Introducing Adversarial Training to Improve the Performance of Model in Natural Language Processing Domain

Anchun Gui[1], Jiashuo Sun[2], Junyu Chen[2] and Tao Chen[1]

[1]315202111540{41,05}
[2]230202111539{64,18}

Abstract

Recently, the concept of adversarial training is getting more and more attention in the field of deep learning. Adversarial training is not only used in the Computer Vision (CV) domain, it also can be applied in the Natural Language Processing (NLP) domain, and obtain good performance. In this paper, we proposed some methods that can be used for adversarial training in the NLP downstream tasks. We give the definition and detailed implements of each methods and compare with their advantages and disadvantages. We used Question Answer (Q&A) matching as our downstream task, the experimental results demonstrate the effectiveness of adversarial training, which significantly improves the generalization ability of the model. Furthermore, adversarial training can be used in many tasks which indicates its expansibility.

1 Introduction

Adversarial samples (Goodfellow, Shlens, and Szegedy 2014) is the input samples which were deliberately added subtle perturbation. A good adversarial samples can use subtle perturbation to make a difference on results. The basic principle of adversarial training (Szegedy et al. 2013) is to construct some adversarial samples by adding perturbation and feed into the model for training to improve the robustness of the model when encountering adversarial samples. While adversarial training boosts the robustness, it is widely accepted by computer vision researchers that it is at odds with generalization, with classification accuracy on non-corrupted images dropping as much as as 10% on CIFAR-10 (Krizhevsky, Hinton et al. 2009), and 15% on Imagenet (Deng et al. 2009). Surprisingly, people observe the opposite result for language models, showing that adversarial training can improve both generalization and robustness. Although adversarial training can make the model perform better, it lengthen the model's training time.

However, in the natural language processing domain, the input is a discrete sequence of words, generally presented in the form of one-hot vector. If the perturbation is directly performed on the raw text, then the scale and direction of the disturbance may be meaningless. Because the set of high-dimensional one-hot vectors does not admit infinitesimal perturbation, we can use the perturbation on continuous word embeddings instead of discrete word inputs.

Specifically, since large-scale pre-trained models (e.g. BERT (Devlin et al. 2018)) are gradually applied to natural language processing tasks (e.g information extraction (Wan and Xiao 2008), text classification (Baker and McCallum 1998)), adversarial training can be simply applied to the Embedding layer of the pre-trained model. Since the adversarial training is to update the parameters by accumulating the gradient in backpropagation (Rumelhart, Hinton, and Williams 1986), we can only use Deep Learning technology instead of traditional Machine Learning. The commonly used adversarial training methods are FGM (Fast Gradient Method), PGD (Projected Gradient Descent), FreeAT (Free Adversarial Training), YOPO (You Only Propagate Once) and FreeLB (Free Large-Batch) (Goodfellow, Shlens, and Szegedy 2014; Madry et al. 2017; Shafahi et al. 2019; Zhang et al. 2019; Zhu et al. 2019), we will introduce in detail in related work.

In this paper, we use QA matching as our downstream task to demonstrate the effictiveness of adversarial training. Intelligent QA matching infers the precise needs of users by understanding the deep semantics of natural language questions raised by users, and does not need to be re-screened by users when obtaining results, which improves the efficiency of information acquisition while improving user experience. We will demonstrate results based on experiments.

2 Related Work

Recently, adversarial training (Goodfellow, Shlens, and Szegedy 2014) has gradually become a hot research topic and received more and more researchers' attention. Adversarial training is implemented by adding random minor disturbance into the input of model, which may fools model's prediction. Actually, it is the addition of those disturbances, which force model adjusting weight to adapt to this change, thus the robustness of the model is promoted by this special training procedure.

We know that the general principle of adversarial training is to optimize the following Max-Min objective function,

$$\min_{\theta} E_{(x,y)\sim\mathcal{D}} \left[\max_{\Delta x \in \Omega} L(x + \Delta x, y; \theta) \right]$$

where, $\mathcal{D}$ refers to data set, x, y refers to the input and output

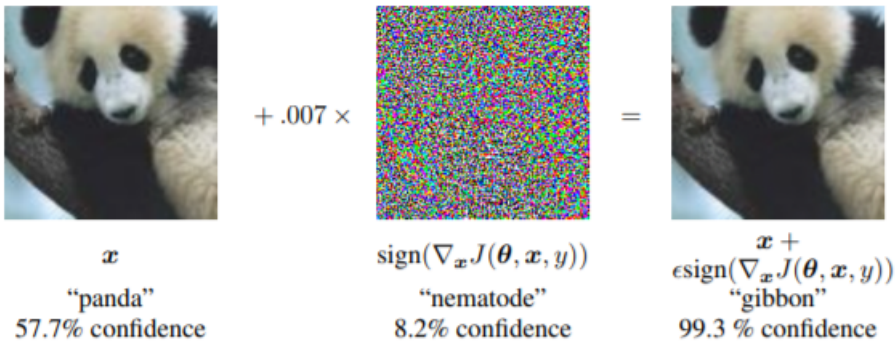

Figure 1: Adversarial samples

of model respectively, θ represents the parameters of model, Δx represents the disturbance mixing with x, $L(x, y; \theta)$ corresponding to objective function. For the objective function:

- Maximize the inner layer disturbance means to find the disturbance that maximizes the loss function. Simply speaking, the tiny disturbance should make the neural network as confused as possible.
- The outer layer is the minimization formula for the optimization of the neural network, that is, when the disturbance is fixed, we train the neural network model to minimize the loss of training data so that making the model have certain robustness to adapt to the disturbance.

A lot of efforts have been made to determine the perturbation. In terms of the motivation behind the research, the current research directions on adversarial training can be divided into the following two aspects.

Gradient Disturbance FGM (Miyato, Dai, and Goodfellow 2017) proposed to regard the model's backpropagation gradient as the perturbation Δ mix with the model's input. Specifically, we can make the perturbation $\Delta x = \epsilon \nabla_x L(x, y; \theta)$. It is worth noting that $\nabla_x L(x, y; \theta)$ is normalized in case of avoiding Δx is expanding beyond expectation. Generally,

$$\Delta x = \epsilon \frac{\nabla_x L(x, y; \theta)}{\|\nabla_x L(x, y; \theta)\|}$$

or

$$\Delta x = \epsilon sign(\nabla_x L(x, y; \theta))$$

the latter normalization Method should be called FGSM (Goodfellow, Shlens, and Szegedy 2014).

Since FGM calculates the adversarial perturbation by adjusting ϵ, the results obtained may not be optimal, and there is also a linear assumption issue in FGSM and FGM. Therefore, in order to solve these problems, PGD (Madry et al. 2017) was proposed. PGD, which based on multiple iterations, projects the disturbance to the specified range and approaching the optimal disturbance gradually.

$$g_t = \nabla_{X_t}(L(f_\theta(X_t), y))$$

where g_t represents the loss in t time step.

$$X_{t+1} = \prod_{X+S}(X_t + \epsilon(g_t/\|g_t\|))$$

The input at time step $t+1$ is calculated from the input at time step t. Notice that if the disturbance goes beyond a certain range, it maps back to the specified range S.

Efficiency In the process of calculating the PGD (Madry et al. 2017), both the gradient of parameters and the gradient of output will be calculated in each time step. However, only the gradient of parameters is used in gradient descent, while only the gradient of input is used in gradient elevation, which is actually a great waste. A natural question is, can we use both the gradient of the parameter and input simultaneously? This is the core idea of FreeAT (Shafahi et al. 2019). The gradient calculation formula of the model is

$$g_\theta = E_{(x,y)\in B}\left[\nabla_\theta l(x + \delta, y, \theta)\right]$$

In addition, the adversarial gradient of the model can be calculated by

$$g_{adv} = \nabla_x l(x + \delta, y, \theta)]$$

To be specific, although FreeAT still adopts PGD training strategy, that is, K times of gradient is calculated for each min-batch sample, but the gradient obtained each time is used to update both disturbance and parameters.

The starting point of YOPO (Zhang et al. 2019) is to use neural network to reduce the computation amount of gradient calculation. From the perspective of PMP, anti-disturbance is only related to the first layer of neural network. Therefore, the paper proposes to fix the front base layer, calculate the gradient of the first layer only, and update the disturbance accordingly.

Like FreeAT, FreeLB (Zhu et al. 2019) intends to use both gradients more efficiently. However, different from FreeAT, the parameters of a FreeLB are not updated every time the gradient is raised.

$$g_t = g_{t-1} + \frac{1}{K} E_{(Z,y)\in B}\left[\nabla_\theta L\left(f_\theta\left(X + \delta_{t-1}\right), y\right)\right]$$

In this way, the parameters of a FreeLB are updated by using the gradient of the parameters accumulated after K steps.

Compared with PGD, FreeLB simplifies the gradient calculation procedure. Unfortunately, compared with the gradient calculation of FreeAT, the improved computational efficiency of FreeLB is not obvious.

However, it is worth mentioning that the advantage of FreeLB is not only in efficiency, but in its excellent effect. Since FreeLB uses the gradient of multi-step accumulation to update, the gradient estimation is more accurate.

3 Proposed Solution

In this section, we mainly demonstrate that how to introduce adversarial training into the practical problems we need to deal with. Here, we firstly describe the downstream task to be solved in brief, the domain-specific (real estate industry in particular) question and answer matching task, which involves scenarios such as: when customers are interested in some houses, they will ask relevant questions to the relevant real estate brokers, then the corresponding real estate brokers will answer those questions. Thus, we can access the multiple rounds of dialogue data.

Our goals are as follows: given a fragment of communication of a certain length, the fragment contains the questions of a client and subsequently broker replies with several messages. One of these subsequent broker messages needs to be identified as the answer to the client question. The difficulties and challenges of this issue are:

1. Random and fragmented for some chatting fragments, and language styles vary from region to region.
2. The model is required to have good generalization and be able to deal with various short and long dialogues.

Our modeling process for this problem can be roughly divided into the following five parts:

3.1 Problem Definition

Given a dataset including M data samples, where the i-th data item $(x^{(i)}, y^{(j)}, \widehat{y}_i)_{j=1}^{K}$ consists of a customer question (source text) $x^{(i)}$, a set of agent answers $y^{(j)}$ and the label $\widehat{y}_i$. Specifically, every data item is divided into K part where K is the number of agent answers corresponding to customer questions, and j-th sample item is one of the agent answers. Under this setting, a question and answer matching model is to learn the mapping from the source text $x^{(j)}$ to the target sequence $y^{(j)}$ and predict whether the $x^{(j)}$ and $y^{(j)}$ are matching.

3.2 Data Preprocessing

Since the original data comes from the actual scenes in the industry, we need to clean the original data firstly, such as, deduplication, data type transformation and incomplete data completion, etc. Generally speaking, taking the data after some rules cleaning as the input of the model can reduce the influence of data noise on the model on the one hand, and enable the model to mine the potential features inside the data on the other hand. Mastering these features may play a great role in improving the generalization ability of the model.

3.3 Model Building

In this work, we choose Nezha (Wei et al. 2019), a open source language model from Huawei, as our backbone network. The model structure is shown in Figure 2. Therefore we need to preprocess the original data to meet the input form of Nezha model. Specifically, we splice the client's question with the corresponding broker's reply, which is segmented by a special token [SEP] in vocabulary. It is a very common form of the input in BERT-like models.

After the processed data as the input into the backbone network, we can get the vector representation of the "query-reply" from feature extractor. Then, in order to judge whether the answer matches the question, we need to add a classifier on the top of the backbone network. Here, we simply use a multilayer perceptron (MLP) whose output size is the same with the number of categories in this problem.

3.4 Adversarial Training

The above two parts can be considered standard procedures for ordinary natural language processing. In this part, we need to introduce adversarial training into the training process of the model. Specifically, our approach is as follows: firstly, calculate the gradient of embedding layer corresponding to the loss objective of the model, then perform some operations, which are mentioned in related work, on the gradient with the original embedding layer as a disturbance. Finally, we need to add the resulting small perturbations to the embedding layer weights and perform gradient descent.

3.5 Training

Given the set of data pairs $(x^{(i)}, y^{(j)}, \widehat{y}_i)_{j=1}^{K}$, the loss function of the question and answer matching is cross entropy loss:

$$Loss = \frac{1}{N} \sum_{i} -(y_i \times log(p_i) + (1 - y_i) \times log(1 - p_i))$$

where N is the word sequence length of target output sequence, and p_i is the model prediction.

4 Experiments

4.1 Dataset

We use the dataset collected by CCF&BAKE[1] from various online questions and answers, which contains approximately 6K samples, each of which contains one customer question and multiple agent answer. Then, a validation set containing 500 samples will be selected from the remaining examples. In order to evaluate our proposed method, we test on 14K samples dataset from CCF&BAKE. The statistic information of which are summarized in Table 1.

4.2 Baselines and Evaluation Metrics

For question and answer matching task, we compare our model with basic NEZHA model, basic Transformer model and Transformer model with FGM adversarial training

[1] https://www.datafountain.cn/competitions/474/datasets

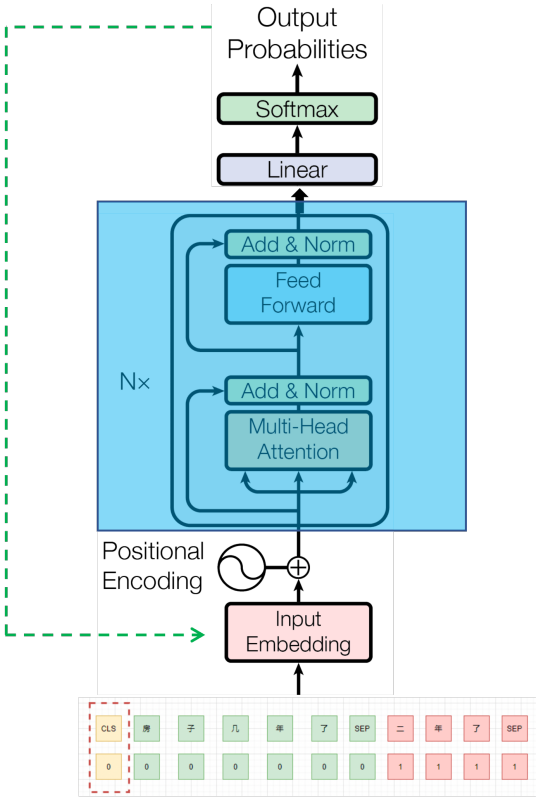

Figure 2: Model architecture

	Questions	Replys
training set	6,000	21,586
validation set	500	1,830
test set	14,000	53,758

Table 1: The statistics of customer questions and agent answers on the dataset.

method (Transformer-AT) as well. We split each data item into multiple training examples, each of which only contains one question and one answer. The baselines are the following state-of-the-art encoder-decoder models:

- **NEZHA** represents a Transformer-based model, an improved model of the BERT(Devlin et al. 2018) model.
- **Transformer**(Vaswani et al. 2017) represents a Attention-based encoder-decoder model.
- **Transformer-AT** represents a standard Transformer model with FGM adversarial training method.

The implementation of Transformer is base on open source tool OpenNMT[2].For adversarial training methods, we use FGM, PGD and FreeAT and add them to the NEZHA model. For model evaluation, we adopt F-measure (F1) as our evaluation metrics for the question and answer matching task. For evaluation, exact match is used for determining whether the predictions are correct.

[2]https://github.com/OpenNMT/OpenNMT-py

4.3 Implementation Details

We set maximal length of source sequence as 128, 20 for target sequence of model output, and 20 for the decoders of all baselines. We choose the 21,129 frequently-occurred Chinese words as our vocabulary. The dimension of the word embedding is 128,The dimension of hidden state in encoder, decoder is 768. The word embedding is used in pre-trained NEZHA model. Words masked probably is 0.15. We use Adagrad as optimization function with learning rate = 0.00005. All the baseline models are trained on a single Tesla P100.

4.4 Results and Analysis

In this section, we present the results of whether to add the adversarial training and different methods of adversarial training, respectively. The results of whether to add the adversarial training are shown in Table 2.

Models	F1 score
Transformer	0.783
Transformer-AT	0.788
NEZHA	0.797
NEZHA-AT	**0.804**

Table 2: The results of whether to add the adversarial training on the dataset.

We first compare our proposed method with the baseline

models.The results show that our model with adversarial training performs better than basic model without adversarial training in question and answer matching task.We further analyze the experimental results of different adversarial training methods on NEZHA model. Table 3 shows the results of different methods of adversarial training.

Models	F1 score
NEZHA	0.797
NEZHA-FGM	0.804
NEZHA-PGD	0.804
NEZHA-FreeAT	**0.806**

Table 3: The results of different methods of adversarial training on the dataset.

It can be observed that different methods of adversarial training can improve the performance of matching, compared to the basic NEZHA model. Meanwhile, FreeAT is the best adversarial training method in our experiment, and achieve 0.9 % average gain than basic NEZHA model.

5 Conclusion

In this paper, we propose to introduce adversarial training into a natural language processing downstream task-specific (i.e., question and answer matching), so that the model can learn some sophisticated features and enhance the robustness of the model. The final goal is improving the performance of the model on task-specific further. We tested our proposed model on QA datasets about the real estate industry, which is a common problem in real world scenarios. Finally the experimental results showed that we achieved a competitive results compared to other basic models and further the limited and shortcomings of our proposed model will be explored in the future.

References

Baker, L. D.; and McCallum, A. K. 1998. Distributional clustering of words for text classification. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, 96–103.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, 248–255. Ieee.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Krizhevsky, A.; Hinton, G.; et al. 2009. Learning multiple layers of features from tiny images.

Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; and Vladu, A. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.

Miyato, T.; Dai, A. M.; and Goodfellow, I. J. 2017. Adversarial Training Methods for Semi-Supervised Text Classification. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Rumelhart, D. E.; Hinton, G. E.; and Williams, R. J. 1986. Learning representations by back-propagating errors. *nature*, 323(6088): 533–536.

Shafahi, A.; Najibi, M.; Ghiasi, A.; Xu, Z.; Dickerson, J.; Studer, C.; Davis, L. S.; Taylor, G.; and Goldstein, T. 2019. Adversarial training for free! *arXiv preprint arXiv:1904.12843*.

Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; and Fergus, R. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention Is All You Need. *CoRR*, abs/1706.03762.

Wan, X.; and Xiao, J. 2008. Single Document Keyphrase Extraction Using Neighborhood Knowledge. In *AAAI*, volume 8, 855–860.

Wei, J.; Ren, X.; Li, X.; Huang, W.; Liao, Y.; Wang, Y.; Lin, J.; Jiang, X.; Chen, X.; and Liu, Q. 2019. NEZHA: Neural Contextualized Representation for Chinese Language Understanding. *CoRR*, abs/1909.00204.

Zhang, D.; Zhang, T.; Lu, Y.; Zhu, Z.; and Dong, B. 2019. You only propagate once: Accelerating adversarial training via maximal principle. *arXiv preprint arXiv:1905.00877*.

Zhu, C.; Cheng, Y.; Gan, Z.; Sun, S.; Goldstein, T.; and Liu, J. 2019. Freelb: Enhanced adversarial training for natural language understanding. *arXiv preprint arXiv:1909.11764*.