Hyperparameter Optimization for Improving BERT-Based Irony Sentence Recognition

Renjian Hou

Department of Electrical Engineering and Automation, Xiamen University of Technology, Xiamen, Fujian, 361024, China
2210613128@stu.xmut.edu.cn

Abstract. Irony is a figure of speech in which the words are employed with an intended meaning that differs from their literal meaning. The ability to recognize and interpret ironic sentences can prevent misunderstandings in conversations and enhance effective communication. With the continuous improvement of Natural Language Processing (NLP) systems, many issues related to semantic recognition and human-computer dialogue have been largely resolved. However, since ironic sentences have complex semantics and are not easily understood, there are still difficulties in identifying this type of sentences. it can be seen that Irony Detection is of great importance. The experiment of this paper is to establish a recognition model for ironic sentences. Experiments use pytorch library and Bidirectional Encoder Representations from Transformers (BERT) model to train the ironic sentence recognition model. Through neural network and deep learning, the model is successfully trained. Furthermore, the work focuses on examining how various combinations of hyperparameters affect performance. The output model is able to identify sarcastic sentences with success, and it will contribute to the growth of different NLP systems by serving as a useful recognition model.

Keywords: Irony Detection, Deep Learning, Neural Network.

1 Introduction

In order to enable computers to comprehend, interpret, and produce human language, the multidisciplinary field of natural language processing (NLP) integrates strategies and tactics from computer science, artificial intelligence, linguistics, and other fields [1]. Over time, NLP has evolved from simple rules and statistical methods to complex deep learning models that allow machines to achieve near or even surpass human levels of performance in language understanding and generation.

An ironic statement is a figure of speech that expresses the intention to ridicule or criticize by using words that are contrary to their actual meaning [2]. This expression is often used to expose a contradiction or irrationality in something, or to criticize or satirize a phenomenon, person, or behavior. These kinds of statements, which are very common in social media, can cause great harm to the target. In addition, to understand the ironic sentences, it is necessary to understand the meaning of the sentences, if there is no understanding, it will cause misunderstanding and communication barriers. Moreover, the literal meaning and the true meaning of sarcasm are often confused and difficult to recognize by existing NLP systems. Therefore, the judgment of ironic sentences is crucial to improve NLP ability. It is also important to ensure that the model does not generate ironic statements. So, this is an important question [3].

Over the past decade, the NLP field has undergone several important changes. Early on, models relied on a lot of manual feature engineering and shallow learning models. Subsequently, Convolutional neural networks (CNNs) and recurrent neural networks (RNNs) became popular in a variety of natural language processing (NLP) applications, including sentiment analysis, machine translation, and question-answering systems, with the advent of deep learning [4].

In recent times, the focus of NLP research has shifted to transformer architecture and pre-trained language models, such as Bidirectional Encoder Representations from Transformers (BERT), Generative Pre-training Transformer (GPT), and their variations [5,6]. These models, through unsupervised pre-training of large-scale corpora, are able to capture rich linguistic regularities and knowledge, which are then fine-tuned on specific downstream tasks. This approach has led to breakthroughs on a variety of language tasks such as sentiment analysis, named entity recognition, text classification, semantic similarity, and more and machine translation. In this article, the author will train a model for judging ironic statements using pytorch. The optimal optimizer selection is determined by adjusting the learning rate and weight attenuation, and the final output model is tested.

2 Related Work

There are several pre-trained language models developed in previous arts. BERT, Generative Pre-trained Transformer (GPT) and other models have greatly promoted the development of the NLP field, and these models have reached the best performance of their time on multiple NLP tasks [6]. For example, OpenAI's Contrastive Language-Image Pre-training (CLIP) models show how visual and linguistic processing can be combined to generate images or understand their content [7].

Moreover, small sample learning and meta-learning strategies are proposed previously. Small sample learning and meta-learning have emerged as key areas of research to address the data scarcity issue. These techniques aim to provide models with the ability to rapidly adjust to novel tasks using minimal amounts of labeled data.There is also a branch of work for cross-language NLP. With increasing globalization, the importance of cross-language NLP (enabling models to process and understand multiple languages) is increasing. For example, models such as Multilingual BERT (mBERT) can support multi-language understanding and generation, facilitating the acquisition and exchange of cross-language information [8].

3 Method

3.1 Dataset

The data set was taken from a hugging face paper. As there were datasets related to ironic statements, the author didn't find them on Hugging Face, so the author extracted the datasets from papers that did similar experiments.

There are a total of 39,780 data sets in the dataset, of which 21,301 are non-ironic statements with a label of 0. The remaining 18,479 are satirical sentences. Obviously it could be observed that the labels are not equal to each other. This may cause instability of model training in the process of model training. At the same time, there are not only emoji in the data set, but also the existence of "#", which will confuse the training of the model, because in the extraction of features, "#" will directly affect, and the addition of # will directly change the judgment of the model [9].

3.2 Data Preprocessing

Since there are more than 3000 more data sets with label 0 than with label 1 in the data set, this work first deletes the data with label 0 to the same number as the number with label 1. Later, since label and feature are combined in the data set, the author divides the data set into two txt files, one is label.txt and the other is feature.txt. Third, as mentioned above, since the feature contains emoji and "#", this work will delete these two symbols in the data reading.

A tokenization method called WordPiece is used in the BERT model. It splits the original text into smaller units. These units are called tokens. WordPiece begins with a vocabulary of full words and gradually learns to break up common word suffixes or uncommon words into smaller fragments [10]. The principle of this method includes the following steps.

First, text standardization, converting all text to lower case, and possibly some processing to remove or convert special characters.

Second, text segmentation, dividing text into tokens according to Spaces and punctuation marks. The purpose of this segmentation is to divide the text roughly along word boundaries.

Third, WordPiece segmentation, this algorithm will further divide these words into smaller units.

Fourth, converted to ID, after being divided into sub-words, each sub-word will be converted into a unique ID. Since BERT's rain training vocabulary includes tens of thousands to a hundred thousand different tokens. This allows the model to quickly find the corresponding word embed by these ids.

3.3 BERT

BERT model is used in this paper as the model. The BERT model's benefits are that it has high universality and applicability, and BERT is the mainstream model with a huge library. BERT is a model based on Transformer structure changes. The differences between BERT and conventional Transformer are as follows:

Firstly, BERT uses bi-directional Transformer encoder. This means that each token looks not only at the token before it, but also at the token after it. This fully bidirectional attention mechanism enables BERT to acquire the deep semantics of the entire input sequence.

Secondly, BERT model mainly makes use of the encoder part of Transformer, which is specialized to Transformer architecture. It is not designed to handle series-to-sequence tasks, but to generate a stable language representation.

is used in this paper as the model. The BERT model's benefits Thirdly, a substantial amount of unlabeled text is used to pre-train BERT models, which help them acquire a general representation of the language before honing it for a particular task. Therefore, simple fine-tuning can be adapted to text tasks, reducing the dependence on large amounts of annotated data. BERT's pre-training includes the learning of responsible language relationships, enabling it to outperform traditional models on specific tasks.

Fourthly, BERT has high flexibility and wide applicability, BERT architecture allows it to easily adapt to a variety of language processing tasks.

4 Result

In the experiments carried out on Pycharm software, python version 3.10.0, the model trained three epochals, the optimizer was AdamW, and the model used pre-trained BERT.

There are often many hyperparameters in the model, which is very difficult to adjust, so this study was done. In this experiment, there are 2 hyperparameters, including learning rate and weight decay. Their results are respectively demonstrated in Table 1 and Table 2. After testing the model, the Loss was low and the accuracy was high.

Table 1. Result comparison under various learning rates.

Lr	Weight	Average Loss	Average accuracy
5e-5	0.02	0.64%	99.86%
4e-5	0.02	0.43%	99.94%
3e-5	0.02	0.43%	99.94%
2e-5	0.02	0.25%	99.96%
1e-5	0.02	0.37%	99.94%
1e-4	0.02	1.42%	99.82%
1e-3	0.02	69.32%	50%
1e-2	0.02	69.33%	50%

Among them, the learning rate controls the step parameter when the optimizer updates the model weight, and determines the magnitude of weight adjustment in each iteration. An excessive amount of oscillation during the model training process could result from setting the learning rate too high. Even before the best results are obtained, a setting that is too low will cause the model training process to be too sluggish. Subsequently, it was discovered that while the results were generally good between 5e-5 and 1e-4, they were poor outside of that range. The table shows that the optimal learning rate is 2e-5, with the best average loss and accuracy.

Weight attenuation can help the model learn smoother and more generalizing features and prevent the model from overfitting. However, if the weight attenuation is too high, The data cannot be fully fitted, and the model's learning capacity will be constrained. As a result, the loss function decreases very slowly during the training of the model. If the setting is too low, the model will be too sensitive to overfit. Let the model perform well on training data, but poorly with limited generalization ability on fresh data that has never been seen before. It was determined from the experiment that there was little variation in the results within the 0.0002-0.2 range, indicating that this hyperparameter was not crucial to the model's training. The data cannot be fully fitted, and the model's learning capacity will be constrained.

Table 2. Result comparison under various weights.

Lr	Weight	Average Loss	Average accuracy
1e-5	0.02	0.37%	99.94%
1e-5	0.002	0.20%	99.94%
1e-5	0.0002	0.30%	99.95%
1e-5	0.2	0.32%	99.94%

5 Conclusion

This paper employs PyCharm as the integrated development environment for programming, leveraging its robust features to facilitate the coding process. For the implementation of ironic models, the PyTorch library is utilized. PyTorch's flexibility and efficiency make it an excellent choice for developing and training complex neural network models. During the model training phase, a meticulous approach to hyperparameter tuning has been adopted. The optimal parameters identified for the model include a 2e-5 learning rate and a 0.002 weight decay. The convergence and generalization abilities of the model depend heavily on these parameters. To sum up, the acknowledgment of ironic sentences is a complex but fascinating challenge for NLP. With the right combination of advanced models, hyperparameter tuning, and a focus on continuous improvement, there is great potential for future models to significantly enhance our ability to process and understand the rich and varied tapestry of human language. It is expected that the models developed in this research will contribute to the advancement of NLP capabilities. As the understanding of irony is a complex aspect of human language, improving the recognition of ironic sentences can lead to more nuanced and human-like interactions between humans and artificial intelligence systems.

References

1. Khurana, D., Koli, A., Khatter, K., & Singh, S.: Natural language processing: State of the art, current trends and challenges. Multimedia tools and applications, **82**(3), 3713-3744 (2023).
2. Potamias, R. A., Siolas, G., & Stafylopatis, A. G.: A transformer-based approach to irony and sarcasm detection. Neural Computing and Applications, **32**(23), 17309-17320 (2020).
3. Zhang, S., Zhang, X., Chan, J., & Rosso, P.: Irony detection via sentiment-based transfer learning. Information Processing & Management, **56**(5), 1633-1644 (2019).
4. Shrestha, A., & Mahmood, A.: Review of deep learning algorithms and architectures. IEEE access, **7**, 53040-53065 (2019).
5. Devlin, J., Chang, M. W., Lee, K., & Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018).
6. Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. Improving language understanding by generative pre-training, 1-12 (2018).
7. Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., ... & Sutskever, I.: Learning transferable visual models from natural language supervision. In International conference on machine learning. 8748-8763 (2021).
8. Pires, T., Schlinger, E., & Garrette, D.: How multilingual is multilingual BERT?. arXiv preprint arXiv:1906.01502 (2019).
9. Zhang, M., Zhang, Y., & Fu, G.: Tweet sarcasm detection using deep neural network. In Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: technical papers. 2449-2460 (2016).
10. Song, X., Salcianu, A., Song, Y., Dopson, D., & Zhou, D.: Fast wordpiece tokenization. arXiv preprint arXiv:2012.15524 (2020).