YNU-oxz at SemEval-2020 Task 5: Detecting counterfactuals based on Ordered Neurons LSTM and Hierarchical Attention Network

Xiaozhi Ou, Shengyan Liu, Hongling Li*
School of Information Science and Engineering
Yunnan University, Yunnan, P.R. China

Abstract

This paper describes the system and results of our team's participation in SemEval-2020 Task5: Modelling Causal Reasoning in Language: Detecting Counterfactuals, which aims to simulate counterfactual semantics and reasoning in natural language. This task contains two subtasks: Subtask1–Detecting counterfactual statements and Subtask2–Detecting antecedent and consequence. We only participated in Subtask1, aiming to determine whether a given sentence is counterfactual. In order to solve this task, we proposed a system based on Ordered Neurons LSTM (ON-LSTM) with Hierarchical Attention Network (HAN) and used Pooling operation for dimensionality reduction. Finally, we used the K-fold approach as the ensemble method. Our model achieved an F1 score of 0.7040 in Subtask1 (Ranked 16/27).

1 Introduction

In recent years, in order to simulate counterfactual semantics and reasoning in natural languages, the introduction of counterfactuals in natural language understanding systems has received more and more research attention. According to the problem of counterfactual conditionals (Goodman, 1947), counterfactual statements can be transformed into opposing statements with true premises and results. Counterfactual statements describe events that did not occur or cannot occur, and the consequences that may occur after the event. More specifically, counterfactuals describe events that are contrary to facts and involve common sense, understanding, and reasoning. Therefore, whether the system can correctly identify counterfactual natural language statements is very important. In SemEval-2020 Task 5: Modelling Causal Reasoning in Language–Detecting Counterfactuals (Yang et al., 2020). The organizer has designed two subtasks. Subtask1 requires determining whether a given sentence is counterfactual. Solving this problem is the basis for all downstream counterfactual related causal reasoning analysis using natural language. Subtask2 aims to locate the premises and results in of counterfactuals, and to point out that causal insights are inherent characteristics of counterfactuals. This is to further test the causal knowledge conveyed in counterfactual statements.

In this competition, we only participated in Subtask1. We used a deep learning method to build a neural network model based on Ordered Neuron LSTM (ON-LSTM) (Shen et al., 2019) and Hierarchical Attention Network (HAN) (Wang et al., 2019). The difference between Ordered Neurons LSTM (ON-LSTM) and original LSTM (Hochreiter and Schmidhuber, 1997) is that encodes the hierarchical structure of sentences into features to enhance the expression ability of LSTM (Wang et al., 2018). The Hierarchical Attention Network (HAN) model has two important characteristics. The first is layering, the word-level layer and the sentence-level layer, which conforms to the document structure; the second is the attention mechanism, which can give dynamic weight according to the content when weighting. In addition, we have added the Pooling operation, and in the Pooling layer, we used Average Pooling and

*Corresponding author: `honglingli66@126.com`

Max Pooling operations to compress the amount of parameters and reduce overfitting. Finally, considering the reason for the small dataset and data imbalance, we used the K-fold approach as the ensemble method.

The rest of the paper is structured as follows. Section 2 introduces related work to detect counterfactual statements. Section 3 describes the dataset and our model. In Section 4, we analyze the experimental results. We summarize and looked forward to future work in Section 5.

2 Related Work

Counterfactual statements that describe events that did not occur and their consequences. Counterfactuals have been studied in many different fields. Logicians and philosophers are concerned with the literal logical relationship between the cause and effect of counterfactual form and the result (Goodman, 1947). In contrast, political scientists often conduct counterfactual thought experiments on policies, or other aspects of society and evaluate them (Tetlock and Belkin, 1996). Despite extensive research in the field of counterfactual thinking, counterfactuals have not been extensively studied in computational linguistics. First, the base rate of counterfactual statements is lower; we found that only 2% of status updates and 1% of tweets on Facebook contain counterfactual statements. Second, counterfactual statements can take many forms of natural language [1]. They can use or not use explicit if- or then- clauses (for example, consider "if I haven't met him, then I'll be better" rather than "I hope I haven't met him").

In this section, we review some studies and briefly discuss their findings. Son et al. (2017) created an anti fact tweet dataset and explored the method of detecting anti fact using rule-based and supervised statistical methods. Counterfactual Examples for Bias (CEB) is designed for non-experts to discover potential biases. Using counterfactual examples is a technique for developing a fairer model (Kusner et al., 2017; Sokol and Flach, 2019; Wachter et al., 2017). Myers et al. (2020) proposed a preliminary design of an interactive visualization tool, CEB, in order to reveal the bias in the neural network (NN), which is commonly used in artificial intelligence. CEB combines counterfactual examples with the abstraction of neural network decision-making processes, enabling non-experts to discover biases. In relational classification, counterfactual conditionals can be viewed as a subset of Condition type of Contingency class in the Penn Discourse Treebank (PDTB) (Prasad et al., 2008) or the Condition relation of Rhetorical Structure Theory (RST) (Mann and Thompson, 1987). Moreover, many researchers have tried to use PDTB and RST for end-to-end discourse relationship analysis (Biran and McKeown, 2015; Lin et al., 2009; Ji and Eisenstein, 2014).

3 Methodology and Data

3.1 Data description

In this task, we only used the official provided training dataset. In Subtask1, the purpose is to distinguish whether a given sentence is counterfactual, so data labels are divided into two categories: if the sentence is counterfactual, it is represented by label 1, otherwise it is represented by label 0. Counterfactuals describe events that are contrary to facts. In addition, we also conducted statistics on the categories of the training dataset. The training dataset contains 13,000 instances, of which there are 1454 counterfactual instances and 11546 factual instances, the ratio is about 88.82% : 11.18%. We found that the category labels were severely imbalanced, especially the amount of label 1 only accounts for 11.18%. But in this task, we did not take effective measures to solve the imbalance of data.

3.2 ON-LSTM with HAN and Pooling

Our proposed network architecture is shown in Figure 1. Our model is based on Ordered Neurons LSTM (ON-LSTM) with Hierarchical Attention Network (HAN) and adds Pooling operations. Next, we briefly describe the details of the system.

- **Input Layer:** This layer mainly inputs all pre-processed text data into the model.

[1]Just looking for words like "if" will not produce useful results; only 2% of tweets containing "if" are counterfactual

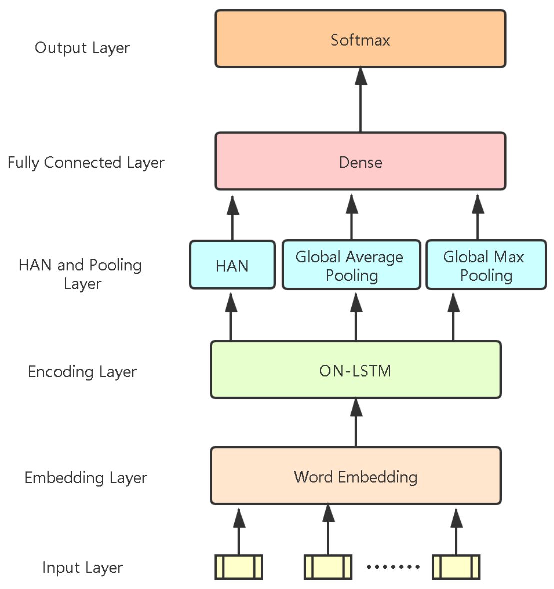


Figure 1: Architecture based on ON-LSTM with HAN and Pooling

- **Embedding Layer:** This layer vectorizes the words input by the pre-trained word vector model in the existing dictionary.
- **Encoding Layer:** In the encoding layer, we use ON-LSTM to encode vectorized text. ON-LSTM aims at the characteristics of grammatical layering for natural language. From the perspective of the grammatical structure, the cell states are arranged in an orderly manner according to the grammatical level, and then different update rules are implemented according to the different grammatical levels, so as to achieve the retention of higher grammatical level information. In other words, ON-LSTM integrates the hierarchical structure (tree structure) into the LSTM through a specific ordering, thereby allowing the LSTM to automatically learn the hierarchical structure information to express richer information. ON-LSTM is a new variant of LSTM (Shen et al., 2019). The main difference between them is that the update mechanism from $\hat{c_t}$ to c_t is different. The update formula for the entire ON-LSTM is as follows:

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \tag{1}$$

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \tag{2}$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \tag{3}$$

$$\hat{c_t} = \tanh(W_c x_t + U_c h_{t-1} + b_c) \tag{4}$$

$$\tilde{f_t} = \overrightarrow{c_s}(softmax(W_{\tilde{f}} x_t + U_{\tilde{f}} h_{t-1} + b_{\tilde{f}}) \tag{5}$$

$$\tilde{i_t} = \overleftarrow{c_s}(softmax(W_{\tilde{i}} x_t + U_{\tilde{i}} h_{t-1} + b_{\tilde{i}}) \tag{6}$$

$$w_t = \tilde{f_t} \circ \tilde{i_t} \tag{7}$$

$$c_t = w_t \circ (f_t \circ c_{t-1} + i_t \circ \hat{c_t}) + (\tilde{f_t} - w_t) \circ c_{t-1} + (\tilde{i_t} - w_t) \circ \hat{c_t} \tag{8}$$

$$h_t = o_t \circ \tanh(c_t) \tag{9}$$

Among them, the f_t, i_t, o_t are called forget gate, input gate, output gate respectively, the input is the historical information h_{t-1} and the current information x_t. $\overrightarrow{c_s}$ and $\overleftarrow{c_s}$ are cumsum() operations in the right and left directions, respectively. The newly introduced $\tilde{f_t}$ and $\tilde{i_t}$ represent the master forget gate and master input gate respectively. w_t represents a vector where the intersection part is 1 and the rest is all 0. In this way, the high-level information remains a considerable long distance, while the low-level information may be updated at each step of input, thereby embedding the hierarchical structure through information grading.

- **HAN and Pooling Layer:** In this layer, we use HAN and Pooling operations. HAN uses a hierarchical Attention architecture to apply two levels of attention to word-level and sentence-level, respectively. When weighting, dynamic weight can be given according to the content, so that the words biased to counterfactual language are more prominent, so as to classify sentences. The core structure of the HAN network consists of two parts. The bottom part of the model is a word encoder plus an Attention layer based on word encoding. The upper part of the model is a sentence encoder and a sentence-based Attention layer. At the same time, because the output of ON-LSTM is three-dimensional, we use Global Average Pooling and Global Max Pooling operations to reduced three-dimensional to two-dimensional.
- **Fully Connected Layer:** This layer extracts and integrates useful information.
- **Output Layer:** This layer uses a softmax activation function and classifies and predicts the final aggregated information.

3.3 K-folding ensemble

In this task, due to the small dataset, we use the K-fold method to improve the performance of the model. The design idea of this method comes from K-fold cross-validation, we randomly divide the source data into K parts and use the K-1 subsets to do the training, the remaining subset is the validation set, and then this process is repeated K times. Finally, the K results are summed and do an averaging operation on the accumulated objects to get the final output. From the K-fold idea, we know that when using K-fold, the value of K needs to be greater than 1. The purpose of performing the K-fold ensemble is to train to different data sets during each fold training process, and to extract different features during the model feature extraction process, which can further improve the generalization ability of the model.

4 Experiment and results

4.1 Data preprocessing

We perform some simple preprocessing operations on the data, such as using NLTK to download stopwords and removing stopwords[2], replacing repeated full stop, question marks, and exclamation points with single instance punctuation marks with the special mark ”repeat”. Replaced all uppercase letters with lowercase letters. Restored abbreviated words. Lemmatization, using WordNetLemmatizer to restore language vocabulary to its general form (which can express complete semantics). The stopwords are removed to delete some meaningless words, and the purpose of lemmatization is to reduce the number of unknown words. Here we have not considered that some modal verbs should be deleted due to stopwords affecting performance, which may be one of the reasons for the low effectiveness of our model. Finally, since the official did not provide a validation set, we randomly selected 1,000 instances from the official training set as the validation set.

4.2 Experiment setting

In our model, the pre-trained word embedding we used is FastText [3], which is provided by Mikolov et al. (2017). It is a 2 million word vector trained using subword information on Common Crawl with 600B tokens, and its dimension is 300 (crawl-300d-2M.vec). For the training data we used 10-fold, we set the batch size to 256 and epoch to 20. In our model, for the ON-LSTM layer, we set the hidden unit to 128. To prevent the model from overfitting, a Dropout layer was added between the ON-LSTM layer and the HAN layer, and the Dropout was set to 0.5. At the Pooling layer, we used AveragePooling1D () and

[2]http://www.nltk.org/nltk_data
[3]https://fasttext.cc/docs/en/english-vectors

MaxPooling1D (), and set pool_size to 2. At the fully connected layer, we set the hidden unit to 256 and selected Relu as it's activation function. Finally, we added the Dropout layer and the BatchNormalization layer, where Dropout was set to 0.5. In the output layer, we used the sigmoid activation function for binary classification. The loss function of this model is binary cross-entropy (unweighted), and the optimizer is Adam (standard parameters of Adam optimizer in keras.opimizers).

4.3 Result

This Subtask1 evaluates the classification system by calculating the F1 score. The left side of Table 1 shows the results of different models running without the K-fold method. We can observe that the F1 score of the ON-LSTM model is higher than that of LSTM and GRU, which are 0.109 and 0.036, respectively. This shows that the ON-LSTM model is effective in detecting counterfactuals, because there are conditional and logical relations in counterfactual sentences, and there are complex hierarchical structures in sentences. The ON-LSTM model can learn the syntactic structure of sentences, and make use of the order information of neurons, so that the model can encode the hierarchical structure of sentences, so as to obtain more abundant semantic information, enhance the expression ability of LSTM and effectively improve the performance of the model.

Model	K value	F1	Run	K value	F1
LSTM+HAN	k=0	0.503	ON-LSTM+HAN(run_1)	k=0	0.612
GRU+HAN	k=0	0.576	ON-LSTM+HAN(run_2)	k=5	0.637
ON-LSTM+HAN(run_1)	**k=0**	**0.612**	**ON-LSTM+HAN(run_3)**	**k=10**	**0.704**

Table 1: Running results of different model(Left). Running results of different K values(Right)

The right side of Table 1 shows the running results of different K values of the same model. We can observe that run_1 is the result of not using the K-fold method, while run_2 and run_3 are the results of using the K-fold method. For run_2, the performance of the model is improved by 0.134, which shows that the K-fold method greatly improves the performance of the model. For run_3, we adjusted the K value slightly on the basis of run_2. In our experiment, the maximum value of K is 10, and the value of K depends on the situation of different tasks.

Team Name	F1	Recall	Precision
HIT	0.9090(1)	0.9190	0.9000
BUT-FIT	0.9030(2)	0.9070	0.9000
YNU_OXZ	**0.7040(16)**	**0.6610**	**0.7520**

Table 2: Official leaderboard results

Table 2 shows the results of ours and the top two teams on the official leaderboard, the best team obtained an F1 score of 0.9090, and the F1 score of our model is only 0.7040 (Rank16 / 27). It can be seen from the confusion matrix that our model is more focused on predicting label 0, the prediction correct rate reached 0.97, and mispredict a lot of label 1 as label 0, which may be because the class label in the training dataset is heavily imbalanced (label 0 accounts for 88.82% in training set), and our model fails to effectively solve this problem. The confusion matrix of the model prediction results in subtask1 is shown in Appendix A.

5 Conclusion

This paper introduces the general idea and specific scheme of our participation in Subtask1 in SemEval-2020 Task5, which aims to identify which of the two sentences is a counterfactual statement. We use the model based on the combination of ON-LSTM with HAN and Pooling for classification, and used the K-fold approach to ensemble which could improve model performance. We found that the data is seriously unbalanced. Undersampling or oversampling the data can effectively solve the problem of data imbalance, and transfer learning can effectively improve the performance of the model. We will explore these methods in the future.

References

Or Biran and Kathleen McKeown. 2015. Pdtb discourse parsing as a tagging task: The two taggers approach. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 96–104.

Nelson Goodman. 1947. The problem of counterfactual conditionals. *The Journal of Philosophy*, 44(5):113–128.

S Hochreiter and J Schmidhuber. 1997. Long short-term memory. 9:1735–1780.

Yangfeng Ji and Jacob Eisenstein. 2014. Representation learning for text-level discourse parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13–24.

Matt J Kusner, Joshua Loftus, Chris Russell, and Ricardo Silva. 2017. Counterfactual fairness. In *Advances in Neural Information Processing Systems*, pages 4066–4076.

Ziheng Lin, Min-Yen Kan, and Hwee Tou Ng. 2009. Recognizing implicit discourse relations in the penn discourse treebank. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 343–351.

William C Mann and Sandra A Thompson. 1987. *Rhetorical structure theory: A theory of text organization*. University of Southern California, Information Sciences Institute Los Angeles.

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2017. Advances in pre-training distributed word representations. *CoRR*, abs/1712.09405.

Chelsea M Myers, Evan Freed, Luis Fernando Laris Pardo, Anushay Furqan, Sebastian Risi, and Jichen Zhu. 2020. Revealing neural network bias to non-experts through interactive counterfactual examples. *arXiv preprint arXiv:2001.02271*.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind K Joshi, and Bonnie L Webber. 2008. The penn discourse treebank 2.0. In *LREC*. Citeseer.

Yikang Shen, Shawn Tan, Alessandro Sordoni, and Aaron C. Courville. 2019. Ordered neurons: Integrating tree structures into recurrent neural networks. abs/1810.09536.

Kacper Sokol and Peter A Flach. 2019. Counterfactual explanations of machine learning predictions: Opportunities and challenges for ai safety.

Youngseo Son, Anneke Buffone, Joe Raso, Allegra Larche, Anthony Janocko, Kevin Zembroski, H Andrew Schwartz, and Lyle Ungar. 2017. Recognizing counterfactual thinking in social media texts. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 654–658.

Philip E Tetlock and Aaron Belkin. 1996. *Counterfactual thought experiments in world politics: Logical, methodological, and psychological perspectives*. Princeton University Press.

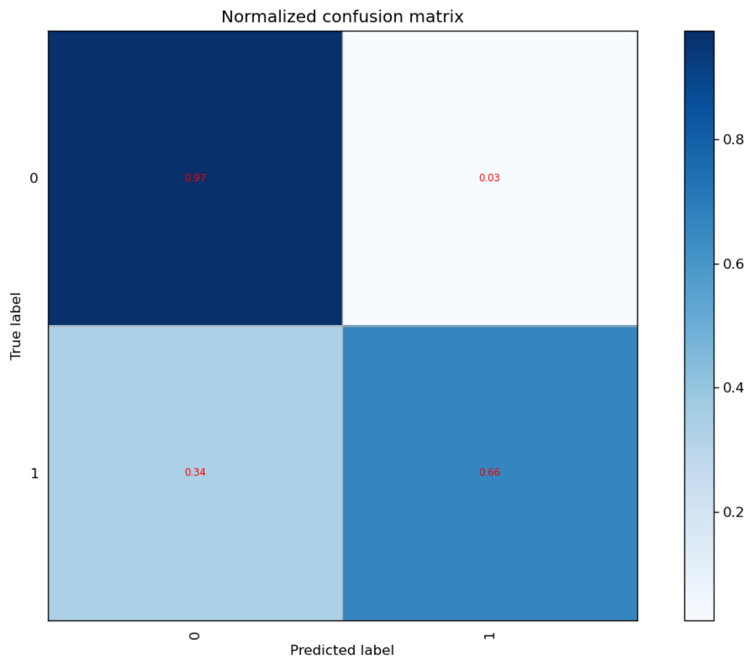
Sandra Wachter, Brent Mittelstadt, and Chris Russell. 2017. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harv. JL & Tech.*, 31:841.

Jin Wang, Bo Peng, and Xuejie Zhang. 2018. Using a stacked residual lstm model for sentiment intensity prediction. *Neurocomputing*, 322:93–101.

Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. 2019. Heterogeneous graph attention network. abs/1903.07293:2022–2032.

Xiaoyu Yang, Stephen Obadinma, Huasha Zhao, Qiong Zhang, Stan Matwin, and Xiaodan Zhu. 2020. SemEval-2020 task 5: Counterfactual recognition. In *Proceedings of the 14th International Workshop on Semantic Evaluation (SemEval-2020)*, Barcelona, Spain.

A Appendix A : Confusion matrix of model for subtask1


Normalized confusion matrix
True label
Predicted label