Sesame Street Ensemble: A Mixture of DistilBERT Experts

Stanford CS224N Default Project

Tyler Consigny
Department of Symbolic Systems
Stanford University
tconsign@stanford.edu

Abstract

In this project, I attempt to finetune a pre-trained DistilBERT model to better handle an out of domain QA task. As there are only a few training examples from these outside domains, I had to utilize various techniques to create more robust performance: 1) implemented a mixture of local experts architecture and 2) finetuned a number of hyperparameters to perform best over this few shot learning task. Specifically, a separate DistilBERT model was finetuned on each of the in-domain datasets to act as an expert. The finetuning approaches focused on reinitializing a variable amount of final transformer blocks and training for a longer period. These two approaches were then synthesized to produce the final model.

1 Introduction

The task of question and answering is one that points to many of the most intriguing questions at the heart of artificial intelligence. To perform well on these tasks, a model needs to 'understand' the text in some manner. With the recent development of large-scale pre-trained models such as BERT, performance on these tasks has received a positive jolt. However, it has been observed that performance does not generalize well beyond the training distribution [1]. This marks a drastic bifurcation between human and model abilities. Humans are able to quickly apply knowledge to newly seen circumstances. This level of robustness has not been achieved in our NLP models as of yet. The approach in this paper attempts to address this problem in a small way. By implementing and synthesizing the approaches proposed in other papers [2,3,4], I attempted to make progress in this goal of robustness of QA models.

In this project, a model is developed that is given a question and paragraph pair and returns a start and end location that represents the span of the answer in the provided paragraph. This format is based upon that used in the SQuAD 2.0 dataset [5]. The goal is to have performance transfer to datasets that were not used to train the model (or only with a few hundred training examples). If this transfer of performance is achieved, it demonstrates that the model is better suited for the real world where examples often come from distributions that do not strictly resemble those previously trained on. There have been various techniques proposed to address this task such as meta-learning, domain adversarial training, and mixture-of-experts.

I chose to utilize two approaches that I thought would build on each other: mixture-of-experts [3] and few sample fine-tuning techniques [2]. Specifically, a pre-trained DistilBERT [6] model was finetuned on each of the in-domain datasets and the outputs of each 'expert' were weighted by a gating network to produce a final output. This was combined with finetuning processes like training for longer and reinitializing a certain number of final transformer blocks. Ultimately, these two approaches were combined to produce the final proposed model. However, I found that both of these approaches actually had a detrimental effect on performance. The possible cause of this will be explored later in the paper.

2 Related Work

2.1 DistilBERT

As transfer learning from large-scale pre-trained models becomes more pervasive in NLP, the restrictions of storage and compute become more prevalent as well. The creation of DistilBERT attempts to mitigate some of these issues [6]. This smaller pre-trained general-purpose language model provides a well-performing baseline that can then be fine-tuned to perform on other downstream tasks (sentiment classification, SQuAD, etc). While pre-training on the same corpus as BERT, the authors leverage various techniques to condense the size of the model and increase the inference speed while retaining much of the performance. These techniques include knowledge distillation, a novel triple loss function, and various architecture, initialization, and hyperparamter choices.

In this project, a pre-trained DistilBERT model is finetuned on the in-domian datasets to provide a baseline model for the QA task. The reduced size of DistilBERT made finetuning and storage more feasible (especially while using multiple expert models).

2.2 Mixture-of-Experts

The mixture-of-experts approach is an ensemble method that attempts to divide up the input space by having separate expert models for different areas of the input regions. Discussions of this method can be found in Hinton et. al. [3] and Masoudnia et. al [4]. The task here is to learn two sets of parameters: those of the experts models and those of the gating function that weights the outputs of each expert. In this paper, each expert is a pre-trained DistilBERT model that is finetuned on one of the in-domain datasets. The gating function used is a multi-layer perceptron.

2.3 Few Sample Finetuning

Zhang et. al. [2] identified three sub-optimal hyperparameter and architecture choices within common practices involving BERT. Two of these are relevant in this paper: training time and reinitialization of a variable amount of final transformer blocks. These observations were combined with the mixture-of-experts procedure form above.

3 Approach

3.1 Baseline

Following the approach found at [7], I finetuned a pre-trained DistilBERT model to act as baseline model. The implementation used also performs tasks such as chunking the question-paragraph pair and caching the tokenized representations of the data. The baseline was trained on examples from all of the in-domain datasets

3.2 Mixture-of-Experts

The main alteration to the architecture of the model took place through the implementation of mixture of local experts as described in Hinton et. al.[3]. Specifically, I trained 3 separate DistilBERT models to act as 'experts' for each of the in-domain datasets. Alongside these 'expert' models, I trained a multi-layer perceptron (MLP) to act as the gating function to divide up responsibility to each expert. Through a softmax, the gating function output a probability to each expert model so that,

$$\sum_{k=1}^{3} g_k(x) = 1$$

where x is the input vector and k is an expert model.

These probabilities, $g_k(x)$, are then used to 'weight' the respective probability distributions outputted by each expert model. Thus, we will have:

$$p(y|x) = \sum_i g_i(x) f_i(x)$$

where $f_i(x)$ is the conditional distribution output by the i-th expert.

In this specific implementation, this process takes the form of weighting the start and end location logits that are outputted by each expert for a given example. The weighted start and end logits of each expert are then summed together to produce the final outputted logits. The cross-entropy loss function

$$-logp_{start}(i) - logp_{end}(j)$$

where i and j are the gold start and end locations

remained. It is now simply computed on the logits outputted from the mixture-of-experts process. The backpropogated error was used to alter the parameters of the gating function to produce a network that optimized the weight assigned to each expert for all examples in the training set. In regards to the expert models, these were trained one by one on their respective in-domain dataset. They were then retrieved and frozen during the training of the gating network. The architecture of this approach is shown below.

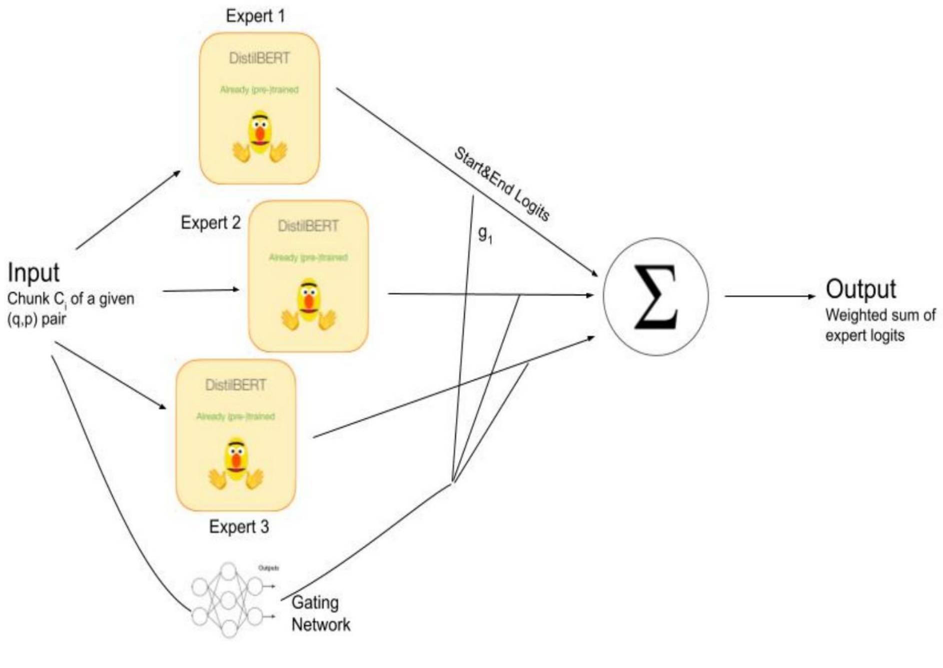


Figure 1: The model architecture used for the implementation of mixture-of-experts

Overall, this approach seemed a good fit for the robust QA task because the dataset contains several different regimes which have different relationships between input and output. The goal was that this ensemble of local expert models would be able to better capture these different relationships. In doing this, it was hoped that a more robust system would be achieved that could better describe a novel data distribution by selectively combining the distributions encapsulated by the three experts.

3.3 Few Sample Finetuning

Inspired by Zhang et. al. [3], the next step I took was to finetune the hyperparameters and architecture to perform well on few sample datasets. The authors focus on three main choices: the gradient debiasing correction within the ADAM optimizer, the random reinitialization of layers before finetuning, and the amount of training time. As the baseline already utilized a dibiasing correction, I focused on the latter two. First off, I experimented with reintizializing a variable number of transformer layers before finetuning. Zhang et. al. found that randomly reinitializing 1-6 layers before finetuning leads to faster training and better performance. The type of reinitialization was dependent upon the type of layer at hand (i.e. Linear layer reinitialized with a normal distribution, Layer Norm layer reinitialized to ones, etc). Second, it was also found that the standard three epochs used for BERT fine-tuning is not optimal. Increasing the training time lead to increased performance on the majority of benchmarks. Presumably, the model was still under-fitting the training data. Following suit, this approach was attempted and various trails with increased number of epochs were undergone.

Finally, these two methods were synthesized into a single approach. A number of the final transformer blocks of each expert were reinitialized and the finetuning time for the experts was increased. The hope was that the benefits of both methods would come through in the final model.

4 Experiments

4.1 Data

The data used will be drawn from six datasets: three in-domain datasets (such as SQuAD [5] and NewQA [8]) and three out-of-domain datasets (such as DuoRC [9] and RACE [10]). The in-domain datasets contain 50,000 training examples whereas the out-of-domain contain only 127 training examples. This disparity is why the robustness of the model is essential. The model needs to use these scarce examples to few-shot learn and enable the ability to perform in these rarely seen domains. All the training sets are in the format of triples: context, question, and answer. During training and evaluation, the conventional input format of [CLS]q[SEP]p[SEP] is used.

4.2 Evaluation method

Two metrics were used to evaluate the performance on the QA task: F1 and EM. When evaluating on the validation and test sets, the maximum F1 and EM scores that were achieved are reported (based on the various answers that human annotators gave). The EM and F1 scores are then averaged across the entire evaluation or test dataset to obtain the final scores.

4.3 Experimental details

To train the baseline, we used the given model configuration and hyperparameters in [7].

Multiple approaches were attempted to improve on this baseline. The number of training epochs during expert finetuning was varied in the range of 3-5. The number of final transformer blocks that were reinitialized varied from 2-4.

Because of memory and computing limitations, the expert models were finetuned individually and then loaded when training the gating network. The gating network was an MLP that consisted of three fully connected layers with a hidden layer size of 768. The ReLU activation function was used in the hidden layers. The output was through a softmax that gave the weights for each expert. During a specific run, the use mixture-of-experts, reinitialization, and number of epochs were all variable. All or none could be utilized.

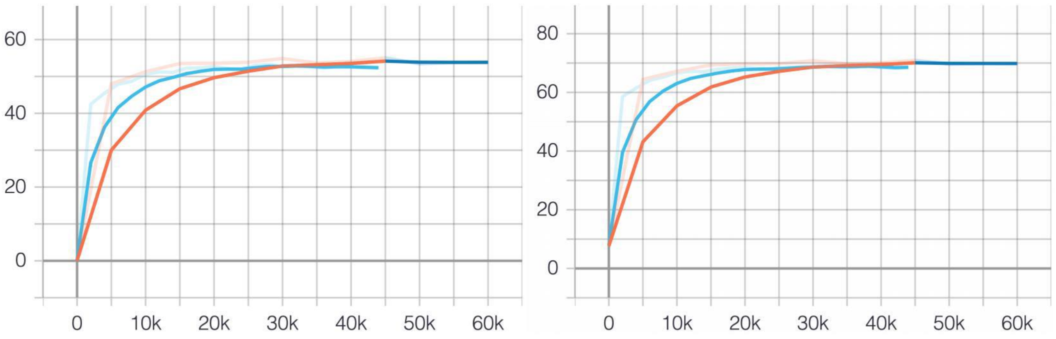
4.4 Results

The tables below describe the results of the different approaches on the oo-domain (out-of-domain) val and test sets as they compare to the baseline. The quantitative metrics used are EM and F1.

As the scores show, the methods attempted in this paper do not show an improvement upon the baseline performance. On the oo-domain dev set, the mixture-of-experts implementation performs very close to the baseline but has a slight negative effect on EM and F1. Training for 4 epochs did not show any improvement over training for the standard 3. Reinitialization of the final 4 transformer blocks also showed a decline in performance. As seen in Figure 2, the EM and F1 scores on the training set increased more quickly for the MoE + Reinit-4 model than the baseline. However, this model plateaued quite early and ended up with no performance improvement over the baseline model.

Results (Validation)		
Model	EM	F1
Baseline	33.246	48.432
MoE	31.952	47.129
4 epochs	33.246	48.432
Reinit-4 Layers	31.475	47.055
MoE + Reinit-4	33.034	47.248

Table 1: EM and F1 results on the oo-domain dev set



(a) EM scores on the training set

(b) F1 scores on the training set

Figure 2: EM and F1 scores on the training set.

Orange = Baseline, Blue = Baseline + 4 epochs, Teal = MoE + Reinit-4

Similar to the performance on the oo-domain dev set, my implementation of the methods in this paper do not show an improvement in performance on the oo-domain test set. The synthesized model that was proposed here, MoE + Reinit-4, performs close to the baseline but with a small performance decrements.

Results (Test)		
Model	EM	F1
Baseline	40.275	59.187
MoE	39.768	58.768
MoE + Reinit-4	40.013	58.781

Table 2: EM and F1 results on the oo-domain test set

This decrease in performance was unexpected but there are possible reasons that may be pointed to for an explanation. For instance, it may be that the gating network for the mixture-of-experts model did not sufficiently learn the correct mapping from input to expert weights. In future work, it may be useful to explore alternate architectures and hyperparameters for this gating function. This will be further explored in the Analysis section below. It may also be that the expert models did not partition into distinct local input regions. If this did not happen, many of the benefits of localization of experts would not be felt in the performance. Alternatively, the lost generalization benefits of training a single DistilBERT model on all three in-domain datasets may not have been repaid through the localization of expert models.

As the results in Zhang et. al. [2] showed an increased performance from reinitializing transformer blocks in BERT, it was surprising that these improvements were not also seen here. The BERT and DistilBERT models were similar enough that I assumed these results would transfer. One possible reason may be that the generalized knowledge from pre-training that was lost during reinitialization had a more damaging effect than the domain knowledge gained during finetuning. Attempting different reinitialization techniques and/or training on larger training sets may be enlightening.

5 Analysis

To better understand why the final model (MoE + Reinit-4) does not succeed, it is insightful to look at specific examples. Take the example found at id a47ddebcc3b74dafa5251ef1137b5160,

"Where did Cassidy find strength after the tragedy?"

The ground truth answer is, "in the words of a Harry Potter film". The predicted answer from the model is, "words of a Harry Potter film". This is close but not the ideal answer. Analyzing the response of each expert helps to understand why this occurred.

Expert Outputs		
Expert	Predicted Start	Gating Weight
SQuAD Expert	"in"	0.15
NewQA Expert	"in"	0.21
Natural Questions Expert	"words"	0.64

Table 3: The word with the highest probability for each expert's outputted start logits and the weight assigned to each expert by the gating network. All networks has the correct end location prediction.

From this, we see that two of the expert models actually predicted the correct answer span. However, these two correct predictions were outweighed by the model with the incorrect start prediction. This lends support to the theory that the gating network has not correctly learned a useful mapping from input to expert weights. This issue may be compounded by the possibility that the input spaces of each of the in-domain datasets are not sufficiently distinct. If this is the case, the experts would not be partitioned to distinct areas of the input space when finetuned.

6 Conclusion

In this project, I attempted to utilize various techniques to create a model that performed robustly on a QA task with out-of-domain examples. Specifically, I implemented a version of mixture-of-experts and applied various finetuning procedures to these expert models. These attempts did not produce performance improvements. Possible explanations for this negative outcome are proposed.

In future work, I see various paths towards improving the approach in this paper. One option would be to use data augmentation methods on the few training examples given from the out-of-domain datasets. Using this increased quantity of examples, an attempt could made to train experts on these datasets alongside the experts for the in-domain datasets. To improve the MoE implementation, one could create a more focused method for grouping in-domain examples and experiment with new gating network architectures.

References

[1] R Thomas McCoy, Ellie Pavlick, and Tal Linzen. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Association for Computational Linguistics* (ACL), 2019.

[2] Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q Weinberger, and Yoav Artzi. Revisiting few-sample bert fine-tuning. *arXiv preprint arXiv:2006.05987*, 2020.

[3] Geoffrey E. Hinton, Michael I. Jordan, S. Nowlan, and R. Jacobs. Adaptive mixtures of local experts. *Neural Computation*, 3:79–87, 1991.

[4] Masoudnia, Saeed Ebrahimpour, Reza. (2014). Mixture of experts: A literature survey. *Artificial Intelligence Review*. 42. 10.1007/s10462-012-9338-y.

[5] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100, 000+ questions for machine comprehension of text. *CoRR*, abs/1606.05250, 2016.

[6] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

[7] `https://web.stanford.edu/class/cs224n/project/default-final-project-handout-robustqa-track.pdf`

[8] Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. Newsqa: A machine comprehension dataset. *ACL 2017*, page 191, 2017.

[9] Amrita Saha, Rahul Aralikatte, Mitesh M. Khapra, and Karthik Sankaranarayanan. DuoRC: Towards Complex Language Understanding with Paraphrased Reading Comprehension. In *ACL*, 2018.

[10] Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. RACE: Large-scale reading comprehension dataset from examinations. In *EMNLP*, 2017.