Improve NLP models by adding Adversarial Training

CS388 Fa21 - Natural Language Processing Final Project

Hsuan-Wei Chen
EID: hc29434

Abstract

The goal is to use adversarial training to improve the model for better understanding the true meaning of the context. I find out that NLP models which are designed for the SQuAD v1.1 dataset can't really deal with SQuAD v2.0 dataset in 2018. However, people using adversarial training methods make models perform better F1 scores on question answering problems now. I train my model on IFLYTEK and TNEWS datasets, plus the adversarial sets that I make for these two datasets. By adding the Fast Gradient Method (an adversarial training method) I get 2.4% and 3.2% improved, for gradient penalty, I get 2.8% and 2.6% improved, respectively. In this project, I am going to analysis how adversarial examples influence our models and how FGM and gradient penalty make improvement.

1 Introduction

For the confrontation in deep learning, generally it has two meanings. One is Generative Adversarial Network, GAN, the other is Adversarial Training, which focuses on the model's robustness under small perturbation.

In recent years, along with the development of deep learning, adversarial examples have gotten more and more attention. In the CV field, we need to enhance the robustness of the model through adversarial attacks and defenses against the model. For example, we have to prevent the model from recognizing red light into green light because of some random noize in the autopilot system. In the NLP field, similar adversarial training also exists, but the adversarial training in NLP is more like a regulation method to improve the generalization ability of the model.

In the analysis part, I will briefly describe adversarial examples, and use the model (ELECTRA-small model) that the course provides to train both SQuAD v1.1 and v2.0 datasets, then analyze the outcomes after the models have been evaluated. Which include the model's performance and examples of the general mistakes that the model makes. Furthermore, talks about how adversarial examples produce and how it affects our models.

For the implementation section, it is a great chance for me to attempt to deal with NLP tasks in my native language, Chinese. Which I think will be harder than undering English. Moreover, I will explain how I implement the Fast Gradient Method and gradient penalty to show these two methods can improve a model's performance dealing with NLP tasks. Also, with logical explanation.

Result section is mainly about my experiments, presenting the improvement that adversarial training can make.

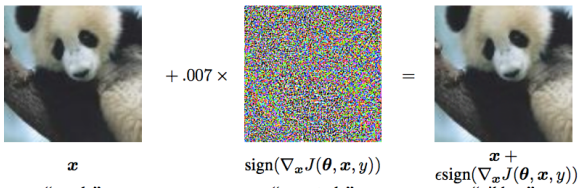
In Discussion, I came up with some problems that I face in the Chinese text, and some ideas for improvement to make models better understanding Chinese.

2 Analysis

2.1 Adversarial examples

To understand adversarial training, we must

understand adversarial samples first. Adversarial examples look exactly the same for humans, but totally different for the prediction of the model.



Samples in CV, form《Intriguing properties of neural networks》

Sample in NLI:

Original Text Prediction: **Entailment**
Premise: *A runner wearing purple strives for the finish line*
Hypothesis: *A **runner** wants to head for the finish line.*
Adversarial Text Prediction: **Contradiction**
Premise: *A runner wearing purple strives for the finish line*
Hypothesis: *A **racer** wants to head for the finish line.*

After understanding adversarial examples, we can associate that adversarial attack is to make as many adversarial examples as they could, adversarial defense is to let models recognize more adversarial examples.

2.2 SQuAD datasets

SQuAD v1.1 contains 100,000+ question-answer pairs on 500+ articles, in SQuAD v2.0, it combines the questions in v1.1 with over 50,000 unanswerable questions written adversarially by crowdworkers to look similar to answerable ones. I trained the ELECTRA-small model using SQuAD v1.1 and v2.0 datasets and evaluated both of them.

	EM Score	F1 Score
SQuAD v1.1	78.69	86.33
SQuAD v2.0	50.98	55.62

performance of ELECTRA-small model

As we can see, with only changing into the dataset with adversarial examples, the F1 score dropped approximately 31%.

There are some mistakes that I found in the prediction file from the model trained by SQuAD v2.0.

Paragraph:
"*... A computational problem is understood to be a task that is in principle amenable to being solved by a computer, which is equivalent to stating that the problem may be **solved by mechanical application of mathematical steps, such as an algorithm.***"

Question: "*What cannot be solved by mechanical application of mathematical steps?*"
Gold Answer: (none)
Model's Answer: ***algorithm***

In the SQuAD v2.0 dataset, there are 86821 questions with answers and 43498 questions without answers. The model needs to determine whether the question is answerable or not, that is to say, we need a verifier to be added into our encoder and decoder system. What I do is to combine my verifier with my decoder, but in my prediction file, my model judges 99.93% of these questions as answerable. Which makes me confirm that adversarial examples really challenge my model.

2.3 How adversarial attack works

2.3.1 adversarial attack

The progress of constructing adversarial examples is called adversarial attack, these examples will cause our model to make wrong judgment. This is because our model did not really learn the perfect rules to judge, the judge boundary is different from the real life decision boundary. Deep learning models are widely used due to their ability to automatically learn features. However, for the kind of self-learning based on data, the features obtained are not necessarily the characteristics we want. The model's understanding of data is very different from humans' minds.

2.3.2 algorithm classification

Based on the model access rights, it can be divided into white-box attacks and black-box attacks. White-box attacks need to obtain detailed information such as the structure and parameters of the model. The black-box attack does not need the model's knowledge, just access the model to obtain the corresponding output of the input.

According to the attack target setting, it can be divided into targeted attack and untarget attack. Untarget attack is designed to cause the output of the model an arbitrary error prediction that deviates from the correct prediction. Target attack aims to make the output of the model a specific result.

For text granularity when adding disturbance, it can be divided into character-level, word-level and sentence-level attacks. Character-level attacks are realized by inserting, deleting or replacing characters and swapping the sequence of characters. Word-level attacks are mainly implemented by replacing words, establishing the vocabulary based on synonyms, similar words, misspellings, etc. Sentence-level attacks are mainly implemented by text retelling or inserting sentences.

The attack strategy can be divided into Image-to-Text, optimization-based attacks, important-based attacks and neural network based attacks. Image-to-Text generates adversarial samples by mapping text data to continuous space, and uses some classic algorithms in the image field, such as FGSM, JSMA, etc. The optimization-based attacks express the attack as a constrained optimization problem, which is solved by using optimization techniques, such as gradient optimization and genetic algorithm optimization. Important-based attacks usually use gradient or text characteristics to design a scoring function to lock keyworks, then add disturbances through text editing. Neural network based attack training the neural network model automatically learns the characteristics of the adversarial sample, thereby implementing the automatic generation of the adversarial sample.

3 Implementation

The idea of Adversarial defence is to let our model recognize adversarial examples correctly. It constructs some adversarial examples into the original dataset, hoping to enhance the model's robustness to adversarial examples. Therefore, improve the model's performance.

3.1 Making Datasets

By referring to 《Generating Adversarial Examples in Chinese Texts Using Sentence-Pieces 》, I manually construct the adversarial examples for the following datasets that I am going to use, IFLYTEK and TNEWS. IFLYTEK is a long text classification datasets with 12,133 train data, 2,599 for validation and 2,600 for test. TNEWS is a short text classification dataset with 53,360 train data, 10,000 for validation and 10,000 for test. I make my adversarial by grabbing 20% of the samples in the above data sets. (For IFLYTEK: 2400 adversarial examples for train data, 520 for validation and 520 for test) I produce it mainly by using character-level and word-level attack.

Original text:
Sentence: “上課時學生的手機響不停, 老師一怒之下把手機摔了, 家長拿發票讓老師賠償”
Gold label: news_education
Adversarial text:
Sentence: “課堂上學生的手機響不停, 教師一怒之下把手機摔了, 家長拿發票讓教師賠償”
Gold label: news_education

The meaning of this sentence is “Student’s phone keeps ringing in the class, teacher breaks the phone in a moment of anger, parents bring the receipt to let teacher compensate”. For the keyword in this sentence to classify to education is “上課時” and “老師”, which means “in the class” and “teacher”. Therefore, I change it into “課堂上” and “教師”, which is another way to say in Chinese and make no difference for the meaning.

3.2 Min-Max

Adversarial training can be write uniformly in the following format:

$$min\ \theta\ E(x,y)\sim D\ [max\ \Delta x \in \Omega\ L(x+\Delta x,y;\theta)]$$

D means dataset, x is input, y is label, θ is the model's parameter, $L(x,y;\theta)$ is the loss of one sample, Δx is the perturbation and Ω is the perturbation space. This uniformly format was first proposed by the paper《Towards Deep Learning Models Resistant to Adversarial Attacks》. It can be understand by the steps below:

1. Inject the perturbation Δx into x, the goal for Δx is to make $L(x+\Delta x,y;\theta)$ as big as possible, which is to make the current model's prediction error.
2. Δx can't be too huge, otherwise it can't show that it is similar to other samples. For the conventional constraint, $\| \Delta x \| \leq \epsilon$, ϵ is constant.
3. After constructing all the adversarial examples $x+\Delta x$ for every sample, use $(x+\Delta x,y)$ as the dataset to minimize the loss to update the model's parameters θ.
4. Repeatedly execute the steps form 1 to 3.

3.3 Fast Gradient Method

How do we compute Δx? It's goal is to make $L(x+\Delta x,y;\theta)$ larger, and we know that we use gradient descent to decrease the loss, on the contrary, we raise the gradient to increase the loss.

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

In order to prevent Δx being too large, we have to normalize $\nabla x\ L(x,y;\theta)$.

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

This is so-called Fast Gradient Method (FGM), it was first propose in《Explaining and Harnessing Adversarial Examples》by Goodfellow.

3.4 Gradient Penalty

For gradient penalty, it is an idea to make the gradient of the loss of our model to be as close as zero. At the point that the gradient of the loss closer to zero is more stable and not easy to interfere.

We can also look at the gradient penalty from the perspective of the L constraint (Lipschitz constraint). One of the methods to implement L constraint is using Spectral Normalization, but it will operate the weight in every layer of our model. Theoretically, this will cause our model to have less expression ability and decrease the performance. On the other hand, gradient penalty makes the model satisfy L constraint instead of every layer, and satisfying L constraint can effectively resist the attack by adversarial examples.

Suppose that we already get Δx, when we update θ, we have to consider the expansion of $L(x+\Delta x,y;\theta)$.

$$min\ \theta\ E(x,y)\sim D\ [\ L(x+\Delta x,y;\theta)\ ]$$
$$\approx min\ \theta\ E(x,y)\sim D\ [\ L(x,y;\theta)+$$
$$\langle \nabla x\ L(x,y;\theta),\Delta x\rangle\ ]$$

The gradient of θ:

$$\nabla \theta\ L(x,y;\theta)+\langle \nabla\theta\ \nabla x\ L(x,y;\theta),\Delta x\rangle$$

Replace $\Delta x = \epsilon\ \nabla x\ L(x,y;\theta)$:

$$\nabla \theta\ L(x,y;\theta)+\epsilon\langle \nabla\theta\ \nabla x\ L(x,y;\theta),\nabla x\ L(x,y;\theta)\rangle$$
$$\approx \nabla \theta\ (L(x,y;\theta)+\frac{1}{2}\epsilon\|\nabla x L(x,y;\theta)\|^2)$$

This show that if we impose the perturbation $\epsilon\nabla x L(x,y;\theta)$ to our input, it is roughly equivalent to add gradient penalty $\frac{1}{2}\epsilon\|\nabla x L(x,y;\theta)\|^2$ to loss.

4 Result

NLP is different from CV, the input of NLP is text, it's a one-hot vector. The Euclidean distance of two different one-hot vector is always $\sqrt{2}$. Theoretically, there did not exist any perturbation. Therefore, I tried to add the perturbation into my embedding layer. I use IFLYTEK and TNEWS as my datasets and a Chinese BERT base model to train. On the CLUE leaderboard, the BERT base model got 60.29 and 56.58 on these two datasets respectively. After adding adversarial training, I

got 62.68 and 57.49, increasing 2.4% and 1.1% respectively. For the gradient penalty, I got 62.32 and 57.91. Basically, gradient penalty can get actually the same score as FGM.

	IFLYTEK	TNEWS
BERT	60.29	56.58
BERT+FGM	62.68	57.69
BERT+GP	62.32	57.91

For the second experiment, I am using IFLYTEK and TNEWS datasets with the adversarial sets which I made by myself that I mentioned in 3.1 to train my model. I got 2.4% and 3.2% improve by adding FGM, 2.8% and 2.6% improve by adding gradient penalty, respectively.

	IFLYTEK + Adversarial sets	TNEWS + Adversarial sets
BERT	50.23	47.36
BERT+FGM	52.57	50.55
BERT+GP	53.01	49.92

5 Discussion

Although FGM and gradient penalty actually improve the model, but the improvment is less than what I expect. There are several points that I think are the key issues causing my result.

1. I do my best to make the adversarial sets as natural as I can, which is to make the keyword of my data into another form. But, the frequency of some words are not rich enough to let my model learn.
2. The best way to produce adversarial sets is to use the neural network base attack for automatically producing adversarial examples. I have tried to use this method, but the sentence that I produced did not have the same meaning as the original text. For example, “美” means “beautiful”, “美術” means "art", which is the keyword for education. “美元” means dollar, keyword for finance and plenty of phrase that begins with “美” can be label in different field. This cause my automatic producing method comes up with “美照” which means beautiful photo and should belong to entertainment but is showing in the sentence which the label in game. Furthermore, there are many other phrases that do not even exist in Chinese or are not logical for the sentence. It could be my problem if my producing method is wrong, or maybe Chinese is actually hard to produce adversarial examples in this way.
3. Because I could only make adversarial examples by myself, it is hard to have a great amount of adversarial examples to enrich the diversity and frequency of my data. Especially after this experiment, if I want to have an incredible improvement of my model for understanding Chinese, the diversity of the phases can be the key point to deal in Chinese NLP tasks.

5 Conclusions

My model saw an improvement on IFLYTEK and TNEWS datasets by using FGM and gradient penalty. Besides, it shows an improvement on the datasets with adversarial sets as well. By understanding the techniques of adversarial attack and adversarial training methods are really important. Additionally, there are several adversarial training methods such as PGD, FreeAT, YOPO FreeLB and SMART that can deal with different adversarial attacks under different kinds of datasets. Consequently, adversarial training can actually improve NLP models, and also improve under the datasets with adversarial sets.

References

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow and Rob Fergus 2014. Intriguing properties of neural networks. *ArXiv*, arXiv: 1312.199

Robin Jia, Percy Liang 2017. Adversarial Examples for Evaluating Reading Comprehension Systems.

ArXiv, arXiv: 1707.07328

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras and Adrian Vladu 2019. Towards Deep Learning Models Resistant to Adversarial Attacks. *ArXiv*, arXiv: 1706.06083

Ian J. Goodfellow, Jonathon Shlens, Christian Szegedy 2015. Explaining and Harnessing Adversarial Examples. *ArXiv*, arXiv: 1412.6572

Linyang Li, Yunfan Shao, Demin Song , Xipeng Qiu† , Xuanjing Huang 2020. Generating Adversarial Examples in Chinese Texts Using Sentence-Pieces. *ArXiv*, arXiv: 2012.14769

John X. Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, Yanjun Qi 2020. TextAttack: A Framework for Adversarial Attacks, Data Augmentation, and Adversarial Training in NLP. *ArXiv*, arXiv: 2005,05909