Leveraging BERT with Mixup for Sentence Classification (Student Abstract)

Amit Jindal,[1] **Dwaraknath Gnaneshwar,**[1] **Ramit Sawhney,**[2] **Rajiv Ratn Shah**[3]
[1]Manipal Institute of Technology
{amitj646, dwarakasharma} @gmail.com
[2]Netaji Subhas Institute of Technology
ramits.co@nsit.net.in
[3]Indraprastha Institute of Information Technology, Delhi
rajivratn@iiitd.ac.in

Abstract

Good generalization capability is an important quality of well-trained and robust neural networks. However, networks usually struggle when faced with samples outside the training distribution. Mixup is a technique that improves generalization, reduces memorization, and increases adversarial robustness. We apply a variant of Mixup called Manifold Mixup to the sentence classification problem, and present the results along with an ablation study. Our methodology outperforms CNN, LSTM, and vanilla BERT models in generalization.

Introduction

Deep neural networks are powerful mathematical models that can approximate any function and have shown excellent results in data-rich problems that are extremely difficult to solve without parameterization. Usually, the network's parameters are orders of magnitude larger than training samples, and inputs that are outside training distribution may challenge the model's capabilities. One way around this problem is to augment the training data with random or systematic transformations to make sure the model trains on samples from the vicinity distribution along with the original distribution. Mixup is a data-agnostic data augmentation method that is proven to be effective in introducing samples from vicinity distribution.

Mixup (Zhang et al. 2017; Verma et al. 2018) has shown improvements in the accuracy of image classification models in Computer Vision and has also proven to be effective in NLP applications, (Guo, Mao, and Zhang 2019) trained a CNN based classifier on various sentiment classification datasets with mixup and found an increase in the generalization capabilities of the model.

We extend the concept and try to improve on it by using a technique called manifold mixup along with BERT on the sentence classification problem. We empirically prove that using BERT with input mixup and manifold mixup gives us substantial gains compared to the baseline vanilla BERT implementation.

Our main contribution is an analysis of the effects of input mixup and manifold mixup on transformer-based models. We argue that manifold mixup is a good regularization technique while also improving the model's generalization capabilities. Experimental results show the effectiveness of manifold mixup in classification tasks.

Methodology

Mixup is a data-agnostic augmentation technique that constructs virtual training examples by interpolating the training samples. In this paper, we propose two variants of Mixup, namely Input Mixup and Manifold Mixup for sentence classification. In Input Mixup, we interpolate the word embeddings of a sentence pair. Interpolation encourages the model to learn more efficiently since Mixup has broadened the training distribution. The following describes the above process.

$$\widetilde{x} = \lambda x_i + (1 - \lambda) x_j \tag{1}$$

$$\widetilde{y} = \lambda y_i + (1 - \lambda) y_j \tag{2}$$

where x_i, x_j are the sentence embedding tensors and y_i, y_j are one-hot label vectors. The parameter $\lambda \in [0,1]$ is distributed according to a Beta distribution: $\lambda \sim \beta(\alpha, \alpha)$.

Manifold Mixup, meanwhile, interpolates the hidden representations of the training examples generated by BERT. Manifold Mixup improves generalization as it leverages interpolations in deeper hidden layers, which capture higher-level information to provide additional training signal and also improves the hidden representation and decision boundaries of neural networks at multiple layers. In particular, let f_k be the hidden representation at layer k. Notably, the mixup process is defined as:

$$\widetilde{x} = \lambda f(x_i)_k + (1 - \lambda) f(x_j)_k \tag{3}$$

$$\widetilde{y} = \lambda y_i + (1 - \lambda) y_j \tag{4}$$

Training with Manifold Mixup involves selecting a layer k from a set of layers S in our neural network and interpolating the hidden representation at that layer. We do this stochastically for every mini-batch and backpropagate gradients through the entire computational graph, including layers before the mixup layer k. In the case where $S = 0$, Manifold Mixup reduces to the input mixup.

Experiments

To test the effectiveness of manifold mixup, we implemented it with a pre-trained BERT model as the baseline. We

Experiments error rates					
Model	**IMDB**	**SST-1**	**MR**	**TREC**	**SUBJ**
CNN	7.67	55.0	21.61	8.8	7.59
LSTM	12.85	53.6	21.67	13.5	8.60
BERT	6.46	46.75	**10.83**	2.62	2.20
BERT + Input Mixup	6.17	45.55	12.81	2.41	1.505
BERT + Manifold Mixup	**6.02**	**44.20**	11.94	**2.20**	**1.501**

Table 1: Test error (%) of the testing methods using BERT. Best results highlighted in Bold.

compare the results of Manifold mixup with Input mixup, baseline model on five datasets. The CNN and LSTM experiments results and details are taken from (Guo, Mao, and Zhang 2019) and are compared to our models. The baseline BERT model used is the BERT base model, which has 12 transformer blocks, 12 attention heads, and 110 million parameters. The following datasets are used in our experiments. All models are trained on Nvidia Tesla K80 GPU.

- **IMDB** is a binary sentiment classification dataset.
- **MR** is a movie review dataset for detecting positive/negative reviews.
- **TREC** is a question dataset to categorize a question into six question types
- **SUBJ** is a subjectivity detection dataset for classifying a sentence as being subjective or objective
- **SST-1** is the Stanford Sentiment Treebank with five categories label

Table 2: Test error (%) Manifold Mixup for different sets of eligible layers S on IMDB

S	IMDB
{0}	6.17
{0, 1}	6.10
{0, 1, 2}	6.27
{0, 1, 2, 3}	6.21
{0, 1, 2, 3, 4}	6.15
{0, 1, 2, 3, 4, 5}	6.22
{0, 1, 2, 3, 4, 5, 6}	6.09
{0, 1, 2, 3, 4, 5, 6, 7 }	**6.02**
{0, 1, 2, 3, 4, 5, 6, 7, 8}	6.23
{0, 1, 2, 3, 4, 5, 6, 7, 8, 9}	6.25
{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10}	6.28
{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11}	6.40

The results of all experiments are summarized in Table 1. Table 3 shows the sensitivity of Input Mixup and Manifold Mixup to the hyper-parameter α.

Ablation study

In the context of neural networks, ablation studies are done by systematically removing parts of a neural network and observing the effect on the performance of the network. We present our results of an ablation study on the number of layers in manifold mixup in Table 2. We noticed that ablation studies were only instrumental on larger datasets, our ablation experiments on MR, SUBJ did not yield any interesting or varying results.

Table 3: Test error (%) of Input Mixup and Manifold Mixup for different α values on MR dataset

α	Input Mixup	Manifold Mixup
0.5	12.91	12.53
1	13.66	11.96
1.5	12.81	11.96
2	13.85	11.94

We also found that for small datasets, higher α values performed better than smaller α values.

Conclusion

We show that manifold mixup can be a viable regularization technique that brings a slew of improvements to the generalization capabilities of the model. Empirical results suggest that BERT with manifold and input mixup outperforms the already excellent performance of vanilla BERT. Future research directions include studying the results of mixup with pruning techniques, how manifold mixup NLP models would react to adversarial examples, and the feasibility of manifold mixup augmented adversarial training as an adversarial defense technique.

References

Guo, H.; Mao, Y.; and Zhang, R. 2019. Augmenting data with mixup for sentence classification: An empirical study. *arXiv preprint arXiv:1905.08941*.

Verma, V.; Lamb, A.; Beckham, C.; Najafi, A.; Mitliagkas, I.; Courville, A.; Lopez-Paz, D.; and Bengio, Y. 2018. Manifold mixup: Better representations by interpolating hidden states. *arXiv preprint arXiv:1806.05236*.

Zhang, H.; Cisse, M.; Dauphin, Y. N.; and Lopez-Paz, D. 2017. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*.