

Application of Semi-supervised Adversarial Training Pre-trained Neural Network in Stance Detection

by Xudong Huang

Submission date: 07-Sep-2022 11:52PM (UTC+0100)

Submission ID: 185791264

File name: 25338_Xudong_Huang_Application_of_Semi-supervised_Adversarial_Training_Pre-trained_Neural_Network_in_Stance_Detection__1573825361.docx (407.56K)

Word count: 9310

Character count: 51930



Prifysgol Cymru
Y Drindod Dewi Sant
University of Wales
Trinity Saint David

Application of Semi-supervised Adversarial Training Pre-trained Neural Network in Stance Detection

by

Huang Xudong
(2011454)

Supervisor: Dr. Terry Walcott

Project submitted as part of the requirements for
the award of MSc Data and Artificial Intelligence

September 2022

Abstract: In recent years, the application of adversarial attack and defense in the field of natural language processing has received widespread attention. At the same time, the pre training model plays an important role in the field of natural language processing. This paper focuses on the optimization of the pre training model of the confrontational training in the position detection task. The Gan Bert model and its derivative model are trained on the position detection data set, and the advantages and disadvantages of the model are discussed and some improvements are made. The results show that the Gan Bert model has better performance than the Bert model in the dataset with more unlabeled samples, and the mt-gan-bert model has no greater performance improvement than the gan-bert model in the multitask dataset. Based on mt-dnn and gan-bert migration training, this paper puts forward suggestions for improving the mt-gan-bert model.

Key words: Stance detection Semi-supervised Adversarial Training Pre-trained Neural Network GAN-BERT

Acknowledgements

I would like to express my deepest gratitude to Dr. Terry Walcott, my supervisor for his support, knowledge, patience and encouraging during this process. I am extremely grateful for your assistance and suggestions throughout my thesis. At the same time, I would like to send my sincere thanks to all the teachers in the computer department who have helped me in my master's study.

Last but not the least important, I owe more than thanks to my family, my mother and father for all the encouragement. Thank you for helping me during this process

Introduction

With the rapid development of information technology, network has become an important way for more and more people to get information. This information is transmitted in text, pictures, audio and video. This phenomenon has doubled since the advent of social media and its rapid expansion worldwide. Various forms of social networks, such as blogs, forums and microblogs, connect people around the world through information. As a result, the content of network information expands rapidly. Corresponding network information carriers are also growing almost endlessly. Text data tends to grow the fastest and most out of order. These text data contain valuable feedback on products, services, policies and news. These data provide policy makers with public opinion data that is highly relevant to their interests. This makes part of the network text data extremely valuable to relevant practitioners. But extracting effective information from this confusing text data is a very challenging task. If these data are analyzed directly by human, it will bring enormous time and money pressure to the corresponding enterprises or organizations.

Therefore, for enterprises and departments, the best way to extract effective information from this confusing data is to automatically understand and analyze the text. Text understanding is one of the two main core tasks in natural language processing. Natural language processing can be classified into four categories in application: sequence labeling, classification tasks, sentence relationship judgment, and generative tasks. Classification tasks are commonly used in network text data processing. Among them, affective analysis and position detection are two commonly used tasks for network text classification.

Emotional analysis and position detection are two similar text classification tasks. Both are considered excessive steps in text understanding. The difference between the two is reflected in the understanding of the text content. For emotional analysis, the main extract is the emotional-related text description in the text content. For position detection, the relationship between supporting and opposing views and topics in the text needs to be analyzed. In other words, in position detection, researchers pay more attention to the cognitive dimension of the attitudes and beliefs expressed in the text. (Stance Detection and Analysis in Social Media) In most tasks, the position detection task is more complex than the emotional analysis task, and to some extent, the position analysis can be considered a further task of emotional analysis.

In short, position detection is a text classification task that classifies a piece of text by understanding the expression of positions and judgments about an indefinite proposition. Position detection is widely used in the classification and recognition of network text data, such as election/referendum results prediction, information

retrieval, rumor detection and false news detection. Therefore, recent research has a high interest and heat for position detection.

research status

Early Research on Position Detection

Compared with emotional analysis, the study of position detection started relatively late. The main features of the early position detection research are the text style and labeling characteristics of the dataset used, as well as the type and characteristics of the position detection classification.

Earlier research projects had some articles that were collected and classified. In Hasan's and Ng's articles, they classify the pre-2013 position detection research projects into three categories: 1. Congressional debates,(Get out the vote: Determining support or opposition from Congressional floor-debate transcripts) 2. Corporate meetings and discussions(Support or oppose? Classifying positions in online debates from reply activities and opinion expressions), and 3. Online social, political and ideological debates(Predicting stance in ideological debate with rich linguistic knowledge). And online debates about products. Then in 2014, position detection experiments for tweets appeared(Identifying users with opposing opinions in Twitter debates), and the number of studies on tweets increased dramatically. Research on network ideologies and social debates has gradually emerged. In addition, early position detection studies were not popular because they lacked a large number of datasets. Among them, the exploration of the use of machine learning algorithm and statistical algorithm in position detection provides a lot of ideas for future research. But what really limits the development of position detection is the paucity of datasets.

This changed after several major competitions after 2016. The three most important games for position detection are (1) SemEval-2016 shared task: English tweet position detection(SemEval-2016 task 6: Detecting stance in tweets), (2) NLPCC-ICCPOL-2016 Chinese microblog position recognition shared task(Overview of nlpcc shared task 4: Stance detection in chinese microblog), and 3.2017 Spanish and Catalan tweet position detection shared task(Overview of the task on stance and gender detection in tweets on Catalan independence at IberEval 2017). These three contests provide a convenient social media chat dataset for Twitter position detection in three languages. In addition, these three competitions attracted more people to explore better position detection algorithms. It provides more ideas for further research.

Research Status of Position Detection

Position detection has been developed to a more in-depth study in many fields and directions. Therefore, for a more comprehensive description of the development of position detection, the research content of position detection is often divided into several directions. There are three main classification criteria: target type, text granularity, and classifier method. It can be divided into single target, multiple target and cross target according to different target types. The text granularity can be divided into sentence level, text level and debate text. Classifier methods are divided into traditional machine learning, subject model, in-depth learning and two-stage methods. Since this paper mainly discusses classifier methods, only the classification of classifier methods is introduced.

In the early stages of research, most of the stand detection methods were based on traditional machine learning. Models based on these traditional machine learning methods often need to be combined with linguistic rules to construct feature combinations manually. They need to have a certain understanding of linguistic knowledge, but also need to consume a lot of manpower to extract these linguistic features. Text features can be divided into three research directions: simple text features, emotional features and mixed features.

1. Simple Text Features: This series of features includes lexical features, word frequency-inverse document ratings, word vectors, topic model-related features, named entities, dependencies, syntax rules, and so on. Sun et al. (Exploring various linguistic features for stance detection) demonstrated that these linguistic features can improve the performance of stand detection by analyzing the role of semantic, lexical, morphological and syntax features in the task of stand detection on Chinese Weibo.

2. Emotional characteristics: Emotional detection is a similar text categorization task for position detection. Its characteristics can provide certain user emotional tendencies for position detection, which are related to user emotional tendencies to a certain extent. To demonstrate the relationship between emotional information and the tendency to stand for a given goal. Emotional dictionaries are an important means of combining emotional characteristics with position preferences. Mohamad(Stance and sentiment in tweets) et al. first published a label dataset that combines emotional information with position tags. Based on this dataset, Ebrahimid(A joint sentiment-target-stance model for stance classification in tweets) et al. incorporated emotional polarity into their goals and positions, modeled their interactions using a graph model, and established a logarithmic linear frontal affection-goal-position joint model.

3. In addition to these more easily acquired features, there are some features that are not reflected in the text to help position detection. For example, Din Yujie(Stance detection in Chinese microblogs via fusing multiple text features) et al. used the bag model and word vectors and word vectors which were integrated into synonym

dictionary to do position detection using SVM, RF and decision tree respectively. Finally, they fused all the classifiers in the later stage, and obtained the best results of Chinese Weibo position detection task (NLPcc2016 Task4). LI Zi-yi et al. used convolution to mine similar text position information under the same user and added attention mechanism to enhance text user-perceived representation. They proposed a user-related position detection model (USDm) and obtained good experimental results in H&N14 dataset.

For a model, an effective classification of known text features is required. In this aspect, the classifier of the model needs to be studied. In position detection, the use of classifiers is similar to that of common NLP classifiers. SVM is the most commonly used and often set as the baseline model classifier. The second most commonly used traditional machine learning classifier is logistic regression. In addition, there are Naive Bayesian, Decision Tree, ANN, Maximum Entropy classifiers.

Position Detection Based on Theme Model

Theme model is a classifier commonly used in opinion mining tasks such as feature recognition. This method can associate the target with the important keywords in the text and solve some problems that are difficult to judge by other methods such as text implication. However, due to the sparse and discrete textual features in thematic models, more training corpus is required, and the performance is poor in multi-target position detection tasks. Gottipati(Staying informed: supervised and semi-supervised multi-view topical analysis of ideological perspective) et al. proposed an implicit thematic model that integrates information extraction and emotional analysis into the model, uses the hierarchy of arguments in the Encyclopedia of Debate dataset to construct themes to infer topics and positions in the field of social and political debate, and finally demonstrates the advantages and disadvantages of the model through qualitative and quantitative assessments.

Position Detection Based on Deep Learning

As a better way than traditional machine learning in most cases, in-depth learning has a different application in natural language processing than computer vision. There are two common training methods for position detection based on in-depth learning. One is to randomly initialize or use corpuses such as Google News and Crisis Encyclopedia embedded with training words, the other is to use the pre-training language model to learn the background knowledge contained in large-scale unsupervised corpus.

Convolutional neural networks are undoubtedly the most widely used for in-depth learning. Convolutional neural network was originally used in computer vision. Because CNN can effectively extract the prominent features of training set, it has a good effect in text classification tasks such as text emotional classification, false

news detection and so on. But some scholars task convolution neural networks have poor ability to capture grammar and syntax information of the sentence. Turn to RNN as the main classifier. However, due to RNN's inability to capture long-distance dependencies and the disappearance of gradients, short-term and long-term memory networks have been proposed and widely used in text classification problems. LSTM networks are even used as application layers by many set models.

Another very popular processing method in natural language processing is attention mechanism, which was first proposed in the field of visual images. It has been widely used in deep learning-based natural language processing in recent years, and is often used to form new set models combining other deep learning models. In addition, the large number of attention-based transformer models proposed by Google in 2017 have attracted considerable attention in natural language processing, and in 2018, a pre-training model, bert, built by multiple transformer models, has greatly improved the model performance of natural language processing tasks.

Antagonistic Neural Network for Natural Language Processing

Recent advances in deep neural networks have led to significant performance optimization in a range of different areas. However, with more in-depth study by researchers, more and more problems emerge. One of these concerns is whether the robustness of the in-depth learning model is good enough. Robustness is a very important index of the neural network, which indicates whether the deep learning network model is stable enough. However, more and more studies indicate that the in-depth learning network model incorrectly judges some inputs under the influence of small perturbations to the input data. This aspect of research is also known as resistance machine learning. There are two main branches in this field, one is Resistance Attacks, designed to generate resistance samples for deceptive target models; The other is defense against resistance, which aims to build models that are less vulnerable to attack.

Antagonistic neural networks were originally used in image processing. Initial advocates of antagonistic neural networks believe that due to the discreteness of natural language processing data, antagonistic neural networks are more difficult to apply to natural language processing. Lantao Yu(Seqgan: Sequence generative adversarial nets with policy gradient) proposed seqgan in 2016, using the Monte Carlo method of reinforcement learning to solve Gan. This reinforcement learning method handily solves the problem of backpropagation failure of discrete data. Subsequently, a series of studies have divided the study of NLP's defensive attack into four granular attacks: character-level attack, word-level attack, sentence-level attack and mixed-level attack. In addition to granularity, attackers'mastery of target information divides text attacks into three categories: white, gray, and black. Operator selection for fine-grained text attacks can be divided into gradient-based attacks and importance-based attacks.

With the development and application of DNNs in the field of NLP, the research on text data countermeasures and poisoning attacks has also received widespread attention. According to the different granularity of text data perturbation, the existing NLP related attack methods are classified, mainly including character level attack, single word level attack and sentence level attack. Comparing the above three kinds of attack perturbations with different granularity, the character level and sentence level perturbations produce poor text quality, The former will destroy its syntax and reduce the readability of the confrontation text, but the attack success rate is high; The latter makes a great difference between the counter text and the original input, and the attack effect is relatively general. While the word level attack has a better performance in the control of the quality and effectiveness of the counter text and the success rate of the attack

Character level attack

The character level attack mainly adds disturbances to the characters in the original input, and the common methods include character deletion, modification, addition, replacement, etc. the quality of the counter text generated by the character level disturbance is poor, which destroys its syntax [Combating adversarial misspellings with robust word recognition], so this kind of attack method is less applied.

In the text classification task, Gao et al. [Black-box generation of adversarial text sequences to evade deep learning classifiers] proposed an algorithm deepwordbug, which can generate hostile text sequences and mislead the deep learning model by taking advantage of the security vulnerabilities in the NLP model. This algorithm can effectively generate micro text perturbations in black box settings, thus forcing the NLP model to misclassify text input. The author also designed a new scoring strategy for the deepwordbug algorithm, To find the most important words to modify, so that the NLP model makes wrong predictions. The deepwordbug algorithm applies simple character level conversion to the highest ranked words to minimize the editing distance of disturbance and change the original classification. The results show that deepwordbug reduces the prediction accuracy of the current latest NLP model, and reduces the performance of the model by generating antagonistic sequence text.

Word level attack

Unlike character level attacks, word level attacks perturb words in the original input. The main method is word replacement, including word replacement methods based on word vector similarity [Is bert really robust? a strong baseline for natural language attack on text classification and entailment], synonyms [Generating natural language adversarial examples through probability weighted word saliency], semaphores [Word-level textual adversarial attacking as combinatorial optimization], language model scores [Generating fluent adversarial examples for natural

languages]. Word level attacks have better performance in resisting text quality and effectiveness control and attack success rate. In addition, using language models for control, The smoothness and fluency of the generated confrontation text can also be guaranteed.

Among the word level attack methods, typical white box attacks such as the shared word replacement attack and textfooler attack need to obtain the detailed information of the target model; While common black box attacks, such as word substitution attacks based on semantic primitives, use the mobility of the text to attack. In addition, researchers also study other attack methods in white box scenarios and black box scenarios, such as attacks based on relationship analysis perturbation. This section will introduce them from three aspects: word level white box attacks, black box attacks and other attacks.

Sentence level attack

The sentence level attack regards the original input of the whole sentence as the object of disturbance, aiming to generate a counter text with the same semantics as the original input (at least the real label of the current task does not change), but makes the threatened model judge the change. Common sentence level attack methods include re decoding after encoding, adding irrelevant sentences, paraphrasing, etc. the following will introduce the sentence level attack methods

Wang et al. [Cat-gen: Improving robustness in nlp models via controlled adversarial text generation] proposed a controlled adversarial text generation (catgen) model, which generates adversarial text through controllable attributes unrelated to known task tags given the input text. Compared with many existing adversarial text generation methods, this method can generate more diverse and smooth adversarial text, And it has strong robustness to model retraining and different model architectures. Therefore, the generated combat text can be used for combat training to improve the model, but how the model implicitly extracts attributes needs to be explored. Therefore, Li et al. [Bert-attack: Adversarial attack against bert using bert] proposed a more efficient combat text generation method, bertgattack, By using a pre trained mask language model to produce high-quality combat text, the author contrasts Bert with the model finely adjusted in the downstream task and other DNNS, so as to mislead the prediction results of the target model. This method is superior to the most advanced attack strategy in terms of success rate and interference percentage. At the same time, the generated combat text is smooth and semantically unchanged, but in terms of interference rate, attack success rate There is still much room for improvement in grammatical correctness and semantic consistency

NLP oriented defense method

The basic purpose of generating confrontation texts for NLP models is to use these confrontation texts to enhance the robustness of the models, so as to prevent malicious confrontation texts from interfering with model predictions. This section mainly introduces text defense methods in the NLP field, which are divided into confrontation training, privacy protection defense and detection defense. Because of the endless emergence of attack methods, the defense work against texts has become more important and difficult. At present, the defense research around NLP still needs further exploration.

Adversarial training is an important way to enhance the robustness of neural networks. In the process of adversarial training, some adversarial texts with small perturbations will be mixed in the training samples, and then the neural network will adapt to such perturbations, so as to be robust to adversarial texts. In the NLP field, Miyato et al. [Adversarial training methods for semi-supervised text classification] introduced adversarial and virtual adversarial training into text classification, and by adding regular terms to the model, so that the output distribution of a sample is the same as the output distribution after perturbation. This method has a good defense effect in text classification tasks, and can use a large number of unlabeled samples to regularize the model through confrontation training to improve the effect. In addition, in order to overcome the distribution difference problem, the framework obtains antagonistic sentences by adding some small perturbations to each word in the sentence, and generates a character level word representation for the words in the sentence.

Text privacy protection defense

In order to protect the privacy security of NLP model and prevent user information from being stolen, Li et al. [Towards robust and privacy-preserving text representations] proposed an unbiased representation privacy protection learning method. The author used the method of Ganin et al. [Domain-adversarial training of neural networks] to train a deep model with adversarial learning ability to explicitly hide individual privacy information. At the same time, it still supports high-quality NLP tasks. Experimental results show that the protection method can hide most of the user's personal information and achieve the purpose of data protection in NLP tasks.

For the problem of text data privacy protection, Coavoux et al. [Privacy-preserving neural representations of text] trained a classifier that predicts binary attributes, applied the generative adversarial network (GAN) to text privacy protection, and directly optimized the probability of training samples based on the character level language model. The experimental results show that the privacy protection on the trustpilot dataset has been improved, while the accuracy against the original text has

only slightly decreased, It improves the tradeoff between practicality and privacy of text representation

To sum up, the privacy protection method in NLP can effectively protect text data information and prevent the harm caused by the theft of user privacy. However, its practicality in the physical world still needs further research

Defense based on confrontation detection

In addition to enhancing the robustness of the training process and the privacy security of text data through adversary training and privacy protection technology, the defense against NLP models can also achieve resistance to model attacks caused by synonyms or spelling errors by detecting adversary texts

Gao et al. [Black-box generation of adversarial text sequences to evade deep learning classifiers] used Python automatic correction toolkit v0.3.0 to detect malicious input. Python automatic correction tool improved the prediction accuracy of the model from an average of 13.34% to an average of 74.17%, Belinkov et al. [Synthetic and natural noise both break neural machine translation] used Google's spell checker to correct spelling errors caused by hostile methods. Google's spell checker improved the bilingual evaluation understudy (Bleu) score of the French corpus from 16.7 to 21.4, and the Bleu score of the German corpus from 18.6 to 25.0, Li et al. [Textbugger: Generating adversarial text against real-world applications] used context aware spell checking operations to achieve similar effects. However, the experimental results show that this detection method is effective for character level modifications, but partially effective for word level attacks, which may be caused by differences in modification strategies, Spell checking methods are also not applicable to confrontation texts based on other languages.

Methodology

Because the deep neural network mentioned above is vulnerable to attack whether it is applied to image processing or natural language processing. Therefore, in order to effectively prevent the deep neural network from failure in the case of relatively simple attacks. Confrontational defense has become a hot research direction of researchers. Compared with the field of image processing, the antagonistic defense means in natural language processing can often improve the robustness of NLP models more effectively, that is, to some extent, improve the performance of models in test sets. Therefore, in recent NLP competition tasks, adversarial defense has become a major means to obtain high scores. Adversarial defense methods mainly include adversarial training, privacy protection defense and detection defense. Confrontational training is considered as one of the most effective defense methods among these methods. The basic idea of this method is to use the

confrontation text obtained by the attack method and the normal text to mix them, and then use them as a new training data set to fine tune the target model. Basic idea of confrontation training min max formula:

$$\min_{\theta} E_{(x,y) \sim D} [\max_{r_{adv} \in S} L(\theta, x + r_{adv}, y)]$$

That is, it is necessary to find a set of counter samples in the sample space within a certain disturbance range that maximizes loss. This counter sample is composed of the original sample X and the disturbance term r_{adv} . And the min function indicates that in the face of the sample set with such data distribution, the expected loss of the model on the counter sample set should be minimized by updating the model parameters. (Adversarial training methods for semi-supervised text classification)

As far as the current antagonistic training means are concerned. Most of them obtain new false text samples by applying limited perturbation to the original text samples, which are provided to the discriminator for recognition. There are generally two mainstream methods to add restrictions to the disturbance. One is that the disturbance added to the text can not be recognized by people to a certain extent, and the other is to replace the content of the original text sample through antagonistic word conversion and similar methods to form new synonymous sentences or text approximate sentences.

Either of these two methods is based on the idea of forming a model against sample attacks through certain perturbations mentioned above. The simplest method based on this idea is to use gradient rise to find the best disturbance. The most commonly used basic method for this practice is the fast gradient method proposed by Goodfellow in 2017. This method generally has four steps: 1. Calculate the normal loss in the forward direction, and calculate the normal parameter grad in the backward direction. 2. Calculate the disturbance r according to the confrontation training formula and grad, and attack the embedding weight. 3. Calculate adv loss and adv grad according to the model weight after the attack. 4. Restore the embedding initial weight, accumulate the original grad and adv grad, and update the model weight.

The formula of disturbance is as follows:

$$r_{adv} = \epsilon \cdot g / \|g\|_2$$

$$g = \nabla_x L(\theta, x, y)$$

Where θ is the parameter of the model, X is the input of the model, y is the target related to x , and L is the cost for training the neural network.

Although the antagonistic training method based on gradient can improve the generalization of text very conveniently and effectively. However, this training

method is easy to be attacked by antagonistic word replacement and suffer a large performance loss. Therefore, the research on the confrontational training of offensive word substitution is another very popular research direction of confrontational training.

This method is usually based on very effective countermeasures or deep learning of the original text. This method is aimed at the training of antagonistic attack methods. As mentioned above, adversarial attack methods are often divided into character level attack, word level attack, sentence level attack and mixed level attack. Among them, antagonistic word replacement is one of the most effective methods to attack high-level neural models(Defending pre-trained language models from adversarial word substitutions without performance sacrifice). Therefore, AWS based confrontational training is one of the most widely used training ideas.

Randomization is a very effective method for AWS adversarial defense. It was originally used in computer vision as a defense measure, but because text input is discrete, ye et al. Proposed in 2020 to replace words in sentences with synonyms to achieve randomization in NLP. Researchers also pay attention to the method of replacing synonyms of words in sentences. Heuristic attack methods replace each word attack model iteratively, which often makes it difficult for static defense strategies to defend against this dynamic process. Attacking based on the attention score of each word in the sentence is also a more effective attack method.

However, this method is often aimed at improving the robustness of the model, which will lose the prediction accuracy of the model to a certain extent. In addition, most of these defense methods either have strong limitations on the attack space to prove their robustness, which requires a lot of computing resources in the training and reasoning process.

Due to the limitation of equipment, this paper can only choose the method with low requirements for computing resources for experiments.

Technical and theoretical knowledge

Improving the robustness of neural networks is a very popular and important trend in artificial intelligence research in recent years. Since the concept of antagonistic neural networks was proposed, researchers have improved their understanding of the robustness of neural networks to a new level. Deeper and wider neural networks have become better choices for complex problems. This chapter mainly introduces the relevant technical knowledge involved in the position detection and classification task, and the available antagonistic training methods.

Antagonistic neural network can improve the generalization ability of object model

to a certain extent. For natural language processing, more antagonistic training methods are black box antagonistic training such as FGM. However, when the number of training set data is small, the black box confrontation training is difficult to help the generalization ability of the object model. One of the better solutions is the semi supervised method. Improve model generalization with less data.

Semi-supervised GANs

Semi-supervised GANs is a semi supervised generative GAN model first proposed by salimans in 2016. The "real" samples are divided into one of the target classes, and the generated samples are divided into $k + 1$ classes. Specifically, it is assumed that D and G represent a discriminator and a generator, respectively. And P_D and P_G represent the real data distribution and the generated samples, respectively. If it is necessary to train a semi-supervised k-class classifier, the target of D needs to be expanded to the following extent.

P_M is defined as the possibility that a common sample X in the model M is considered to be a wrong class. P_M is the probability that a normal sample is considered the correct class. And the loss function of D is defined as:

$$L_D = L_{D_{sup}} + L_{D_{unsup}}$$

$$L_{D_{sup}} = -E_{x,y \sim p_d} \log[p_m(\hat{y} = y | x, y \in (1, \dots, k))]$$

$$L_{D_{unsup}} = -E_{x \sim p_d} \log[1 - p_m(\hat{y} = y | x, y = k + 1)] \\ - E_{x \sim G} \log[p_m(\hat{y} = y | x, y = k + 1)]$$

Where L is expressed as the error of the error instances misallocated to the true k classifications. L indicates the error in identifying the correct instance as in the error category or as not identifying the error instance.

In addition, in order for G to be able to generate samples similar to those sampled in the true distribution p_d . G should produce data as close as possible to real data statistics, that is, the average sample generated by G in a ruffian should be similar to the real prototype sample, so the loss of feature matching for G should be:

$$L_{G_{feature\ matching}} = \left\| E_{x \sim p_d} f(x) - E_{x \sim G} f(x) \right\|_2^2$$

At the same time, G loss needs to consider the error caused by the false samples correctly identified by D , i.e.

$$L_{G_{unsup}} = -E_{x \sim G} \log[1 - p_m(\hat{y} = y | x, y = k + 1)]$$

The $\mathcal{G}$ loss is $L_{\mathcal{G}} = L_{\mathcal{G}_{feature\ matching}} + L_{\mathcal{G}_{unsup}}$

Bert model

The Bert model is a pre training language proposed by Google in 2017 (Attention is all you need). The proposed method has greatly promoted the progress of NLP technology. At the same time, it has gradually formed a new language processing process: pre training using large-scale and high-quality corpus of multi-human languages, building a language representation model suitable for a variety of NLP tasks, and fine tuning specific tasks on this basis to achieve wide application.

Since the Bert model is built based on transformer, its model input is also different from other models. In Bert, the input vectors are summed by three different embedding methods, which are:

1. Wordpiece embedding: vector representation of the word itself. Wordpiece refers to dividing a word into a limited group of common sub word units, which achieves a good balance between the effectiveness of the word and the flexibility of the character.
2. Position embedding: encode the word position information into a feature vector. Due to the lack of time-series neural networks such as RNN or LSTM in the Bert structure, the position information of the sequence cannot be obtained, so a position embedding needs to be built.
3. Segment embedding: vector representation used to distinguish two sentences.

In order to make the Bert model flexibly adapt to various downstream tasks related to text analysis, it is necessary to generate a continuous and fixed length text stream (starting with the classification mark [CLS] and separated by [SEP] at the end of the sentence) by splicing a single sentence in the document, and randomly mask the tokens in the sentence as input in a certain proportion.

network structure

Although Bert is based on transformer, it only uses the encoder part of transformer, and its overall framework is stacked by the encoder of multi-layer transformer. The function of the encoder is to obtain the semantic features of the text, such as the context features and the syntactic dependencies, which is attributed to the fact that it is composed of N multi head self attention layers and feedforward neural network layers. For the input $x = \{X_1, X_2, \dots, X_i\}$ (sentence sequence, X_i represents the

ith word), the output of each sub layer in the encoder is 1. By summing the sub layers adjacent to the normalized connection, the output word embedding vector list is transferred to the next layer.

In this encoder, the main function of each attention is to re encode the target word through the correlation between the target word and all the words in the sentence. Therefore, the calculation of each attention includes three steps: calculating the degree of correlation between words, normalizing the degree of correlation, and obtaining the code of the target word by weighted summation of the degree of correlation and the codes of all words.

The self attention layer is created to solve the problem that network structures such as RNN and LSTM, which are often used to process serialized data, cannot speed up parallel computation in GPU. In addition, the self attention model can also solve the problem of long-distance dependence in the traditional RNN model. Through attention, the distance between two words that are far away can be reduced to 1 to directly calculate the relevance of words.

Calculation process

For each token encoding, the semantic information of the received sequence x is divided into the following steps:

1. Perform matrix transformation on X to obtain Q, K, V
2. Calculate self attention function;
3. Project Q, K, V into H subspace through independent linear transformation matrix $W_1, \dots, W_h$
4. The feature vector Z is generated by stitching h different attention results.

The formula is expressed as follows:

$$Z(Q, K, V) = \text{Concat}(\text{head}_1, \dots, \text{head}_h)W^0$$

Where, q is the output of the upper sub attention layer, and K and V are the attention vector sets output by the encoder.

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Feedforward neural network layer: to enhance the expression ability of transformer, calculate the nonlinear hierarchical characteristics:

$$\text{FN}(Z) = \max(0, ZW_1 + b_1)W_2 + b_2$$

Through this layer, the output vector Z in the formula is changed nonlinearly and the

vector length remains unchanged.

Summation and normalization layer: calculate the standardized output of each layer of data:

$$\mu = \frac{1}{d} \sum_{i=1}^d x_i, \sigma = \sqrt{\frac{1}{d} \sum_{i=1}^d (x_i - \mu)^2 + \varepsilon}$$
$$x_i' = f(g \frac{x_i - \mu}{\sigma} + b)$$

Where, f is a nonlinear function, g and b are model parameters. The short cut idea in the residual network is adopted to add the input and output x_i and output x_i' of each layer to realize the effective superposition of the network structure and make the result smoother.

Training tasks

(1) masked language model

Some words are masked randomly, and then the word is predicted by context. In Bert, 15% of wordpiece tokens will be randomly masked, 80% of these 15% tokens will be replaced by [mask], 10% will be replaced by a random word, and 10% will remain the same. This design makes the model have the ability to capture the context relationship, and can be conducive to token level tasks such as sequence annotation.

(2) next sentence prediction

For 50% of the sentences in the corpus, select the corresponding next sentence to form the upper and lower sentences together as the positive sample; The remaining 50% of the sentences randomly select a sentence other than the next sentence to form the upper and lower sentences as the negative sample. This setting is beneficial to sense level tasks.

(3) Fine tuning test stage:

The Bert series model includes a general language representation model and a process of fine tuning the model for various NLP tasks. In the last stage, based on the multi lingual universal language representation model obtained by pre training, the word vector of each word in the text fused with the context semantics can be dynamically learned. In the fine-tuning trial stage, it is necessary to train on the new corpus according to the specific NLP task, and refine the network structure (update the parameters of the model) to obtain the final model.

Taking a text two classification problem as an example, we represent the input text as a sequence of [CLS] + (masked) tokens after preprocessing. First, each word (including [CLS]) is output as a word vector with a length of 768 through the pre training model, and a full connection layer is added outside the model. After dot multiplication of the embedding matrix and the output vector, it is converted into the dimension of the word, and then the probability of classifying the sentence sequence is calculated through the softmax layer, Thus, a sentence text classifier is constructed.

The fine-tuning of multi classification problems is similar to the above method. The so, Max function is used to judge the probability of each category, and the 768 dimensional word vector is output as an n-dimensional real vector (n = |L| is the number of classification labels, and the sum of all elements in the vector is 1). The calculation formula is as follows:

$$P_i = \frac{e^{V \cdot c_i}}{\sum_{j=1}^{|L|} e^{V \cdot c_j}}$$

Where, V is the weight matrix (including the pre training model parameters and the new task specific parameters, and the weights are adjusted according to the training data), C is the [CLS] output of the multi label classification, and the inner product $V \cdot C$ represents the scores under different classifications. Use softmax to calculate the conditional probability P_L , and select the category with the highest probability as the classification label.

Multi-Task Deep Neural Networks

In many cases, the accuracy of using only the pre training model to complete the downstream tasks cannot meet the requirements of application. There are many ways to improve the performance of the model. In addition to putting forward excellent methods, deepening and widening the neural network (deep learning), increasing the number of data sets (pre training model) and increasing the objective function (multi task learning) are all means that can be used to improve the effect. For position detection, its classification is often based on the position classification of a topic, and the mainstream training data set often contains classification tasks of multiple topics. In this case, the application of multi task learning mechanism can more effectively train sentence level text classification models and improve the accuracy of language text classifiers.

Multi-task learning is similar to solving multi-objective optimization problems in mathematics. It is a learning mode and model structure in which multiple tasks share model parameters. The difference from traditional single task learning is that MTL models multiple related tasks at the same time. Refer to the following figure:

This is a transfer learning method. Through joint (simultaneous) training of multiple related tasks, it can provide additional training information for the main task and enhance the generalization power of the model, that is, the adaptability of the model to unknown samples. The effectiveness of this method lies in:

Training samples can be increased, and the noise of different tasks is different. Learning multiple tasks at the same time can obtain more general expression
The model can judge whether the learned features are really effective through other tasks.

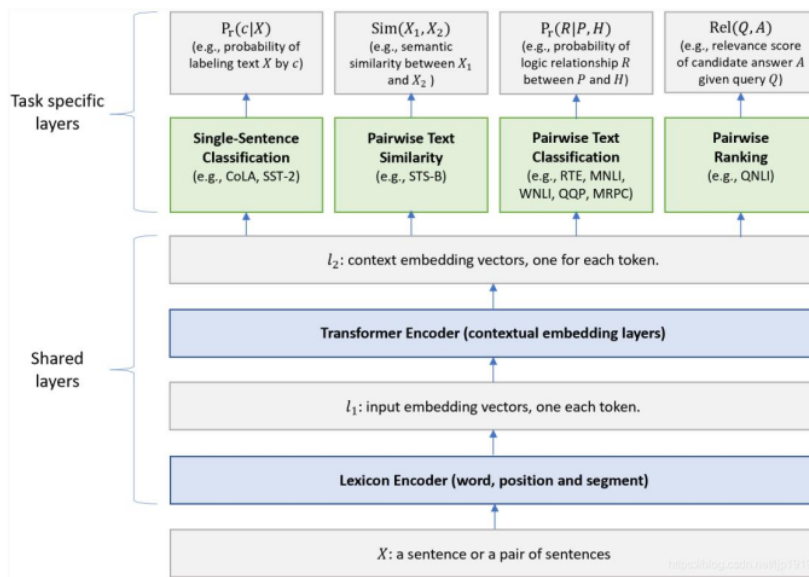
Features that are difficult to learn in one task can be learned in other tasks

Multi-task learning tends to make models pay attention to information expression that other models also pay attention to

To some extent, it can be regarded as regularization

To some extent, multitasking learning has received great attention in recent years, just like pre training learning. Therefore, the combination model of the two should be applied. Xiaodong Liu proposed a multi task learning pre training model based on the Bert model in 2019(Multi-task deep neural networks for natural language understanding).

The architecture of mt-dnn is shown in Figure 1 above. The bottom layer is shared among all tasks, and the top layer represents the output of a specific task. Input x is a word sequence (a sentence or a pair of sentence pairs combined together). First, it is expressed as a series of embedding vectors in l_1 , and each word corresponds to a vector. Then it is input to the feature extractor transformer, which generates a series of context representations (context embedding sequences) in l_2 through self attention.



Lexicon Encoder (l_1) :

Is a token of length M . The first character is always the [CLS] character. If there are two sentences, add a [SEP] character in the middle

Lexicon encoder is a word vector layer that maps input x into a series of embedded vectors. There are also operations such as position encoding and segment encoding.

Transformer Encoder (l_2) :

Use transformer to map the input vector representation x into the context representation C . Which shares parameters among different tasks. Bert needs to learn the representation method through pre training, and then use it for other tasks through fine-tuning, while MT-DNN uses multi-task objectives to learn its representation.

The training program of mt-dnn includes two stages: pre training and multi task fine tuning. The pre training phase follows the Bert model. The parameters of lexicon encoder and transformer encoder are learned through two unsupervised prediction tasks: masked language modeling and next sentence pre prediction.

In the multi task fine-tuning phase, we use minibatch based stochastic gradient descent (SGD) to learn model parameters (that is, parameters of all shared layers and task specific layers). The algorithm flow is shown in the following figure. For each epoch, select a mini batch BT (among the 9 GLUE tasks), and then update the model parameters of the specific task K. This approximately optimizes the sum of all multitasking.

Perform Mini batch on the data set of each task to get D_T . B_T is a mini batch of task T, and then uses this data to calculate the loss value of classification, regression

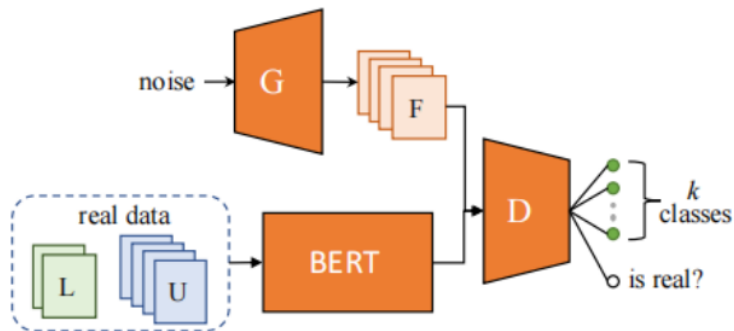
and rearrangement tasks, and then uniformly calculates the gradient descent and updates the model parameters.

Result

The position detection task for tweets is a natural language classification task that needs to process a large amount of data. When hot public opinion appears, a large number of relevant comments often appear in a short time. In addition, in the long run, with the development of hot issues, the public's view of the same event will also change. In this case, we want to analyze the changing public opinion and update the model in time, but labeling the data set often takes a lot of time. While ssgan can alleviate the lack of labels to a certain extent, and also improve the robustness of the model.

model building

Danilo published a Bert model combining ssgans in 2020. Because of the discreteness of text data, general antagonistic training is based on embedding. The model adds the ssgans framework in the fine-tuning stage of the Bert model. A discriminator D for generating text samples by classification and a reverse acting generator G are introduced. Where G is a multi-layer perceptron that receives the 100 dimensional noise vector extracted with input n and outputs the generated sample H . Discriminator D is another MLP, which receives original text samples and generates text samples, and outputs them as $K + 1$ logits vectors.



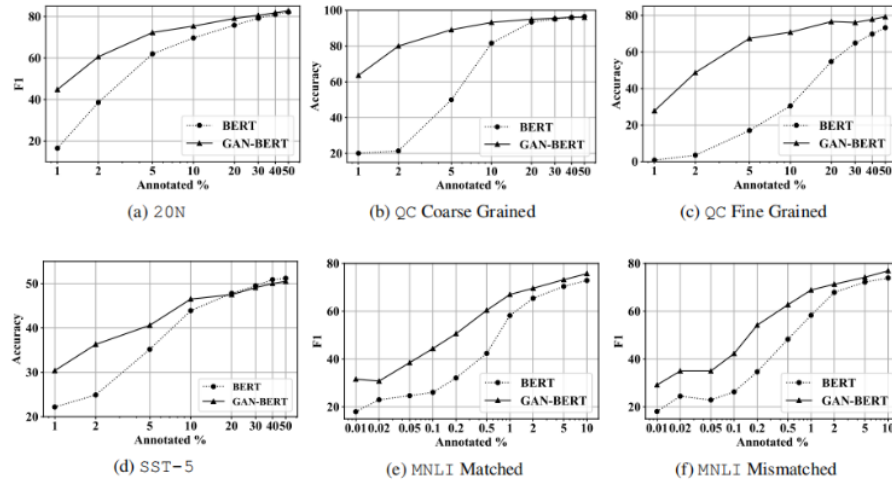
In the forward step of this model, the classifier classifies the input text samples. For

real examples, D should classify them into one of K classes. For generated samples, D shall classify them into $K + 1$ category. In the process of its discovery and propagation, the samples generated by G have an impact on $loss_D$ and $loss_G$. When discriminator D cannot find the samples generated by G , D will be punished, and vice versa. When updating the discriminator D , the model will fine tune the Bert weight. After the training is completed, the generator will be discarded and the Bert model will complete the remaining classification work.

Conjecture

From the original Gan Bert, the author gives some experimental results. It can be seen from the figure that compared with the Bert model, the Gan Bert model has better performance when most of the data sets have fewer labeled samples. However, with the increase of the proportion of the labeled samples in the samples, the gap between them decreases rapidly and tends to be close.

From this result, it can be seen that the performance of Gan Bert model on data sets that are all labeled text samples is not good, but it can improve the text classification performance of Bert model to a certain extent in the case of less training samples. In the task of position detection, which is prone to sample imbalance, the Bert model is optimized to a certain extent.



Discussion

It can be seen from experiments that the application of gan-bert model in multi task data sets is relatively complex, and it needs to train different task data sets. The

original author of gan-bert model proposed mt-gan-bert model in 2021 by combining mt-dnn model with ssgan. This model is very similar to gan-bert model, and Bert is still used as the classification model. However, in the forward step of this model, the batch belonging to the t-th task is extracted for a gan-bert training process, and then the batch of one of the tasks is selected for a gan-bert training process. That is, the Bert model is trained by the gan-bert method through the data set batches of multiple tasks, so as to obtain better performance.

Similarly, the original text is trained and recognized on several different data sets. Finally, the following five different data tables are obtained.

Task	BERT model		MT-DNN		Difference	
	MF1	ACC	MF1	ACC	MF1	ACC
HaSpeeDe	77.79%	80.13%	77.73%	80.67%	-0.06	+0.53
AMI A	83.70%	83.87%	84.70%	84.93%	+1.01	+1.07
AMI B	80.41%	80.64%	84.97%	85.13%	+4.56	+4.48
DANKMEMES	72.96%	73.17%	74.77%	74.83%	+1.81	+1.67
SENTIPOLC 1	73.35%	73.03%	72.04%	72.55%	-1.30	-0.48
SENTIPOLC 2	63.85%	67.84%	59.91%	64.34%	-3.94	-3.50

Task	BERT ₂₀₀		GAN-BERT ₂₀₀		Difference	
	MF1	ACC	MF1	ACC	MF1	ACC
HaSpeeDe	61.62%	67.23%	62.70%	67.97%	+1.08	+0.73
AMI A	64.34%	65.53%	69.03%	69.03%	+4.69	+3.50
AMI B	52.80%	51.94%	56.09%	55.98%	+3.29	+4.04
DANKMEMES	53.38%	53.00%	56.42%	56.67%	+3.03	+3.67
SENTIPOLC 1	61.23%	65.70%	63.62%	67.97%	+2.39	+2.27
SENTIPOLC 2	41.42%	49.66%	48.69%	54.18%	+7.27	+4.51

Task	BERT ₅₀₀		GAN-BERT ₅₀₀		Difference	
	MF1	ACC	MF1	ACC	MF1	ACC
HaSpeeDe	63.50%	68.93%	63.33%	69.30%	-0.18	+0.37
AMI A	70.24%	71.17%	72.48%	72.60%	+2.24	+1.43
AMI B	56.70%	56.65%	60.71%	58.52%	+4.01	+1.87
DANKMEMES	56.58%	57.00%	58.43%	56.17%	+1.85	-0.83
SENTIPOLC 1	58.94%	66.87%	63.02%	66.88%	+4.08	+0.01
SENTIPOLC 2	43.12%	51.92%	48.67%	54.89%	+5.55	+2.97

Task	BERT ₂₀₀		MT-GAN-BERT ₂₀₀		Difference	
	MF1	ACC	MF1	ACC	MF1	ACC
HaSpeeDe	61.62%	67.23%	63.22%	64.17%	+1.60	-3.07
AMI A	64.34%	65.53%	69.10%	68.70%	+4.76	+3.17
AMI B	52.80%	51.94%	48.76%	48.28%	-4.04	-3.66
DANKMEMES	53.38%	53.00%	51.34%	52.67%	-2.04	-0.33
SENTIPOLC 1	61.23%	65.70%	63.56%	66.58%	+2.33	+0.88
SENTIPOLC 2	41.42%	49.66%	45.45%	52.09%	+4.03	+2.43

Task	BERT ₅₀₀		MT-GAN-BERT ₅₀₀		Difference	
	MF1	ACC	MF1	ACC	MF1	ACC
HaSpeeDe	63.50%	68.93%	62.83%	67.93%	-0.67	-1.00
AMI A	70.24%	71.17%	71.81%	73.83%	+1.57	+2.67
AMI B	56.70%	56.65%	58.48%	55.46%	+1.78	-1.20
DANKMEMES	56.58%	57.00%	54.64%	55.00%	-1.94	-2.00
SENTIPOLC 1	58.94%	66.88%	65.29%	70.48%	+6.35	+3.60
SENTIPOLC 2	43.12%	51.92%	49.79%	56.18%	+6.67	+4.26

model structure

MT-DNN

Mt-dnn is a pre training model based on Bert improvement proposed by Microsoft Asia Research Institute and Microsoft Redmond voice dialogue research group in 2019. This model has better performance than Bert model in small sample multitask dataset. However, mt-dnn is not a perfect model. Mt-dnn has a lower classification ability for the same text dataset than Bert model under a certain degree of text adversarial attacks. That is, the resilience of mt-dnn model to adversarial attacks is lower than that of Bert model. This anti common sense situation occurs in most text tasks. Therefore, it is more meaningful to conduct antagonistic training on mt-dnn than that on Bert. Since the performance of the model obtained by multitasking training with Bert is poor, applying ssgans to mt-dnn may obtain better performance.

On the other hand, mt-dnn itself has poor antagonistic recovery. While ssgans will be affected by the actual samples and the model itself during training. Therefore, the performance of the generator trained by ssgans may be worse than that of the

generator trained by Bert. If the generator trained by the text features obtained by Bert is used to train mt-dnn of different tasks by classification, the model applied to a single task in a multi task may improve the performance of the multi task model in a single task to a certain extent

Conclusion

With the arguments above, it come to the conclusion that the Gan Bert model has better performance than the Bert model in the dataset with more unlabeled samples, and the mt-gan-bert model has no greater performance improvement than the gan-bert model in the multitask dataset. Based on mt-dnn and gan-bert migration training, this paper puts forward suggestions for improving the mt-gan-bert model.

AHMED, A. & XING, E. (2010), "Staying informed: supervised and semi-supervised multi-view topical analysis of ideological perspective", in Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, pp. 1140-1150.

BAO, R., WANG, J. & ZHAO, H. (2021), "Defending pre-trained language models from adversarial word substitutions without performance sacrifice", arXiv preprint arXiv:2105.14553,.

BELINKOV, Y. & BISK, Y. (2017), "Synthetic and natural noise both break neural machine translation", arXiv preprint arXiv:1711.02173,.

COAVOUX, M., NARAYAN, S. & COHEN, S. B. (2018), "Privacy-preserving neural representations of text", arXiv preprint arXiv:1808.09408,.

DIAN, Y. J., JIN, Q. & WU, H. M. (2017), "Stance detection in Chinese microblogs via fusing multiple text features", Computer Engineering and Applications, Vol. 53 No. 21, pp. 77-84.

EBRAHIMI, J., DOU, D. & LOWD, D. (2016), "A joint sentiment-target-stance model for stance classification in tweets", in Proceedings of COLING 2016, the 26th international conference on computational linguistics: Technical papers, pp. 2656-2665.

GANIN, Y., USTINOVA, E., AJAKAN, H., GERMAIN, P., LAROCHELLE, H., LAVIOLETTE, F., MARCHAND, M. & LEMPITSKY, V. (2016), "Domain-adversarial training of neural networks", The journal of machine learning research, Vol. 17 No. 1, pp. 2096-2030.

GAO, J., LANCHANTIN, J., SOFFA, M. L. & QI, Y. (2018), "Black-box generation of adversarial text sequences to evade deep learning classifiers", in 2018 IEEE Security and Privacy Workshops (SPW), IEEE, pp. 50-56.

HASAN, K. S. & NG, V. (2013), "Stance classification of ideological debates: Data, models, features, and constraints", in Proceedings of the sixth international joint conference on natural language processing, pp. 1348-1356.

HASAN, K. S. & NG, V. (2012), "Predicting stance in ideological debate with rich linguistic knowledge", in Proceedings of COLING 2012: Posters, pp. 451-460.

HSIEH, Y., CHENG, M., JUAN, D., WEI, W., HSU, W. & HSIEH, C. (2019), "On the robustness of self-attentive models", in Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 1520-1529.

HSIEH, Y., CHENG, M., JUAN, D., WEI, W., HSU, W. & HSIEH, C. (2019), "On the robustness of self-attentive models", in Proceedings of the 57th Annual Meeting of

the Association for Computational Linguistics, pp. 1520-1529.

JIN, D., JIN, Z., ZHOU, J. T. & SZOLOVITS, P. (2020), "Is bert really robust? a strong baseline for natural language attack on text classification and entailment", in Proceedings of the AAAI conference on artificial intelligence, pp. 8018-8025.

KOROTEEV, M. V. (2021), "BERT: A review of applications in natural language processing and understanding", arXiv preprint arXiv:2103.11943,.

LI, J., JI, S., DU, T., LI, B. & WANG, T. (2018), "Textbugger: Generating adversarial text against real-world applications", arXiv preprint arXiv:1812.05271,.

LI, L., MA, R., GUO, Q., XUE, X. & QIU, X. (2020), "Bert-attack: Adversarial attack against bert using bert", arXiv preprint arXiv:2004.09984,.

LI, Y., BALDWIN, T. & COHN, T. (2018), "Towards robust and privacy-preserving text representations", arXiv preprint arXiv:1805.06093,.

LIU, X., HE, P., CHEN, W. & GAO, J. (2019), "Multi-task deep neural networks for natural language understanding", arXiv preprint arXiv:1901.11504,.

MIYATO, T., DAI, A. M. & GOODFELLOW, I. (2016), "Adversarial training methods for semi-supervised text classification", arXiv preprint arXiv:1605.07725,.

MOHAMMAD, S. M., SOBHANI, P. & KIRITCHENKO, S. (2017), "Stance and sentiment in tweets", ACM Transactions on Internet Technology (TOIT), Vol. 17 No. 3, pp. 1-23.

MOHAMMAD, S., KIRITCHENKO, S., SOBHANI, P., ZHU, X. & CHERRY, C. (2016), "Semeval-2016 task 6: Detecting stance in tweets", in Proceedings of the 10th international workshop on semantic evaluation (SemEval-2016), pp. 31-41.

MURAKAMI, A. & RAYMOND, R. (2010), "Support or oppose? Classifying positions in online debates from reply activities and opinion expressions", in Coling 2010: Posters, pp. 869-875.

PRUTHI, D., DHINGRA, B. & LIPTON, Z. C. (2019), "Combating adversarial misspellings with robust word recognition", arXiv preprint arXiv:1905.11268,.

RAJADESINGAN, A. & LIU, H. (2014), "Identifying users with opposing opinions in Twitter debates", in International conference on social computing, behavioral-cultural modeling, and prediction, Springer, pp. 153-160.

REN, S., DENG, Y., HE, K. & CHE, W. (2019), "Generating natural language adversarial examples through probability weighted word saliency", in Proceedings of the 57th

annual meeting of the association for computational linguistics, pp. 1085-1097.

SUN, Q., WANG, Z., ZHU, Q. & ZHOU, G. (2016), "Exploring various linguistic features for stance detection", *Natural Language Understanding and Intelligent Applications*. Springer, pp. 840-847.

TAULÉ, M., MARTÍ, M. A., RANGEL, F. M., ROSSO, P., BOSCO, C. & PATTI, V. (2017), "Overview of the task on stance and gender detection in tweets on Catalan independence at IberEval 2017", in *2nd Workshop on Evaluation of Human Language Technologies for Iberian Languages, IberEval 2017, CEUR-WS*, pp. 157-177.

THOMAS, M., PANG, B. & LEE, L. (2006), "Get out the vote: Determining support or opposition from Congressional floor-debate transcripts", *arXiv preprint cs/0607062*,.

VASWANI, A., SHAZEER, N., PARMAR, N., USZKOREIT, J., JONES, L., GOMEZ, A. N., KAISER, Ł. & POLOSUKHIN, I. (2017), "Attention is all you need", *Advances in neural information processing systems*, Vol. 30.

WANG, T., WANG, X., QIN, Y., PACKER, B., LI, K., CHEN, J., BEUTEL, A. & CHI, E. (2020), "Cat-gen: Improving robustness in nlp models via controlled adversarial text generation", *arXiv preprint arXiv:2010.02338*,.

XU, R., ZHOU, Y., WU, D., GUI, L., DU, J. & XUE, Y. (2016), "Overview of nlpcc shared task 4: Stance detection in chinese microblogs", *Natural language understanding and intelligent applications*. Springer, pp. 907-916.

YU, L., ZHANG, W., WANG, J. & YU, Y. (2017), "Seqgan: Sequence generative adversarial nets with policy gradient", in *Proceedings of the AAAI conference on artificial intelligence*, pp.

ZANG, Y., QI, F., YANG, C., LIU, Z., ZHANG, M., LIU, Q. & SUN, M. (2019), "Word-level textual adversarial attacking as combinatorial optimization", *arXiv preprint arXiv:1910.12196*,.

ZHANG, H., ZHOU, H., MIAO, N. & LI, L. (2020), "Generating fluent adversarial examples for natural languages", *arXiv preprint arXiv:2007.06174*,.

ZHAO, Z., DUA, D. & SINGH, S. (2017), "Generating natural adversarial examples", *arXiv preprint arXiv:1710.11342*,.

Application of Semi-supervised Adversarial Training Pre-trained Neural Network in Stance Detection

GRADEMARK REPORT

FINAL GRADE

/100

GENERAL COMMENTS

Instructor

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

PAGE 10

PAGE 11

PAGE 12

PAGE 13

PAGE 14

PAGE 15

PAGE 16

PAGE 17

PAGE 18

PAGE 19

PAGE 20

PAGE 21

PAGE 22

PAGE 23

PAGE 24

PAGE 25

PAGE 26

PAGE 27

PAGE 28

PAGE 29
