Text Smoothing: Enhance Various Data Augmentation Methods on Text Classification Tasks

Anonymous ACL submission

Abstract

Before entering the neural network, a token is generally converted to the corresponding one-hot representation, which is a discrete distribution of the vocabulary. Smoothed representation is the probability of candidate tokens obtained from a pre-trained masked language model, which can be seen as a more informative substitution to the one-hot representation. We propose an efficient data augmentation method, termed **text smoothing**, by converting a sentence from its one-hot representation to a controllable smoothed representation. We evaluate text smoothing on different benchmarks in a low-resource regime. Experimental results show that text smoothing outperforms various mainstream data augmentation methods by a substantial margin. Moreover, text smoothing can be combined with those data augmentation methods to achieve better performance.

1 Introduction

Data augmentation is a widely used technique, especially in the low-resource regime. It increases the size of the training data to alleviate overfitting and improve the robustness of deep neural networks. In the field of natural language processing (NLP), various data augmentation techniques have been proposed. One most commonly used method is to randomly select tokens in a sentence and replace them with semantically similar tokens to synthesize a new sentence (Wei and Zou, 2019; Kobayashi, 2018; Wu et al., 2019). (Kobayashi, 2018) proposes contextual augmentation to predict the probability distribution of replacement tokens by using the LSTM language model and sampling the replacement tokens according to the probability distribution. (Wu et al., 2019) uses BERT's (Devlin et al., 2018) masked language modeling (MLM) task to extend contextual augmentation by considering deep bi-directional context. (Kumar et al., 2020) further propose to use different types of transformer based pre-trained models for conditional data augmentation in the low-resource regime.

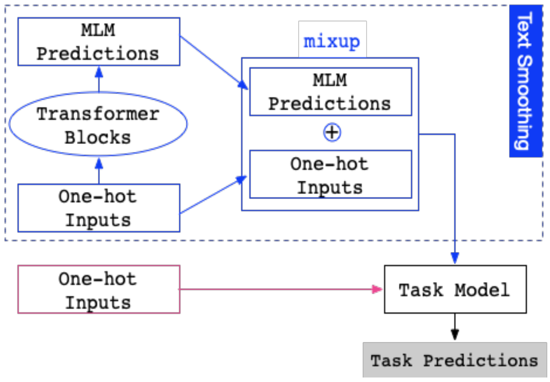

Figure 1: The blue part demonstrates the use of text smoothing data augmentation for downstream tasks, and the red part directly uses the original input.

MLM takes masked sentences as input, and typically 15% of the original tokens in the sentences will be replaced by the [MASK] token. Before entering MLM, each token in sentences needs to be converted to its one-hot representation, a vector of the vocabulary size with only one position is 1 while the rest positions are 0. MLM outputs the probability distribution of the vocabulary size of each mask position. Through large-scale pre-training, it is expected that the probability distribution is as close as possible to the ground-truth one-hot representation. Compared with the one-hot representation, the probability distribution predicted by pre-trained MLM is a "smoothed" representation, which can be seen as a set of candidate tokens with different weights. Usually, most of the weights are distributed on contextual-compatible tokens. Multiplying the smooth representation by the word embedding matrix can obtain a weighted summation of the word embeddings of the candidate words, termed smoothed embedding, which is more informative and context-rich than the one-hot's embedding obtained through lookup operation. Therefore, the use of smoothed representation instead of one-hot representation as the input of the model can be seen as an efficient weighted data augmentation method. To get the smoothed representation of all the tokens of the entire sentence with only one forward process in MLM, we do not explicitly mask the input. Instead, we turn on the dropout of MLM and dynamically randomly discard a portion of the weight and hidden state at each layer.

An unneglectable situation is that some tokens appear more frequently than others in similar contexts during pre-training, which will cause the model to have a preference for these tokens. This is harmful for downstream tasks such as fine-grained sentiment classification. For example, given “The quality of this shirt is average .", the “average" token is most relevant to the label. The smoothed representation through the MLM at the position of “average" is shown in Figure 2. Although the probability of “average" is the highest, more probabilities are concentrated on tokens conflict with the task label, such as “high", “good" or “poor”. Such a smoothed representation is hardly a good augmented input for the task. To solve this problem, (Wu et al., 2019) proposed to train label embedding to constraint MLM predict label compatible tokens. However, under the condition of low resources, it is not easy to have enough label data to provide supervision. We get inspiration from the practical data augmentation method mixup (Zhang et al., 2017) in the computer vision field. We interpolate the smoothed representation with the original one-hot representation. Through interpolation, we can enlarge the probability of the original token, and the probabilities are still mostly distributed on the context-compatible words, as shown in the figure 2.

We combine the two stages as **text smoothing**: obtaining a smooth representation through MLM and interpolating to constrain the representation more controllable. To evaluate the effect of text smoothing, we perform experiments with low-resource settings on three classification benchmarks. In all experiments, text smoothing achieves better performance than other data augmentation methods. Further, we are pleased to find that text smoothing can be combined with other data augmentation methods to improve the tasks further. To the best of our knowledge, this is the first method to improve a variety of mainstream data augmentation methods.

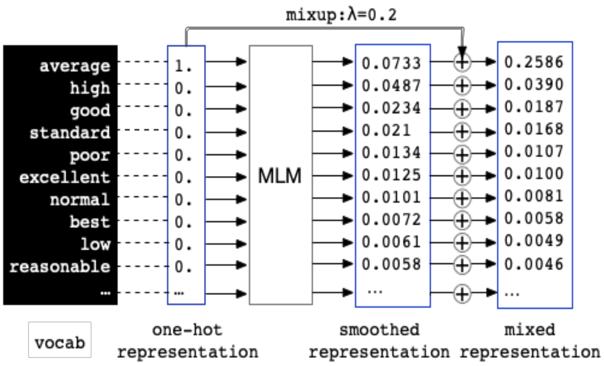

Figure 2: Interpolation of the smoothed representation and the original one-hot representation.

2 Related Work

Various NLP data augmentation techniques have been proposed and they are mainly divided into two categories: one is to modify raw input directly, and the other interferes with the embedding (Miyato et al., 2016; Zhu et al., 2019). The most commonly used method to modify the raw input is the token replacement: randomly select tokens in a sentence and replace them with semantically similar tokens to synthesize a new sentence. (Wei and Zou, 2019) directly uses the synonym table WordNet(Miller, 1998) for replacement. (Kobayashi, 2018) proposes contextual augmentation to predict the probability distribution of replacement tokens with two causal language models. (Wu et al., 2019) extends contextual augmentation with BERT’s (Devlin et al., 2018) masked language modeling (MLM) to consider bi-directional context. (Gao et al., 2019) softly augments a randomly chosen token in a sentence by replacing its one-hot representation with the distribution of the vocabulary provided by the causal language model in machine translation. Unlike (Gao et al., 2019), we use MLM to generate smoothed representation, which considers the deep bi-directional context more adequately. And our method has better parallelism, which can efficiently obtain the smoothed representation of the entire sentence in one forward process. Moreover, we propose to constrain smoothed representation more controllable through interpolation for classification tasks.

3 Our Method

3.1 Smoothed Representation

We use BERT as a representative example of MLM. Given a downstream task dataset, namely $\mathcal{D} = \{t_i, p_i, s_i, l_i\}_{i=1}^{N}$, where N is the number of

```python
sentence = "My favorite fruit is pear ."
lambd = 0.1 # interpolation hyperparameter
mlm.train() # enable dropout, dynamically mask
tensor_input = tokenizer(sentence, return_tensors="pt")
onehot_repr = convert_to_onehot(**tensor_input)
smoothed_repr = softmax(mlm(**tensor_input).logits[0])
interpolated_repr = lambd * onehot_repr + (1 - lambd) * smoothed_repr
```

Listing 1: Codes to implement text smoothing in PyTorch

instances, t_i is the one-hot encoding of a text (a single sentence or a sentence pair), p_i is the positional encoding of t_i, s_i is the segment encoding of t_i and l_i is the label of this instance. We feed the one-hot encoding t_i, positional encoding p_i as well as the segment encoding s_i into BERT, and fetch the output of the last layer of the transformer encoder in BERT, which is denoted as:

$$\overrightarrow{t_i} = \mathrm{BERT}(t_i) \quad (1)$$

where $\overrightarrow{t_i} \in \mathcal{R}^{\text{seq\_len,emb\_size}}$ is a 2D dense vector in shape of [sequence_len, embedding_size]. We then multiply $\overrightarrow{t_i}$ with the word embedding matrix $W \in \mathcal{R}^{\text{vocab\_size,embed\_size}}$ in BERT, to get the MLM prediction results, which is defined as:

$$\mathrm{MLM}(t_i) = \mathrm{softmax}(\overrightarrow{t_i} W^T) \quad (2)$$

where each row in $\mathrm{MLM}(t_i)$ is a probability distribution over the token vocabulary, representing the context-compatible token choices in that position of the input text learned by pre-trained BERT.

3.2 Mixup Strategy

The mixup (Zhang et al., 2017) is defined as:

$$\tilde{x} = \lambda x_i + (1 - \lambda) x_j \quad (3)$$

$$\tilde{y} = \lambda y_i + (1 - \lambda) y_j \quad (4)$$

(x_i, y_i) and (x_j, y_j) are two feature-target vectors drawn at random from the training data, and $\lambda \in [0, 1]$. In text smoothing, the one-hot representation and smoothed representation are derived from the same raw input, their lables are identical and the interpolation operation will not change the label. So the mixup operation can be simplified to:

$$\widetilde{t_i} = \lambda \cdot t_i + (1 - \lambda) \cdot \mathrm{MLM}(t_i) \quad (5)$$

t_i is the one-hot representation, $\mathrm{MLM}(t_i)$ is the smoothed representation, $\widetilde{t_i}$ is the interpolated representation and λ is the balance hyperparameter to control interpolation strength. In the downstream tasks, we use interpolated representation instead of the original one-hot representation as input.

	SST-2	SNIPS	TREC
Train	20	70	60
Dev	20	70	60
Test	1821	700	500

Table 1: Data statistics in low-resource regime settings.

4 Experiment

4.1 Baseline Approaches

EDA(Wei and Zou, 2019) consists of four simple operations: synonym replacement, random insertion, random swap, and random deletion.
Back Translation (Shleifer, 2019) translate a sentence to a temporary language (EN-DE) and then translate back the previously translated text into the source language (DE-EN).
CBERT (Wu et al., 2019) masks some tokens and predicts their contextual substitutions with pre-trained BERT.
BERTexpand, BERTprepend (Kumar et al., 2020) conditions BERT by prepending class labels to all examples of given class. “expand" a the label to model vocabulary, while “prepend" without.
GPT2context (Kumar et al., 2020) provides a prompt to the pre-trained GPT model and keeping generating until the EOS token.
BARTword, BARTspan (Kumar et al., 2020) conditions BART by prepending class labels to all examples of given class. BARTword masks a single word while BARTspan masks a continuous chunk.

4.2 Experiment Setting

Our experiment strictly follows the settings in the (Kumar et al., 2020) paper on three text classification datasets downloaded from the links [1].
SST-2 (Socher et al., 2013) is a movie reviews sentiment classification task with two labels.
SNIPS (Coucke et al., 2018) is a task of over 16,000 crowd-sourced queries distributed among 7 user intents of various complexity.

[1] SST-2 and TREC: https://github.com/1024er/cbert_aug, SNIPS: https://github.com/MiuLab/SlotGated-SLU/tree/master/data/snips

Method	SST-2	SNIPS	TREC	Avg.
No Aug	52.93 (5.01)	79.38 (3.20)	48.56 (11.53)	60.29(6.58)
EDA	53.82 (4.44)	85.78 (2.96)	52.57 (10.49)	64.06(5.96)
BackTrans.	57.45 (5.56)	86.45 (2.40)	66.16 (8.52)	70.02(5.49)
CBERT	57.36 (6.72)	85.79 (3.46)	64.33 (10.90)	69.16(7.03)
BERTexpand	56.34 (6.48)	86.11 (2.70)	65.33 (6.05)	69.26(5.08)
BERTprepend	56.11 (6.33)	86.77 (1.61)	64.74 (9.61)	69.21(5.85)
GPT2context	55.40 (6.71)	86.59 (2.73)	54.29 (10.12)	65.43(6.52)
BARTword	57.97 (6.80)	86.78 (2.59)	63.73 (9.84)	69.49(6.41)
BARTspan	57.68 (7.06)	87.24 (1.39)	67.30 (6.13)	70.74(4.86)
Text smoothing	**59.37(7.79)**	**88.85(1.49)**	**67.51(7.46)**	**71.91 (5.58)**

Table 2: Evaluating data augmentation methods on different datasets in a low-resource regime.

Method	SST-2	SNIPS	TREC	Avg.
EDA	59.66 (5.57)	87.53 (2.31)	55.95 (7.90)	67.71 (5.26)
+ text smoothing	**64.84(6.82)**	**88.54(3.03)**	**67.68(9.70)**	**73.69(6.52)**
BackTrans.	60.60 (7.40)	86.04 (2.20)	64.57 (7.48)	70.40 (5.70)
+ text smoothing	**61.66(7.62)**	**88.72(1.99)**	**69.17(10.51)**	**73.19(6.7)**
CBERT	60.10 (4.57)	86.85 (2.06)	63.56 (8.09)	70.17 (4.91)
+ text smoothing	**61.65(6.65)**	**88.18(2.85)**	**67.84(9.70)**	**72.56(6.4)**
BERTexpand	59.85 (6.16)	86.12 (2.45)	62.67 (7.59)	69.55 (5.40)
+ text smoothing	**62.04(7.93)**	**89.49(2.05)**	**65.89(7.48)**	**72.47(5.82)**
BERTprepend	60.28 (5.80)	86.86 (2.46)	65.20 (6.88)	70.78 (5.05)
+ text smoothing	**62.75(7.14)**	**88.04(1.92)**	**68.07(7.30)**	**72.95(5.45)**
GPT2context	57.46 (4.96)	84.10 (2.39)	46.47 (12.80)	62.68 (6.72)
+ text smoothing	**60.66(6.72)**	**87.68(1.60)**	**59.13(11.33)**	**69.16(6.55)**
BARTword	86.98(1.96)	60.99(7.15)	61.29(10.00)	69.76(6.37)
+ text smoothing	**62.67(7.40)**	**88.50(2.10)**	**67.75(6.50)**	**72.97(5.33)**
BARTspan	87.34(2.17)	63.42(5.58)	62.47(8.11)	71.08(5.29)
+ text smoothing	**62.37(7.18)**	**89.06(2.18)**	**70.89(6.81)**	**74.11(5.39)**

Table 3: The effect of text smoothing combined with other data augmentation methods in low-resource regime.

TREC (Li and Roth, 2002) contains six question types collected from 4,500 English questions.

We randomly subsample 10 examples per class for each experiment for both training and development set to simulate a low-resource regime. Data statistics of the three datasets are shown in Table 1. Following (Kumar et al., 2020), we replace numeric class labels with their text versions.

We first compare the effects of text smoothing and baselines data augmentation methods on different datasets in a low-resource regime. Then we further explore the effect of combining text smoothing with each baseline method. Considering that the amount of data increases to 2 times after combination, we expand the data used in the baseline experiments to the same amount for the fairness of comparison. All experiments are repeated 15 times to account for stochasticity and results are reported as Mean (STD) accuracy on the full test set.

4.3 Experimental Results

As shown in Table2, text smoothing brings the largest improvement to the model on the three datasets compared with other data augmentation methods. Compared with training without data augmentation, text smoothing achieves an average improvement of 11.62% on the three datasets, which is significant. The previously best method is BARTspan, which is exceeded by Text smoothing with 1.17% in average.

Moreover, we are pleased to find that text smoothing can be well combined with various data augmentation methods, further improving the baseline data augmentation methods. As shown in Table3, text smoothing can bring significant improvements of 5.98%, 2.79%, 2.39%, 2.92%, 2.17%, 6.48%, 3.21%, 3.03% to EDA, BackTrans, CBERT, BERTexpand, BERTprepend, GPT2context, BARTword, and BARTspan, respectively. To the best of our knowledge, this is the first method to improve a variety of mainstream data augmentation methods.

5 Conclusoins

This article proposes text smoothing, an effective data augmentation method, by converting sentences from their one-hot representations to controllable smoothing representations. In the case of a low data regime, text smoothing is significantly better than various data augmentation methods. Furthermore, text smoothing can further be combined with various data augmentation methods to obtain better performance.

References

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190*.

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*.

Fei Gao, Jinhua Zhu, Lijun Wu, Yingce Xia, Tao Qin, Xueqi Cheng, Wengang Zhou, and Tie-Yan Liu. 2019. Soft contextual data augmentation for neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5539–5544.

Sosuke Kobayashi. 2018. Contextual augmentation: Data augmentation by words with paradigmatic relations. *arXiv preprint arXiv:1805.06201*.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2020. Data augmentation using pre-trained transformer models. *arXiv preprint arXiv:2003.02245*.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

George A Miller. 1998. *WordNet: An electronic lexical database*. MIT press.

Takeru Miyato, Andrew M Dai, and Ian Goodfellow. 2016. Adversarial training methods for semi-supervised text classification. *arXiv preprint arXiv:1605.07725*.

Sam Shleifer. 2019. Low resource text classification with ulmfit and backtranslation. *arXiv preprint arXiv:1903.09244*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y 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.

Jason Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196*.

Xing Wu, Shangwen Lv, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019. Conditional bert contextual augmentation. In *International Conference on Computational Science*, pages 84–95. Springer.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. 2017. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*.

Chen Zhu, Yu Cheng, Zhe Gan, Siqi Sun, Tom Goldstein, and Jingjing Liu. 2019. Freelb: Enhanced adversarial training for natural language understanding. *arXiv preprint arXiv:1909.11764*.