GLUE Benchmark:

Individual Project Report

NLP for Data Science

DATS 6312

Mariko McDougall

12-09-2021

Individual Final Report

1. Introduction

The General Language Understanding Evaluation benchmark (GLUE) is a compilation of natural language datasets and tasks designed with the goal of testing models on a variety of different language challenges.

Historically, many NLP models were trained and tested on very specific datasets. This method often produced well performing models when run on similar data, but had very poor performance when applied to corpuses that varied from their original training data. Additionally, these models could only be used to perform the specific task they were built for, with very little cross-utility to different tasks, even when using the same data.

With the advent of transfer learning, it became possible to create a model which could be used for multiple tasks, across a variety of input datasets. Through this, models that have a deeper understanding of language can be created, with broad cross-utility and applicability in a variety of natural language contexts.

The 9 tasks in the GLUE dataset represent a diverse set of challenges, from grammatical parsing to context-reliant pronoun identification, from an array of text contexts including news reports, public forums and wikipedia pages.

To create a high-performing model capable of understanding and generating natural language, we integrate three state-of-the-art models together with an ensemble learning model to capture information from the three disparate models on each task. With this method, we capitalize on the unique strengths of each individual model, while mitigating most weaknesses.

My specific portion of the project entailed researching and training DeBERTa, coding and implementing the ensemble learning, writing sections of the group report and presenting our findings.

2. Description of your individual work

DeBERTa

The model that I researched and trained for the project was DeBERTa, a BERT-based model produced by Microsoft Dynamics and Microsoft Research in October 2021. DeBERTa is an abbreviation of **D**ecoding-**E**nhanced BERT with disentangled **A**ttention, and as the name implies there are two major innovations in this model compared to previous similar innovations: an Enhanced Decoding mask and Disentangled attention. Both of these innovations stem from an enhanced tracking of the position of words within a given sentence.

Ensemble Modeling

To integrate the predictions from all models and select the most probable answer, we used ensemble learning. Ensemble learning is a meta approach to modeling that leverages the power of multiple predictive models to make predictions that are more accurate than any model in the ensemble. Whereas many approaches would compare the outputs of each model and simply take the best result, ensemble learning takes the class predictions from all models and uses all of them as an input. This integration, rather than exclusion of less accurate models allows the ensemble learner to glean additional information from the weaker models as well, rather than discarding it.

In this project, we use both Random Forests and Gradient Boosting ensemble methods, and compare and contrast their results. Gradient boosting and random forests are very similar models, both are decision tree-based and mainly vary with how the output of the trees are combined. Where random forests build all trees simultaneously and independently (horizontally), gradient boosting builds them sequentially, with each tree being built in a chain. Both methods have their own strengths and weaknesses, and so we thought to run both, and compare the outputs.

3. Portion of the work that you did on the project in detail

Training DeBERTa

Using the training code that Stefani sourced and modified, I trained DeBERTa on all 9 GLUE tasks. For each task, I initially started with a batch size of 10, and a maximum model length of 512, however for most tasks this resulted in a crash due to excessive data being sent to the CUDA. These two parameters were reduced incrementally for each task until training could be completed without error.

Coding and Training Ensemble Models

Once all three models had been trained, with Stefani training XLNet and Arathi training ELECTRA, all of the trained models' weights were saved to our github repo. From there, I loaded all of the saved models into my ec2 instance, and performed the ensemble modeling.

The code I wrote for the ensemble method was based on a previous group project that I completed for Cloud Computing, DATS 6450.

In the modified version of this code, for each task in the GLUE dataset, each trained transformer model was loaded from the saved weights. The GLUE validation dataset was then tokenized for each specific model, then the class/continuous variables were predicted using the loaded model. The accuracy for these predictions were also calculated.

The predictions for all three models were concatenated into a single dataframe. This was then split into training and testing sets, and the training set and corresponding labels were ingested into the ensemble model. The models were trained on the training set, and tested on the testing set, and the accuracy scores were calculated and saved.

Presenting our Findings

For the final project report, each group member was responsible for researching and reporting on their respective model. My personal section was the DeBERTa model, as well as the majority of the ensemble model. Additionally, I contributed to the content of other sections of the paper, with exception to the sections on ELECTRA and XLNET.

Additionally, I also contributed to the creation of the powerpoint presentation, and I anticipate presenting the section on DeBERTa and the ensemble methods, as well as responding to questions.

4. Results

DeBERTa

DeBERTa			
Corpus	**Metric**	**Baseline**	**Tuned**
Single-Sentence Tasks			
CoLA	*Matthew's correlation*	0.000	0.621
SST-2	*Accuracy*	0.491	0.935
Similarity and Paraphrase Tasks			
MRPC	*Accuracy and F1*	0.316 f1: 0.000,	0.865 f1: 0.903
QQP	*Accuracy and F1*	0.632 f1: 0.000	0.906 f1: 0.875
STS-B	*Pearson and Spearman*	P:0.055 S:0.067	P:0.868 S:0.868
Inference Tasks			
MNLI	*Accuracy*	0.354	0.875
MNLI-MM	*Accuracy*	0.330	0.872
QNLI	*Accuracy*	0.495	0.915
RTE	*Accuracy*	0.527	0.668
WNLI	*Accuracy*	0.563	0.437

In nearly all tasks, there was a strong improvement over baseline when the DeBERTa was trained on the tasks prior to evaluating on the validation set. Interestingly, this is not true for all tasks; WNLI performs slightly better on the baseline untrained model compared to the trained model. This is an interesting quirk of the WNLI dataset which is known to be mildly adversarial: several training examples have very similar, but opposite, examples in the testing and validation set. This often leads to lower scores on the WNLI dataset, to the point where it is specifically addressed in the GLUE dataset's FAQ.

Ensemble Learning

		Transformers			**Ensemble Methods**	
Corpus	**Metric**	**Electra**	**XLNet**	**Deberta**	**Random Forest**	**Grad Boost**
Single-Sentence Tasks						
CoLA	*Matthew's correlation*	0.607	0.400	0.621	**0.688**	0.657
SST-2	*Accuracy*	0.917	0.940	0.935	**0.958**	0.954
Similarity and Paraphrase Tasks						
MRPC	*Accuracy and F1*	0.882 f1:0.915	**0.892** **f1:0.923**	0.865 f1:0.903	0.878 f1:0.912	0.854 f1:0.893
QQP	*Accuracy and F1*	0.900 f1:0.866	0.874 f1:0.833	0.906 f1:0.875	0.909 f1:0.878	**0.910** **f1:0.879**
STS-B	*Pearson and Spearman*	P:0.873 S:0.872	P:0.893 S:0.889	P:0.868 S:0.868	P:0.894 S:0.889 *Note: Regressor	**P:0.897** **S:0.892** ***Note: Regressor**
Inference Tasks						
MNLI	*Accuracy*	0.817	0.857	0.875	**0.877**	**0.877**
MNLI-MM	*Accuracy*	0.821	0.858	0.872	0.875	**0.877**
QNLI	*Accuracy*	0.889	0.878	0.915	0.916	**0.924**
RTE	*Accuracy*	0.682	0.740	0.668	**0.726**	0.702
WNLI	*Accuracy*	0.465	0.563	0.437	0.500	**0.636**

The ensemble methods outperform the individual transformers in 8/10 GLUE tasks, with XLNet outperforming them in the MRPC and RTE tasks.

When comparing the gradient Boost and the Random Forest and Gradient Boosting we see similar results from each model, often with Gradient Boosting slightly outperforming the Random Forest. While Gradient Boost performed slightly better overall, it is interesting to note that Random forest performed better in all single-sentence tasks, and might be better suited for applications that are similar in structure.

5. Summary and Conclusions

Overall, our ensemble model performed much better than we had initially hoped. While we initially anticipated scores that were roughly on par with the best model for each category, we did not fully anticipate that it would perform better in most cases than any of the state-of the art models. Even the relatively humble random forest performed better than the individual transformer models when given their predictions as inputs.

That being said, the utility of ensemble methods for this purpose are limited. While they may be able to predict a binary class, such as if a sentence is grammatically correct or not, or grade how similar two sentences are, it is unlikely that ensemble methods could generate text like a transformer could. Thus, while an ensemble model might be good for text ingestion and parsing, it may not be useful for many other NLP tasks, such as question response or summarization.

Improvements could be made to our development pipeline in several ways.

One potential way to increase performance would be to include additional transformer models, as long as the new models are distinct from those used in the current ensemble. Another BERT-like model, for instance, may not increase performance by a significant margin, due to its similarity to DeBERTa. However, a distinct model, such as GPT-2 may improve performance due to the increase in dissimilar processing.

One key limitation that we faced in the project was a short training time duration necessitated by the time-out duration of the GWU VPN. For security reasons, the ec2 instances supplied by the university could only be connected to using a GWU IP address, either from a local network or using a VPN. The VPN, unfortunately, had a maximum duration of 12 hours after which the connection would be terminated and any running process would likewise terminate. Thus, all training intervals needed to be completed in under 12 hours, which was surprisingly problematic when performing fine-tuning of such large models.

6. Percentage of Code from the Internet

Script	Internet Code	Modified Lines	Original Lines	Calculated Value
Single Model Training	32	12	13	44.4%
Evaluate Loop	26	4	10	61.1%
Train and Test Ensemble	51	18	76	26.0%
Total	**109**	**34**	**99**	**36.1%**

7. References

GLUE Explained: Understanding BERT Through Benchmarks
Dataset description sheet
GLUE: A MULTI-TASK BENCHMARK AND ANALYSIS PLATFORM FOR NATURAL LANGUAGE UNDERSTANDING
DeBERTa Original Paper
Hugging Face Transformers Examples
Data Science Central