From Random to Supervised: A Novel Dropout Mechanism Integrated with Global Information

Hengru Xu[1,†]   **Shen Li[2,†]**   **Renfen Hu[3]**   **Si Li[1]**   **Sheng Gao[1,‡]**

[1]{xuhengru, lisi, gaosheng}@bupt.edu.cn
[2]shen@deeplycurious.ai
[3]irishere@mail.bnu.edu.cn

[1] SICE, Beijing University of Post and Telecommunication
[2] Deeplycurious.ai
[3] Institute of Chinese Information Processing, Beijing Normal University

Abstract

Dropout is used to avoid overfitting by randomly dropping units from the neural networks during training. Inspired by dropout, this paper presents GI-Dropout, a novel dropout method integrating with global information to improve neural networks for text classification. Unlike the traditional dropout method in which the units are dropped randomly according to the same probability, we aim to use explicit instructions based on global information of the dataset to guide the training process. With GI-Dropout, the model is supposed to pay more attention to inapparent features or patterns. Experiments demonstrate the effectiveness of the dropout with global information on seven text classification tasks, including sentiment analysis and topic classification.

1 Introduction

Recently, neural networks have achieved remarkable results in natural language processing (NLP). Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) are two popular types of neural network architectures and both of them are widely applied to various NLP tasks. CNN is known for its strong ability in extracting position-invariant features and RNN is highlighted in modeling sequences (Yin et al., 2017). In sentence classification tasks, models based on CNN or RNN aim to represent sentences as appropriate embeddings, which are supposed to encode semantic features for the classification.

However, with the consideration of computational complexity and spatial limitation, neural networks are often trained via mini-batch in which global information is gathered implicitly rather

---

† Hengru Xu and Shen Li contributed equally to this work.
‡ Corresponding author.

than explicitly. To facilitate the learning process, Li et al. (2017) extract global semantic features from the training dataset, and encode them into CNN filters with a novel initialization mechanism. This approach gains significant improvements in sentiment analysis and topic classification tasks.

Unlike most of machine learning methods, the advantage of neural networks is extracting features with less need of feature engineering. In general, the stronger ability of a model to learn features automatically, the better performance it will achieve. However, during the training process, neural networks tend to focus on some distinctive words or phrases but ignore other noteworthy patterns, which may result in overfitting, especially in a small dataset. To avoid this problem, dropout is proposed (Hinton et al., 2012; Srivastava et al., 2014). The key idea of dropout is to randomly drop units from the neural network during training and use a smaller weight of these units in the test.

Inspired by the above works, we propose a novel dropout method guided by global information (GI-Dropout). In our method, we force the model to pay more attention to features that are inapparent or with low frequency by dropping words that are prominent and easy to learn. Unlike the traditional dropout method where neurons are dropped randomly with the same probability, we encode global information into dropout. Specifically, we drop words based on their importance which are calculated from training data via a novel Naïve Bayes (NB) weighting technique.

With this dropout method, neural networks tend to extract not only the obvious features but also the unobvious features which are also helpful for the classification. By integrating our method into a classic CNN model for text classification (Kim, 2014) and a novel self-attentive RNN (Lin et al., 2017), we observe significant improvements in

various benchmarks.[1] The advantages of our approach are as follows:

1. Global information is directly obtained from the training data without any external resources;

2. GI-Dropout is simple but effective, and could be easily applied to other DNN models;

3. The computation brought by our method is relatively small, resulting in little additional training cost.

2 Related Work

Recently, neural networks dominate the state-of-the-art results on a wide range of NLP tasks. For text classification, Kim (2014) proposes a classical one-layer CNN which is very efficient for feature extraction, and it is considered as a strong baseline for various sentiment and topic classification tasks. Following this work, Yin and Schütze (2015) introduce multichannel variable-size convolution, and Zhang et al. (2016b) exploit different pre-trained word embeddings (e.g. word2vec and GloVe). Zhang and Wallace (2017) improve the CNN model by optimizing hyper-parameters and provide a detailed sensitivity analysis.

RNNs also achieve comparable performance in this area. Tang et al. (2015) show that gated RNN performs well on document-level sentiment classification. Lin et al. (2017) propose a enhanced model to extract an interpretable sentence embedding by introducing self-attention mechanism and yields a significant performance gain compared with other sentence embedding methods.

Yin et al. (2017) make a systematic comparison of CNNs and RNNs, showing that both of the networks can provide complementary information for text classification tasks, while which architecture performs better depends on how important it is to semantically understand the global/long-range semantics.

To improve the semantic understanding abilities of the models, some works aim to encode prior knowledge into the networks. For example, Hu et al. (2016) present a framework that encapsulates the logical structured knowledge into a neural network. Li et al. (2017) encode global semantic features into the convolutional filters instead of

initializing them randomly, which helps the filters focus on learning useful n-grams.

Another effective method to facilitate learning process is to exploit dropout mechanism. Apparently, if a model pays too much attention to a few distinct patterns, it can easily give rise to an overfitting, especially in a small dataset. Hinton et al. (2012) introduce Binary (regular) Dropout, showing that it can prevent co-adaptation of neurons by randomly dropping units from the neural networks during training, so as to reduce overfitting. Later Srivastava et al. (2014) show that multiplying outputs of the neurons by a random variable drawn from Gaussian distributions works as well, or perhaps better than regular dropout. Ba and Frey (2013) present standout, an adaptive dropout method, where each variable's dropout probability is calculated by a binary belief network, which can be trained jointly with the neural networks. Kingma et al. (2015) introduce variational dropout, a generalization of Gaussian dropout where the dropout rates are also learned during training.

The existing dropout methods are often based on mathematics or learned jointly with the downstream task, where global information is not explicitly utilized. Different from previous works, we focus on how to utilize global information to help model training via dropout. As depicted in Figure 3, GI-Dropout is introduced at the beginning of the baseline models, which is different from prior dropout methods which aim at controlling units in the networks rather than input words in the texts.

In this work, we use the global information to guide dropout method by dropping words based on their importance. Hence, neural networks are able to extract not only the obvious features but also the unobvious features which are also helpful for the classification.

3 Our method

The intuition behind our method is straightforward. Since neural networks aim to capture semantic features and classify sentences by the features, we encourage models to share more attention to unobvious features by dropping words according to their importance. Some features are so distinctive that model can learn them easily. However, a sentence may have more than one feature that can contribute to class prediction. For

---

[1]We release source codes at https://gitlab.com/xusong19 960424/global_cnn.

Figure 1: Top 30 key words of each class in Customer Review dataset

Figure 2: GI-Dropout probability and rank in SST-1 with $\beta = 0.95$.

instance, in "The story is sad and very boring", "boring" is of strong polarity and indicates negative emotion. Neural networks may not be sensitive to other features like "sad" which is also helpful for the sentiment classification, due to the very strong impact of "boring". In GI-Dropout, a word of higher importance score has greater possibility to be dropped. Thus, models are forced to learn unobvious features and will achieve better performance in prediction.

3.1 Importance Score

Firstly, we compute an importance score for each word. Intuitively, word "unique" is much more important than "movie" for determining polarities of reviews. Naïve Bayes (NB) weighting is an effective technique for determining the importance of words (Martineau and Finin, 2009; Wang and Manning, 2012; Li et al., 2017). The NB weight r of word w in class c is calculated as follows:

$$r_c^w = \frac{(n_c^w + \alpha)/\|n_c\|_1}{(n_{\tilde{c}}^w + \alpha)/\|n_{\tilde{c}}\|_1} \quad (1)$$

where n_c^w is the count of word w in class c, $n_{\tilde{c}}^w$ is the count of word w in the other classes, $\|n_c\|_1$ is the count of all the word occurrences in class c, $\|n_{\tilde{c}}\|_1$ is the count of all the word occurrences in the other classes, α is a smoothing parameter and is set as 1 in this paper.

To avoid low-frequency words being recognized as important words, we propose an improved NB weighting method based on (1):

$$r_c^w = \frac{(n_c^w + \alpha)/\|n_c\|_1}{(n_{\tilde{c}}^w + \alpha)/\|n_{\tilde{c}}\|_1} \times \log_\beta n_c^w \quad (2)$$

where $\log_\beta n_c^w$ is introduced as a frequency factor. The base β is a hyperparameter.

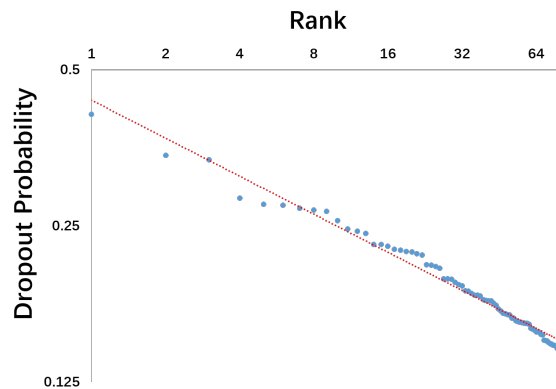For positive class in movie review dataset (MR), the scores of words like "unique" and "warm"

should be large since they appear much more frequently in positive texts than in negative texts. As for neutral words like "the" and "movie", their scores should be small. For a word w, we select the max score of it as its importance score:

$$r^w = max(r_{c_0}^w, r_{c_1}^w, ..., r_{c_n}^w) \quad (3)$$

In Figure 1, we show top 30 key words of each class in customer review dataset (CR). We aim to drop these key words with higher probabilities and encourage the model to pay more attention to other unobvious features.

3.2 Dropout Probability

As shown in 3.1, we compute words importance scores with the whole training data. It is a simple yet effective way to represent the global information. After obtaining the scores, we compress them into $[0, 1)$. The GI-Dropout probability of word w is:

$$p(r) = \frac{e^r - 1}{e^r + 1} \quad (4)$$

where r is the importance score of w calculated via (2). A word would not be ignored when its probability is 0.

The β in (2) is a key parameter. As shown in Figure 2, after tuning β, the GI-Dropout probability of a word and its probability rank follow Zipf's Law. Zipf (1935) states that given a sample of words, the frequency of any word is inversely proportional to its rank in the frequency table. Replacing the frequency with GI-Dropout probability, we can get a variant of Zipf's Law. The experiments will show that setting β to this value in SST-1 is not a coincidence.
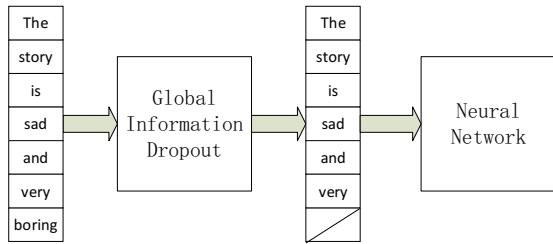
Figure 3: GI-Dropout. In this case, the word embedding of "boring" is dropped and set to zero vector while "sad" is not.

3.3 GI-Dropout Method

As illustrated in Figure 3, we implement a GI-Dropout layer before the neural network. Models without our dropout method can be viewed as the special case in which all the words are not dropped in GI-Dropout layer, i.e. dropout probabilities of all words are 0.

In this paper, every word in training data has a score to measure its importance via the novel NB weighting method, as well as a dropout probability calculated by the proposed scale function. During training, the words will be dropped according to their dropout probabilities.

The way to implement our dropout method is very straightforward. In embedding layer, we get the word embedding e_i of word w_i after looking it up in the embedding table. After that, this word can be dropped according to its GI-Dropout possibility. For word w_i, we set the e_i to zero vector if it needs to be dropped. Through this method, the neural network will not learn features from words whose embeddings are zero vectors. It is worth noted that the dropout probabilities of words differ from each other, which is different from the traditional dropout method where all the neurons are dropped according to the same probability. The dropout probabilities which are encoded with global information, guide the model to share attention to unobvious patterns.

In traditional dropout method, all the neurons are used in testing, but their weights are scaled down by a factor p (same with p in training) since a part of units emit nothing to the next layer during training. While in our method, during evaluation and testing, dropout probabilities of all the words are set to 0 so as to use all the patterns, and scaling is not needed.

Dataset	c	l	N	V	Test
MR	2	20	10662	18765	CV
SST-1	5	18	11855	17836	2210
SST-2	2	19	9613	16185	1821
Subj	2	23	10000	21323	CV
TREC	6	10	5952	9592	500
CR	2	19	3775	5340	CV
MPQA	2	3	10606	6246	CV

Table 1: Datasets summary. c: Number of target classes. l: Average sentence length. N: Dataset size. V: Vocabulary size. Test: Test set size (CV means there is no standard train/test split and thus 10-fold CV is used).

4 Experiments

CNN-non-static proposed by Kim (2014) is considered as a very strong baseline in sentence classification. Self-attentive RNN proposed by Lin et al. (2017) also achieves outstanding performance in many sentence classification tasks. We adopt these two models to evaluate GI-dropout.

4.1 Datasets

Following (Kim, 2014), we evaluate the performance of the proposed approach on various datasets. We use the same seven datasets with (Kim, 2014), including both sentiment analysis and topic classification tasks:

MR: Movie reviews sentiment datasets[2].

SST-1: Stanford Sentiment Treebank with 5 sentiment labels (Socher et al., 2013)[3]. The data consists of phrases-level and sentence-level instances. To keep same with (Kim, 2014), we train the model on both phrases and sentences but only test on sentences.

SST-2: SST-1 data with binary labels.

Subj: Subjective or objective classification dataset (Pang and Lee, 2005).

TREC: 6-class question classification dataset (Li and Roth, 2002) [4].

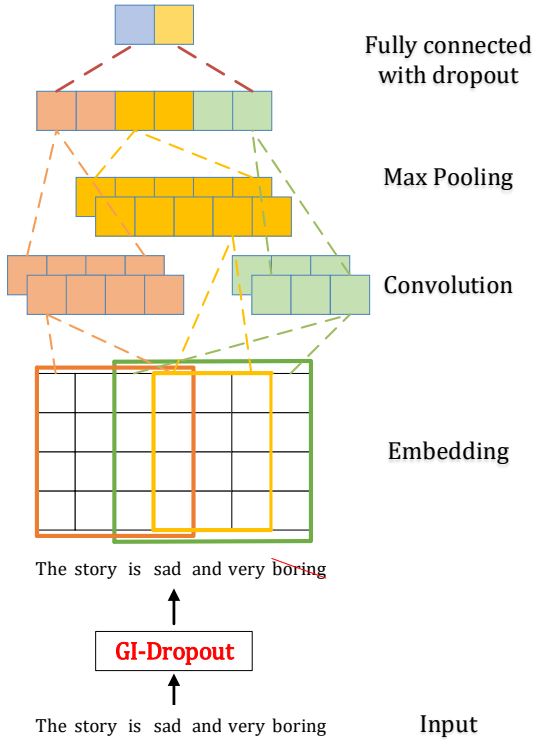**CR**: Customer products review dataset (Hu and Liu, 2004) [5].

Figure 4: CNN architectures with GI-Dropout.

MPQA: Opinion polarity detection dataset (Wiebe et al., 2005).

The statistics of the datasets can be seen in Table 1.

4.2 CNN Model

CNNs use filters to capture semantic features of n-grams. After that, max-pooling is introduced to force the network to capture the most useful local features produced by convolutional layers (Collobert et al., 2011). A simple CNN model in (Kim, 2014) consists of the embedding layer, one convolution and pooling layer, and one fully connected layer. Four model variations are provided in (Kim, 2014), and we choose the CNN-non-static model as our baseline. The hyperparameters of the CNN are described in Table 2. The architecture of the model integrated with GI-Dropout is shown in Figure 4.

4.3 Self-attentive RNN Model

Long Short-Term Memory (LSTM) is a specific recurrent neural network (RNN) architecture which is good at modeling temporal sequences and

Parameters	Values
Word embeddings	GoogleNews-negative300 [6]
Fine-tune	Yes
Convolution	1-d
Filter size	[3, 4, 5]
Filter numbers	300
Activation function	ReLU
Pooling method	max-over-time
MLP dropout rate	0.5

Table 2: CNN configuration.

can capture long-range dependencies (Sak et al., 2014). Attention mechanism, first proposed in (Bahdanau et al., 2014), has become an integral part of sequence modeling. The self-attentive RNN proposed by Lin et al. (2017) consists of a bidirectional LSTM (biLSTM) and the self-attention mechanism. Self-attention mechanism is used to replace the max pooling or averaging step after the biLSTM. Multiple hops of attention are performed to extract semantic features in different aspects of the sentence.

In brief, suppose we have a sentence of n tokens, and let the hidden unit number for each unidirectional LSTM be u. After the biLSTM layer, we can get H, which have the size of n-by-$2u$. The attention mechanism takes the whole LSTM hidden states H as input, and outputs a vector of weights a,

$$a = softmax(w_{s2} \tanh(W_{s1} H^T)) \quad (5)$$

where W_{s1} is a weight matrix with a shape of d_a-by-$2u$, and W_{s2} is a vector of parameters with size d_a which is a hyperparameter.

To extract r different aspects of the sentence, Lin et al. (2017) present multiple hops of attention, i.e. extend the w_{s2} into a r-by-d_a matrix and note it as W_{s2}. In the end, the annotation vector a becomes annotation matrix A.

$$A = softmax(W_{s2} \tanh(W_{s1} H^T)) \quad (6)$$

The sentence embedding is:

$$M = AH \quad (7)$$

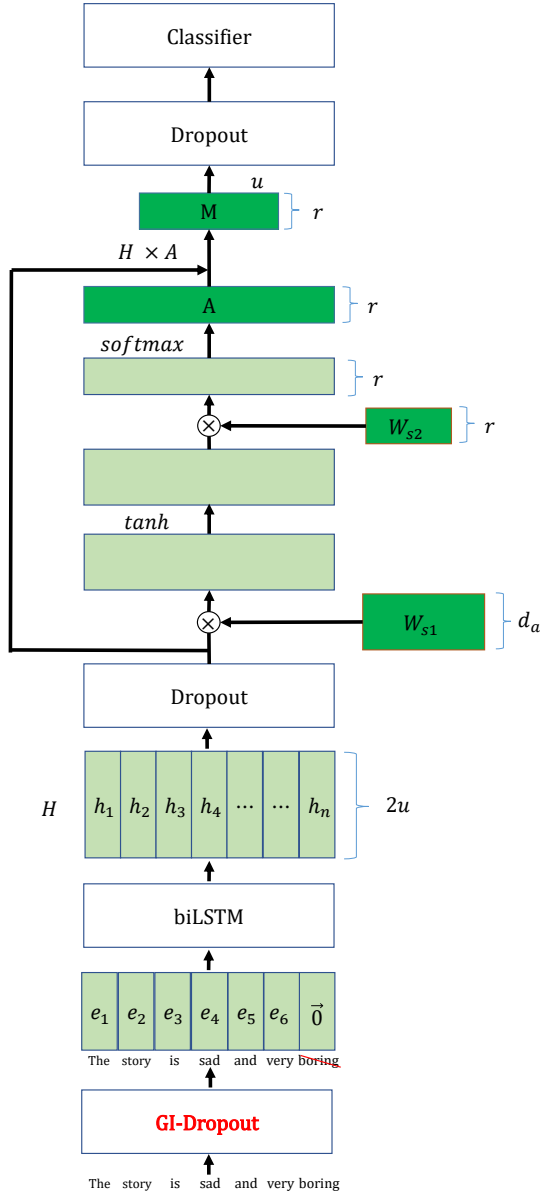Then the paper uses two layer 2-layer MLP with ReLU activation function to predict the label of

---

[6] A widely used publicly available word2vec 300-dimension vectors which were trained on 100 billion words from Google News in (Mikolov et al., 2013) way.

Figure 5: Self-attentive RNN architectures.

Parameters	Values
Word embeddings	Glove-300 [7]
Fine-tune	Yes
biLSTM hidden units	300
d_a	350
r	4
MLP Activation	ReLU
MLP dropout rate	0.5

Table 3: Self-attentive RNN configuration.

of texts. We also use a 2-layer ReLU output MLP with 2000 hidden units. During training we use a 0.5 dropout rate on the MLP. The hyperparameters are described in Table 3.

4.4 Experiment Settings

We apply our method to two baseline models. For fair comparison, we use the same hyperparameters settings with two baselines for training and testing. For datasets that do not have test sets, we split them for cross-validation with fixed random seeds. We train all the models using early stopping and set timedelay to 10.

4.5 Effectiveness of GI-Dropout

Results on 7 datasets are listed in Table 4. Experiments show that the models with GI-Dropout outperform both CNN and self-attentive RNN baselines by a significant margin.

To test whether global information makes key contribution, we conduct another experiment in which all words are dropped according to the same probability at the GI-Dropout layer. Grid search method is used to find the best result which is listed in "Dropout-same-prob" row.

The one-layer CNN provides a very strong baseline. The first line is the result of CNN-non-static model in (Kim, 2014). We reproduce the experiment results in "CNN-baseline" row.

Table 4 shows that by simply dropping all the words according to the same probability, the model gains slight improvements against CNN baseline on all the datasets except in MPQA. Similarly, it achieves improvements compared with RNN baseline on most datasets.

the sentence. Besides, a penalization term is introduced to encourage the diversity of summation weight vectors across different hops of attention.

Since Lin et al. (2017) do not provide source codes, we reproduce the model and integrate dropout layers into the model as shown in Figure 5. We perform a grid search to get the best baseline hyperparameters with which the model can achieve the state-of-the-art accuracy in most of the datasets. This model uses a bidirectional LSTM with 300 dimensions of hidden states in each direction. In self attention part, d_a is 350 and the coefficient of the penalization term is 1. r is set to 4 considering the size of datasets and the length

---

[7] A widely used publicly available 300-dimension word embeddings (Pennington et al., 2014).

578

Model	MR	SST-1	SST-2	Subj	TREC	CR	MPQA
CNN-non-static	81.5	48.0	87.2	93.4	93.6	84.3	89.5
CNN-reproduce	81.4	47.8	87.5	93.0	92.4	84.3	89.6
CNN-Dropout-same (p)	81.5(0.1)	48.5(0.1)	87.6(0.1)	93.5(0.2)	92.9(0.1)	84.5(0.5)	87.4(0.1)
CNN-GI-Dropout (β)	**81.9**(0.87)	**49.0**(0.95)	**88.1**(0.98)	**93.4**(0.91)	**93.2**(0.83)	**85.1**(0.87)	**89.8**(0.98)
RNN-baseline	82.1	49.7	89.7	93.6	92.6	84.1	89.6
RNN-Dropout-same (p)	82.2(0.2)	51.9(0.1)	90.1(0.1)	93.9(0.1)	93.4(0.2)	84.2(0.1)	**89.7**(0.1)
RNN-GI-Dropout (β)	**82.5**(0.87)	**54.1**(0.95)	**90.4**(0.95)	**94.2**(0.98)	**94.8**(0.95)	**84.7**(0.91)	**89.7**(0.98)
MVCNN	-	49.6	89.4	93.9	-	-	-
MGNC-CNN	-	48.7	88.3	94.1	95.5	-	-
CNN-Rule	81.7	-	89.3	-	-	85.3	-
Semantic-CNN	82.1	50.8	89.0	93.7	94.4	86.0	89.3
combine-skip	76.5	-	-	93.6	92.2	80.1	87.1
DSCNN	82.2	50.6	88.7	93.9	95.6	-	-
Paragraph Vector	74.8	48.7	87.8	90.5	91.8	78.1	74.2
NBSVM	79.4	-	-	93.2	-	81.8	86.3
Tree LSTM	-	51.0	88.0	-	-	-	-

Table 4: Effectiveness of GI-Dropout. Dropout-same means dropping units with the same probability. Results also include: MVCNN (Yin and Schütze, 2015), MGNC-CNN (Zhang et al., 2016b), CNN-Rule (Hu et al., 2016), Semantic-CNN (Li et al., 2017), combine-skip (Kiros et al., 2015), combine-skip (Kiros et al., 2015), DSCNN (Zhang et al., 2016a), Paragraph Vector (Le and Mikolov, 2014), NBSVM (Wang and Manning, 2012) and Tree LSTM (Tai et al., 2015).

β	CNN	RNN
0.98 ($10^{-0.01}$)	48.8	51.9
0.95 ($10^{-0.02}$)	**49.0**	**54.1**
0.91 ($10^{-0.04}$)	48.0	51.8
0.87 ($10^{-0.06}$)	48.1	52.4
0.83 ($10^{-0.08}$)	47.4	51.4

Table 5: β and accuracy in SST-1.

Top-k	CNN baseline	GI-Dropout in CNN
0	87.5	88.1
50	87.1	87.9
100	86.7	87.9
200	86.1	87.5
500	84.7	86.6
1000	81.7	84.0

Table 6: Accuracy decline when removing top-k apparent words in SST-2.

By integrating our GI-Dropout mechanism, the model further improves the performance significantly on both CNN and RNN models. Compared with Dropout-same, there is a clear advantage that results on all of the datasets have been improved.

With the comparison between GI-Dropout and Dropout-same, we are convinced that GI-Dropout benefits from global information which provides explicit semantic information to guide the training process.

Even when compared with other models with complex architectures, GI-Dropout models achieve the best accuracy on most datasets, especially in SST-1 and SST-2.

4.6 Further Analysis of Our Method

With GI-Dropout, we drop words according to their importance scores. The higher score of a word, the greater chance it is to be ignored. We further analyze why GI-Dropout works so well, and the relationship between β and accuracy.

GI-dropout helps models to learn inapparent features. To test whether the method indeed helps models to learn the inapparent features, we conduct experiments where the top-k apparent words (with highest important scores) were removed from test cases in SST-2. Results are shown in Table 6. We can observe that the CNN base-

line model is more sensitive to the apparent features and GI-dropout can still have relatively good results even when we remove top 1000 apparent words. Thus, the model is supposed to pay more attention to the inapparent features with the help of GI-Dropout.

GI-dropout helps models to reduce the overfitting for the apparent features. The frequent words can easily induce the model to focus on limited features and activate a part of units with large score. This can be seen by analyzing the cases which the proposed model makes a correct prediction and the baseline makes a incorrect prediction:

(1) *provide -lrb- s -rrb- nail-biting suspense and credible characters <u>without</u> relying on technology-of-the-moment technique or* **pretentious**[8] *dialogue.*

(2) *the screenplay <u>sabotages</u> the movie's* **strengths** *at almost every juncture.*

(3) *this is <u>cool</u>, <u>slick</u> stuff, ready to <u>quench</u> the thirst of an audience that* **misses** *the summer blockbusters.*

The baseline model is prone to focus only on the prominent features, e.g. the **"pretentious"** (negative) in case (1), **"strengths"** (positive) in case(2) and **"miss"** (negative) in case (3), and then make wrong predictions. Even though there are some important words indicating the opposite polarity, e.g. **"without"** in case (1), **"sabotages"** in case (2) , **"cool"**, **"slick"** and **"quench"** in case (3), the model can not make use of these features efficiently.

By integrating our GI-dropout method, the model can learn not only the obvious features, e.g. **"strengths"**, but also the less obvious features e.g. **"sabotages"**. Thus, it makes correct predictions in all the above cases.

The relationship between β and accuracy. Another thing should be noticed is the value of β in Equation 2. As shown in Figure 2, the probability of a word and its rank follow Zipf's Law when β is 0.95 in SST-1. Actually, for each dataset, there is an appropriate β value for Equation 2 that can approximate the dropout probability and its rank with a Zipfian distribution. We assume that the β setting in accord with Zipf's Law could have an important positive effect on the model perfor-

mance. To examine this hypothesis, we further test the influences of different β values on the CNN and RNN model. As expected, Table 5 shows that the models achieve the best results for both CNN and RNN in SST-1 with β setting to 0.95.

5   Conclusion

This paper proposes GI-Dropout, a novel dropout method which utilizes global information and guides neural networks to extract not only obvious features but also unobvious features.

This idea is inspired by dropout in which units are dropped randomly in training according to the same probability. Unlike traditional dropout method, we aim to use global information to guide our dropout based on the importance of the words.

By integrating this mechanism, we encode global information explicitly into model via a novel Naïve Bayes Weighting method. We discover that model can be sensitive to some inapparent patterns, which is of great help to the classification. Experimental results demonstrate the effectiveness of GI-Dropout on multiple text classification tasks. In addition, our method requires few external resources and relatively small calculation. It is simple but effective and could be easily applied to other NLP tasks.

References

Jimmy Ba and Brendan Frey. 2013. Adaptive dropout for training deep neural networks. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3084–3092. Curran Associates, Inc.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa.

---

[8]Words in **bold** denote the apparent features with high importance scores, e.g. "pretentious" appears 159 times in **positive** texts and 5 in **negative** texts. Words with <u>underline</u> represent unobvious features that also contribute to the class prediction.

2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537.

Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177. ACM.

Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard Hovy, and Eric Xing. 2016. Harnessing deep neural networks with logic rules. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 2410–2420.

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

Diederik P Kingma, Tim Salimans, and Max Welling. 2015. Variational dropout and the local reparameterization trick. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2575–2583. Curran Associates, Inc.

Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. In *Advances in neural information processing systems*, pages 3294–3302.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *International Conference on Machine Learning*, pages 1188–1196.

Shen Li, Zhe Zhao, Tao Liu, Renfen Hu, and Xiaoyong Du. 2017. Initializing convolutional filters with semantic features for text classification. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1884–1889.

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.

Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. *arXiv preprint arXiv:1703.03130*.

Justin Martineau and Tim Finin. 2009. Delta tfidf: An improved feature space for sentiment analysis.

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.

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.

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.

Haşim Sak, Andrew Senior, and Françoise Beaufays. 2014. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In *Fifteenth annual conference of the international speech communication association*.

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.

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.

Kai Sheng Tai, Richard Socher, and Christopher D Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 1556–1566.

Duyu Tang, Bing Qin, and Ting Liu. 2015. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 1422–1432.

Sida Wang and Christopher D Manning. 2012. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*, pages 90–94. Association for Computational Linguistics.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language resources and evaluation*, 39(2-3):165–210.

Wenpeng Yin, Katharina Kann, Mo Yu, and Hinrich Schütze. 2017. Comparative study of cnn and rnn for natural language processing. *arXiv preprint arXiv:1702.01923*.

Wenpeng Yin and Hinrich Schütze. 2015. Multichannel variable-size convolution for sentence classification. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*, pages 204–214.

Rui Zhang, Honglak Lee, and Dragomir Radev. 2016a. Dependency sensitive convolutional neural networks for modeling sentences and documents. In *Proceedings of NAACL-HLT*, pages 1512–1521.

Ye Zhang, Stephen Roller, and Byron C Wallace. 2016b. Mgnc-cnn: A simple approach to exploiting multiple word embeddings for sentence classification. In *Proceedings of NAACL-HLT*, pages 1522–1527.

Ye Zhang and Byron Wallace. 2017. A sensitivity analysis of (and practitioners guide to) convolutional neural networks for sentence classification. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 253–263.

George K Zipf. 1935. The psychology of language. *NY Houghton-Mifflin*.