \\
&\geq \mathop{\mathbb{E}}_{B\sim\mathcal{N}(\mu,\sigma)}[\log \mathcal{L}(f_\theta(E(\mathbf{x}|z)), y)] - KL(\mathcal{N}(\mu, \sigma)||\mathcal{P}(B)) \\
&= \mathcal{L}(B, \theta)
\end{aligned}$$

where $\mathcal{P}(B)$ is the prior distribution and $\mathcal{N}(\mu, \sigma)$ denotes the Gaussian approximate posterior with parameters μ and σ. $\mathcal{L}(B, \theta)$ is used as the relaxed evidence lower bound of marginal log likelihood $\log \mathcal{L}(f_\theta(\mathbf{x}), y))$. Here, we follow Kingma et al. (2015); Louizos et al. (2017) to choose the prior distribution $\mathcal{P}(B)$ as the "improper log-scaled uniform distribution" to guarantee that the regularization term $D_{KL}(\mathcal{N}(\mu, \sigma)||\mathcal{P}(B))$ only depends on dropout ratio α, i.e. irrelevant to μ. Formally, we write the prior distribution as follows:

$$\mathcal{P}(\log |B_{ij}|) = const \rightarrow \mathcal{P}(|B_{ij}|) \propto \frac{1}{|B_{ij}|} \tag{6}$$

Since there exists no closed-form expression for such KL-divergence, we follow Louizos et al. (2017) to approximate it by the following formula with minimum variance:

$$\begin{aligned}
D_{KL} &= -k_1\sigma(k_2 + k_3 \log \alpha) + \frac{1}{2}\log(1 + \frac{1}{\alpha}) + k_1 \\
k_1 &= 0.63576 \quad k_2 = 1.87320 \quad k_3 = 1.48695
\end{aligned} \tag{7}$$

By adopting the improper log-uniform prior, more weights are compressed towards zero, and the KL-divergence is negatively correlated with dropout ratio α. Intuitively, the dropout ratio α_i is an redundancy indicator for i_{th} word in the vocabulary, with larger α_i meaning less performance loss caused by dropping i_{th} word. During training, we use re-parameterization trick (Kingma and Welling, 2013) to sample embedding weights from the normal distribution to reduce the Monte-Carlo variance in Bayesian training.

3.3 Vocabulary Selection

After optimization, we can obtain the dropout ratio α_i associated with each word w_i. We propose to select vocabulary subset based on the dropout ratio by using a threshold α_T. Therefore, the remaining vocabulary subset is described as follows:

$$\hat{\mathbb{V}} = \{w_i \in V | \alpha_i < \alpha_T\} \tag{8}$$

where we use $\hat{\mathbb{V}}$ to denote the subset vocabulary of interest, by adjusting α_T we are able to control the selected vocabulary size.

4 Experiments

We compare the proposed vocabulary selection algorithm against several strong baselines on a wide range of text classification tasks and datasets.

4.1 Datasets & Architectures

The main datasets we are using are listed in Table 2, which provides an overview of its description and capacities. Specifically, we follow (Zhang et al., 2015; Goo et al., 2018; Williams et al., 2018) to pre-process the document classification datasets, natural language understanding dataset and natural language inference dataset. We exactly replicate their experiment settings to make our method comparable with theirs. Our models is implemented with TensorFlow (Abadi et al., 2015). In order to evaluate the generalization ability of VVD selection algorithm in deep learning architectures, we study its performance under different established architectures (depicted in Figure 3). In natural language understanding, we use the most recent attention-based model for intention tracking (Goo et al., 2018), this model first uses BiLSTM recurrent network to leverage left-to-right and right-to-left context information to form the hidden representation, then computes self-attention weights to aggregate the hidden representation and predicts user intention. In document classification, we mainly follow the CNN architecture (Kim, 2014) to extract n-gram features and then aggregate these features to predict document category. In natural language inference, we follow the popular ESIM architecture (Williams et al., 2018; Chen et al., 2017) us-

Datasets	Task	Description	#Class	#Train	#Test
ATIS-flight (Tur et al., 2010)	NLU	Classify Airline Travel dialog	21	4,478	893
Snips (Coucke et al., 2018)		Classify inputs to personal voice assistant	7	13,084	700
AG-news (Zhang et al., 2015)	DC	Categories: World, Sports, etc	4	120,000	7,600
DBPedia (Lehmann et al., 2015)		Categories: Company, Athlete, Album, etc	14	560,000	70,000
Sogou-news (Zhang et al., 2015)		Categories: Sports, Technology, etc	5	450,000	60,000
Yelp-review (Zhang et al., 2015)		Categories: Review Ratings (1-5)	5	650,000	50,000
SNLI (Bowman et al., 2015)	NLI	Entailment: Contradict, Neutral,Entail	3	550,152	10,000
MNLI (Williams et al., 2018)		Multi-Genre Entailment	3	392,702	10,000

Table 2: An overview of different datasets under different classification tasks including description and sizes.

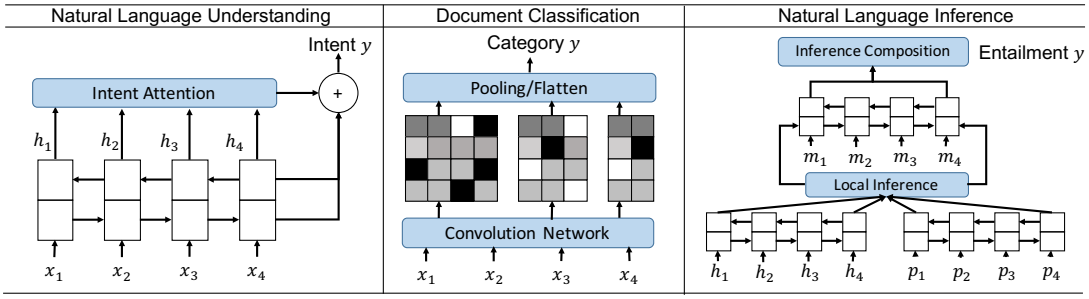

Figure 3: The neural network architecture overview of different NLP tasks.

ing the Github implementation[3]. In this structure, three main components input encoding, local inference modeling, and inference composition are used to perform sequential inference and composition to simulate the interaction between premises and hypothesis. Note that, we do not apply the syntax-tree based LSTM proposed in (Chen et al., 2017) because we lost the parse tree (Klein and Manning, 2003) after the vocabulary compression, instead, we follow the simpler sequential LSTM framework without any syntax parse as input. Besides, the accuracy curve is obtained using the publicly available test split rather than the official online evaluation because we need to evaluate lots of times at different vocabulary capacity.

4.2 Baselines

Here we mainly consider the following baselines:

Frequency-based (task-agnostic) This approach is already extensively talked about in section 1, its basic idea is to rank the word based on its frequency and then set a threshold to cut off the long tail distribution.

TF-IDF (task-agnostic) This algorithm views the vocabulary selection as a retrieval problem (Ramos et al., 2003), where term frequency is viewed as the word frequency and document frequency is viewed as the number of sentences where such word appears. Here we follow the canonical TF-IDF approach to compute the retrieval score as follows:

$$tfidf(w, D) = tf(w)^{\lambda} * (\log \frac{N}{n_w})^{1-\lambda} \quad (9)$$

where $tf(w)$ denotes the word frequency, λ is the balancing factor, N denotes the number of sentences and n_w denotes the number of sentences in which w appears. We rank the whole vocabulary based on the $tfidf$ and cut off at given threshold.

Group Lasso (task-aware) This baseline aims to find intrinsic sparse structures (Liu et al., 2015; Park et al., 2016; Wen et al., 2016) by grouping each row of word embedding. The regularization objective is described as follows, which aims at finding the row-wise sparse structure:

$$\mathcal{L}_{reg} = \sum_i (\sum_j W_{ij}^2)^{\frac{1}{2}} \quad (10)$$

After optimized with the above regularization, we use a threshold-based selection strategy on the row-norm of embedding matrix, the selected vocabulary is described as $\hat{\mathbb{V}} = \{w_i \in \mathbb{V} | ||W_i||_2 > \beta_T\}$, where β_T is the threshold.

4.3 Experimental Results

Here we demonstrate our results in natural language understanding, document classification, and

[3] https://github.com/coetaur0/ESIM

Datasets / Reported Accuracy	Accuracy	Vocab	Methods	AUC	Vocab@-3%	Vocab@-5%
Snips / 96.7 (Liu and Lane, 2016)	95.9	11000	Frequency	77.4	81	61
	95.9		TF-IDF	77.6	81	62
	95.6		Group Lasso	82.1	77	52
	96.0		VVD	**82.5**	**52**	**36**
ATIS-Flight / 94.1 (Goo et al., 2018)	93.8	724	Frequency	70.1	33	28
	93.8		TF-IDF	70.5	34	28
	93.8		Group Lasso	72.9	30	26
	94.0		VVD	**74.8**	**29**	**26**
AG-news / 91.1 (Zhang et al., 2015)	91.6	61673	Frequency	67.1	2290	1379
	91.6		TF-IDF	67.8	2214	1303
	91.2		Group Lasso	68.3	1867	1032
	91.6		VVD	**70.5**	**1000**	**673**
DBPedia / 98.3 (Zhang et al., 2015)	98.4	563355	Frequency	69.7	1000	743
	98.4		TF-IDF	71.7	1703	804
	97.9		Group Lasso	71.9	768	678
	98.5		VVD	**72.2**	**427**	**297**
Sogou-news / 95.0 (Zhang et al., 2015)	93.7	254495	Frequency	70.9	789	643
	93.7		TF-IDF	71.3	976	776
	93.6		Group Lasso	73.4	765	456
	94.0		VVD	**75.5**	**312**	**196**
Yelp-review / 58.0 (Zhang et al., 2015)	56.3	252712	Frequency	74.0	1315	683
	56.3		TF-IDF	74.1	1630	754
	56.5		Group Lasso	75.4	934	463
	57.4		VVD	**77.9**	**487**	**287**
SNLI / 86.7 (Williams et al., 2018)	84.1	42392	Frequency	72.2	2139	1362
	84.1		TF-IDF	72.8	2132	1429
	84.6		Group Lasso	73.6	1712	1093
	85.5		VVD	**75.0**	**1414**	**854**
MNLI / 72.3 (Williams et al., 2018)	69.2	100158	Frequency	78.5	1758	952
	69.2		TF-IDF	78.7	1656	934
	70.1		Group Lasso	79.2	1466	711
	71.2		VVD	**80.1**	**1323**	**641**

Table 3: Experimental Results on various NLP tasks and datasets on the proposed metrics in subsection 2.3. Bold accuracy means the result is statistically significantly better than the competitors.

natural language inference separately in Table 3. From these tables, first of all, we can observe that VVD is able to maintain or even improve the reported accuracy on DC and NLU tasks, the accuracy of VVD is reported under dropping out the words with dropout rate larger than 0.95. The exception is in NLI (Williams et al., 2018), where the common approach uses GloVe (Pennington et al., 2014) for initialization, and we use random initialization, which makes our model fall slightly behind. It is worth noting that Frequency-based/TF-IDF methods are based on the model trained with cross entropy, while both Group-Lasso and VVD modify the objective function by adding additional regularization. It can be seen that VVD is performing very similar to the baseline models on DC and NLU tasks, while consistently outperforming the baseline methods (with random initialized embedding) on more challenging NLI and Yelp-Review tasks, that said, VVD can also be viewed as a generally effective regularization technique to sparsify features and alleviate the over-fitting problem in NLP tasks. In terms of the vocabulary selection capability, our proposed VVD is demonstrated to outperform the competing algorithms in terms of both AUC and Vocab@-X% metrics consistently over different datasets as shown in Table 3. In order to better understand the margin between VVD and frequency-based method, we plot their accuracy-vocab curves in Figure 4, from which we can observe that the accuracy curves start from nearly the same accuracy with the full vocabulary, by gradually decreasing the budget $\hat{V}$, VVD decreases at a much lower rate than the competing algorithms, which clearly reflects its superiority under limited-budget scenario. From the empirical result, we can conclude that: 1) the retrieval-based selection algorithm can yield

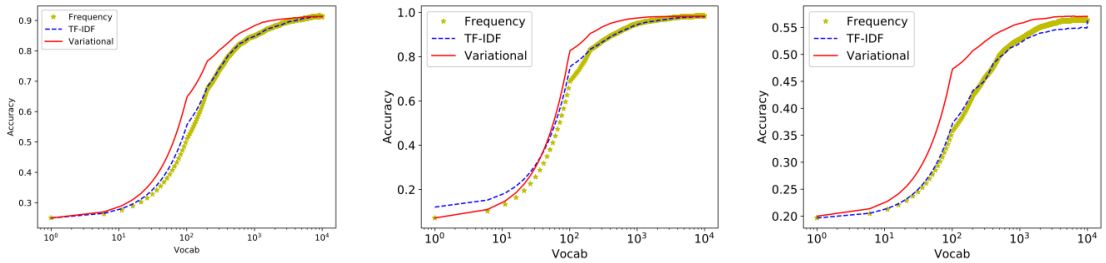

Figure 4: The accuracy-vocab curve of VVD, TF-IDF and frequency-based baseline, the datasets used are AG-news, DBPedia and Yelp-review respectively.

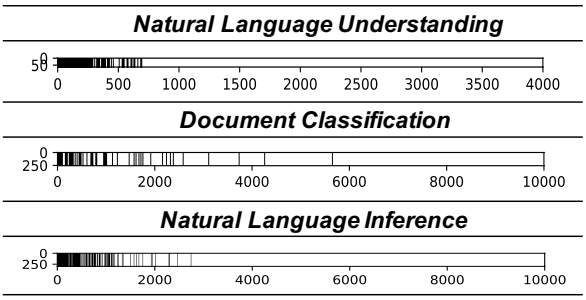

Figure 5: The vocabulary selection spectrum of our proposed VVD algorithm on different NLP tasks.

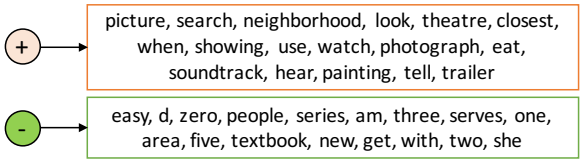

Figure 6: The vocabulary selected (+)/unselected (-) by VVD compared to frequency-based baseline.

marginal improvement over the AUC metric, but the vocab@-X% metric deteriorates. 2) group-lasso and VVD algorithm directly considers the connection between each word and end classification accuracy; such task-awareness can greatly in improving both evaluation metrics. Here we show that NLU datasets are relatively simpler, which only involves detecting key words from human voice inputs to make decent decisions, a keyword vocabulary within 100 is already enough for promising accuracy. For DC datasets, which involve better inner-sentence and inter-sentence understanding, hundred-level vocabulary is required for most cases. NLI datasets involve more complicated reasoning and interaction, which requires a thousand-level vocabulary.

Figure 7: The vocabulary cloud of Snips NLU dataset.

Case Study To provide an overview of what words are selected, we depict the selection spectrum over different NLP tasks in Figure 5, from which we observe that most of the selected vocabulary are still from the high-frequency area to ensure coverage, which also explains why the frequency-based algorithm is already very strong. Furthermore, we use the Snips dataset (Coucke et al., 2018) to showcase the difference between the vocabularies selected by VVD and by frequency-based baseline. The main goal of this dataset is to understand the speaker's intention such as "BookRestaurant", "PlayMusic", and "SearchLocalEvent". We show the selected/unselected words by our algorithm in Figure 6 under a vocabulary budget of 100, it is observed that many non-informative but frequent functional words like "get", "with", and "five" are unselected while more task-related but less frequent words like "neighborhood", "search", "theatre" are selected. More vividly, we demonstrate the word cloud of the selected vocabulary of Snips (Coucke et al., 2018) in Figure 7.

4.4 Discussion

Here we will talk about some potential issues posed when training and evaluating VVD.

Training Speed Due to the stochasticity of VVD, the training of text classification takes longer than canonical cross entropy objective. More importantly, we observe that with the in-

crease the full vocabulary size, the convergence time of VVD also increases sub-linearly but the convergence time of Cross Entropy remains quite consistent. We conjecture that this is due to the fact that the VVD algorithm has a heavier burden to infer the drop out the probability of the long tail words. Therefore, we propose to use a two-step vocabulary reduction to dramatically decrease VVD's training time, in the first step, we cut off the rare words without having any harm on the final accuracy, then we continue training with VVD on the shrunk vocabulary. By applying such a hybrid methodology, we are able to decrease the training time dramatically.

Evaluation Speed As we know, at each vocabulary point, the network needs to perform once evaluation on the whole test set. Therefore, it is not practical to draw each vocabulary size from 1 to V and perform V times of evaluation. Given the limited computational resources, we need to sample some vocabulary size and estimate the area under curve relying on only these points. Uniformly sampling the data points are proved wasteful, since when the accuracy curve will converge to a point very early, most of the sampled point is actually getting equivalent accuracy. Therefore, we propose to increase the interval exponentially to cover more samples at extremely low vocabulary size. For example, given the total vocabulary of 60000, the interval will be split into 1, 2, 4, 8, 24, 56, ..., 60K. Using such sampling method achieve a reasonably accurate estimation of ROC with only $O(log(|V|))$ sample points, which is affordable under many cases.

5 Related Work

Neural Network Compression In order to better apply the deep neural networks under limited-resource scenarios, much recent research has been performed to compress the model size and decrease the computation resources. In summary, there are mainly three directions, weight matrices approximation (Le et al., 2015; Tjandra et al., 2017), reducing the precision of the weights (Hubara et al., 2017; Han et al., 2015) and sparsification of the weight matrix (Wen et al., 2016). Another group of sparsification relies on the Bayesian inference framework (Molchanov et al., 2017; Neklyudov et al., 2017; Louizos et al., 2017). The main advantage of the Bayesian sparsification techniques is that they have a small number of hyperparameters compared to pruning-based methods. As stated in (Chirkova et al., 2018), Bayesian compression also leads to a higher sparsity level (Molchanov et al., 2017; Neklyudov et al., 2017; Louizos et al., 2017). Our proposed VVD is inspired by these predecessors to specifically tackle the vocabulary redundancy problem in NLP tasks.

Vocabulary Reduction An orthogonal line of research for dealing similar vocabulary redundancy problem is the character-based approaches to reduce vocabulary sise (Kim et al., 2016; Zhang et al., 2015; Costa-Jussà and Fonollosa, 2016; Lee et al., 2017), which decomposes the words into its characters forms for better handling open world inputs. However, these approaches are not applicable to character-free languages like Chinese and Japanese. Moreover, splitting words into characters incurs potential lose of word-level surface form, and thus needs more parameters at the neural network level to recover it to maintain the end task performance (Zhang et al., 2015), which contradicts with our initial motivation of compressing the neural network models for computation- or memory-constrained scenarios.

6 Conclusion

In this paper, we propose a vocabulary selection algorithm which can find sparsity in the vocabulary and dynamically decrease its size to contain only the useful words. Through our experiments, we have empirically demonstrated that the commonly adopted frequency-based vocabulary selection is already a very strong mechanism, further applying our proposed VVD can further improve the compression ratio. However, due to the time and memory complexity issues, our algorithm and evaluation are more suitable for classification-based application. In the future, we plan to investigate broader applications like summarizaion, translation, question answering, etc.

7 Acknowledgement

The authors would like to thank the anonymous reviewers for their thoughtful comments. This research was sponsored in part by NSF 1528175. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *International Conference on Learning Representations*.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 632–642.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017. Enhanced LSTM for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1657–1668.

Nadezhda Chirkova, Ekaterina Lobacheva, and Dmitry Vetrov. 2018. Bayesian compression for natural language processing. *arXiv preprint arXiv:1810.10927*.

Marta R. Costa-Jussà and José A. R. Fonollosa. 2016. Character-based neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 2: Short Papers*.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibault Lavril, Maël Primet, and Joseph Dureau. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *CoRR*, abs/1805.10190.

Manaal Faruqui, Yulia Tsvetkov, Dani Yogatama, Chris Dyer, and Noah A. Smith. 2015. Sparse overcomplete word vector representations. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 1491–1500.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. 2010. A note on the group lasso and a sparse group lasso. *arXiv preprint arXiv:1001.0736*.

Chih-Wen Goo, Guang Gao, Yun-Kai Hsu, Chih-Li Huo, Tsung-Chieh Chen, Keng-Wei Hsu, and Yun-Nung Chen. 2018. Slot-gated modeling for joint slot filling and intent prediction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, volume 2, pages 753–757.

Song Han, Huizi Mao, and William J Dally. 2015. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *International Conference on Learning Representations*.

Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. 2017. Quantized neural networks: Training neural networks with low precision weights and activations. *The Journal of Machine Learning Research*, 18(1):6869–6898.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1746–1751.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. 2016. Character-aware neural language models. In *AAAI*, pages 2741–2749.

Diederik P Kingma, Tim Salimans, and Max Welling. 2015. Variational dropout and the local reparameterization trick. In *Advances in Neural Information Processing Systems*, pages 2575–2583.

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *International Conference on Learning Representations*.

Dan Klein and Christopher D Manning. 2003. Fast exact inference with a factored model for natural language parsing. In *Advances in neural information processing systems*, pages 3–10.

Quoc V. Le, Navdeep Jaitly, and Geoffrey E. Hinton. 2015. A simple way to initialize recurrent networks of rectified linear units. *CoRR*, abs/1504.00941.

Jason Lee, Kyunghyun Cho, and Thomas Hofmann. 2017. Fully character-level neural machine translation without explicit segmentation. *TACL*, 5:365–378.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. Dbpedia–a

large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6(2):167–195.

Baoyuan Liu, Min Wang, Hassan Foroosh, Marshall Tappen, and Marianna Pensky. 2015. Sparse convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 806–814.

Bing Liu and Ian Lane. 2016. Attention-based recurrent neural network models for joint intent detection and slot filling. In *Interspeech 2016, 17th Annual Conference of the International Speech Communication Association, San Francisco, CA, USA, September 8-12, 2016*, pages 685–689.

Christos Louizos, Karen Ullrich, and Max Welling. 2017. Bayesian compression for deep learning. In *Advances in Neural Information Processing Systems*, pages 3288–3298.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1412–1421.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Dmitry Molchanov, Arsenii Ashukha, and Dmitry P. Vetrov. 2017. Variational dropout sparsifies deep neural networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 2498–2507.

Kirill Neklyudov, Dmitry Molchanov, Arsenii Ashukha, and Dmitry P Vetrov. 2017. Structured bayesian pruning via log-normal multiplicative noise. In *Advances in Neural Information Processing Systems*, pages 6775–6784.

Jongsoo Park, Sheng Li, Wei Wen, Ping Tak Peter Tang, Hai Li, Yiran Chen, and Pradeep Dubey. 2016. Faster cnns with direct sparse convolutions and guided pruning. *International Conference on Learning Representations*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Juan Ramos et al. 2003. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, volume 242, pages 133–142.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

Andros Tjandra, Sakriani Sakti, and Satoshi Nakamura. 2017. Compressing recurrent neural network with tensor train. In *2017 International Joint Conference on Neural Networks, IJCNN 2017, Anchorage, AK, USA, May 14-19, 2017*, pages 4451–4458.

Gokhan Tur, Dilek Hakkani-Tür, and Larry Heck. 2010. What is left to be understood in atis? In *Spoken Language Technology Workshop (SLT), 2010 IEEE*, pages 19–24. IEEE.

Sida Wang and Christopher Manning. 2013. Fast dropout training. In *international conference on machine learning*, pages 118–126.

Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. 2016. Learning structured sparsity in deep neural networks. In *Advances in Neural Information Processing Systems*, pages 2074–2082.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 1112–1122.

Dani Yogatama, Manaal Faruqui, Chris Dyer, and Noah A. Smith. 2015. Learning word representations with hierarchical sparse coding. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pages 87–96.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657.