The 33rd Conference on Computational Linguistics and Speech Processing (ROCLING 2021)
Taoyuan, Taiwan, October 15-16, 2021. The Association for Computational Linguistics and Chinese Language Processing

CYUT at ROCLING-2021 Shared Task: Based on BERT and MacBERT

Xie-Sheng Hong
Department of CSIE
Chaoyang University of Technology
Taichung, Taiwan
kk6251433@gmail.com

Shih-Hung Wu
Department of CSIE
Chaoyang University of Technology
Taichung, Taiwan
shwu@cyut.edu.tw

Abstract

This paper present a description for the ROCLING 2021 shared task in dimensional sentiment analysis for educational texts. We submitted two runs in the final test. Both runs use the standard regression model. The Run1 uses Chinese version of BERT as the base, and in Run2 we use the early version of MacBERT that Chinese version of RoBERTa-like BERT model, RoBERTa-wwm-ext. Using powerful pre-training model of BERT for text embedding to help train the model.

Keywords:
Natural Language Processing, Dimensional Sentiment Analysis, Deep learning

1 Introduction

The ROCLING 2021 Shared Task was inherited from a task about dimensional sentiment analysis task for Chinese words at IALP2016. The task is extended to include both word- and phrase-level dimensional sentiment analysis. It explores the sentence-level dimensional sentiment analysis task on educational texts.

In view of structured information such as attendance, in-class participation have been extensively studied to predict students' learning performance. For this reason, the organizers of task wanted participants to use unstructured information such as self- evaluation comments written by students. Using dimensional sentiment analysis to identify valence-arousal ratings from texts. To analyze the affective states contained in them to help illuminate students' affective states.

In the three training sets provided, there are several sentences, phrases, or words with their corresponding real-valued or average scores for both valence and arousal dimensions. The two dimensions range from 1 (highly negative or calm) to 9 (highly positive or excited) where valence represents the degree of positive and negative sentiment, and arousal represents the degree of calm and excitement(Yu et al., 2016). For example, the questions and answers used in this shared task are shown below:

Input:

今天教了許多以前沒有學過的東西，所以上起課來很新鮮

Output:

Valence: 6.8

Arousal: 5.2

In short, the specific goal of this shared task is to input a sentence and let our proposed system predict the score on two indexes, Valence and Arousal.

Therefore, this task can be defined as a two-objective regression task. We used the Chinese versions of the BERT and RoBERTa pre-training models to construct the regression models. In the experiment, we chose different motivation functions in the two tests. We also adjusted some of the parameters in order to find a more suitable method. The rest of the paper will give the details of our method, show the experiment setting and results, also discussions on the results, In the final section, we will give conclusion and future works.

2 Method

For comparison, we fine-tune two Transformer Models, bert-base-chinese and RoBERTa-wwm-ext. The former uses the official Chinese version of BERT provided by Google(Devlin et al., 2019), while the latter uses the Chinese version of the RoBERTa-like model proposed by State Key Laboratory of Cognitive Intelligence, iFLYTEK Research (HFL)(Cui et al., 2019, 2020). We fine-tun the two models, and combine two different activation functions, ReLU and LeakyReLU, to build the regression model. In this section, we present our ideas and attempts on the models, as well as the methods and procedures used to build them.

2.1 Model-1: BERT

In Run1, our system is based on the standard Chinese version of the BERT pre-training model proposed by Google. BERT is a deep, two-way unsupervised language representation trained using only plain text corpus. Unlike word2vector(Mikolov et al.) and GloVe(Pennington et al., 2014), which do not use context.Transformers is a new simple network structure proposed by Google, which is based only on attention mechanisms and does not require recursion and convolution at all. The results of the two translator tasks presented in their study show that the model can improve considerably with this network.Also, it is easier to perform parallelization, and the training time required for the model is significantly reduced(Vaswani et al., 2017).BERT takes into account the context in which a particular word appears each time it is used in an article. Therefore, even if the same word occurs repeatedly, BERT can generate different word vectors according to different contexts(Devlin et al., 2019).

We consider that the text data used for training the model is clean and does not contain tags such as <br>, which are not useful for model training. So, we just organize the training data and convert it into a data form that can be read by the BERT model. In this part, we use Pytorch(Paszke et al., 2019) and call HuggingFace(Wolf et al., 2020) to fine-tune and build the whole model.

2.2 Model-2: Chinese-RoBERTa-wwm

As a comparison with Run1, our system is built using a RoBERT-like model called RoBERTa-wwm-ext in Run2. First, the original version of RoBERTa (Robustly optimized BERT approach)(Liu et al., 2019), which can be simply understood as an enhanced optimization of the original BERT model. It was jointly proposed by Facebook and the University of Washington. RoBERTa has the following main improvements over the original BERT. It uses a larger number of model parameters, a larger batch size, and increases the training data. In the model training process, RoBERTa adopts a dynamic mask, so that the model generates a new mask pattern for each input sequence. In this way, the model can gradually adapt to different mask patterns as the data is input. Then, considering the controversy over the validity of the NSP task used on BERT(Devlin et al., 2019; Lample and Conneau, 2019; Joshi et al., 2020), RoBERTa has also adapted the NSP task.

MacBERT is a new pre-training model proposed by the HFL after improving the models same proposed by them, such as Chinese-BERT-wwm and Chinese-RoBERTa-wwm(Cui et al., 2020, 2019). Therefore, in this paper we call the RoBERTa-wwm-ext model as the early version MacBERT. The model's full name is RoBERTa Whole Word Masking Extended data. The "wwm" here refers to the updated version released by the author of the original BERT in 2019, named Whole Word Masking. It mainly mitigates the drawbacks in original BERT's Wordpice. If the masked WordPiece token belongs to a whole word, then all the WordPiece tokens will be masked, so that it forms a complete word, not just WordPices in the training task. It is beneficial to design more powerful models(Wu et al., 2016; Cui et al., 2019).

The model we use is called RoBERTa-like because this pre-trained model is made by the original author by integrating the advantages of RoBERTa and BERT-wwm. In essence, it is not RoBERTa, but a BERT model trained according to the training method similar to RoBERTa. They used the wwm strategy for mask instead of Dynamic masking in the pre-training phase, eliminated the NSP loss, and adjusted the length of MAX Len and the number of training steps(Cui et al., 2019, 2020).

As Model-1, we also used Pytorch and called HuggingFace to fine-tune and build the whole model.

2.3 ReLU and LeakyReLU

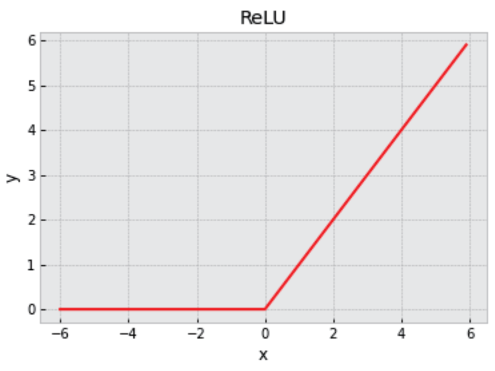
In the current study, we used two activation functions, ReLU and LeakyReLU, to construct the regression models for our two systems. In order to make a larger difference between the two tests, we used LeakyReLU in the BERT-based model and used common ReLU in RoBERTa-like model.

ReLU (rectified linear activation function) is a widely used activation function in deep neural networks(Ramachandran et al., 2017). It can be as a piecewise linear function that modifies the negative part to zero and keeps the positive part. In other words, the value of ReLU is zero when it is smaller than zero and remains the same when it is larger(Nair and Hinton, 2010; Sun et al., 2014). It is a non-saturated excitation function, which has many advantages over saturated functions such as sigmoid and tanh.

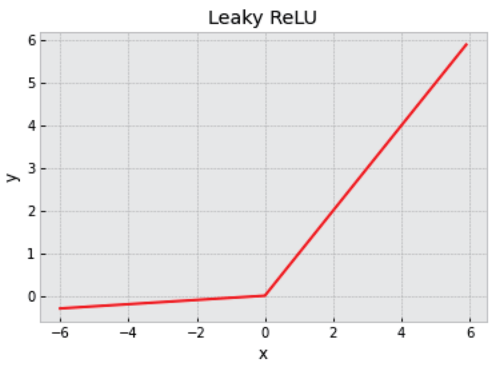
Since ReLU can maintain the original state when the output is above zero, this property can keep the gradient invariant and can effectively mitigate the vanishing gradient and exploding gradient problems(Clevert et al., 2016; Xu et al., 2015)that easily occur in the sigmoid and tanh functions. In addition, ReLU has an important property that the result of activation by it is sparse. Although some scholars have question(Xu et al., 2015), it is generally accepted that the sparse property of ReLU can lead to excellent performance(Glorot et al., 2011; He et al., 2014). The reason is that the sparsity of ReLU can separate the features in the data, make the dense features sparse, and make the features linearly separable(Glorot et al., 2011). Therefore, ReLU can learn the features of data more flexibly and effectively.

LeakyReLU is a variant of ReLU. The biggest difference with ReLU is that LeakyReLU is given a non-zero slope in the negative part, the negative part is no longer set to zero at all times. This change solves the dying ReLU problem. It refers to the fact that when the activation value of ReLU is always negative, the gradient obtained is also always zero. As a result, the neuron can no longer learn, like it is "dead" . The adjustment solves this problem, so that it has the advantage of ReLU, but also solves some of the original shortcomings of ReLU(Xu et al., 2015; Maas, 2013).

The detailed shapes of the two models can be seen in Figure 1 (a) and Figure 1 (b).



(a) ReLU Function



(b) LeakyReLU Function

Figure 1: ReLU and LeakyReLU Function

3 Experiments

Figure 2 shows the overall flow of our study, which can be roughly divided into the following processes. We first do a simple preprocessing of the raw data to be used for training the model, and build a dataloader to facilitate the training.Then we construct a BERT/RoBERTa-like neural network model and train it.After the training, the test data are also organized into a dataloader and given to the model for prediction.Finally, the prediction results of the model are organized into a prescribed format.

In this section, we describe in detail the various settings of our system and analyze the results and errors.

3.1 Parameters and Setting

For Run1, we use a batch size of 16 to run the training. We use AdamW(Loshchilov and

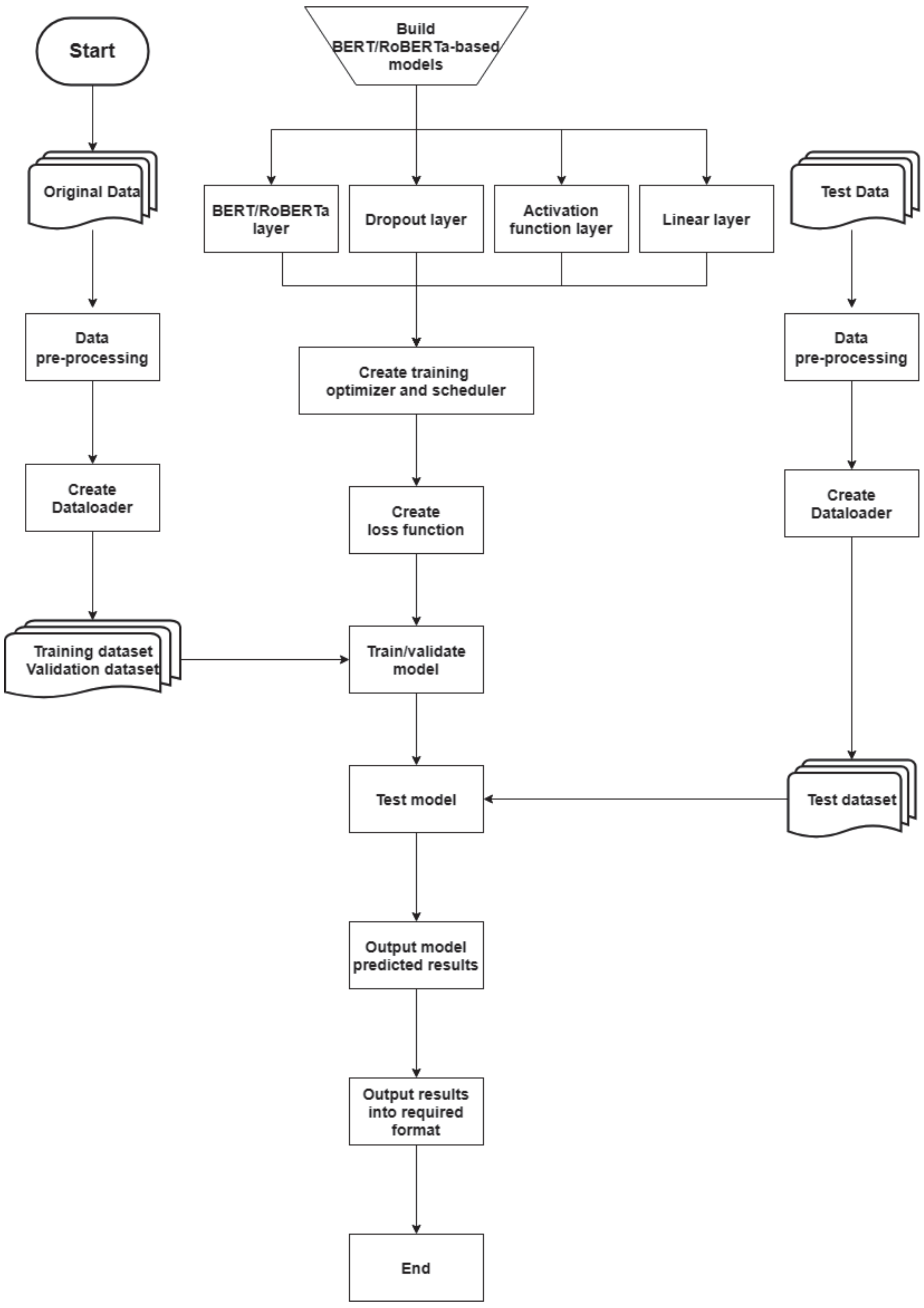


Figure 2: The flow chart of our study

Hutter, 2019) as the optimizer of the model, which is a variant of another classical optimizer, Adam(Kingma and Ba, 2017). The authors of AdamW propose simple modifications to improve the weight decay and improve its generalization performance. We set the learning rate of the optimizer to 2e-5, correct_bias to False, and keep the rest of the relevant parameters as default. Run2 is the same as Run1 except that the batch size is changed to 32. Finally, the maximum length of the training data is set to 200 for both.

For the model architecture, Run1 and Run2 are basically the same. Both of them use two linear layers, two activation layers and two dropout layers with p=0.2 for Run1, with 0.1 for Run2, to avoid overfitting.

During the model training, Run1 and Run2 are trained for 100 epochs by default. A parameter called ”patience” is set as the threshold. The training is terminated early if the model does not improve its loss for three consecutive epochs. Run1 is automatically terminated at the 60th epoch, and Run2 is manually terminated at 26 epochs to avoid overfitting.

3.2 Official Run and Fixed Run Result

Table 1 show the prediction results of our system for the two targets in two Runs. Since we made a mistake when submitting the final test, the predicted answers were misaligned with the questions in our submitted results. As a result, we got unexpected predicted scores from the organizer’s validation. The scores is named ”Official” in tables. After fixing the program, we verified the model prediction with the same verification method as the organizer, mean absolute error and pearson correlation coefficient, and got a different result. The new results are shown in Table 1, too.

3.3 Result and Error Analysis

From Section 3.3, our system returned to a more normal value after correcting the error. Thanks to the power of BERT, our system is able to perform to a certain extent even if we do not do too much complex processing of training data or neural network models.

Briefly summarizing Tables 1, Run2 (RoBERTa-wwm-ext + ReLU) is better than Run1 (BERT + LeakyReLU) in both objectives. It seems to show that the RoBERTa-wwm-ext trained with more pretraining data is still better than the original BERT, even with the LeakyReLU assist, when the other settings are similar.

Table 3 shows the performance of our system in the two Runs after the correction in comparison with the other groups’ systems. It is found that our system is able to have a similar degree of correctness as the other groups after the correction. After looking at the results for each group and ours, we were surprised to find that the systems in each group performed slightly worse on the ”Arousal” index than on the ”Valence” index.

Therefore, we list the prediction results and questions of ”Arousal” that are partially better (loss<=0.001) and worse (loss>=3) on our Run2-system to facilitate our comparison:

Better cases:

我覺得老師講的有一點快，需要時間消化
Error = 0.00070

我認爲可以稍微補充以及多舉些例子，好讓我們比較容易理解
Error = -0.00078

能把今天的課程學好，未來應該很受用
Error = 0.00094

Worse cases:

今天收穫很多
Error = 3.197

受益良多，頗有趣
Error = 3.054

內容生動有趣
Error = 3.158

上課時老師善用舉例，讓我有更具體的思考邏輯，對於整個架構也比較了解
Error = -3.317

Looking at our prediction results with Table 1 and Table 3, we found that our system seemed to have a higher recognition rate for the more complete sentences.As you can see from the example we gave, if the student’s description of the emotional state is more detailed, the more accurate our system is in predicting the answer. When the student describes the situation in a more concise manner, our system is most likely unable to make

Run	Valence MAE	Valence r	Arousal MAE	Arousal r
Run1-Official	1.695	-0.017	1.177	0.040
Run2-Official	1.685	0.007	1.252	-0.021
Run1-Fixed	0.674	0.870	0.901	0.531
Run2-Fixed	0.600	0.900	0.877	0.565

Table 1: Official and Fixed Run Result

Run	Valence MAE	Valence r	Arousal MAE	Arousal r
Run1	0.512	0.887	0.666	0.753
Run2	0.462	0.911	0.652	0.774

Table 2: Development Result

Run	Valence MAE	Valence r	Arousal MAE	Arousal r
CYUT-Run1-Fixed	0.674	0.870	0.901	0.531
CYUT-Run2-Fixed	0.600	0.900	0.877	0.565
NCU-NLP-Run1	0.625	0.900	0.938	0.549
NCU-NLP-Run2	0.611	0.904	0.989	0.582
ntust-nlp-1-Run1	0.684	0.912	0.906	0.607
ntust-nlp-1-Run2	0.586	0.901	0.885	0.585
ntust-nlp-2-Run1	0.654	0.905	0.880	0.581
ntust-nlp-2-Run2	0.667	0.913	0.866	0.616
SCUDS-Run1	0.953	0.694	1.054	0.375
SCUDS-Run2	0.975	0.667	1.039	0.354
SoochowDS-Run1	2.421	0.073	1.327	0.051
SoochowDS-Run2	1.073	0.584	1.125	0.228

Table 3: Comparison of results with other groups

a correct prediction. It is worth noting that, as we have shown, there are also some sentences where our system is unable to make a correct prediction although there is a more detailed description. For this, we have the following inference.

Table 2 shows the results of the validation of our model on the development set after the training. However, it is important to note that even in the development set with standard answers, ”Arousal” still only has a Pearson correlation coefficient of about 0.7. We believe that this may be the reason for the poor performance of the predictions of ”Arousal”. Perhaps the information we provide to the model for learning may not be relevant enough to the answer we are trying to predict. Considering this study, we only did simple pre-processing and Tokenizer on the text data used to train the model. We did not filter the key features in the text. Hence, our model may need more narratives to predict the answer and cannot make judgments based on key features alone. This makes it difficult for our model to predict short sentences. We think this is one of the parts of our system that needs to be improved in the future, filtering out the key features beforehand to improve the feature strength. Enhance the relevance of the model in training and prediction.

4 Conclusion and Future Works

In this paper, we describe our proposed approach on ROCLING 2021 shared task in dimensional sentiment analysis for educational texts. Our system is based on the Chinese version of the BERT and RoBERTa-wwm-ext models. Although the results were not good in the official run because of a program error, after fixing the error, our system has a standard result. It is worth noting that we only used some standard methods and parameters and do not use any complex methods. But because of this, our system’s shortcomings are also obvious. As ”Arousal” problem this time, the direct use of training data may not be sufficient to train the model completely. In the future, we can adjust the pre-processing part of the data, such as finding the key features in the sentences in advance, or increasing the training data. For the neural network model, we can try to add classical neural networks such as LSTM(Hochreiter and Schmidhuber, 1997) or GRU(Chung et al., 2014) for training, and improve the depth of the model in the future.

References

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling.

Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. 2016. Fast and accurate deep network learning by exponential linear units (elus).

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Shijin Wang, and Guoping Hu. 2020. Revisiting pre-trained models for Chinese natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 657–668, Online. Association for Computational Linguistics.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Ziqing Yang, Shijin Wang, and Guoping Hu. 2019. Pre-training with whole word masking for chinese bert. *arXiv preprint arXiv:1906.08101*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Deep sparse rectifier neural networks. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 315–323, Fort Lauderdale, FL, USA. PMLR.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2014. Spatial pyramid pooling in deep convolutional networks for visual recognition. *Lecture Notes in Computer Science*, page 346—361.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A method for stochastic optimization.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization.

Andrew L. Maas. 2013. Rectifier nonlinearities improve neural network acoustic models.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space.

Vinod Nair and Geoffrey E. Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, page 807—814, Madison, WI, USA. Omnipress.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Prajit Ramachandran, Barret Zoph, and Quoc V. Le. 2017. Searching for activation functions.

Yi Sun, Xiaogang Wang, and Xiaoou Tang. 2014. Deeply learned face representations are sparse, selective, and robust.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Huggingface's transformers: State-of-the-art natural language processing.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation.

Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li. 2015. Empirical evaluation of rectified activations in convolutional network.

Liang-Chih Yu, Lung-Hao Lee, Shuai Hao, Jin Wang, Yunchao He, Jun Hu, K. Robert Lai, and Xuejie Zhang. 2016. Building Chinese affective resources in valence-arousal dimensions. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 540–545, San Diego, California. Association for Computational Linguistics.