An Approach to Boost the Baseline Performance on TRIP

Ziyan Wang
University of Michigan
ziywang@umich.edu

Jim Yang
University of Michigan
yzr@umich.edu

Abstract

BERT is a pre-trained model that can be fine-tuned to create state-of-the-art models for various natural language processing tasks, however, it fails to perform well on TRIP, which is a benchmark dataset with tiered tasks for commonsense reasoning. In this report, We introduce a sub-task training approach, which aims to tune the encoding layers in the Bert model on another dataset which contains the similar domain knowledge with TRIP, then feed the tuned Bert model into the TRIP dataset.

1 Introduction

A lot of recent research activities have been working on commonsense reasoning in natural language understanding. Many large-scale benchmark datasets were developed and online leaderboards encourage researchers to solve them. Research scientists have derived large-scale language models (LMs) pre-trained on massive amounts of online texts (Peters et al., 2018a; Brown et al., 2020), and nowadays, the best models can already achieve human-level performance or even better in language understanding tasks such as commonsense inference (Bowman et al., 2015a). However, besides just achieving these tasks with a high performance, some scientists are interested in how well machines understand the tasks that they are solving, in other words, whether machines can perform verifiable reasoning as humans do (Storks et al., 2021). There is an important concern that if widespread bias in language benchmarks would lead to superficial correlations between context and class labels (Schwartz et al., 2017; Gururangan et al., 2018), thus allows systems to bypass reasoning and achieve artificially high performance (McCoy et al., 2019; Niven and Kao, 2019). As a result, it is not clear whether machines really solve the problem or can perform verifiable reasoning like humans.

In order to help solving the above problem, (Storks et al., 2021) produced a benchmark targeting physical commonsense reasoning calls Tiered Reasoning for Intuitive Physics (`TRIP`). In addition to a high-level end task for story plausibility, it also provides a common proxy task for commonsense reasoning problems. This benchmark includes dense annotations for each story capturing multiple tiers of reasoning beyond the end task. A tiered evaluation is proposed by (Storks et al., 2021) from these annotations. Given a pair of stories that differs only by one sentence which makes one of them implausible, the language model has to jointly identify three things: the plausible story, a pair of conflicting sentences in the implausible story and the underlying physical states that cause the conflicts. Such systematic evaluation aims to figure out whether a high-level plausibility prediction can be verified based on lower-level understanding.

(Storks et al., 2021)'s work has shown a problem, which is what we are interested in this project. They have shown that although large language models are able to perform really well on high end task, these models have poor performance on jointly supporting their predictions with the proper evidence. As a result, such high end task predictions are not really related to language model's understanding of how the world works. Their work provides baseline performance of BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019) and DeBERTa (He et al., 2021). Limited by time and group size, we only focused on the BERT pre-trained model for this project.

The problem we try to solve in this project is to improve the baseline performance in the TRIP paper. As shown in Figure 1, We introduce a subtask training method on the Bert model, which aims to update the parameters for the encoding layers of pre-trained Bert model, and make it closer to the domain knowledge of TRIP dataset.

The result shows that our method did not outper-

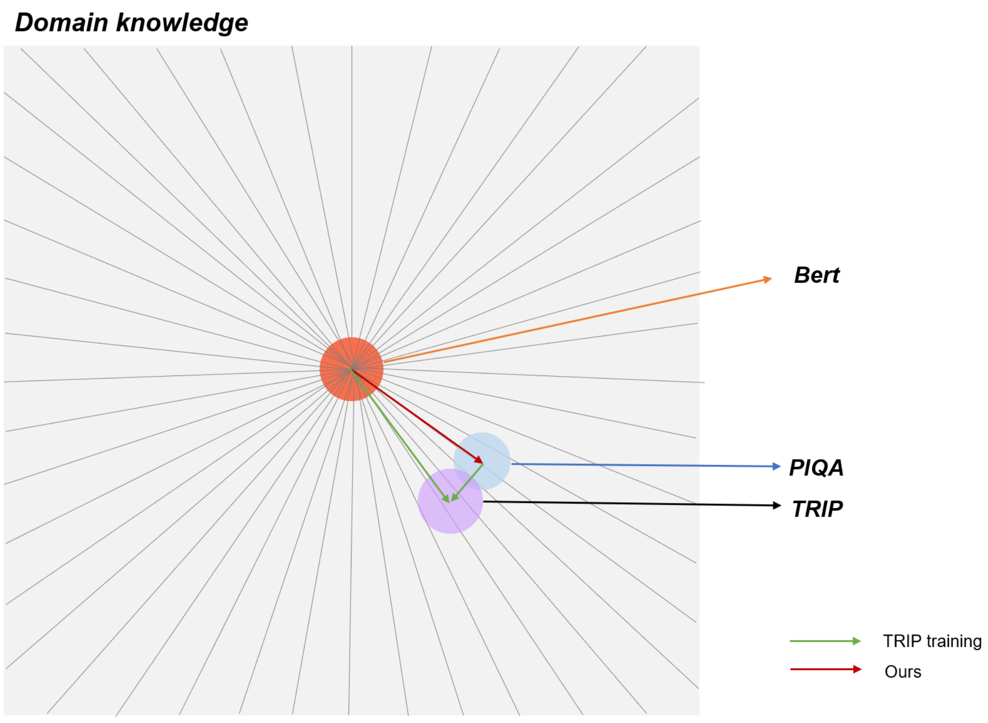


Figure 1: Summary of idea

form the baselines in the original paper. We will compare our performance with the baselines by using the same evaluation metric in the TRIP paper, and we will also discuss the future improvements that may improve the benchmarks further.

2 Related Work

We briefly review three pre-trained language models: BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019) and DeBERTa (He et al., 2021), for which the TRIP paper used to power several variations for the proposed reasoning system.

2.1 BERT

Language model pre-training is effective for improving a lot of natural language processing tasks (Dai and Le, 2015; Peters et al., 2018b), including but not limited to sentence-level tasks such as natural language inference (Bowman et al., 2015b; Williams et al., 2018) and paraphrasing (Dolan and Brockett, 2005), and token-level tasks such as named entity recognition and question answering (Sang and Meulder, 2003). However, the existing two approaches for apply pre-training (fine-tuning and feature-based) both used unidirectional language models to learn general language representations, and the authors of BERT found such techniques restrict the power of the pre-trained representations. So they proposed BERT, which stands for **B**idirectional **E**ncode **R**epresentations from **T**ransformers.

The BERT model is better than the previous approaches because it reduces the unidirectionality constraint by using a "masked language model" (MLM) pre-training objective as shown in Figure 2. In this method, some tokens from the input will be masked randomly by the masked language model. The objective is predicting the original vocabulary id of the masked word only based on its context. This approach is not similar to left-to-right language model pre-training, but its objective enables the representation to fuse the left and right context, thus pre-training a deep bidirectional transformer is possible. Besides MLM, the authors of BERT proposed a task called "next sentence prediction", which jointly pre-trains text-pair representations. BERT is the first fine-tuning based representation model to achieve state-of-the-art performance on a large suite of tasks, and it advances the state of the art for many NLP tasks.

2.2 RoBERTa

The inspiration of RoBERTa is that also self-training models like BERT (Devlin et al., 2019) have gained great performance, it is still hard to determine which parts of the approaches really con-

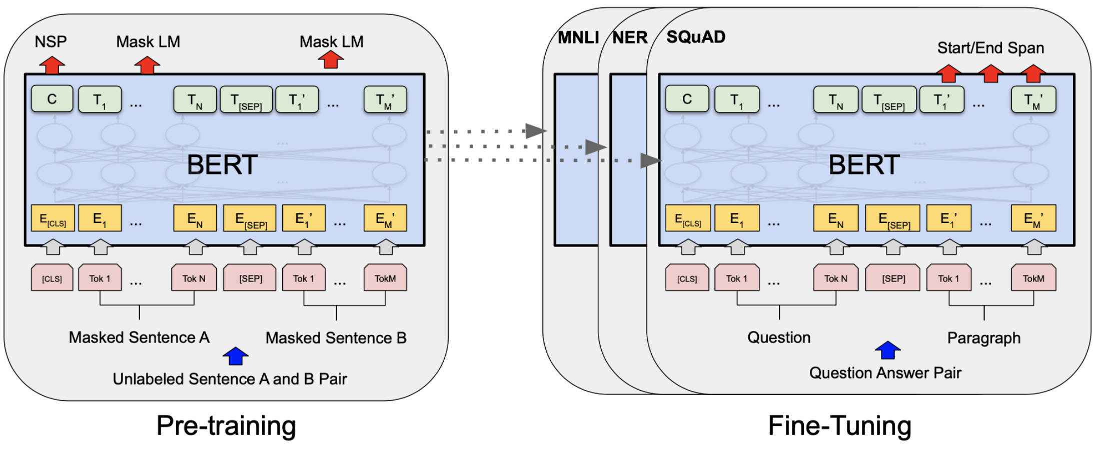


Figure 2: Overall pre-training and fine-tuning procedures for BERT (Devlin et al., 2019).

tribute to the performance gain. In addition, training is expensive in computation and thus limits the amount of tuning. Such training is usually done with private datasets that have various sizes, which limit the possibility to test the effects of the modeling advances.

RoBERTa has a evaluation of the effects of hyperparameter tuning and training set size. The authors of this paper found that BERT was extremely undertrained, so they proposed this robustly optimized pre-training approach called RoBERTa. A number of modifications were applied. The model was trained for long time with more data and larger batches. The next sentence prediction objective was removed, and it was trained on longer sequences. Lastly, the model dynamically changes the masking pattern applied to the training data. With these modifications the performance of pertrained BERT models were substantially improved. This improved pre-training procedure reaches state-of-the-art results on several natural language processing tasks, such as GLUE, RACE and SQuAD.

2.3 DeBERTa

DeBERTa stands for **D**ecoding-**e**nhanced BERT with disentangled **a**ttention. It improves the BERT and RoBERTa models by applying two novel approaches: one is a disentangled attention mechanism, the other is an enhanced mask decoder.

Comparing to BERT which uses a vector containing the sum of its word embedding and position embedding to represent each word in the input layer, DeBERTa represents each word using two vectors for word embedding and position embedding respectively. Similar to BERT, DeBERTa is pre-trained using masked language modeling (MLM), which is introduced above. Although the disentangled attention method already considers the contents and relative positions of the context words, the absolute positions of these words are not considered. However, these positions are important for prediction. To solve this problem, when the model decodes the masked words based on the aggregated contextual embeddings of word contents and positions, absolute word position embeddings are incorporated right before the softmax layer. A comparison between the decoding layers of BERT and DeBERTa is shown in Figure 3.

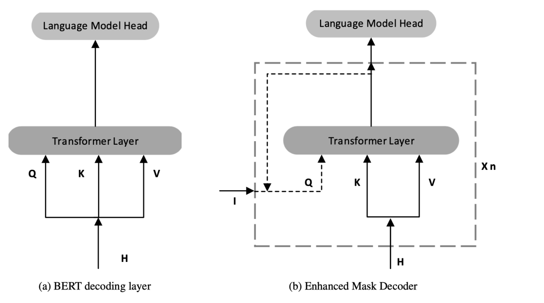


Figure 3: Comparison of the decoding layer (He et al., 2021)

3 Approaches

3.1 Sub-task Training

3.1.1 Overview

In order to improve the baseline performance the author reported in the TRIP paper (Storks et al.,

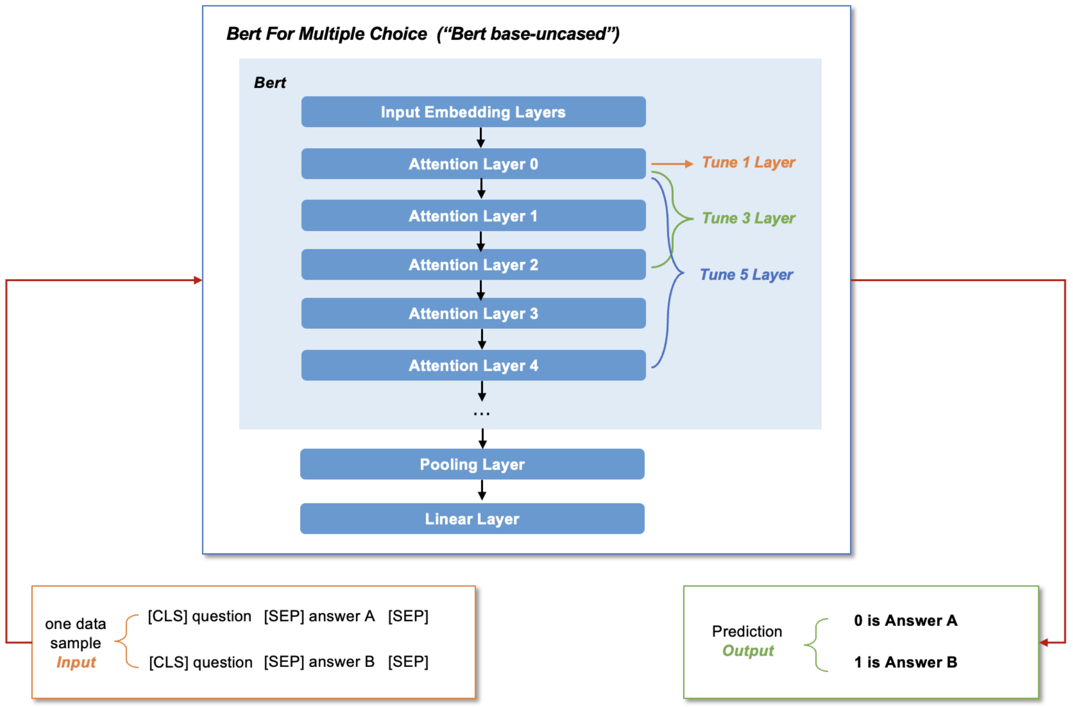


Figure 4: Model Structure

2021), we proposed a sub-task training method, which aims to have a better pre-trained model before training on the TRIP dataset. Figure 1 shows the idea of our work. As discussed in the Bert paper (Devlin et al., 2019) and the previous section, the pre-trained Bert model was trained inside the general domain knowledge, i.e.the dataset is too huge to contain all the topics in the world. Therefore, when this pre-trained model is trained on the TRIP dataset which is inside a specific domain knowledge, it will keep updating its weights during training and make it close to the domain knowledge. For instance, in the figure 1, the longer green line represents the training process in the TRIP paper, and makes the weights of Bert model close to the TRIP's domain knowledge. However, there may be a risk while moving to the specific domain knowledge since it may get stuck at some local minimums, then intuitively the training process will not be representative, which will lead to a poor performance in the end. Therefore, our idea is that, before feeding the original pre-trained Bert model into the TRIP dataset, we may use or define some sub-tasks and tune the Bert model by training on a dataset which shares the same or similar domain knowledge with TRIP. Like the red line in the Figure 1, we will first do a sub-task training on another dataset, and make the weights for the encoding layers of Bert close to the other domain knowledge. Then we will use the new Bert model to do the same training process in the TRIP paper. We believe that in this approach, the weights of the Bert model are closer to the TRIP's domain knowledge than the original ones, which will lead a better performance.
Figure 4 shows the whole process for the sub-task training. Given a sample from another dataset, we will train our model by controlling the freeze of the weights inside the Bert's encoding layers. And the following sub-sections will discuss these in detail.

3.1.2 Dataset

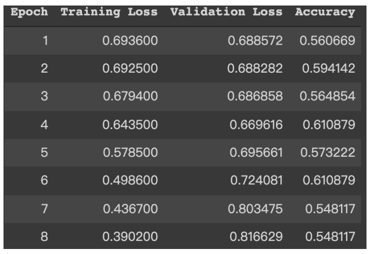
The dataset we used is the Physical Interaction: Question Answering (PIQA) (Bisk et al., 2020), which is a multiple choice dataset which aims to evaluate language representations on their knowledge of physical commonsense. PIQA has 16113 training samples, 1838 validation samples and 3084 samples for testing. Each sample contains a goal(question) q and two possible solutions $sol1$, $sol2$, and the label is just 0 or 1, 0 means the correct answer is $sol1$ and 1 means the correct answer is $sol2$. Since we could not access the labels for the testing samples, so we will only use validation data in the PIQA dataset for evaluating the model.

ModelClass	Iteration Per Seconds(it/s)
Official	**0.91**
Ours	0.74

Table 1: Speed for diffent model class

Name	Value
Learning Rates	$1e-4$
Epochs	8
Batch Size	6
Weight Decay	$1e-2$
GPUs	4

Table 2: Hyperparameters



Epoch	Training Loss	Validation Loss	Accuracy
1	0.693600	0.688572	0.560669
2	0.692500	0.688282	0.594142
3	0.679400	0.686858	0.564854
4	0.643500	0.669616	0.610879
5	0.578500	0.695661	0.573222
6	0.498600	0.724081	0.610879
7	0.436700	0.803475	0.548117
8	0.390200	0.816629	0.548117

Figure 5: Training statistics for tune-3-layer

3.1.3 Model selection and Class selection

Considering the time and the cost in the AWS SageMaker(discussed in Section 7.2), we will use the bert-base-uncased for all the models in this project. Note that TRIP paper used the bert-large-uncased, so for the consistency we will use the performance for bert-base-uncased while reproducing the TRIP result.
Since PIQA is a multiple choice dataset, we use the class AutoModelForMultipleChoice, which contains the Bert model with a linear layer as the classifier. We also try to compare the difference between the provided class with other versions of implementations. Therefore, we re-implement a model which do the same function for the AutoModelForMultipleChoice. However, as shown in Table 1, the official one is the best in terms of the speed.

3.1.4 Preprocessing

We first manually calculate the token numbers for each possible input with the format: [CLS] question [SEP] {solution1 or solution2 } [SEP], and only include the samples that either the length of their token numbers is shorter than a max length threshold of 50. After this filtering operation, we will have 2070 samples for training, and 239 samples for validation.
Next step is to transfer the samples to the required input of the Bert model. We used the tokenizer from the bert-base-uncased model, then parsed the our training samples with the format: tokenizer([goal, goal], [sol1, sol2]). This will generate the input_ids, attention_masks and token_type_ids for the samples. In next step, for each batch of samples of input_ids, attention_masks and token_type_ids, we created a data collator which will pad our sentences in the current batch of samples into the same length.

3.1.5 Optimizer

We used the AdamW (Loshchilov and Hutter, 2019) provided by the HuggingFace (Wolf et al., 2020) as our optimizer. The hyperparameter we used is shown in the next sub-sections. Furthermore, since our aim here is to make the weights of the attention layers in the pre-trained Bert Model closer to the TRIP's domain knowledge, so we need to freeze some layers' parameters in the optimizer, i.e. we only want the weights of the chosen layers to be updated through the sub-task training. As shown in Figure 4, we tried three different experiments that we only select the first one, three and five attention layers as the non-frozen layers. We will also not freeze the layers of the initial three embedding layers (word_embedding, position_embedding and token_type_embedding), and the last classifier layer will also not be frozen.

3.1.6 Hyperparameters

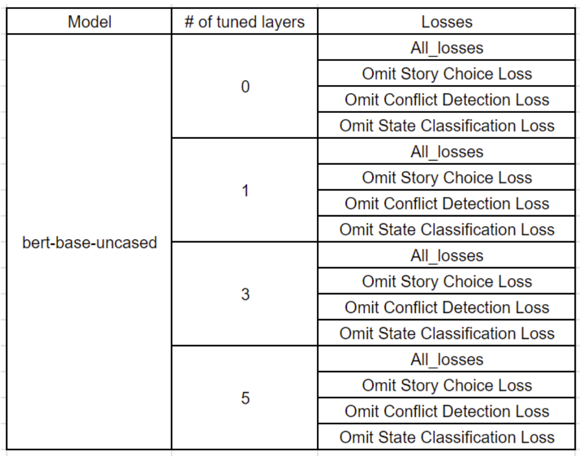
We used the following hyperparameters which are shown in Table 2. We applied a grid search when choosing the hyperparameters, and stopped when we find our model is converged through training. We did not tune the hyperparameters further based on the time and the cost issue which mentioned in Section 7.2.

3.2 Experiments

In order to compare the performance for how many layers that we need to freeze, we tuned three

Model	Accuracy(%)
Bert-large(PIQA's paper)	66.8
Bert-base	52.1
Bert-base-tune-1	50.6
Bert-base-tune-3	61.1
Bert-base-tune-5	56.5

Table 3: Performance for diffent models on PIQA



Model	# of tuned layers	Losses
bert-base-uncased	0	All_losses
		Omit Story Choice Loss
		Omit Conflict Detection Loss
		Omit State Classification Loss
	1	All_losses
		Omit Story Choice Loss
		Omit Conflict Detection Loss
		Omit State Classification Loss
	3	All_losses
		Omit Story Choice Loss
		Omit Conflict Detection Loss
		Omit State Classification Loss
	5	All_losses
		Omit Story Choice Loss
		Omit Conflict Detection Loss
		Omit State Classification Loss

Figure 6: Experiments Set-ups

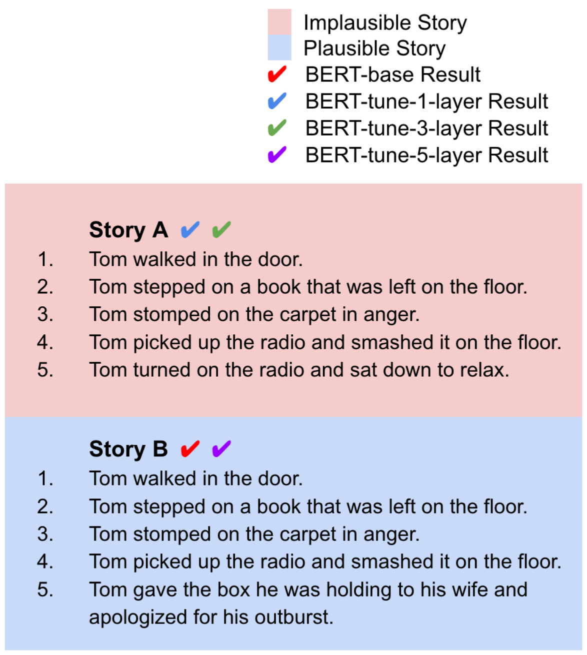
models, and each model will only tune one layer, three layers, and five layers respectively. Figure 5 shows the training statistics for the tune-three-layers model. Now, under the hyperparameter combinations, their performance on the PIQA dataset are shown in Table 3. As reported in the PIQA paper, the result for the Bert-large is roughly 0.66, and in our experiment the performance for Bert-base is about 0.54, so we believe our tuned model is relatively representative, which means that we successfully make the Bert model closer to the TRIP's domain knowledge, as our desired in this section.

4 Experiments

In this final project, we did sixteen experiments to compare the performance on 4 different models, including the BERT-base model and 3 self-tuned model, using 4 different loss functions as (Storks et al., 2021) specified. The comparisons of these results are shown in Figure 6. The 4 loss functions are: L_p for precondition classification, L_f for effect classification, L_c for conflicting sentence detection and L_s for story choice classification.

5 Results & Evaluation

All results are shown in Table 4. In the first section, we can observe that all four models reached low consistency and zero verifiability while maintaining high end task accuracy, reaching around 76% for both BERT-base and BERT-tune-5-layer. BERT-tune-1-layer reaches the lowest accuracy of only 71.2% but the highest consistency of 6.3%. When we omit the story classification loss, as in the second section, both consistency and verifiability increased significantly. BERT-tune-1-layer's



🟥 Implausible Story
🟦 Plausible Story
✔ BERT-base Result
✔ BERT-tune-1-layer Result
✔ BERT-tune-3-layer Result
✔ BERT-tune-5-layer Result

Story A ✔ ✔
1. Tom walked in the door.
2. Tom stepped on a book that was left on the floor.
3. Tom stomped on the carpet in anger.
4. Tom picked up the radio and smashed it on the floor.
5. Tom turned on the radio and sat down to relax.

Story B ✔ ✔
1. Tom walked in the door.
2. Tom stepped on a book that was left on the floor.
3. Tom stomped on the carpet in anger.
4. Tom picked up the radio and smashed it on the floor.
5. Tom gave the box he was holding to his wife and apologized for his outburst.

Figure 7: One example of model inference on a pair of stories. The story highlighted as red is implausible, and the one highlighted in blue is plausible.

consistency was increased by 16.8%, and the verifiability for BERT-tune-5-layer was increased by 5%. At the same time, all model's end task accuracy dropped. The accuracy for BERT-tune-3-layer dropped 5.4%.

When we omit the conflict detection loss in the third section, the end task accuracy for all models dropped significantly. All their accuracies dropped for more than 30%. The original BERT has the highest accuracy of 44.4% and BERT-tune-1-layer only has 42.2%. In addition, all their consistencies and verifiabilities dropped to zero. In the last section, when the state classification loss was omitted, the models' accuracies only dropped a little comparing to the first section, and the verifiabilities remained zero. BERT-tune-5-layer has the highest accuracy of 77.5%, while the original BERT and BERT-tune-1-layer reaches 72.6%. However, model consistencies were improved a lot. The consistency for BERT-tune-1-layer was improved from 2.6% to 10.5%.

Other than that, we also did an inference check. We randomly select one sample, and generate the prediction by all four models. As shown in Figure 7, Story A is implausible since the conflicts in sentence 4 & 5, so which means the ground-truth label should be Story B. by observing the prediction results, original Bert model and Bert-tune-5-layer

Model	Accuracy(%)	Consistency(%)	Verifiability(%)
All Losses			
original-bert	76.1	4.0	0.0
bert-tune-1-layer	74.1	2.6	0.0
bert-tune-3-layer	71.2	6.3	0.0
bert-tune-5-layer	75.8	5.4	0.0
Omit Story Choice Loss			
original-bert	70.4	18.5	4.0
bert-tune-1-layer	68.9	19.4	4.0
bert-tune-3-layer	65.8	17.9	4.0
bert-tune-5-layer	73.0	20.0	5.0
Omit Conflict Detection Loss			
original-bert	44.4	0.0	0.0
bert-tune-1-layer	42.2	0.0	0.0
bert-tune-3-layer	43.0	0.0	0.0
bert-tune-5-layer	42.5	0.0	0.0
Omit State Classification Loss			
original-bert	72.6	9.7	0.0
bert-tune-1-layer	76.9	10.5	0.0
bert-tune-3-layer	72.6	8.5	0.0
bert-tune-5-layer	77.5	8.0	0.0

Table 4: End and tier task metrics for tiered classifiers on the test set on the test set of TRIP trained on varied combinations of loss functions.

model correctly predict the results, while the other two models make the wrong prediction. This result is consistent to the performance when we include all losses together.

6 Discussion

By observing the performance and some inference tests above, we can conclude that our approaches have some improvement when we omit the State Classification Loss, but none of them can achieve the improvements for other cases. Therefore, in this section we will try to analyze the possible reasons.

Our approach had some improvements when we omit the State Classification Loss, but as long as we include this loss our approach achieve a worse performance. we think reason may due to the following reasons. Under the cost issue which is shown in the 7.2, first, when doing the sub-task training, we only find a proper hyperparameter combinations, and we did not do further investigation like random search or grid search with smaller steps around the current hyperparameters to further improve the results. Therefore, our tuned model may not be as informative enough to represent the domain knowledge. Second, we restrict our samples to 50 tokens, which only covers 20% of the original PIQA dataset (Bisk et al., 2020). Therefore, our tuned model may achieve the overfitting, which will lead to a poor result when predicting another dataset. Third, the domain knowledge for PIQA and TRIP are described as similar, so that means even tough our tuned model get closer to the TRIP datasets, but it may also get closer to other domain knowledge, like the blue region in the Figure 1. Therefore, to some extent, in our model, the ratio for other domain knowledge may higher than the TRIP's domain knowledge, which will significantly affect the later performance since it will have a higher weights.

7 Future Directions & Conclusion

7.1 From our approaches

As we disucssed in the previous section, we may modify our model by the following three points of views: First, we will do a grid search to achieve a better hyperparameter combinations, as mentioned in the PIQA leader-board, if we can achieve an accuracy for 0.8, then we believe this tuned model can achieve a better performance for TRIP dataset. Second, we will try to include more samples in the PIQA dataset to make the model more robust. Third, we may first do a investigation on TRIP dataset, then generate a key-words list which may cover the domain knowledge of TRIP. Then we apply this key-words list on the PIQA dataset or other similar datasets, filter out the samples that do not belong to this domain knowledge. Then we do the same things in the final project to get the results.

We believe that the above three approaches may further improve this method, and then improve the baseline performance.

7.2 From other methods

There are some other methods may also improve the baseline performance. For instance, if we can fist pre-train a model which can learn the precondition and result for each action, then use this model as a teacher while training on the TRIP, we may outperform the baselines. In another point of view, we may use the basic EDA method (Wei and Zou, 2019) to augment the training data with random insertion and random deletion, aiming to improve model robustness for word order and stacking. This may also contribute for an improvements for the baselines.

7.3 Conclusion

To sum up, it is a hard on-going research topic, and we will keep investigating it.

8 Division of Work

8.1 Work

We divide this project into several sub-components with following, and each of us leads some of them:

1. Idea brainstorming (Ziyan);
2. Trip paper reproduce (Jim);
3. Presentation (Ziyan);
4. Report preparation (Jim);
5. Sub-task training (Ziyan);
6. Running experiments (Jim);
7. Result Inference (Jim);

For each sub-component, the leader will take roughly 60% of the work, and the other one will finish the work asked by the leader.

8.2 Cost

1. Colab Pro (Jim) $10 for one month;
2. AWS SageMaker (Ziyan) $1000 for total three days;

Jim paid the Colab Pro for one month for reproducing the TRIP result. However, we moved onto the Greatlakes due to the GPU in Colab Pro did not have enough memory.

Ziyan paid the AWS SageMaker for three days for the sub-task training. It contains 4 Tesla V100 GPUs and be able to do the sub-task training with Bert-base model.

Acknowledgements

Since in semester, Ziyan is very busy for Ph.D. application and two research projects, so he did not have time to do the sub-task training on the Great Lakes because the speed and the queue waiting are really time-consuming. So, Ziyan used his own budget to pay a AWS SageMaker to run the task, which saves nearly 60 % of time.

The code for this project can be found on this link: https://github.com/Urheen/595-final

References

Yonatan Bisk, Rowan Zellers, Ronan LeBras, Jianfeng Gao, and Yejin Choi. 2020. PIQA: reasoning about physical commonsense in natural language. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7432–7439. AAAI Press.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015a. A large annotated corpus for learning natural language inference. *CoRR*, abs/1508.05326.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015b. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 632–642. The Association for Computational Linguistics.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Andrew M. Dai and Quoc V. Le. 2015. Semi-supervised sequence learning. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 3079–3087.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing, IWP@IJCNLP 2005, Jeju Island, Korea, October 2005, 2005*. Asian Federation of Natural Language Processing.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R. Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers)*, pages 107–112. Association for Computational Linguistics.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: decoding-enhanced bert with disentangled attention. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

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. *CoRR*, abs/1907.11692.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 3428–3448. Association for Computational Linguistics.

Timothy Niven and Hung-Yu Kao. 2019. Probing neural network comprehension of natural language arguments. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4658–4664. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018a. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 2227–2237. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018b. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 2227–2237. Association for Computational Linguistics.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning, CoNLL 2003, Held in cooperation with HLT-NAACL 2003, Edmonton, Canada, May 31 - June 1, 2003*, pages 142–147. ACL.

Roy Schwartz, Maarten Sap, Ioannis Konstas, Leila Zilles, Yejin Choi, and Noah A. Smith. 2017. The effect of different writing tasks on linguistic style: A case study of the ROC story cloze task. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017), Vancouver, Canada, August 3-4, 2017*, pages 15–25. Association for Computational Linguistics.

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

Jason W. Wei and Kai Zou. 2019. EDA: easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the*

2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019, pages 6381–6387. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

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. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, EMNLP 2020 - Demos, Online, November 16-20, 2020*, pages 38–45. Association for Computational Linguistics.