BERT in Stance Detection

Gu chun, Li youquan, Lv xinkai

Abstract

Fake News Challenge is a competition held on $fakenewschallenge.org$, and the competition aims to find the best way for stance detection. We reproduce the top three models in Fake news challenge using Pytorch and Python3, while the original code is written by using other packages. Further more, we propose our new BERT model for stance detection, which achieves a significant 4.43% improvement over the winner of Fake News Challenge in test set. Our code is available at https://github.com/SuLvXiangXin/nlp_final

1 Introduction

Recently, as the novel coronavirus break out, more and more rumors and fake news appear continuously, which brings out a lot of confusion and so there is an impediment for people to have a sense on what's going on.

Therefore, it is of great importance for us to find an accurate and efficient way to eal with those fake news, and it's a natural thought that we could use artificial intelligence, particularly machine learning and natural language processing. By doing these, hopefully, we can help people better determine whether a story is a fact or a hoax.

2 Related work

Stance detection Stance detection is a well-known task in natural language processing, which aims to detect an attitude towards a certain topic or claim. Initially, it focused on parliamentary debates (Thomas et al., 2006) and debating portals (Somasundaran and Wiebe, 2009). Nowadays, latest works have shifted to the domain of social media. With this shift in domains, the definition of input expands from parliamentary debates and debating portals to tweets (Gorrell et al., 2018) and news articles (Pomerleau and Rao, 2017). In past years, stance detection has became a groundwork of many other tasks such as fake news detection (Pomerleau and Rao, 2017). Over the years, many approaches were proposed to solve this task. A common approach is to treat this task as a sentence-level classification task, which is similar to sentiment analysis (Socher et al., 2013). A typical approach is to decompose the task into feature representation and classification.

Feature representation In the early years, we usually extracted features manually. The baseline (Galbraith et al.) of the fake news challenge uses overlap words, refuting words, polarity and co-occurence to design features. (Riedel et al., 2017) uses term frequency(TF) and term frequency-inverse document frequency(TF-IDF) to represent the headline and body pairs of the provided data. Nowadays, we can also use word2vec to automatically extract features to avoid complex and tedious feature design.

Classification method On the one hand, we can use classical machine learning models such as decision trees (Pan et al.) for classification. On the other hand, with the development of deep learning, models such as MLP (Riedel et al., 2017), CNN, RNN and Transformer are gradually applied to classification problems.

3 Method

3.1 SOLAT in the SWEN

The stance detection system of SOLAT in the SWEN is an ensemble model based on an 50/50 weighted average between gradient-boosted decision trees and a deep convolutional neural network. The schematic diagram of it is shown in Figure 1.

3.1.1 Deep Learning Model

The first model used by the team applies several different neural networks used in deep learning.

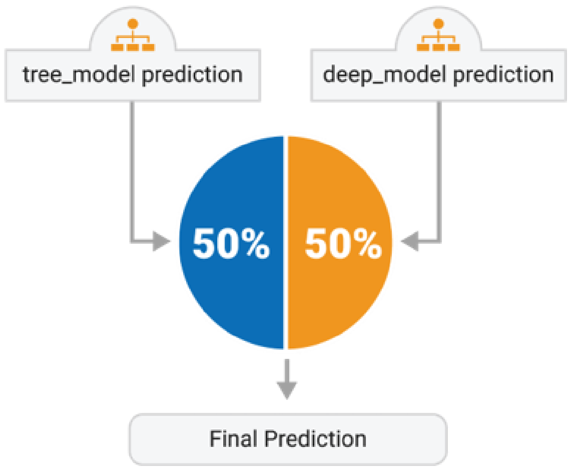

Figure 1: Schematic diagram of SOLAT in the SWEN's system

This model applies a one-dimensional convolutional neural network (CNN) on the headline and body text, represented at the word level using the Google News pretrained vectors. The output of this CNN is then sent to a multi-layer perceptron (MLP) which gives us a 4-class output, including "agree," "disagree," "discuss," and "unrelated". The model is regularized using dropout (p=0.5) in all convolutional layers and all hyperparameters of this model were set to sensible defaults. The architecture of this model is shown in Figure 2.

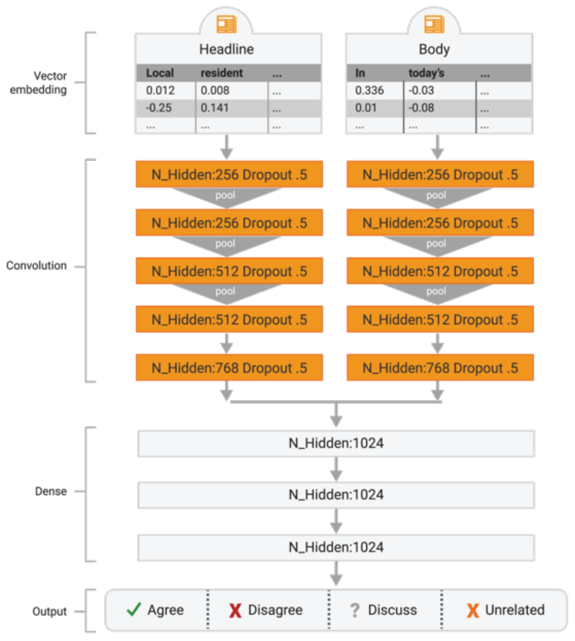

Figure 2: Diagram of deep learning model

3.1.2 Gradient-Boosted Decision Trees

Another model employed in the ensemble is a Gradient-Boosting Decision Trees (GBDT) model. The input of the model is text-based feature derived from the headline and body of an article, which are then fed into Gradient Boosted Trees to predict the relation between the headline and the body. Furthermore, XGBoost is chosen to be the implementation of this model. The diagram of the decision tree model is shown in Figure 3. Features used in the model include the following contexts:

- The number of overlapping words between the headline and body text.
- The similarities measured by the word count, 2-grams and 3-grams.
- The similarities measured after transforming these counts with term frequency-inverse document frequency (TF-IDF) weighting and Singular Value Decomposition (SVD).
- Word2Vec implemented by GoogleVec pretrained.
- Polarity implemented by SentimentIntensityAnalyzer pretrained.
- Cos-similarity between each head and body

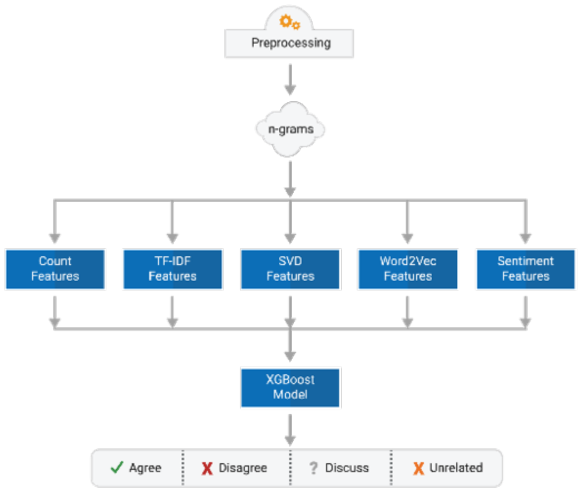

Figure 3: Diagram of decision tree model

3.2 Athene (UKP Lab)

The stance detection of Athene consists of a series of feature extraction and they use 5 MLPs to build ensemble model as their classifier. The schematic diagram of Athene's system is shown in Figure 4.

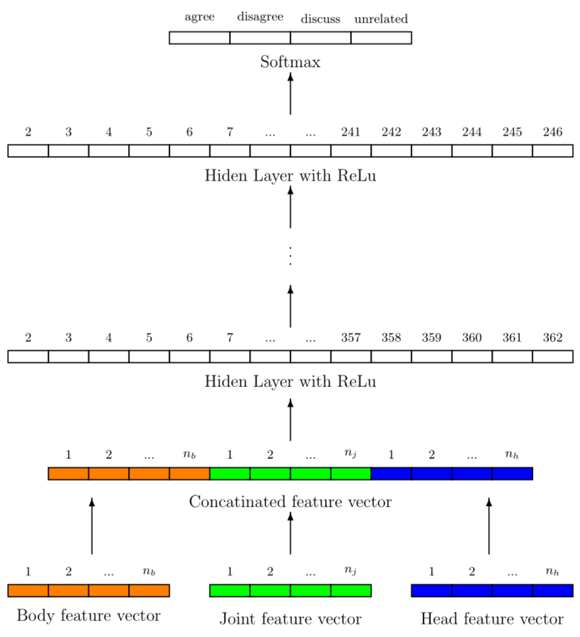

Figure 4: Schematic diagram of Athene's system

3.2.1 Representations and features

The feature engineering consists of the following content.

- Use original features from the fnc-1 baseline, which contains overlap, refuting, polarity and hand.
- Use frequency-inverse document frequency (TF-IDF) to represent each headline and body.
- Implement non-negative matrix factorization and calculate the cosine distance between the headline and body vectors.
- Implement Latent Dirichlet Allocation(LDA) based on the 5000 most important words in dataset and returns feature vector of cosine distance between headline and body.
- Use all data to build Tfidf-Matrix and create LSI(Latent Semantic Indexing)-Model twice, and take the probabilities as the feature vectors.
- Calculate word similarity by comparing the embeddings of the nouns and verbs of the headline and body.

Finally, concatenate the above feature vectors to obtain the final feature vector.

3.2.2 Classifier

The classifier is an ensemble model which consists of 5 MLPs and the prediction is generated by hard voting.

3.3 UCL Machine Reading

The stance detection of (Riedel et al., 2017) consists of lexical and similarity features passed through a multi-layer perceptron (MLP) with one hidden layer. The schematic diagram of UCLMR's system is shown in Figure 5.

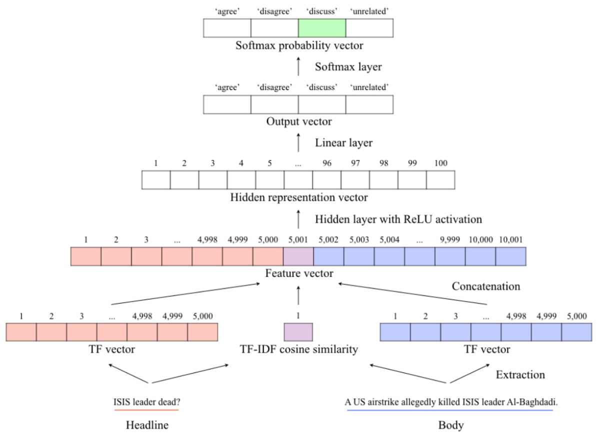

Figure 5: Schematic diagram of UCLMR's system

3.3.1 Representations and features

Firstly, they use term frequency (TF) to represent headline and body, and then concatenate these two vectors with the cosine similarity between l_2-normalized frequency-inverse document frequency (TF-IDF) vectors of the headline and body. They use different vocabularies to construct above vectors. For the TF vectors, they extract the 5000 most frequent words from the training set and exclude the stop words. For the TF-IDF vectors, they extract the 5000 most frequent words from both training set and testing set and also exclude stop words. After concatenating them into a larger vector, they send it to the classifier.

3.3.2 Classifier

The classifier is a MLP with one hidden layer of 100 units and a softmax on the output of the final linear layer. They use ReLU as activation function of the hidden layer and dropout for both hidden layer and the last linear layer.

3.4 Fine-tune BERT

The previous three methods make a lot of effort on feature engineering,which contains plenty of artifacts, and that seems not to be a good way as the artificial feature engineering may ignore the important information. As a result we decide to explore a new way to represent the word/document vector, and the most popular BERT(Bidirectional Encoder Representation from Transformers) which is based on self-attention becomes our first choice.

3.4.1 Representations and features

We choose BERT(Devlin et al., 2018a) as our encoder to acquire the word/document representation. The transformer(Vaswani et al., 2017) block in our network structure is shown in Figure 6. Our BERT uses 12 layers of transformer block, which contains a muti-head self-attention, a residual layer and a MLP. We choose to stack 12 layers of transformer because the pretrained model provided by (Devlin et al., 2018b) has the same structure.For any input document, the output of our encoder is 768 dimension.

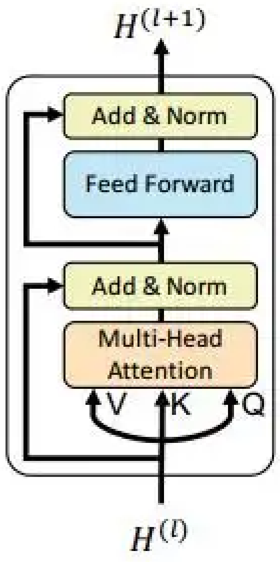

Figure 6: Transformer block(figure comes from network)

3.4.2 Classifer

After we get the document representation, we feed the vector to a simple linear layer which is used to classify the vector into 4 categories (unrelated,agree,disagree,discuss).Because of the imbalanced dataset, we consider giving diverse weight to categories. To be specific, as the number of "related" is far more than other categories, we penalize it by multiply w. To better fit the learning rate, we normalize the given weight to four categories as in eq 2,where gt represents ground truth.

$$L = \sum_{i}^{N} w_{c^{gt}} L(pred, gt) \tag{1}$$

$$\sum_{c \in C} w_c = 1 \tag{2}$$

4 Experiment

4.1 Dataset

In this task, we used the FNC-1 Dataset, which contains labeled articles, each article is made up of a headline and a body text, and a label indicates the stance between headline and body. The stance can be 'agree', 'disagree', 'discuss', and 'unrelated'.

The dataset is derived from the Emergent Dataset created by Craig Silverman. It contains 49972 pieces of the article. As for the stance, 73.1% of them are 'unrelated', 17.8% are 'discuss', 7.4% are 'agree', and 1.7% are 'disagree'.

4.2 Top three team

We complete experiments on FNC-1 dataset. For team "SOLAT in the SWEN", we use a hidden layer size of 256, and Adam as optimizer, 2e-4 as initial learning rate, 1e-4 as weight decay, 128 as batch size. For team "Athene (UKP Lab)", we use Adam as optimizer, 1e-3 as learning rate, 200 as batch size. For team "UCL Machine Reading", we use Adam as optimizer, 1e-2 as learning rate, 500 as batch size, 0.4 as dropout rate. The result is shown in table2

4.3 Our model

Because the pretrained BERT has a max limitation of input sequence length, we randomly select part of each body content and concat with the headline to fit the 512 sequence length. This can be seen as a kind of data augmentation in which we add randomness to our input.In order to avoid damaging the original structure of the pretrained weight, we set the learning rate of our BERT encoder to 1e-5, and we set the learning rate of our classifier to 1e-3 to accelerate the convergence.

We experiment several couple of w (which is designed for the imbalanced dataset) on validation set. We set w of "related"(include agree,disagree,discuss) to be n times of w of "unrelated", and we test n in 1,2,4. The result is shown in table1.

We can see that n=4 is the best choice. So we choose n=2 as our final result, the training procedure is shown in figure7

n	Acc(%)	Score(%)
1	98.88	98.39
2	98.94	98.10
4	98.84	98.66

Table 1: Comparison of different w on validation set

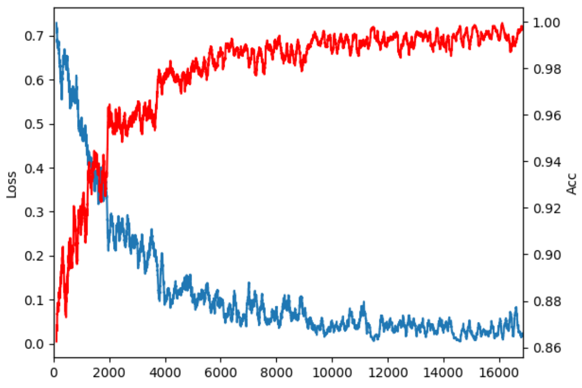

Figure 7: Training process of our model

Finally, we compare our result with the top three team in table2. Obviously,our method has a significant improvement over the top three team.

	Acc(%)	Score(%)
SOLAT in the SWEN	89.2	82.02
Athene (UKP Lab)	89.0	81.73
UCL Machine Reading	88.7	81.74
Ours	**90.9**	**86.45**

Table 2: Comparison of all methods

5 Conclusion

We illustrate using pretrained BERT can get better word representation, which avoid many artifacts in feature engineering.We explore giving diverse weights to diverse categories because of the imbalanced dataset. Although we achieve a significant improvement in testset, our model achieve a high score on validation set,but a relatively low score on testset which might be caused by the different distribution between validation set and test set, and there seems to be a lot of room for improvement in our model's robustness.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018a. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018b. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Byron Galbraith, Humza Iqbal, HJ van Veen, Delip Rao, James Thorne, and Yuxi Pan. https://github.com/FakeNewsChallenge/fnc-1-baseline.

Genevieve Gorrell, Kalina Bontcheva, Leon Derczynski, Elena Kochkina, Maria Liakata, and Arkaitz Zubiaga. 2018. Rumoureval 2019: Determining rumour veracity and support for rumours. *CoRR*, abs/1809.06683.

Yuxi Pan, Doug Sibley, and Sean Baird. https://github.com/Cisco-Talos/fnc-1.

Dean Pomerleau and Delip Rao. 2017. The fake news challenge:exploring how artificial intelligence technologies could be leveraged to combat fake news.

Benjamin Riedel, Isabelle Augenstein, Georgios P. Spithourakis, and Sebastian Riedel. 2017. A simple but tough-to-beat baseline for the fake news challenge stance detection task. *CoRR*, abs/1707.03264.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*, pages 1631–1642.

Swapna Somasundaran and Janyce Wiebe. 2009. Recognizing stances in online debates. In *ACL/IJCNLP*, pages 226–234.

Matt Thomas, Bo Pang, and Lillian Lee. 2006. Get out the vote: Determining support or opposition from congressional floor-debate transcripts. In *EMNLP*, pages 327–335.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.