Learning Variational Word Masks to Improve the Interpretability of Neural Text Classifiers

Hanjie Chen, Yangfeng Ji
Department of Computer Science
University of Virginia
Charlottesville, VA, USA
{`hc9mx, yangfeng`}@virginia.edu

Abstract

To build an interpretable neural text classifier, most of the prior work has focused on designing inherently interpretable models or finding faithful explanations. A new line of work on improving model interpretability has just started, and many existing methods require either prior information or human annotations as additional inputs in training. To address this limitation, we propose the variational word mask (VMASK) method to automatically learn task-specific important words and reduce irrelevant information on classification, which ultimately improves the interpretability of model predictions. The proposed method is evaluated with three neural text classifiers (CNN, LSTM, and BERT) on seven benchmark text classification datasets. Experiments show the effectiveness of VMASK in improving both model prediction accuracy and interpretability.

1 Introduction

Neural network models have achieved remarkable performance on text classification due to their capacity of representation learning on natural language texts (Zhang et al., 2015; Yang et al., 2016; Joulin et al., 2017; Devlin et al., 2018). However, the lack of understanding of their prediction behaviors has become a critical issue for reliability and trustworthiness and hindered their applications in the real world (Lipton, 2016; Ribeiro et al., 2016; Jacovi and Goldberg, 2020). Many explanation methods have been proposed to provide post-hoc explanations for neural networks (Ribeiro et al., 2016; Lundberg and Lee, 2017; Sundararajan et al., 2017), but they are only able to explain model predictions and cannot help improve their interpretability.

In this work, we consider interpretability as an intrinsic property of neural network models. Furthermore, we hypothesize that neural network models with similar network architectures could have

Ex.	Model	Text & Explanation
1	A	An exceedingly clever piece of cinema
	B	An exceedingly clever piece of cinema
2	A	It becomes gimmicky instead of compelling
	B	It becomes gimmicky instead of compelling

Table 1: Model A and B are two neural text classifiers with similar network architectures. They all make correct sentiment predictions on both texts (ex. 1: positive; ex. 2: negative). Two post-hoc explanation methods, LIME (Ribeiro et al., 2016) and SampleShapley (Kononenko et al., 2010), are used to explain the model predictions on example 1 and 2 respectively. Top three important words are shown in pink or blue for model A and B. Whichever post-hoc method is used, explanations from model B are easier to understand because the sentiment keywords "clever" and "gimmicky" are highlighted.

different levels of interpretability, even though they may have similar prediction performance. Table 1 shows explanations extracted from two neural text classifiers with similar network architectures.[1] Although both models make correct predictions of the sentiment polarities of two input texts (positive for example 1 and negative for example 2), they have different explanations for their predictions. In both examples, no matter which explanation generation method is used, explanations from model B are easier to be interpreted regarding the corresponding predictions. Motivated by the difference of interpretability, we would like to investigate the possibility of *building more interpretable neural classifiers with a simple modification on input layers*. The proposed method does not demand significant efforts on engineering network architectures (Rudin, 2019; Melis and Jaakkola, 2018). Also, unlike prior work on improving interpretability (Erion

---

[1]The similarity will be detailed in section 4 and more examples are provided in Table 5.

et al., 2019; Plumb et al., 2019), it does not require pre-defined important attributions or pre-collected explanations.

Specifically, we propose variational word masks (VMASK) that are inserted into a neural text classifier, after the word embedding layer, and trained jointly with the model. VMASK learns to restrict the information of globally irrelevant or noisy word-level features flowing to subsequent network layers, hence forcing the model to focus on important features to make predictions. Experiments in section 5 show that this method can improve model interpretability and prediction performance. As VMASK is deployed on top of the word-embedding layer and the major network structure keeps unchanged, it is *model-agnostic* and can be applied to any neural text classifiers.

The contribution of this work is three-fold: (1) we proposed the VMASK method to learn global task-specific important features that can improve both model interpretability and prediction accuracy; (2) we formulated the problem in the framework of information bottleneck (IB) (Tishby et al., 2000; Tishby and Zaslavsky, 2015) and derived a lower bound of the objective function via the variational IB method (Alemi et al., 2016); and (3) we evaluated the proposed method with three neural network models, CNN (Kim, 2014), LSTM (Hochreiter and Schmidhuber, 1997), and BERT (Devlin et al., 2018), on seven text classification tasks via both quantitative and qualitative evaluations.

Our implementation is available at `https://github.com/UVa-NLP/VMASK`.

2 Related Work

Various approaches have been proposed to interpret DNNs, ranging from designing inherently interpretable models (Melis and Jaakkola, 2018; Rudin, 2019), to tracking the inner-workings of neural networks (Jacovi et al., 2018; Murdoch et al., 2018), to generating post-hoc explanations (Ribeiro et al., 2016; Lundberg and Lee, 2017). Beyond interpreting model predictions, the explanation generation methods are also promising in improving model's performance. We propose an information-theoretic method to improve both prediction accuracy and interpretability.

Explanation from the information-theoretic perspective. A line of works that motivate ours leverage information theory to produce explana-

tions, either maximizing mutual information to recognize important features (Chen et al., 2018; Guan et al., 2019), or optimizing the information bottleneck to identify feature attributions (Schulz et al., 2020; Bang et al., 2019). The information-theoretic approaches are efficient and flexible in identifying important features. Different from generating post-hoc explanations for well-trained models, we utilize information bottleneck to train a more interpretable model with better prediction performance.

Improving prediction performance via explanations. Human-annotated explanations have been utilized to help improve model prediction accuracy (Zhang et al., 2016). Recent work has been using post-hoc explanations to regularize models on prediction behaviors and force them to emphasize more on predefined important features, hence improving their performance (Ross et al., 2017; Ross and Doshi-Velez, 2018; Liu and Avci, 2019; Rieger et al., 2019). Different from these methods that require expert prior information or human annotations, the VMASK method learns global important features automatically during training and incorporate them seamlessly on improving model prediction behaviors.

Improving interpretability via explanations. Some work focuses on improving model's interpretability by aligning explanations with human-judgements (Camburu et al., 2018; Du et al., 2019b; Chen and Ji, 2019; Erion et al., 2019; Plumb et al., 2019). Similarly to the prior work on improving model prediction performance, these methods still rely on annotations or external resources. Although enhancing model interpretability, they may cause the performance drop on prediction accuracy due to the inconsistency between human recognition and model reasoning process (Jacovi and Goldberg, 2020). Our approach can improve both prediction accuracy and interpretability without resorting to human-judgements.

3 Method

This section introduces the proposed VMASK method. For a given neural text classifier, the only modification on the neural network architecture is to insert a word mask layer between the input layer (e.g., word embeddings) and the representation learning layer. We formulate our idea within the information bottleneck framework (Tishby et al., 2000), where the word mask layer restricts the in-

formation from words to the final prediction.

3.1 Interpretable Text Classifier with Word Masks

For an input text $\boldsymbol{x} = [x_1, \cdots, x_T]$, where x_t ($t \in \{1, \ldots, T\}$) indicates the word or the word index in a predefined vocabulary. In addition, we use $\boldsymbol{x}_t \in \mathbb{R}^d$ as the word embedding of x_t. A neural text classifier is denoted as $f_{\boldsymbol{\theta}}(\cdot)$ with parameter $\boldsymbol{\theta}$, which by default takes $\boldsymbol{x}$ as input and generates a probability of output $\boldsymbol{Y}$, $p(\boldsymbol{Y}|\boldsymbol{x})$, over all possible class labels. In this work, beyond prediction accuracy, we also expect the neural network model to be more interpretable, by focusing on important words to make predictions.

To help neural network models for better feature selection, we add a random layer $\boldsymbol{R}$ after the word embeddings, where $\boldsymbol{R} = [R_{x_1}, \ldots, R_{x_T}]$ has the same length of $\boldsymbol{x}$. Each $R_{x_t} \in \{0, 1\}$ is a binary random variable associated with the word type x_t instead of the word position. This random layer together with word embeddings form the input to the neural network model, i.e.,

$$\boldsymbol{Z} = \boldsymbol{R} \odot \boldsymbol{x}, \tag{1}$$

where $\odot$ is an element-wise multiplication and each $\boldsymbol{Z}_t = R_{x_t} \cdot \boldsymbol{x}_t$. Intuitively, $\boldsymbol{Z}$ only contains a subset of $\boldsymbol{x}$, which is selected randomly by $\boldsymbol{R}$. Since $\boldsymbol{R}$ is applied directly on the words as a sequence of 0-1 masks, we also call it the word mask layer in this work.

To ensure $\boldsymbol{Z}$ has enough information on predicting $\boldsymbol{Y}$ while contains the least redundant information from $\boldsymbol{x}$, we follow the standard practice in the information bottleneck theory (Tishby et al., 2000), and write the objective function as

$$\max_{\boldsymbol{Z}} I(\boldsymbol{Z}; \boldsymbol{Y}) - \beta \cdot I(\boldsymbol{Z}; \boldsymbol{X}), \tag{2}$$

where $\boldsymbol{X}$ as a random variable representing a generic word sequence as input, $\boldsymbol{Y}$ is the one-hot output random variable, $I(\cdot; \cdot)$ is the mutual information, and $\beta \in \mathbb{R}_+$ is a coefficient to balance the two mutual information items. This formulation reflects our exact expectation on $\boldsymbol{Z}$. The main challenge here is to compute the mutual information.

3.2 Variational Word Masks

Inspired by the variational information bottleneck proposed by Alemi et al. (2016), instead of computing $p(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z})$, we start from an approximation distribution $q(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z})$. Then, with a few

assumptions specified in the following, we construct a tractable lower bound of the objective in Equation 2 and the detailed derivation is provided in Appendix A.

For $I(\boldsymbol{Z}; \boldsymbol{Y})$ under q, we have $I(\boldsymbol{Z}; \boldsymbol{Y}) = \sum_{\boldsymbol{y}, \boldsymbol{z}} q(\boldsymbol{y}, \boldsymbol{z}) \log(q(\boldsymbol{y}|\boldsymbol{z})/q(\boldsymbol{y}))$. By replacing $\log q(\boldsymbol{y}|\boldsymbol{z})$ with the conditional probability derived from the true distribution $\log p(\boldsymbol{y}|\boldsymbol{z})$, we introduce the constraint between $\boldsymbol{Y}$ and $\boldsymbol{Z}$ from the distribution and also obtain a lower bound of $I(\boldsymbol{Z}; \boldsymbol{Y})$,

$$\begin{aligned} I(\boldsymbol{Z}; \boldsymbol{Y}) &\geq \sum_{\boldsymbol{y}, \boldsymbol{z}} q(\boldsymbol{y}, \boldsymbol{z}) \log p(\boldsymbol{y}|\boldsymbol{z}) + H_q(\boldsymbol{Y}) \\ &= \sum_{\boldsymbol{y}, \boldsymbol{z}, \boldsymbol{x}} q(\boldsymbol{x}, \boldsymbol{y}) q(\boldsymbol{z}|\boldsymbol{x}) \log p(\boldsymbol{y}|\boldsymbol{z}) \\ &\quad + H_q(\boldsymbol{Y}), \end{aligned} \tag{3}$$

where $H_q(\cdot)$ is entropy, and the last step uses $q(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) = q(\boldsymbol{x}) q(\boldsymbol{y}|\boldsymbol{x}) q(\boldsymbol{z}|\boldsymbol{x})$, which is a factorization based on the conditional dependency [2].

Given a specific observation $(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})$, we define the empirical distribution $q(\boldsymbol{X}^{(i)}, \boldsymbol{Y}^{(i)})$ as a multiplication of two Delta functions $q(\boldsymbol{X}^{(i)} = \boldsymbol{x}^{(i)}, \boldsymbol{Y}^{(i)} = \boldsymbol{y}^{(i)}) = \delta_{\boldsymbol{x}^{(i)}}(\boldsymbol{x}) \cdot \delta_{\boldsymbol{y}^{(i)}}(\boldsymbol{y})$. Then, Equation 3 can be further simplified as

$$\begin{aligned} I(\boldsymbol{Z}; \boldsymbol{Y}^{(i)}) &\geq \sum_{\boldsymbol{z}} q(\boldsymbol{z}|\boldsymbol{x}^{(i)}) \log p(\boldsymbol{y}^{(i)}|\boldsymbol{z}) \\ &= \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x}^{(i)})}[\log p(\boldsymbol{y}^{(i)}|\boldsymbol{z})]. \end{aligned} \tag{4}$$

Similarly, for $I(\boldsymbol{Z}; \boldsymbol{X})$ under q, we have an upper bound of $I(\boldsymbol{Z}; \boldsymbol{X})$ by replacing $p(\boldsymbol{Z}|\boldsymbol{X})$ with a predefined prior distribution $p_0(\boldsymbol{Z})$

$$\begin{aligned} I(\boldsymbol{Z}; \boldsymbol{X}) &\leq \mathbb{E}_{q(\boldsymbol{x})}[\mathrm{KL}[q(\boldsymbol{z}|\boldsymbol{x}) \| p_0(\boldsymbol{z})]] \\ &= \mathrm{KL}[q(\boldsymbol{z}|\boldsymbol{x}^{(i)}) \| p_0(\boldsymbol{z})], \end{aligned} \tag{5}$$

where $\mathrm{KL}[\cdot \| \cdot]$ denotes Kullback-Leibler divergence. The simplification in the last step is similar to Equation 4 with the empirical distribution $q(\boldsymbol{X}^{(i)})$.

Substituting (5) and (4) into Equation 2 gives us a lower bound $\mathcal{L}$ of the informaiton bottleneck

$$\begin{aligned} \mathcal{L} =\,& \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x}^{(i)})}[\log p(\boldsymbol{y}^{(i)}|\boldsymbol{z})] \\ &- \beta \cdot \mathrm{KL}[q(\boldsymbol{z}|\boldsymbol{x}^{(i)}) \| p_0(\boldsymbol{z})]. \end{aligned} \tag{6}$$

The learning objective is to maximize Equation 6 with respect to the approximation distribution $q(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}) = q(\boldsymbol{X}, \boldsymbol{Y}) q(\boldsymbol{Z}|\boldsymbol{X})$. As a classification problem, $\boldsymbol{X}$ and $\boldsymbol{Y}$ are both observed and

---

[2] $\boldsymbol{Y} \leftrightarrow \boldsymbol{X} \leftrightarrow \boldsymbol{Z}$: $\boldsymbol{Y}$ and $\boldsymbol{Z}$ are independent given $\boldsymbol{X}$.

$q(\boldsymbol{X}, \boldsymbol{Y})$ has already been simplified as an empirical distribution, the only one left in the approximation distribution is $q(\boldsymbol{Z}|\boldsymbol{X})$. Similarly to the objective function in variational inference (Alemi et al., 2016; Rezende and Mohamed, 2015), the first term in $\mathcal{L}$ is to make sure the information in $q(\boldsymbol{Z}|\boldsymbol{X})$ for predicting $\boldsymbol{Y}$, while the second term in $\mathcal{L}$ is to regularize $q(\boldsymbol{Z}|\boldsymbol{X})$ with a predefined prior distribution $p_0(\boldsymbol{Z})$.

The last step of obtaining a practical objective function is to notice that, given $\boldsymbol{X}_t^{(i)} = \boldsymbol{x}_t^{(i)}$ every $\boldsymbol{Z}_t$ can be redefined as

$$\boldsymbol{Z}_t = R_{x_t} \cdot \boldsymbol{x}_t^{(i)}, \tag{7}$$

where $R_{x_t} \in \{0, 1\}$ is a standard Bernoulli distribution. Then, $\boldsymbol{Z}$ can be reparameterized as $\boldsymbol{Z} = \boldsymbol{R} \odot \boldsymbol{x}^{(i)}$ with $\boldsymbol{R} = [R_{x_1}, \dots, R_{x_T}]$. The lower bound $\mathcal{L}$ can be rewritten with the random variable $\boldsymbol{R}$ as

$$\begin{aligned}\mathcal{L} =& \mathbb{E}_{q(\boldsymbol{r}|\boldsymbol{x}^{(i)})}[\log p(\boldsymbol{y}^{(i)}|\boldsymbol{R}, \boldsymbol{x}^{(i)})] \\ & - \beta \cdot \mathrm{KL}[q(\boldsymbol{R}|\boldsymbol{x}^{(i)})\|p_0(\boldsymbol{R})].\end{aligned} \tag{8}$$

Note that, although β is inherited from the information bottleneck theory, in practice it will be used as a tunable hyper-parameter to address the notorious posterior collapse issue (Bowman et al., 2016; Kim et al., 2018).

3.3 Connections

The idea of modifying word embeddings with the information bottleneck method has recently shown some interesting applications in NLP. For example, Li and Eisner (2019) proposed two ways to transform word embeddings into new representations for better POS tagging and syntactic parsing. According to Equation 1, VMASK can be viewed as a simple linear transformation on word embeddings. The difference is that $\{R_{x_t}\}$ is defined on the vocabulary, therefore can be used to represent the global importance of word x_t. Recall that $R_{x_t} \in \{0, 1\}$, from a slightly different perspective, Equation 1 can be viewed as a generalized method on word-embedding dropout (Gal and Ghahramani, 2016). Although there are two major differences: (1) in Gal and Ghahramani (2016) all words share the same dropout rate, while in VMASK every word has its own dropout rate specified by $q(R_{x_t}|\boldsymbol{x}_t)$, i.e. $1 - \mathbb{E}[q(R_{x_t}|\boldsymbol{x}_t)]$; (2) the motivation of word-embedding dropout is to force a model not to rely on single words for prediction, while VMASK is to learn a task-specific importance for every word.

Another implementation for making word masks sparse is by adding L_0 regularization (Lei et al., 2016; Bastings et al., 2019; Cao et al., 2020), but this regularizer only distinguishes words as important or unimportant, rather than learning continuous importance scores.

3.4 Model Specification and Training

We resort to mean-field approximation (Blei et al., 2017) to simplify the assumption on our q distribution. For $q_\phi(\boldsymbol{R}|\boldsymbol{x})$, we have $q_\phi(\boldsymbol{R}|\boldsymbol{x}) = \prod_{t=1}^T q_\phi(R_{x_t}|\boldsymbol{x}_t)$, which means the random variables are mutually independent and each governed by $\boldsymbol{x}_t$. We use the amortized variational inference (Rezende and Mohamed, 2015) to represent the posterior distribution $q_\phi(R_{x_t}|\boldsymbol{x}_t)$ with using an inference network (Kingma and Welling, 2014). In this work, we adopt a single-layer feedforward neural network as the inference network, whose parameters ϕ are optimized with the model parameters θ during training.

Following the same factorization as in $q_\phi(\boldsymbol{R}|\boldsymbol{x})$, we define the prior distribution $p_0(\boldsymbol{R})$ as $p_0(\boldsymbol{R}) = \prod_{t=1}^T p_0(R_{x_t})$ and each of them as $p_0(R_{x_t}) = \mathrm{Bernoulli}(0.5)$. By choosing this non-informative prior, it means every word is initialized with no preference to be important or unimportant, and thus has the equal probability to be masked or selected. As $p_0(\boldsymbol{R})$ is a uniform distribution, we can further simplify the second term in Equation 8 as a conditional entropy,

$$\max_{\boldsymbol{\theta}, \boldsymbol{\phi}} \mathbb{E}_q[\log p(\boldsymbol{y}^{(i)}|\boldsymbol{R}, \boldsymbol{x}^{(i)})] + \beta \cdot H_q(\boldsymbol{R}|\boldsymbol{x}^{(i)}). \tag{9}$$

We apply stochastic gradient descent to solve the optimization problem (Equation 9). Particularly in each iteration, the first term in Equation 9 is approximated with a single sample from $q(\boldsymbol{R}|\boldsymbol{x}^{(i)})$ (Kingma and Welling, 2014). However, sampling from a Bernoulli distribution (like from any other discrete distributions) causes difficulty in backpropagation. We adopt the Gumbel-softmax trick (Jang et al., 2016; Maddison et al., 2016) to utilize a continuous differentiable approximation and tackle the discreteness of sampling from Bernoulli distributions (Appendix B). During training, We use Adam (Kingma and Ba, 2014) for optimization and KL cost annealing (Bowman et al., 2016) to avoid posterior collapse.

For a given word x_t and its word embedding $\boldsymbol{x}_t$, in training stage, the model samples each r_{x_t} from $q(R_{x_t}|\boldsymbol{x}_t)$ to decide to either keep or zero out the

Datasets	C	L	#train	#dev	#test
IMDB	2	268	20K	5K	25K
SST-1	5	18	8544	1101	2210
SST-2	2	19	6920	872	1821
Yelp	2	138	500K	60K	38K
AG News	4	32	114K	6K	7.6K
TREC	6	10	5000	452	500
Subj	2	23	8000	1000	1000

Table 2: Summary statistics for the datasets, where C is the number of classes, L is average sentence length, and # counts the number of examples in the *train/dev/test* sets.

corresponding word embedding x_t. In inference stage, the model takes the multiplication of the word embedding x_t and the expectation of the word mask distribution, i.e. $x_t \cdot \mathbb{E}[q(R_{x_t}|x_t)]$, as input.

4 Experiment Setup

The proposed method is evaluated on seven text classification tasks, ranging from sentiment analysis to topic classification, with three typical neural network models, a long short-term memories (Hochreiter and Schmidhuber, 1997, LSTM), a convolutional neural network (Kim, 2014, CNN), and BERT (Devlin et al., 2018).

Datasets. We adopt seven benchmark datasets: movie reviews IMDB (Maas et al., 2011), Stanford Sentiment Treebank with fine-grained labels SST-1 and its binary version SST-2 (Socher et al., 2013), Yelp reviews (Zhang et al., 2015), AG's News (Zhang et al., 2015), 6-class question classification TREC (Li and Roth, 2002), and subjective/objective classification Subj (Pang and Lee, 2005). For the datasets (e.g. IMDB, Subj) without standard train/dev/test split, we hold out a proportion of training examples as the development set. Table 2 shows the statistics of the datasets.

Models. The CNN model (Kim, 2014) contains a single convolutional layer with filter sizes ranging from 3 to 5. The LSTM (Hochreiter and Schmidhuber, 1997) has a single unidirectional hidden layer. Both models are initialized with 300-dimensional pretrained word embeddings (Mikolov et al., 2013). We fix the embedding layer and update other parameters on different datasets to achieve the best performance respectively. We use the pretrained

BERT-base model[3] with 12 transformer layers, 12 self-attention heads, and the hidden size of 768. We fine-tune it with different downstream tasks, and then fix the embedding layer and train the mask layer with the rest of the model together.

Baselines and Competitive Methods. As the goal of this work is to propose a novel training method that improves both prediction accuracy and interpretability, we employ two groups of models as baselines and competitive systems. Models trained with the proposed method are named with suffix "-VMASK". We also provide two baselines: (1) models trained by minimizing the cross-entropy loss (postfixed with "-base") and (2) models trained with ℓ_2-regularization (postfixed with "-ℓ_2"). The comparison with these two baseline methods mainly focuses on prediction performance as no explicit training strategies are used to improve interpretability.

Besides, we also propose two competitive methods: models trained with the explanation framework "Learning to Explain" (Chen et al., 2018) (postfixed with "-L2X") and the "Information Bottleneck Attribution" (Schulz et al., 2020) (postfixed with "-IBA"). L2X and IBA were originally proposed to find feature attributions as post-hoc explanations for well-trained models. We integrated them in model training, working as the mask layer to directly generate mask values for input features (L2X) or restrict information flow by adding noise (IBA). In our experiments, all training methods worked with random dropout ($\rho = 0.2$) to avoid overfitting.

More details about experiment setup are in Appendix C, including data pre-processing, model configurations, and the implementation of L2X and IBA in our experiments.

5 Results and Discussion

We trained the three models on the seven datasets with different training strategies. Table 3 shows the prediction accuracy of different models on test sets. The validation performance and average runtime are in Appendix D. As shown in Table 3, all base models have the similar prediction performance comparing to numbers reported in prior work (Appendix E). The models trained with VMASK outperform the ones with similar network architec-

---
[3] https://github.com/huggingface/
pytorch-transformers

Models	Methods	IMDB	SST-1	SST-2	Yelp	AG News	TREC	Subj
CNN	CNN-base	89.06	46.32	85.50	94.32	91.30	92.40	92.80
	CNN-ℓ_2	89.12	46.01	85.56	94.46	91.28	90.62	92.39
	CNN-L2X	78.94	37.92	80.01	83.14	84.36	61.00	82.40
	CNN-IBA	88.31	41.40	84.24	93.82	91.37	89.80	91.80
	CNN-VMASK	**90.10**	**48.92**	**85.78**	**94.53**	**91.60**	**93.02**	**93.50**
LSTM	LSTM-base	88.39	43.84	83.74	95.06	91.03	90.40	90.20
	LSTM-ℓ_2	88.40	43.91	83.36	95.00	91.09	90.20	89.10
	LSTM-L2X	67.45	36.92	75.45	77.12	77.53	46.00	81.80
	LSTM-IBA	88.48	42.99	83.53	94.74	91.14	85.40	89.50
	LSTM-VMASK	**90.07**	**44.12**	**84.35**	**95.41**	**92.19**	**90.80**	**91.20**
BERT	BERT-base	91.80	53.43	92.25	96.42	93.59	96.40	95.10
	BERT-ℓ_2	91.75	52.08	92.25	96.41	93.52	96.80	94.80
	BERT-L2X	71.75	39.23	74.03	87.14	82.59	93.20	86.10
	BERT-IBA	91.66	53.80	92.24	96.27	93.45	96.80	95.60
	BERT-VMASK	**93.04**	**54.53**	**92.26**	**96.80**	**94.24**	**97.00**	**96.40**

Table 3: Prediction accuracy (%) of different models with different training strategies on the seven datasets.

tures but trained differently. The results show that VMASK can help improve the generalization power.

Except the base models and the models trained with the proposed method, the records of other three competitors are mixed. For example, the traditional ℓ_2-regularization cannot always help improve accuracy, especially for the BERT model. Although the performance with IBA is slightly better than with L2X, training with them does not show a constant improvement on a model's prediction accuracy.

To echo the purpose of improving model interpretability, the rest of this section will focus on evaluating the model interpretability quantitatively and qualitatively.

5.1 Quantitative Evaluation

We evaluate the local interpretability of VMASK-based models against the base models via the AOPC score (Nguyen, 2018; Samek et al., 2016) and the global interpretability against the IBA-based models via post-hoc accuracy (Chen et al., 2018). Empirically, we observed the agreement between local and global interpretability, so there is no need to exhaust all possible combinations in our evaluation.

5.1.1 Local interpretability: AOPC

We adopt two model-agnostic explanation methods, LIME (Ribeiro et al., 2016) and SampleShapley (Kononenko et al., 2010), to generate local

explanations for base and VMASK-based models, where "local" means explaining each test data individually. The area over the perturbation curve (AOPC) (Nguyen, 2018; Samek et al., 2016) metric is utilized to evaluate the faithfulness of explanations to models. It calculates the average change of prediction probability on the predicted class over all test data by deleting top n words in explanations. We adopt this metric to evaluate the model interpretability to post-hoc explanations. Higher AOPC scores are better.

For TREC and Subj datasets, we evaluate all test data. For each other dataset, we randomly pick up 1000 examples for evaluation due to computation costs. Table 4 shows the AOPCs of different models on the seven datasets by deleting top 5 words identified by LIME or SampleShapley. The AOPCs of VMASK-based models are significantly higher than that of base models on most of the datasets, indicating that VMASK can improve model's interpretability to post-hoc explanations. The results on the TREC dataset are very close because top 5 words are possible to include all informative words for short sentences with the average length of 10.

5.1.2 Global Interpretability: Post-hoc accuracy

The expectation values $\{\mathbb{E}[q(R_{x_t}|\boldsymbol{x}_t)]\}$ represent the global importance of words (subsection 3.3) for a specific task. To measure the interpretability of a model itself (aka, global interpretability), we adopt

Methods	Models	IMDB	SST-1	SST-2	Yelp	AG News	TREC	Subj
LIME	CNN-base	14.47	7.59	16.50	10.69	5.66	15.28	9.77
	CNN-VMASK	**14.74**	**8.63**	**18.86**	**11.38**	**9.03**	14.81	**12.40**
	LSTM-base	14.34	8.76	17.03	8.72	7.00	11.95	9.67
	LSTM-VMASK	**15.10**	**9.52**	**22.14**	**9.70**	**7.39**	11.97	**11.68**
	BERT-base	10.63	36.00	35.89	6.30	7.00	59.22	13.08
	BERT-VMASK	**12.64**	36.16	**46.87**	6.49	**8.47**	60.37	17.82
SampleShapley	CNN-base	15.53	7.63	13.15	13.57	9.88	14.97	8.84
	CNN-VMASK	15.53	**8.33**	**15.95**	**15.06**	9.98	**15.03**	**12.88**
	LSTM-base	15.80	7.91	22.38	10.55	6.62	11.90	11.66
	LSTM-VMASK	**16.48**	**9.73**	22.52	**10.99**	**7.65**	11.86	**12.74**
	BERT-base	12.97	42.06	43.16	18.06	7.21	57.69	33.22
	BERT-VMASK	**13.18**	**44.57**	**50.44**	18.17	**10.02**	58.26	**34.22**

Table 4: AOPCs (%) of LIME and SampleShapley in interpreting the base and VMASK-based models on the seven datasets.

the post-hoc accuracy (Chen et al., 2018) to evaluate the influence of global task-specific important features on the predictions of VMASK- and IBA-based models. For each test data, we select the top k words based on their global importance scores for the model to make a prediction, and compare it with the original prediction made on the whole input text

$$\text{post-hoc-acc}(k) = \frac{1}{M} \sum_{m=1}^{M} \mathbb{1}[y_m(k) = y_m],$$

where M is the number of examples, y_m is the predicted label on the m-th test data, and $y_m(k)$ is the predicted label based on the top k important words.

Figure 1 shows the results of VMASK- and IBA-based models on the seven datasets with k ranging from 1 to 10. VMASK-based models (solid lines) outperform IBA-based models (dotted lines) with higher post-hoc accuracy, which indicates our proposed method is better on capturing task-specific important features. For CNN-VMASK and LSTM-VMASK, using only top two words can achieve about 80% post-hoc accuracy, even for the IMDB dataset, which has the average sentence length of 268 tokens. The results illustrate that VMASK can identify informative words for model predictions. We also noticed that BERT-VMASK has lower post-hoc accuracy than the other two models. It is probably because BERT tends to use larger context with its self-attentions for predictions. This also explains that the post-hoc accuracies of BERT-

VMASK on the IMDB and SST-1 datasets are catching up slowly with k increasing.

5.2 Qualitative Evaluation

Visualizing post-hoc local explanations. Table 5 shows some examples of LIME explanations for different models on the IMDB dataset. We highlight the top three important words identified by LIME, where the color saturation indicates word attribution. The pair of base and VMASK-based models make the same and correct predictions on the input texts. For VMASK-based models, LIME can capture the sentiment words that indicate the same sentiment polarity as the prediction. While for base models, LIME selects some irrelevant words (e.g. "plot", "of", "to") as explanations, which illustrates the relatively lower interpretability of base models to post-hoc explanations.

Visualizing post-hoc global explanations. We adopt SP-LIME proposed by Ribeiro et al. (2016) as a third-party global interpretability of base and VMASK-based models. Without considering the rectriction on the number of explanations, we follow the method to compute feature global importance from LIME local explanations (subsubsection 5.1.1) by calculating the sum over all local importance scores of a feature as its global importance. To distinguish it from the global importance learned by VMASK, we call it *post-hoc global importance*.

Table 6 lists the top three post-hoc global important words of base and VMASK-based models on the IMDB dataset. For VMASK-based mod-
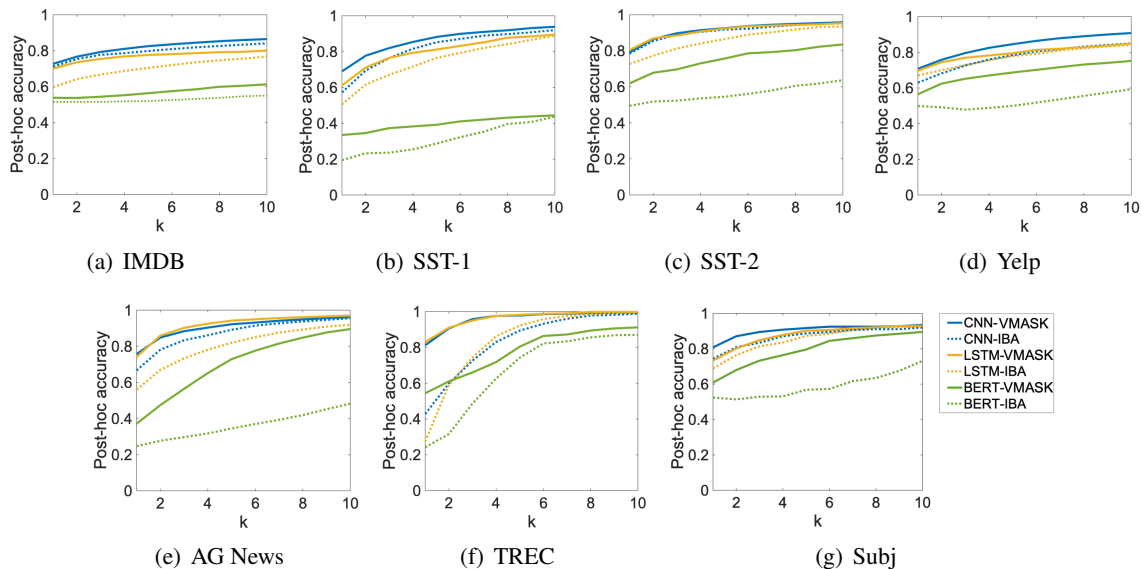
Figure 1: Post-hoc accuracy of VMASK- and IBA-based models on the seven datasets.

els, the global important features selected by SP-LIME are all sentiment words. While for base models, some irrelevant words (e.g. "performances", "plot", "butcher") are identified as important features, which makes model predictions unreliable.

Frequency-importance correlation. We compute the Pearson correlation coefficients between word frequency and global word importance of VMASK-based models in Appendix F. The results show that they are not significantly correlated, which indicates that VMASK is not simply learning to select high-frequency words. Figure 2 further verifies this by ploting the expectation ($\mathbb{E}[q(R_{x_t}|\boldsymbol{x}_t)]$) of word masks from the LSTM-VMASK trained on Yelp and the word frequency from the same dataset. Here, we visualize the top 10 high-frequency words and top 10 important words based the expectation of word masks. The global importance scores of the sentiment words are over 0.8, even for some low-frequency words (e.g. "funnest", "craveable"), while that of the high-frequency words are all around 0.5, which means the VMASK-based models are less likely to focus on the irrelevant words to make predictions.

Task-specific important words. Figure 3 visualizes top 10 important words for the VMASK- and IBA-based models on three datasets via word clouds. We can see that the selected words by VMASK are consistent with the corresponding topic, such as "funnest", "awsome" for sentiment analysis, and "encyclopedia", "spaceport" for news
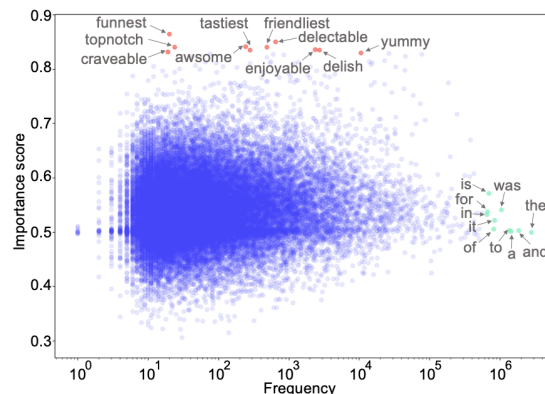

Figure 2: Scatter plot of word global importance and frequency (in log scale) of LSTM-VMASK on the Yelp dataset, where red dots represent top 10 important sentiment words and green dots represent top 10 high-frequency words.

classification, while IBA selects some irrelevant words (e.g. "undress", "slurred").

6 Conclusion

In this paper, we proposed an effective method, VMASK, learning global task-specific important features to improve both model interpretability and prediction accuracy. We tested VMASK with three different neural text classifiers on seven benchmark datasets, and assessed its effectiveness via both quantitative and qualitative evaluations.

Models	Texts	Prediction
CNN-base	Primary plot , primary direction , poor interpretation .	negative
CNN-Vmask	Primary plot , primary direction , poor interpretation .	negative
LSTM-base	John Leguizamo 's freak is one of the funniest one man shows I 've ever seen . I recommend it to anyone with a good sense of humor .	positive
LSTM-Vmask	John Leguizamo 's freak is one of the funniest one man shows I 've ever seen . I recommend it to anyone with a good sense of humor .	positive
BERT-base	Great story , great music . A heartwarming love story that ' s beautiful to watch and delightful to listen to . Too bad there is no soundtrack CD .	positive
BERT-Vmask	Great story , great music . A heartwarming love story that ' s beautiful to watch and delightful to listen to . Too bad there is no soundtrack CD .	positive

Table 5: Examples of the explanations generated by LIME for different models on the IMDB dataset, where the top three important words are highlighted. The color saturation indicates word attribution.

Models	Words
CNN-base	excellent, performances, brilliant
CNN-Vmask	excellent, fine, favorite
LSTM-base	plot, excellent, liked
LSTM-Vmask	excellent, favorite, brilliant
BERT-base	live, butcher, thrilling
BERT-Vmask	powerful, thrilling, outstanding

Table 6: Post-hoc global important words selected by SP-LIME for different models on the IMDB dataset.

Acknowledgments

(a) CNN-Vmask (b) CNN-IBA

(c) LSTM-Vmask (d) LSTM-IBA
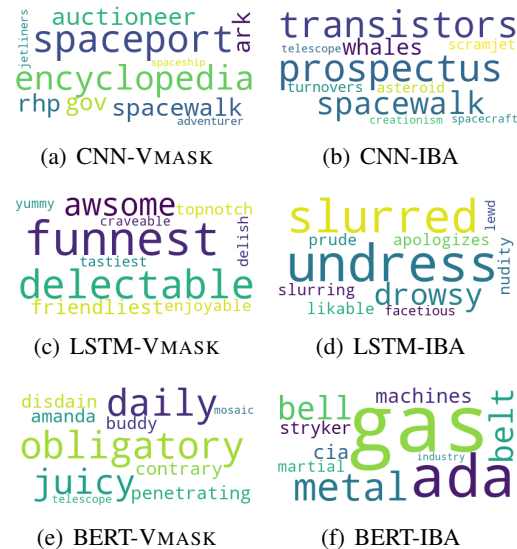
(e) BERT-Vmask (f) BERT-IBA

Figure 3: Word clouds of top 10 important words, where (a) is CNN-Vmask on the AG News dataset, (b) is CNN-IBA on the AG News dataset, (c) is LSTM-Vmask on the Yelp dataset, (d) is LSTM-IBA on the Yelp dataset, (e) is BERT-Vmask on the Subj dataset, and (f) is BERT-IBA on the Subj dataset.

References

Alexander A Alemi, Ian Fischer, Joshua V Dillon, and Kevin Murphy. 2016. Deep variational information bottleneck. *arXiv preprint arXiv:1612.00410*.

Seojin Bang, Pengtao Xie, Heewook Lee, Wei Wu, and Eric Xing. 2019. Explaining a black-box using deep variational information bottleneck approach. *arXiv preprint arXiv:1902.06918*.

Joost Bastings, Wilker Aziz, and Ivan Titov. 2019. Interpretable neural predictions with differentiable binary variables. *arXiv preprint arXiv:1905.08160*.

David M Blei, Alp Kucukelbir, and Jon D McAuliffe. 2017. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877.

Samuel Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 10–21.

Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. 2018. e-SNLI: Natural language inference with natural language explanations. In *Advances in Neural Information Processing Systems*, pages 9539–9549.

Nicola De Cao, Michael Schlichtkrull, Wilker Aziz, and Ivan Titov. 2020. How do decisions emerge across layers in neural models? interpretation with differentiable masking.

Hanjie Chen and Yangfeng Ji. 2019. Improving the interpretability of neural sentiment classifiers via data augmentation. *arXiv preprint arXiv:1909.04225*.

Jianbo Chen, Le Song, Martin J Wainwright, and Michael I Jordan. 2018. Learning to explain: An information-theoretic perspective on model interpretation. *arXiv preprint arXiv:1802.07814*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Chunning Du, Haifeng Sun, Jingyu Wang, Qi Qi, Jianxin Liao, Chun Wang, and Bing Ma. 2019a. Investigating capsule network and semantic feature on hyperplanes for text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 456–465.

Mengnan Du, Ninghao Liu, Fan Yang, and Xia Hu. 2019b. Learning credible deep neural networks with rationale regularization. *arXiv preprint arXiv:1908.05601*.

Gabriel Erion, Joseph D Janizek, Pascal Sturmfels, Scott Lundberg, and Su-In Lee. 2019. Learning explainable models using attribution priors. *arXiv preprint arXiv:1906.10670*.

Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. In *Advances in neural information processing systems*, pages 1019–1027.

Chaoyu Guan, Xiting Wang, Quanshi Zhang, Runjin Chen, Di He, and Xing Xie. 2019. Towards a deep and unified understanding of deep neural models in nlp. In *International Conference on Machine Learning*, pages 2454–2463.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Alon Jacovi and Yoav Goldberg. 2020. Towards faithfully interpretable nlp systems: How should we define and evaluate faithfulness? *arXiv preprint arXiv:2004.03685*.

Alon Jacovi, Oren Sar Shalom, and Yoav Goldberg. 2018. Understanding convolutional neural networks for text classification. *arXiv preprint arXiv:1809.08037*.

Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431, Valencia, Spain. Association for Computational Linguistics.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Yoon Kim, Sam Wiseman, and Alexander M Rush. 2018. A tutorial on deep latent variable models of natural language. *arXiv preprint arXiv:1812.06834*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Diederik P Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *ICLR*.

Igor Kononenko et al. 2010. An efficient explanation of individual classifications using game theory. *Journal of Machine Learning Research*, 11(Jan):1–18.

Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2016. Rationalizing neural predictions. *arXiv preprint arXiv:1606.04155*.

Xiang Lisa Li and Jason Eisner. 2019. Specializing word embeddings (for parsing) by information bottleneck. *arXiv preprint arXiv:1910.00163*.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pages 1–7. Association for Computational Linguistics.

Zachary C Lipton. 2016. The mythos of model interpretability. *arXiv preprint arXiv:1606.03490*.

Frederick Liu and Besim Avci. 2019. Incorporating priors with feature attribution on text classification. *arXiv preprint arXiv:1906.08286*.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Recurrent neural network for text classification with multi-task learning. *arXiv preprint arXiv:1605.05101*.

Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. In *Advances in neural information processing systems*, pages 4765–4774.

Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1*, pages 142–150. Association for Computational Linguistics.

Chris J Maddison, Andriy Mnih, and Yee Whye Teh. 2016. The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712*.

David Alvarez Melis and Tommi Jaakkola. 2018. Towards robust interpretability with self-explaining neural networks. In *Advances in Neural Information Processing Systems*, pages 7775–7784.

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.

W James Murdoch, Peter J Liu, and Bin Yu. 2018. Beyond word importance: Contextual decomposition to extract interactions from lstms. *arXiv preprint arXiv:1801.05453*.

Dong Nguyen. 2018. Comparing automatic and human evaluation of local explanations for text classification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1069–1078.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 115–124. Association for Computational Linguistics.

Gregory Plumb, Maruan Al-Shedivat, Eric Xing, and Ameet Talwalkar. 2019. Regularizing black-box models for improved interpretability. *arXiv preprint arXiv:1902.06787*.

Danilo Jimenez Rezende and Shakir Mohamed. 2015. Variational inference with normalizing flows. *arXiv preprint arXiv:1505.05770*.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144.

Laura Rieger, Chandan Singh, W James Murdoch, and Bin Yu. 2019. Interpretations are useful: penalizing explanations to align neural networks with prior knowledge. *arXiv preprint arXiv:1909.13584*.

Andrew Slavin Ross and Finale Doshi-Velez. 2018. Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. In *Thirty-second AAAI conference on artificial intelligence*.

Andrew Slavin Ross, Michael C Hughes, and Finale Doshi-Velez. 2017. Right for the right reasons: Training differentiable models by constraining their explanations. *arXiv preprint arXiv:1703.03717*.

Cynthia Rudin. 2019. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215.

Wojciech Samek, Alexander Binder, Grégoire Montavon, Sebastian Lapuschkin, and Klaus-Robert Müller. 2016. Evaluating the visualization of what a deep neural network has learned. *IEEE transactions on neural networks and learning systems*, 28(11):2660–2673.

Karl Schulz, Leon Sixt, Federico Tombari, and Tim Landgraf. 2020. Restricting the flow: Information bottlenecks for attribution. *arXiv preprint arXiv:2001.00396*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune bert for text classification? In *China National Conference on Chinese Computational Linguistics*, pages 194–206. Springer.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3319–3328. JMLR. org.

Naftali Tishby, Fernando C Pereira, and William Bialek. 2000. The information bottleneck method. *arXiv preprint physics/0004057*.

Naftali Tishby and Noga Zaslavsky. 2015. Deep learning and the information bottleneck principle. In *2015 IEEE Information Theory Workshop (ITW)*, pages 1–5. IEEE.

Jin Wang, Zhongyuan Wang, Dawei Zhang, and Jun Yan. 2017. Combining knowledge with deep convolutional neural networks for short text classification. In *IJCAI*, pages 2915–2921.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North*

American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 1480–1489, San Diego, California. Association for Computational Linguistics.

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.

Ye Zhang, Iain Marshall, and Byron C Wallace. 2016. Rationale-augmented convolutional neural networks for text classification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing. Conference on Empirical Methods in Natural Language Processing*, volume 2016, page 795. NIH Public Access.

Chunting Zhou, Chonglin Sun, Zhiyuan Liu, and Francis Lau. 2015. A c-lstm neural network for text classification. *arXiv preprint arXiv:1511.08630*.

A Proof of the Two Bounds for the Information Bottleneck

The following derivation is similar to the variational information bottleneck, where the difference is that our starting point is the approximation distribution $q(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z})$ instead of the true distribution $p(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z})$.

The lower bound for $I(\boldsymbol{Z}; \boldsymbol{Y})$.

$$
\begin{aligned}
I(\boldsymbol{Z}, \boldsymbol{Y}) &= \sum_{\boldsymbol{y}, \boldsymbol{z}} q(\boldsymbol{y}, \boldsymbol{z}) \log \frac{q(\boldsymbol{y}, \boldsymbol{z})}{q(\boldsymbol{y}) q(\boldsymbol{z})} \\
&= \sum_{\boldsymbol{y}, \boldsymbol{z}} q(\boldsymbol{y}, \boldsymbol{z}) \log \frac{q(\boldsymbol{y}|\boldsymbol{z})}{q(\boldsymbol{y})} \\
&= \sum_{\boldsymbol{y}, \boldsymbol{z}} q(\boldsymbol{y}, \boldsymbol{z}) \log q(\boldsymbol{y}|\boldsymbol{z}) \\
&\quad + H_q(\boldsymbol{Y}),
\end{aligned} \tag{10}
$$

where $H_q(\cdot)$ represents entropy. Now, if we replace $\log q(\boldsymbol{y}|\boldsymbol{z})$ with the conditional probability derived from the true distribution $\log p(\boldsymbol{y}|\boldsymbol{z})$, we have

$$
\begin{aligned}
&\sum_{\boldsymbol{y}, \boldsymbol{z}} q(\boldsymbol{y}, \boldsymbol{z}) \log q(\boldsymbol{y}|\boldsymbol{z}) \\
&= \sum_{\boldsymbol{y}, \boldsymbol{z}} q(\boldsymbol{y}, \boldsymbol{z}) \log \frac{q(\boldsymbol{y}|\boldsymbol{z}) p(\boldsymbol{y}|\boldsymbol{z})}{p(\boldsymbol{y}|\boldsymbol{z})} \\
&= \sum_{\boldsymbol{y}, \boldsymbol{z}} q(\boldsymbol{y}, \boldsymbol{z}) \log p(\boldsymbol{y}|\boldsymbol{z}) + \mathrm{KL}[q(\boldsymbol{y}|\boldsymbol{z}) \| p(\boldsymbol{y}|\boldsymbol{z})] \\
&\geq \sum_{\boldsymbol{y}, \boldsymbol{z}} q(\boldsymbol{y}, \boldsymbol{z}) \log p(\boldsymbol{y}|\boldsymbol{z}),
\end{aligned} \tag{11}
$$

where $\mathrm{KL}[\cdot \| \cdot]$ denotes Kullback-Leibler divergence. Therefore, we can obtain a lower bound of the mutual information

$$
\begin{aligned}
I(\boldsymbol{Z}, \boldsymbol{Y}) &\geq \sum_{\boldsymbol{y}, \boldsymbol{z}} q(\boldsymbol{y}, \boldsymbol{z}) \log p(\boldsymbol{y}|\boldsymbol{z}) + H_q(\boldsymbol{Y}) \\
&= \sum_{\boldsymbol{y}, \boldsymbol{z}, \boldsymbol{x}} q(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) \log p(\boldsymbol{y}|\boldsymbol{z}) + H_q(\boldsymbol{Y}) \\
&= \sum_{\boldsymbol{y}, \boldsymbol{z}, \boldsymbol{x}} q(\boldsymbol{x}, \boldsymbol{y}) q(\boldsymbol{z}|\boldsymbol{x}) \log p(\boldsymbol{y}|\boldsymbol{z}) + H_q(\boldsymbol{Y}),
\end{aligned} \tag{12}
$$

where the last step uses $q(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) = q(\boldsymbol{x}) q(\boldsymbol{y}|\boldsymbol{x}) q(\boldsymbol{z}|\boldsymbol{x})$, which is a factorization based on the conditional dependency [4].

Since $q(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z})$ is the approximation defined by ourselves, given a specific observation $(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})$, the empirical distribution $q(\boldsymbol{X}^{(i)}, \boldsymbol{Y}^{(i)})$

---

[4] $\boldsymbol{Y} \leftrightarrow \boldsymbol{X} \leftrightarrow \boldsymbol{Z}$: $\boldsymbol{Y}$ and $\boldsymbol{Z}$ are independent given $\boldsymbol{X}$.

is simply defined as a multiplication of two Delta functions

$$
q(\boldsymbol{X}^{(i)} = \boldsymbol{x}^{(i)}, \boldsymbol{Y}^{(i)} = \boldsymbol{y}^{(i)}) = \delta_{\boldsymbol{x}^{(i)}}(\boldsymbol{x}) \cdot \delta_{\boldsymbol{y}^{(i)}}(\boldsymbol{y}). \tag{13}
$$

Then, Equation 12 with $\boldsymbol{X}^{(i)}$ and $\boldsymbol{Y}^{(i)}$ can be further simplified as

$$
\begin{aligned}
I(\boldsymbol{Z}; \boldsymbol{Y}^{(i)}) &\geq \sum_{\boldsymbol{z}} q(\boldsymbol{z}|\boldsymbol{x}^{(i)}) \log p(\boldsymbol{y}^{(i)}|\boldsymbol{z}) \\
&= \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x}^{(i)})}[\log p(\boldsymbol{y}^{(i)}|\boldsymbol{z})]
\end{aligned} \tag{14}
$$

The upper bound for $I(\boldsymbol{Z}; \boldsymbol{X})$.

$$
\begin{aligned}
I(\boldsymbol{Z}, \boldsymbol{X}) &= \sum_{\boldsymbol{x}, \boldsymbol{z}} q(\boldsymbol{x}, \boldsymbol{z}) \log \frac{q(\boldsymbol{x}, \boldsymbol{z})}{q(\boldsymbol{x}) q(\boldsymbol{z})} \\
&= \sum_{\boldsymbol{x}, \boldsymbol{z}} q(\boldsymbol{x}, \boldsymbol{z}) \log \frac{q(\boldsymbol{z}|\boldsymbol{x})}{q(\boldsymbol{z})} \\
&= \sum_{\boldsymbol{x}, \boldsymbol{z}} q(\boldsymbol{x}, \boldsymbol{z}) \log q(\boldsymbol{z}|\boldsymbol{x}) \\
&\quad - \sum_{\boldsymbol{x}, \boldsymbol{z}} q(\boldsymbol{x}, \boldsymbol{z}) \log q(\boldsymbol{z})
\end{aligned} \tag{15}
$$

By replacing $q(\boldsymbol{z})$ with a prior distribution of $\boldsymbol{z}$, $p_0(\boldsymbol{z})$, we have

$$
\sum_{\boldsymbol{x}, \boldsymbol{z}} q(\boldsymbol{x}, \boldsymbol{z}) \log q(\boldsymbol{z}) \geq \sum_{\boldsymbol{x}, \boldsymbol{z}} q(\boldsymbol{x}, \boldsymbol{z}) \log p_0(\boldsymbol{z}). \tag{16}
$$

Then we can obtain an upper bound of the mutual information

$$
\begin{aligned}
I(\boldsymbol{Z}, \boldsymbol{X}) &\leq \sum_{\boldsymbol{x}, \boldsymbol{z}} q(\boldsymbol{x}, \boldsymbol{z}) \log q(\boldsymbol{z}|\boldsymbol{x}) \\
&\quad - \sum_{\boldsymbol{x}, \boldsymbol{z}} q(\boldsymbol{x}, \boldsymbol{z}) \log p_0(\boldsymbol{z}) \\
&= \sum_{\boldsymbol{x}} q(\boldsymbol{x}) \mathrm{KL}[q(\boldsymbol{z}|\boldsymbol{x}) \| p_0(\boldsymbol{z})] \\
&= \mathbb{E}_{q(\boldsymbol{x})}[\mathrm{KL}[q(\boldsymbol{z}|\boldsymbol{x}) \| p_0(\boldsymbol{z})]] \\
&= \mathrm{KL}[q(\boldsymbol{z}|\boldsymbol{x}^{(i)}) \| p_0(\boldsymbol{z})].
\end{aligned} \tag{17}
$$

B Sampling with Gumbel-softmax Trick

Specifically, as r_{x_t} has the probability $1 - p_{x_t}$ and p_{x_t} to take 0 or 1 respectively, we draw samples from a Gumbel(0, 1) distribution for each category $c \in \{0, 1\}$:

$$
s_{x_t}^{(c)} = -\log(-\log u), \ u \sim \mathrm{Uniform}(0, 1), \tag{18}
$$

and then apply a temperature-dependent softmax over the two categories with each dimension calculated as

$$
rs_{x_t}^{(c)} = \frac{\exp((w_{x_t} + s_{x_t}^{(c)})/\tau)}{\sum_c \exp((w_{x_t} + s_{x_t}^{(c)})/\tau)}, \tag{19}
$$

Datasets	$vocab$	$threshold$	$length$
IMDB	29571	5	250
SST-1	17838	0	50
SST-2	16190	0	50
Yelp	45674	10	150
AG News	21838	5	50
TREC	8026	0	15
Subj	9965	1	25

Table 7: Pre-processing details on the datasets. $vocab$: vocab size; $threshold$: low-frequency threshold; $length$: mini-batch sentence length.

where $w_{x_t} = \log(cp_{x_t} + (1-c)(1-p_{x_t}))$, and τ is a hyperparameter called the softmax temperature.

C Supplement of Experiment Setup

Data pre-processing. We clean up the text by converting all characters to lowercase, removing extra whitespaces and special characters. We tokenize texts and remove low-frequency words to build vocab. We truncate or pad sentences to the same length for mini-batch during training. Table 7 shows pre-processing details on the datasets.

Model configurations. We implement the models in PyTorch 3.6. The number of parameters in the CNN, LSTM and BERT are 2652305, 2632405, 109486085 respectively. We tune hyperparameters manually for each model to achieve the best prediction accuracy. We experiment with different kernel numbers ($\{100, \cdots, 500\}$) for the CNN model, hidden states ($\{100, \cdots, 500\}$) for the LSTM model, and other hyperparameters, such as learning rate $lr \in \{1e-4, 1e-3, \cdots, 1\}$, clipping norm $clip \in \{1e-3, 1e-2, \cdots, 1, 5, 10\}$.

Implementation of L2X and IBA.

- The explanation framework of L2X (Chen et al., 2018) is a neural network which learns to generate importance scores $w = [w_1, w_2, \cdots, w_T]$ for input features $x = [x_1, x_2, \cdots, x_T]$. The neural network is optimized by maximizing the mutual information between the selected important features and the model prediction, i.e. $I(x_S; y)$, where x_S contains a subset of features from x. In our experiments, we adopt a single-layer feedforward neural network as the interpreter to generate importance scores for an input text, and multiply each word embedding with its

importance score, $x' = w \odot x$. The weighted word embedding matrix x' is sent to the rest of the model to produce an output y'. We optimize the interpreter network with the original model by minimizing the cross-entropy loss between the final output and the ground-truth label, $\mathcal{L}_{ce}(y_t; y')$.

- We adopt the Readout Bottleneck of IBA which utilizes a neural network to predict mask values $\boldsymbol{\lambda} = [\lambda_1, \lambda_2, \cdots, \lambda_T]$, where $\lambda_t \in [0, 1]$. The information of a feature x_t is restricted by adding noise, i.e. $z_t = \lambda_t x_t + (1 - \lambda_t)\epsilon_t$, where $\epsilon_t \sim \mathcal{N}(\mu_{x_t}, \sigma^2_{x_t})$. And z is learned by optimizing the objective function Equation 2. By assuming the variational approximation $q(z)$ as a Gaussian distribution, the mutual information can be calculated explicitly (Schulz et al., 2020) . We still use a single-layer feedforward neural network as the Readout Bottleneck to generate continuous mask valuues $\boldsymbol{\lambda}$ and construct z for model to make predictions. The Readout Bottleneck is trained jointly with the original model by minimizing the sum of the cross-entropy loss $\mathcal{L}_{ce}(y_t; y)$ and an upper bound $\mathcal{L}_I = \mathbb{E}_x[\text{KL}[p(z|x)\|q(z)]]$ of the mutual information $I(\boldsymbol{Z}; \boldsymbol{X})$. See Schulz et al. (2020) for the proof of the upper bound.

D Validation Performance and Average Runtime

The corresponding validation accuracy for each reported test accuracy is in Table 8. The average runtime for each approach on each dataset is recorded in Table 9. All experiments were performed on a single NVidia GTX 1080 GPU.

E Results of Prediction Accuracy Reported in Previous Papers

Table 10 shows some results of prediction accuracy of base models reported in previous papers.

F Pearson Correlation between Word Frequency and VMASK Expectation Values

Table 11 shows the Pearson correlation coefficients between word frequency and global word importance of VMASK-based models.

Models	Methods	IMDB	SST-1	SST-2	Yelp	AG News	TREC	Subj
CNN	CNN-base	88.82	45.09	84.47	93.70	91.77	92.22	88.96
	CNN-$l2$	88.86	45.02	84.50	94.15	91.78	91.64	88.79
	CNN-L2X	79.76	36.85	80.12	84.10	85.28	62.50	82.54
	CNN-IBA	87.56	41.52	84.79	93.42	91.73	91.40	89.02
	CNN-VMASK	88.98	44.78	85.56	94.34	91.78	92.60	91.60
LSTM	LSTM-base	88.80	41.87	85.21	94.24	91.78	91.15	90.50
	LSTM-$l2$	88.82	41.90	85.22	94.21	91.80	91.17	90.46
	LSTM-L2X	67.72	37.00	77.49	76.50	78.84	47.28	82.07
	LSTM-IBA	88.38	42.48	85.30	94.68	91.93	86.35	90.52
	LSTM-VMASK	90.06	44.96	85.67	94.46	92.28	92.48	92.80
BERT	BERT-base	85.16	50.77	92.66	96.78	93.88	95.13	95.30
	BERT-$l2$	85.05	49.32	92.65	96.80	93.69	95.36	94.87
	BERT-L2X	65.34	38.17	72.68	83.00	80.67	91.32	84.59
	BERT-IBA	85.60	51.02	91.74	96.50	93.24	95.28	96.10
	BERT-VMASK	86.00	51.96	92.74	96.87	94.88	95.49	96.90

Table 8: Validation accuracy (%) for each reported test accuracy.

Models	Methods	IMDB	SST-1	SST-2	Yelp	AG News	TREC	Subj
CNN	CNN-base	8.93	0.75	0.66	31.25	12.47	2.01	0.69
	CNN-$l2$	8.96	0.78	0.65	31.27	12.48	2.00	0.68
	CNN-L2X	11.23	0.92	0.87	33.00	13.62	2.10	0.73
	CNN-IBA	14.21	2.58	1.58	47.80	18.00	2.67	2.15
	CNN-VMASK	12.29	1.17	1.21	38.59	16.16	2.58	1.12
LSTM	LSTM-base	3.81	1.00	1.65	32.12	12.84	1.87	1.34
	LSTM-$l2$	3.83	1.01	1.67	32.13	12.84	1.89	1.34
	LSTM-L2X	5.23	1.02	1.70	35.60	13.20	1.92	1.36
	LSTM-IBA	8.39	1.20	1.85	38.10	15.54	2.01	1.90
	LSTM-VMASK	5.07	1.07	1.73	36.70	13.39	1.96	1.47
BERT	BERT-base	716.87	111.66	51.64	1846.12	1202.09	49.56	73.46
	BERT-$l2$	719.23	111.66	51.65	1846.12	1202.10	50.20	74.49
	BERT-L2X	725.65	110.25	51.70	1845.30	1203.00	52.10	77.00
	BERT-IBA	791.78	115.30	54.12	1885.02	1207.23	60.35	90.00
	BERT-VMASK	774.35	110.79	53.64	1879.36	1205.35	57.48	85.85

Table 9: Average runtime (s/epoch) for each approach on each dataset.

Models	Methods	IMDB	SST-1	SST-2	Yelp	AG News	TREC	Subj
CNN-base	Kim (2014)	-	45.5	86.8	-	-	92.8	93.0
	Wang et al. (2017)	-	-	-	-	86.11	89.33	-
LSTM-base	Du et al. (2019a)	-	45.3	80.6	-	-	86.8	89.3
	Zhou et al. (2015)	-	46.6	86.6	-	-	-	-
	Zhang et al. (2015)	-	-	-	94.74	86.06	-	-
	Liu et al. (2016)	88.5	45.9	85.8	-	-	-	-
BERT-base	Devlin et al. (2018)	-	-	93.5	-	-	-	-
	Sun et al. (2019)	94.6	-	-	97.72	94.75	-	-

Table 10: Results of prediction accuracy (%) collected from previous papers.

Datasets	CNN	LSTM	BERT
IMDB	-0.045	-0.018	-0.016
SST-1	-0.079	-0.070	0.017
SST-2	-0.067	-0.068	-0.023
Yelp	-0.014	0.004	-0.038
AG News	-0.015	-0.040	-0.030
TREC	-0.018	-0.026	0.028
Subj	-0.010	-0.003	0.008

Table 11: Pearson correlation coefficients of VMASK-based models on the seven datasets.