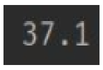
Final Project NLP

318170917 322995358

The algorithm that was used to train the model on the training set is T5-Text-to-Text-Transfer-Transformer. Which was presented in Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer by Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J. Liu on 2020. This model was used for both val and comp predictions. We took the pretrained model and fine-tuned it on our task, for predicting the val and comp document. We found that using ada-factor optimization helped the performance of the model.

The parameters that were chosen were:

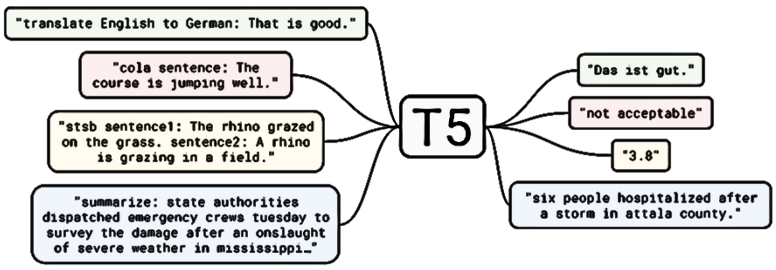
Batch size = 2
learning_rate = 0.0005
num_train_epochs = 10
generation_max_length = 210
gradient_accumulation_steps = 16

The blue score on the val file is: 

```
37.1
```

For this part we initially tried to implement the transformers ourselves, but it was too slow and wasn't good enough, so we used T5 instead.

We tested the T5-BASE, T5-SMALL. The base version performed the best.


"translate English to German: That is good."
"cola sentence: The course is jumping well."
"stsb sentence1: The rhino grazed on the grass. sentence2: A rhino is grazing in a field."
"summarize: state authorities dispatched emergency crews tuesday to survey the damage after an onslaught of severe weather in mississippi…"
T5
"Das ist gut."
"not acceptable"
"3.8"
"six people hospitalized after a storm in attala county."

In order to improve the blue score, we thought that because blue biggest weakness is that it can't understand that 2 words are synonyms. Therefore, getting the roots and modifiers exact words, can help us increase the blue score if we'll replace the predicted roots and modifiers with the real one. This way, we can ensure that the those will get a positive blue score and won’t be considered wrong while having a very similar meaning. So in order to implement it, we tried finding the best method to switch the given and predicted roots and modifiers. We were inspired by a lot of known concepts as we'll elaborate now.

First, Bert-score: that as known, refers to that exact same problem. Bert-score uses cosine similarity between the embedded tokens of the word in the sentences in order to calculate its score, based on the idea that similar words would get a bigger cosine similarity after pretrained embedding. We also read about other methods of similarity score other then cosine similarity for text evaluation and made sure it is the best method to use. We used those ideas to find the similarity between the given modifiers\roots and predicted modifiers\roots.

After finding how similar each pair of predicted and given word is, we faced the problem of how to pick the replacement strategy. As expected, we want to take the max similar for each one, but what about the cases where 2 words got the same argmax? inspired by Active learning's uncertainty method of smallest margin calculation, where we can give a certainty score by calculating how much the model is confident in its prediction we implemented the method and understood this solution is not good enough because each replacement may cause a new conflict over and over again. When, we understood we're facing a "Perfect Match Problem" where each given modifiers\roots could be replaced with one predicted modifiers\roots and via versa and we want the best combination. We explored algorithms that solves this known problem. After trying those ideas we found out that in our case, it is better to able replacement 2 predicted words with the same given one because it increases the changes to get a positive blue score (match) on at list on of them, rather of taking the risk of have a mistake in both of them and have zero matches at the n-gram comprehension in the blue calculation.

In addition, we tried giving the roots and modifiers to the model, from the unlabeled version for the val and comp and using spacy for the training set which we did not have it's roots and modifiers.

The best blue scored version of all this tries was the most basic one, of T5 without the additions so that is what we submitted eventually.

The expected score on the comp file is around 35 because this model was picked according to the val file performance and therefore might not be fitted as best as it was for this file.

We worked together throughout the project.

Bibliography

T5: Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer by Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J. Liu.

Similarity: https://newscatcherapi.com/blog/ultimate-guide-to-text-similarity-with-python

Active learning: https://towardsdatascience.com/introduction-to-active-learning-117e0740d7cc