EECS595 Final Project Proposal: Exploring LSTM Networks Approach on TRIP Dataset

Yingzhuo Yu

1 Introduction

As for the language understanding tasks, large pre-trained language models such as $BERT$, $GPT-3$, $GPT-2$ and $RoBERTa$ have actually achieved impressive end performance. However, when it comes to supporting the predictions by providing evidence besides correctly determining the plausibility of stories, the results achieved by large language models on three tasks(accuracy, consistency and verifiability) based on TRIP dataset are not high as expected (Storks et al., 2021). In order to further explore the underlying reasoning process, this project is going to design and implement a different NLP system that deal with the first two subtasks to improve both accuracy and consistency. We try to explore the new approach by adding LSTM after the first embedding layer and we will make comparisons among different configurations of the new model. Based on the performance of our model pipeline, we are going to compare with the baseline results in Storks et al., 2021 and go deeper into the connection between plausibility classification and the reasoning process.

2 TRIP Dataset

The dataset we use in this project is Tiered Reasoning for Intuitive Physics (TRIP), first introduced by Storks et al. (2021). It's a benchmark aiming at commonsense reasoning. The general task is to classify the plausibility of stories. In the dataset, it includes multiple pairs of stories that are similar mutually and different by only one sentence. Each story contains at least five sentences. Given a pair of stories, systems have to complete tiered tasks. Firstly, it must identify the plausible story from two stories. Then, it have to choose a pair of sentences that conflict mutually. In TRIP dataset, there is only one pair of sentences causing the conflict. Finally, it should figure out the physical states in those two sentences. The whole reasoning process actually reflects how human beings are going to judge the plausibility.

The stories in TRIP dataset are produced from Amazon Mechanical Turk(Storks et al., 2021). Every story is made up of at least five sentences. In this project, we filter out the stories whose length is greater than five. Hired workers are asked to replace the original sentence with the new sentence in the story so as to make the new one become implausible and unrealistic. And this implausibility only comes from the contradiction between the new sentence and the original sentences. In other words, the new sentence itself is plausible and is implausible only if it is connected to another sentence in the story. To reduce subjectivity, for each sentence, it is written in a simple form as "agent-verb-object". It also helps to lay more emphasis on the ability of reasoning for the machine to be trained. For the whole dataset, we split it into three parts, training data, validation data and test data.

3 Previous Work

Physical state change in text is tracked based on ProPara Mishra et al. (2018). Two different kinds of model are implemented to track the location and existence of entities from the beginning of the text to the end. Since the states are sometimes implicit, it requires the machine to have the ability of inferring, which is similar to our project, but demands more. And it concludes that new datasets are required in order to go further in machine reading comprehension of various texts.

The baseline result shown in Storks et al., 2021 indicates that the model performs well in end task but there exists noteworthy disconnection between the plausibility prediction and the reasoning process performance. The score is relatively low in Consistency and Verifiability. Priming large language models based on high-level task to provide supporting evidence of understanding leads to the inconsistent and unverifiable reasoning process (Storks

et al., 2021). Instead, it is suggested that the NLP systems should be trained based on several kinds of lower-level evidence jointly so as to get the final reasoning result.

4 Approach

The whole reasoning model system architecture is shown in Figure 1. The details of each module are described as follows.

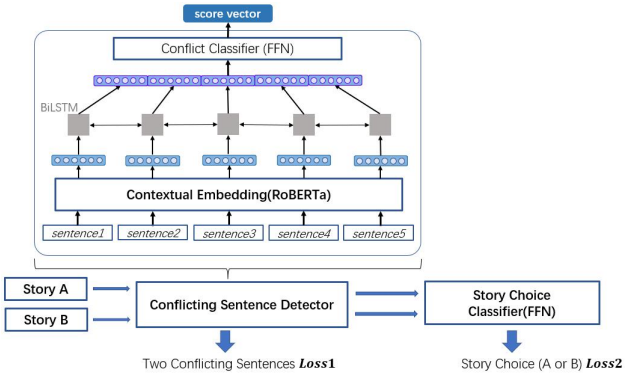

Figure 1: Reasoning System

Contextual Embedding

The first module is the contextual embedding layer. It takes a sentence as the input. We use the corresponding tokenizer to tokenize the sentences for different pretrained language models and we added the special tokens to the encoder. The embedding is based on pre-trained language models and its output is actually the numerical representation of input context. In most cases, the length of the output vector is 768 for base models and 1024 for large models.
In the following section, we will experiment on different language models including RoBERTa, DistilRoBERTa, RoBERTa-large, BERT and BERT-large.

LSTM Layer

We apply Long short-term memory network layer after the contextual embedding layer. Every sentence vector is feed into LSTM. The sequence length is five and we expect this recurrent neural network can help to capture the dependencies between sentences in a story so as to better detect conflict. Before the next module, we first concatenate the output vector for each sentence. The combined vector contains the information of contextual change throughout the story.
And in the following section, we will experiment on different type of recurrent neural networks including LSTM and BiLSTM.

Conflict Classifier

Conflict classifier is actually the feed forward neural network with three hidden layers. The input is the concatenated vector from LSTM layer and a dropout layer is added before feeding into conflict classifier module. And after the first two layers, the size of the output score vector is ten. Since there are totally five sentences in the story, the total probabilities of conflicting sentence pair is $\binom{5}{2} = 10$. The corresponding conflicting pair for each entry is shown below. The final score for each sentence pair

s_0, s_1	s_0, s_2	s_0, s_3	s_0, s_4	s_1, s_2	s_1, s_3s	s_1, s_4	s_2, s_3	s_2, s_4	s_3, s_4

Table 1: conflicting pair correspondence of score vector

is generated by the $softmax$ function. One with highest score represents the predicted potential conflicting pairs. The score vector before the $softmax$ function concatenated with the score vector from another story is the final output of conflicting sentence detector.

Story Choice Classifier

The story choice classifier is another feed forward neural network with no hidden layer. Given the score vectors from two stories, it directly outputs a vector of size two also applied with $softmax$ function. The entry with the highest score shows the implausibility of the story.

Model Training

In this project, we train the parameters of the model architecture through the gradient descent on the Loss

$$L_{total} = \lambda_1 * L_c + \lambda_2 * L_s, \quad (1)$$

where L_c represents the loss for conflicting sentence classification and L_s represents the loss for story choice classification. L_total just sum these two losses up with weights $\lambda_1 + \lambda_2 = 1$.
For the learning rate, we set the initial value of 1e-4 and its value decreases linearly with the process of training. we expect the decreasing learning rate help improve the speed and convergence in gradient descent.

5 Evaluation

5.1 Evaluation Metrics

The evaluation metrics we apply in this project is almost same with the metrics in Storks et al., 2021. Specifically, we have two main tasks, Accuracy and Consistency.

Accuracy

We apply the accuracy as the metrics of Accuracy, namely the proportion of the cases where the implausible story is correctly classified from a pair of stories.

Consistency

We apply the accuracy as the metrics of Consistency: the proportion of the cases where the implausible story is correctly classified from a pair of stories and furthermore, the conflicting sentence pair is selected out correctly.

5.2 Experiment on Loss Weight

Different combinations of λ_1 and λ_2 affect the performance of the model pipeline. For the first experiment, we try totally four combinations: $(0, 1), (0.6, 0.4), (0.8, 0.2), (1, 0)$. For the first case, $\lambda_1 = 0$ indicates that we omit the loss of selecting conflicting sentences while in the last case of $\lambda_2 = 0$, we omit the loss of choosing the correct story. For the remaining cases, we assume that both L_c and L_s are taken into consideration. In this experiment, On one hand, we want to compare the situation when all losses are considered with the situation when story choice loss is omitted. Based on our model architecture, we expect that our model performs best when all losses are considered with appropriate weights. Neither omitting the story choice nor omitting the conflicting sentences choice will decrease the performance of our model. The training plots for each λ combination are shown below.

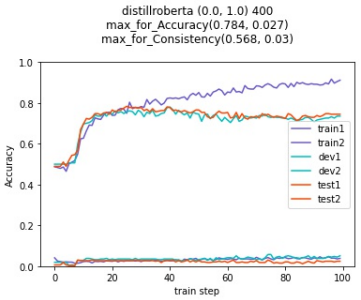

Figure 2: distillroberta with λ_1:0 λ_2:1 (omit story choice)

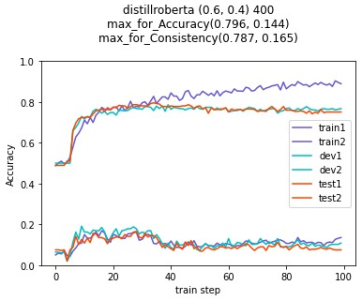

Figure 3: distillroberta with λ_1:0.6 λ_2:0.4

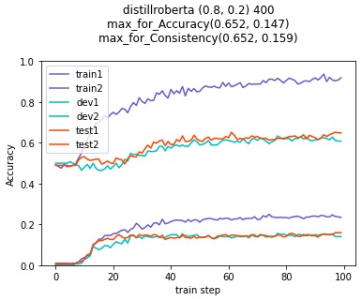

Figure 4: distillroberta with λ_1:0.8 λ_2:0.2

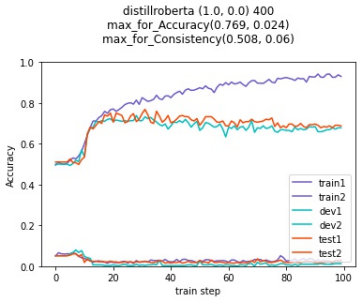

Figure 5: distillroberta with λ_1:1 λ_2:0

The Accuracy and Consistency results are shown in the table below. And we compare with the baseline result in Storks et al., 2021. Out of expectation,

Model	Accuracy	Consistency
All Losses		
BERT(baseline)	78.3%	2.8%
RoBERTa(baseline)	75.2%	6.8%
DeBERTa(baseline)	74.2%	2.2%
DistillRoberta (0.6,0.4)	78.7%	16.5%
DistillRoberta (0.8,0.2)	65.2%	15.9%
Omit Story Choice (0,1)		
BERT(baseline)	73.9%	28.0%
RoBERTa(baseline)	73.6%	22.4%
DeBERTa(baseline)	75.8%	24.8%
DistillRoberta (0,1)	78.4%	3.0%

Table 2: Comparison with baseline result on different weights

the model pipeline performs better on Consistency if all losses are taken into consideration than the case when only the loss of conflicting is considered.

Among two combinations of weights, $\lambda_1 = 0.6$ and $\lambda_2 = 0.4$ is better than $\lambda_1 = 0.8$ and $\lambda_2 = 0.2$ with Accuracy 78.7% and Consistency 16.5%. For the case of all losses, we choose $\lambda_1 = 0.6, \lambda_2 = 0.4$ based on the previous performance.

The low score in Consistency may be caused by omitting the loss of story choice itself. Since in our model architecture, the output of the score vector will not only be used to identify conflicting sentences, but will also be used for the following story choice. Hence, if the story choice is omitted, the parameters in the model will not be punished when mislabeling the story. And as Consistency is based on the performance on Accuracy, the model's score for consistency becomes relatively low.

5.3 Experiment on Pretrained Language Model

For the second experiment, we try out different pretrained large language models for the contextual embedding layer including RoBERTa(Liu et al., 2019) ,BERT(Devlin et al., 2019) and DistilRoBERTa(Sanh et al., 2019) together with corresponding tokenizers. For each language models, we experimented on both of its base version and large version. We are going to compare those models on both Accuracy and Consistency so that we can find whether the language model with more parameters outperforms the others. The results for each model are shown in the table below. We also compare them with the baseline results. Based on the table, we can see that for the case of all loses, almost every pretrained language model performs better than the baseline result both on Accuracy and Consistency. For Accuracy, the highest accuracy is achieved by DistilRoBERTa at about 0.787 while for Consistency, the highest accuracy is achieved by BERT-large at 0.207. However, in the case of omitting story choice loss, the performance is not high as expected, especially for Consistency. The highest accuracy for Consistency is around 0.111 achieved by RoBERTa-base, which is much less than the baseline result. Thus, to make our model pipeline better in understanding story as well as reasoning, being trained on both high-level end task and lower-level predictions task is necessary. Besides, according to the performance of RoBERTa-base and other models, we also find that both the distilled version and the large version perform better. For the distillation version, it is obtained by compressing the knowledge from a large model to a smaller model. And for the large version, the parameter number is larger than the base version. Thus, for both cases, they have better understanding towards story.

Model	Accuracy	Consistency
All Losses		
BERT(baseline)	78.3%	2.8%
RoBERTa(baseline)	75.2%	6.8%
DeBERTa(baseline)	74.2%	2.2%
DistillRoBERTa	**78.7**%	16.5%
RoBERTa-base	69.4%	8.1%
BERT-large	76.3%	**20.7**%
Omit Story Choice		
BERT(baseline)	73.9%	28.0%
RoBERTa(baseline)	73.6%	22.4%
DeBERTa(baseline)	75.8%	24.8%
DistilRoBERTa	78.4%	3.0%
RoBERTa-base	76.2%	11.1%
RoBERTa-large	52.6%	2.4%
BERT-large	74.2%	10.2%

Table 3: Comparison with baseline result on different language models

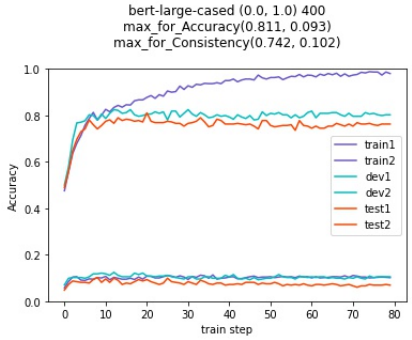

Figure 6: bert-large with λ_1:0 λ_2:1

Here is one of the training plots for BERT-large with story choice omitting. Based on the plot, we can find that the Accuracy for training is increasing with the epoch while for Accuracy on test dataset, the score increases first and then keeps stable. This is the correct tendency showing that the model is priming for the training data and becomes generalized to test data. However, for the accuracy of Consistency, all of the score on training data, validation data and test data are close in value. We may conclude that our model pipeline is better in training for improving high-level task.

5.4 Experiment on LSTM

For the third experiment, we try out different recurrent neural network, basically Long Short-Term

Memory(Sak et al., 2014) and Bidirectional Long Short-Term Memory(Zhou et al., 2016). The object information regarding the conflict can appear in any sentence. Our expectation is that long short-term memory can help to detect dependencies between every two sentences so that the conflict between two sentences can be identified. The results for LSTM layer and BiLSTM layer are shown in the table below. We also compare the results with the baseline results on Accuracy and Consistency.

Model	Accuracy	Consistency
All Losses		
BERT(baseline)	78.3%	2.8%
RoBERTa(baseline)	75.2%	6.8%
DeBERTa(baseline)	74.2%	2.2%
DistilRoBERTa	78.7%	16.5%
$DistilRoBERTa^*$	76.0%	10.2%
Omit Story Choice		
BERT(baseline)	73.9%	28.0%
RoBERTa(baseline)	73.6%	22.4%
DeBERTa(baseline)	75.8%	24.8%
DistilRoBERTa	78.4%	3.0%
$DistilRoBERTa^*$	72.1%	5.4%

Table 4: Comparison with baseline result on different LSTM, $DistilRoBERTa^*$ refers to using LSTM while the default is BiLSTM.

Based on the table, we can see that for the case of all loses, DistilRoBERTa with BiLSTM outperform DistilRoBERTa with LSTM. Since Bidirectional long-short term memory use the sequence information from both directions, the model pipeline can understand the story context through sentences better and know more about the relationship between every two sentences. However, that is not the case when we omit the loss of story choice. DistilRoBERTa with LSTM achieves higher accuracy in Consistency then DistilRoBERTa with BiLSTM. Considering the low score of the consistency compared with the baseline result, the improvement of LSTM on Consistency can be ignored. We can conclude that combining DistilRoBERTa with BiLSTM have the outstanding performance on both Accuracy and Consistency if we include all the losses L_C and L_S.

6 Discussion

Besides all the experiments mentioned before, I also optimize the model by choosing different hidden dimension of the output vector of BiLSTM layer. And here in Table 5, we present the best results for the optimal loss function weight$(0.6, 0.4)$, recurrent neural network layer$(BiLSTM)$ as well as the dropout probability(0.2).

Model	Accuracy	Consistency
All Losses		
BERT(baseline)	78.3%	2.8%
RoBERTa(baseline)	75.2%	6.8%
DeBERTa(baseline)	74.2%	2.2%
BERT-large	**79.9%**	**16.5%**
Omit Story Choice		
BERT(baseline)	73.9%	28.0%
RoBERTa(baseline)	73.6%	22.4%
DeBERTa(baseline)	75.8%	24.8%
BERT-large	**76.3%**	**16.2%**

Table 5: Comparison with baseline result.

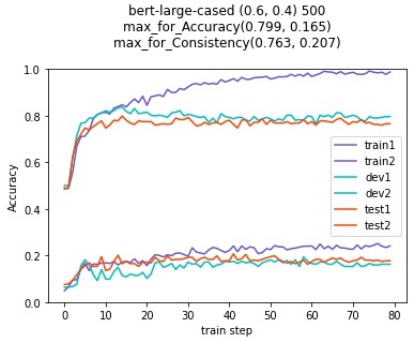

Figure 7: Training plot of bert-large model + BiLSTM with best performance for all Losses

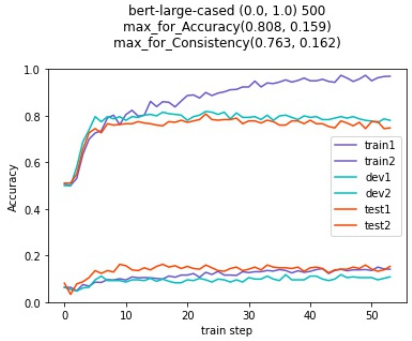

Figure 8: Training plot of bert-large model + BiLSTM with best performance for omitting story choice loss

As is shown in Table 5, our model obtains the noteworthy performance in the case of all losses. When the story choice loss is omitted, the overall understanding of our model pipeline towards the story decreases. This decrease is caused by the feature of our model architecture that the score vectors from BiLSTM layer is directly used for

following story choice.

7 Conclusion

In this work, we proposed a new model architecture that combines pretrained large language models with Bidirectional Long Short-Term Memory. We experimented with different configurations of every module so as to optimize our model. And based on the final result, we conclude that for some certain model architecture, large language models fail to perform well even if we prime it to high-level task only or low-level task only. For training the model, we are supposed to make sure that tasks of all levels should be taken into consideration. The model architecture we propose in this project closely combines two tasks of selecting two sentences causing conflicts and determine the plausibility of the story so that the reasoning process of the model can be strengthened. Compared with the baseline result, we succeed in improving the Consistency and Accuracy of the model in the case of all losses.
Another conclusion is that by applying Bidirectional Long Short-Term Memory, we notice its positive impact on overall model performance. Our expectation of capturing the dependencies between sentences by using BiLSTM may not be reflected on TRIP dataset. And we may assume that it can be further explored if the length of the stories becomes much larger. Also, in Storks et al., 2021, physical states for each object in the sentence are introduced for verifiability task, which is not taken into consideration in this project. In future work, the physical states can be added on the basis of our model pipeline in this project.

8 Work Partition

Since I solo the whole project, I'm responsible for all the works, including designing the model, training and evaluating the system on TRIP Dataset, recording the result, visualizing the training plot as well as writing the final project report.

9 GitHub Code Link

Dataset and all relevant code have been uploaded on GitHub with link: https://github.com/Jason-csc/EECS595-final-project.

References

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

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.

Bhavana Dalvi Mishra, Lifu Huang, Niket Tandon, Wen-tau Yih, and Peter Clark. 2018. Tracking state changes in procedural text: a challenge dataset and models for process paragraph comprehension. *arXiv preprint arXiv:1805.06975*.

Haşim Sak, Andrew Senior, and Françoise Beaufays. 2014. Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108.

Shane Storks, Qiaozi Gao, Yichi Zhang, and Joyce Chai. 2021. Tiered reasoning for intuitive physics: Toward verifiable commonsense language understanding. *arXiv preprint arXiv:2109.04947*.

Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 207–212, Berlin, Germany. Association for Computational Linguistics.