Robust QA with Model Agnostic Meta Learning

Stanford CS224N Default Project

Raghav Samavedam
Department of Computer Science
Stanford University
raghavsa@stanford.edu

Abstract

An ensemble of MAML-type algorithms are finetuned to improve the accuracy of DistilBERT on low-resource data. A weighted expert model is found to have superior test set performance and analysis confirms that MAML-based training algorithms tend to be sensitive to a choice of hyperparameters.

1 Key Information to include

- External collaborators (if you have any): N/A
- Mentor (custom project only): N/A
- Sharing project: N/A

2 Introduction

Deep learning methods have achieved strong performance in a variety of application domains in part due to their ability to create rich representations of their inputs. In many cases, these representations can be fine-tuned to adapt to tasks similar in nature to the original task the representation was trained for. Examples include representations obtained by ResNet, a deep residual neural network trained on the ImageNet dataset for predicting across 20,000 image categories [1], Word2vec, a procedure that embeds a large corpus of words such that semantic relationships are preserved, Node2vec, which embeds nodes in a graph in a way that captures neighborhood similarity and is often used as the initial feature vector to a graph neural network [2], and UniRep, which embeds DNA sequences in a way that captures phylogenetic similarity of different organisms [3].

Beyond word2vec, there is significant interest in learning rich representations of full sentences or paragraphs. Such applications have included sequence-to-sequence models like neural machine translation (NMT) systems, and, for a long time, state-of-the-art performance was achieved by recurrent neural networks (RNNs) and long short-term memory models (LSTMs) [4]. Recently though, with the advent of attention for processing sequences, bidirectional transformers have outperformed recurrent models on a variety of different sequence-to-sequence and sequence representation tasks [5].

One such model, called BERT (Bidirectional Encoder Representations from Transformers), has achieved current state-of-the-art on metrics such as GLUE score, MultiNLI accuracy, and F1 score on the SQuAD v1.1 and v2.0 question answering datasets [6]. BERT is pre-trained using unlabeled natural language data via a masked language model (MLM) method, it is then fine-tuned for next-sentence prediction and question answering tasks (see Figure 1).

Successfully adapting BERT to low-reource natural language domains remains an open problem. Previous approaches have included using multitask [7] and meta-learning [8] fine-tuning procedures. Using a variant of the Model Agnostic Meta Learning (MAML) algorithm from [9], the authors of [8] were able to show that meta learning procedures had a slight advantage in low-resource domain adaptation than multitask models. However the authors of [8] experimented with only a few task

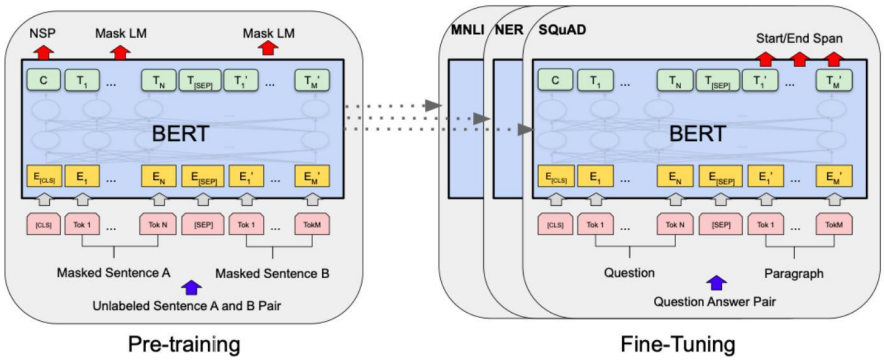

Figure 1: Unsupervised pre-training plus supervised fine-tuning of BERT, image from [6]

distributions $p(T)$ for the MAML algorithm (see Section 4), and while the results did show an improvement over [7], performance for certain task distributions on specific tasks was somewhat counterintuitive.

In this paper, suggestions from a recent paper [10] in the International Conference on Learning Representations (ICLR) are implemented to stabilize training of a MAML-type algorithm on a pre-trained variant of BERT called DIstilBERT. Several task distributions and other MAML-specific hyperparameter initializations are implemented and analyzed and a classifier is trained to predict out-of-domain dataset type to better leverage task-specific fine-tuning.

3 Related Work

3.1 Multi-Task Deep Neural Network (MT-DNN)

Multi-Task models refer to a class of models where tasks share a relatively deep "bulk" set of layers that creates a generic representation of the input. Task-specific "heads," normally much smaller than the shared bulk layer, serve to further process the generic representation to be more tailored to their respective head. In many different application areas, it has been found that training tasks jointly in this fashion can help improve generalization and improve performance by leveraging similarity between tasks [11]. The authors of [7] create a Multi-Task Deep Neural Network (MT-DNN) that uses BERT as a shared text encoder, and task-specifc layers to train for four natural language understanding (NLU) tasks. Some of these tasks are *low–resource*, meaning that there are relatively few training examples associated with them. On these tasks, MT-DNN performs reasonably well - indicating that the representation created by MT-DNN's now-augmented, shared BERT layer is a good initialization that can generalize to data it has not seen a lot of.

3.2 Meta-Learning Algorithms

While MT-DNN performs fairly well on low-resource tasks, the authors of [8] point out that Multi-Task models learn representations that favor high-resource tasks over low-resource ones. This was first noticed in [12], and is visualized in Figure 2.

The authors vary $p(T)$, the task distribution for the MAML algorithm (see Section 4), to be Uniform, Probability Proportional to Dataset Size, and Mixed. The latter indicating that the tasks are initially selected Uniformly, but that over time a certain "target task" is focused on more. The authors use four high-resource tasks and four low-resource tasks and found that in general the meta-learning algorithms outperformed MT-DNN. Probability Proportional to Dataset Size was apparently the most successful task distribution in terms of performance. The authors also tested the efficacy of the transfer learning potential of this MAML-based method and found that it outperformed the MT-DNN baseline when used to pre-train a model for the SciTail dataset.

While this paper is very interesting in that shows that MAML-based approaches can work in low-resource natural language understanding domains, the results of the model for the three task distributions are very similar and do not necessarily indicate that Probability Proportional to Dataset Size

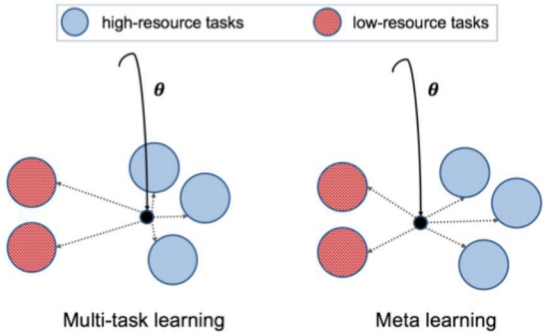

Figure 2: Multi-Task learning may favor high resource tasks over low resource ones, from [8]

is the best option. While performance was better than MT-DNN, it was not by much. Given that the authors indicated that a multitask model learns a learning representation that is biased towards high-resource tasks / domains, I feel that using probability proportional to dataset size would have a similar effect. This would indicate that the most successful MAML-based approach would have similar characteristics to MT-DNN, thus contradicting the supposed representation benefit MAML provides. It is also noteworthy that when using a mixed distribution, which iteratively focuses on a given task, the authors of [8] indicate that performance on the respective task does not increase more than other types of distributions that do not focus on task. This seems counterintuitive and worth looking into further.

3.3 Model Agnostic Meta Learning (MAML)

Finn et al. in [9] developed a highly cited generic method for doing meta learning when certain tasks are low-resource. They benchmarked MAML across several different application areas - sinusoid wave prediction, few shot image classification from the Omniglot and MiniImagenet datasets, and reinforcement learning tasks with the MuJoCo simulator. Details of the algorithm are given in Section 4, see Figure 3 for visualization. For this project, a closely related variant of MAML developed by OpenAI called Reptile [13] is used as it is relatively simple to implement and has similar or superior performance to regular First-Order MAML.

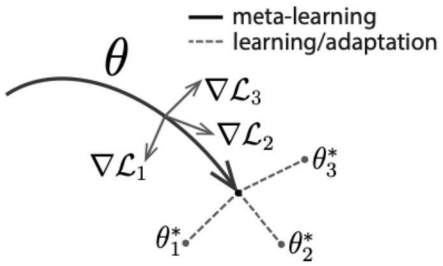

Figure 3: MAML optimizes to find parameters / representation θ that can quickly adapt to new tasks, from [9]

4 Approach

4.1 Background

A description of MAML is given in Figure 4.

As part of the Reptile optimization in [13], the equation from Figure 1:

Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters
1: randomly initialize θ
2: **while** not done **do**
3: Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
4: **for all** $\mathcal{T}_i$ **do**
5: Evaluate $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$ with respect to K examples
6: Compute adapted parameters with gradient descent: $\theta_i' = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$
7: **end for**
8: Update $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'})$
9: **end while**

Figure 4: Model Agnostic Meta-Learning training procedure. From Finn et al.

$$\theta = \theta - \beta \sum_{T_i \in p(T)} \nabla_\theta L_i(f_{\theta_i'}) \tag{1}$$

Can be replaced with:

$$\theta = \theta + \beta \sum_{T_i \in p(T)} (\theta_i' - \theta) \tag{2}$$

Where θ is the "generic" set of parameters, θ_i are the task specific parameters for task i, L_i is the task-specific loss function and f_θ is our given model with parameters θ. Intuitively, the model fine tunes for each task, aggregates over the changes to the model parameters for each task, applies this aggregate to the generic parameters and repeats.

Dou et. al. in [8] vary $p(T)$, the task distribution, to be Uniform, Probability Proportional to Dataset Size, and Mixed.

4.2 Implementation

Code was developed for implementing and testing a MAML / Reptile algorithm on the SquAD Dataset [14]. The learning rate for updating task-specific parameters, α, was not changed from the default rate used to train the DistilBERT transformer. However, the implementation allowed for and experimentation was done on different rates of annealing to the hyperparameter β, which was recommended by [10]. For Mixed task distributions $p(T)$, where the distribution starts uniform or probability proportional to dataset size and then over time changes to favor one task, the update to the distribution - which is done after each main parameter update of θ - is given by as follows:

Let $p(T) = [p_1, p_2, p_3]$ where $p_i \geq 0$ and $p_1 + p_2 + p_3 = 1$ (we have only three domains / tasks we fine-tune on)

If seeking to iteratively focus more on task 1, update $p(T)$ like so:

$p(T) := [\frac{c*p_1}{c*p_1+p_2+p_3}, \frac{p_2}{c*p_1+p_2+p_3}, \frac{p_2}{c*p_1+p_2+p_3}]$ where $c > 1$ another hyperparameter.

K, the number of steps of gradient descent done when adapting θ to task i, is also set as a configurable hyperparameter that can be increased over time - another recommendation of [10] for improving stability.

A novel classifier for predicting dataset membership of an input example was also developed and tested, the idea being that it could help act as a wrapper for three models, each model fine-tuned to give optimal performance on exactly one of the input datasets / domains.

All code for MAML / Reptile updates was developed by the author, the method for varying the rate c to determine how $p(T)$ converges to a certain task and the the classifier for predicting dataset

membership have not been tried in recent MAML-based approaches in natural language processing and understanding.

4.3 Baselines.

The main baseline will be a regularly fine-tuned version of DistilBERT for the out of domain datsets. The code for constructing / training this model is provided in the Robust QA project handout.

5 Experiments

5.1 Data.

Most data comes from the specified *in-domain* datasets: Natural Questions, NewsQA, and SQuAD [14, 15]. A small number of training examples came from the specified *out-of-domain* datasets: RelationExraction, DuoRC and RACE.

5.2 Evaluation.

The evaluation metrics are the same ones used on the SQuAD leaderboard: **Exact Match (EM)** and **F1**.

5.3 Experimental Details

DistilBERT was pre–trained for 3 epochs on the whole corpus of in-domain data. Performance on out-of-domain datasets was then measured. A main result of [10], expanding the number of task-specific updates K as a function of the number of main parameter updates. This helps stabilize the MAML training process, which is known to be very sensitive to a choice of hyper-parameters. Different Annealing rates for the β hyperparameter were tried on fixed task distributions Uniform and Probability Proportional to Dataset Size (PPDS). task-specific F1 was also reported for a PPDS distribution with the best β annealing rate. Hyperparameter optimization was done for mixed task distributions and task-specific F1 is reported for the best observed models for the relation extraction and duorc datasets. Best task-specific F1 is also reported for the race dataset, which did not respond well to any MAML-based training given the ensemble of hyperparameter combinations used. A classifier to predict dataset membership is trained, and accuracy on all three datasets over the number of training iterations completed is reported. This classifier is then used as a wrapper for a Mixture of Experts (MoE) that leverages models each specifically optimized to a given dataset. Predictions from the three models are then combined in two ways and validation F1 and EM is reported for each.

5.4 Results

"Normal DistilBERT" achieved a standard 48.18 F1 and 33.25 EM on out-of-domain test data (RobustQA).

All task distributions performed relatively poorly and got worse over time on the race dataset. Best performance was on the original pre-trained DistilBERT model, which gave an F1 score of 40.88 for the race dataset.

Output from the dataset classifier was used in two ways. In the first approach, the argmax of the classifier's predictions is used to select a task-optimized model (mixed task distribution trained for datasets relation extraction and duorc and the original DistilBERT model for the race dataset). Validation EM was 31.675 and Validation F1 was 47.568 on the entirety of the out-of-domain datasets. A secondary approach was then applied where the softmax function was applied to the classifier's prediction vector to create a set of weights D. The final start and end logits prediction calculated for each example was then expressed as a weighted sum of each task-optimized model's start and logits prediction, with D as the weights. This weighted expert approach got a validation EM of 33.769 and validation F1 of 48.138 on the entirety of the out-of-domain datasets. In terms of validation dataset performance, The model that performed the best was the Mixed-task distribution that focused on the relation extraction task, getting an EM socre of 34.03 and an F1 score of 49.01.

On the held out test dataset, a regular baseline model (pre-trained DistilBERT), the relation extraction optimized model (see Figure 8), and the weighted expert model were evaluated. The baseline got EM = 40.872 and F1 = 58.305, the relation extraction optimized model got an EM = 41.468 and F1 = 58.773, and the weighted expert model had EM = 41.537 and F1 = 59.300.

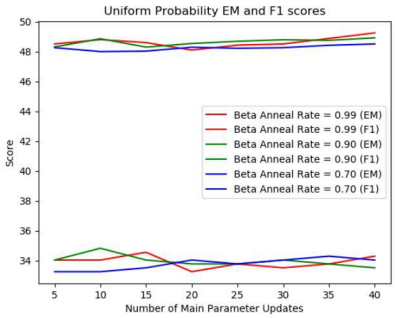

Figure 5: Performance data for Uniform Task Distribution. Improvement seems limited after many main parameter updates, best annealing rate for β seems to 0.99

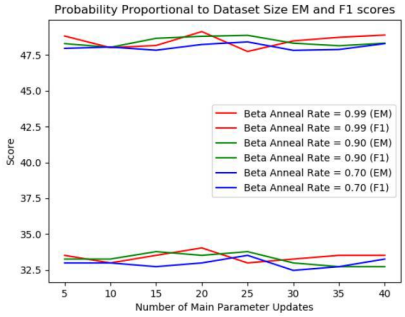

Figure 6: Performance data for PPDS Task Distribution. Improvement seems limited after many main parameter updates, best annealing rate for β seems to 0.99

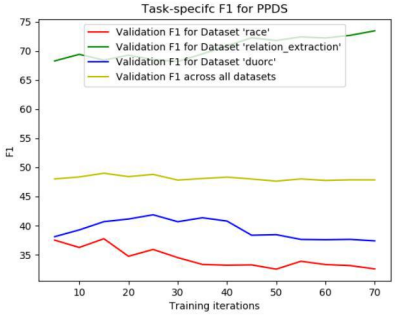

Figure 7: Task-specific performance data for PPDS Task Distribution using an annealing rate of 0.99 to β. Improvement was best for the relation extraction dataset, with gradually worse performance on duorc and race over time.

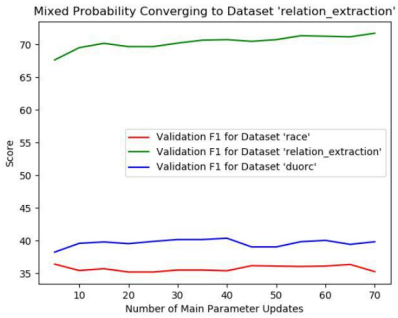

Figure 8: Optimal hyperparameter choice involved setting C = 1.1, K's expansion rate = 1.025, and β's annealing rate = 0.9, with initial value β = 1

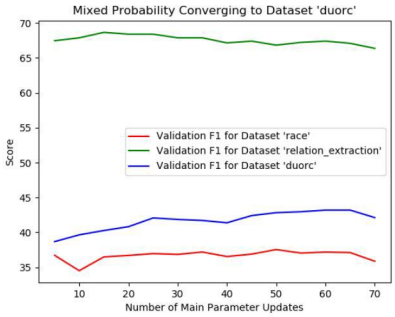

Figure 9: Optimal hyperparameter choice involved setting C = 1.1, K's expansion rate = 1.025, and β's annealing rate = 0.9, with initial value β = 1

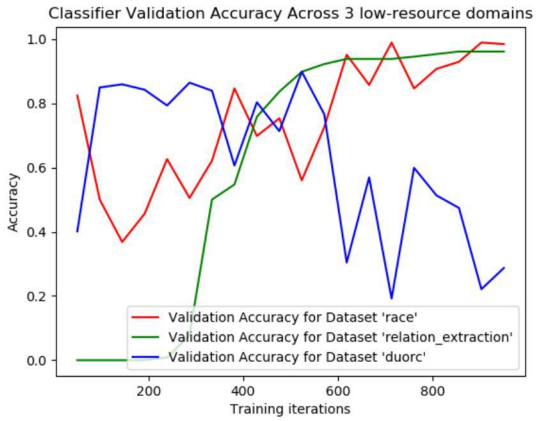

Figure 10: Architecture: Model is 2 layer neural network with 3 hidden units and uses the ReLU activation function. Training was done with the cross entropy loss function.

6 Analysis

The weighted expert model, while not having the best validation set performance, had the best performance on the test set and was significantly better than the baseline model. This indicates that the classifier was able to generalize reasonably well to the test set. Classifier performance (see Figure 10) indicated that getting better at recognizing membership in the duorc dataset was directly correlated

with worse performance on recognizing membership in the race and relation extraction datasets. This result was not unique to the choice of classifier architecture shown, though the architecture for which accuracies are reported had the best max min of the three accuracies (roughly 550 iterations into the training process). Qualitatively speaking, these results would imply that the relation extraction and race datasets were strongly different in nature, as they could be told apart relatively easily by the classifier. This shows in some of the training / validation error in Figure 8, as fine-tuning for the relation extraction dataset did not improve performance for the race dataset. However, these results also imply that the duorc dataset is very similar to both relation extraction and race, but based off of Figure 9, weighting duorc training updates in the main parameter update of MAML had a slight negative effect on the other two datasets' performance.

It should be noted how hard it was to train a model that was specialized to the race dataset, as any choice of hyperparameters for a MAML fine-tuning algorithm to train for maximum performance on the race dataset did poorly. In practice, making the step size of the main parameter, β, equal to a very small number like 0.1 or 0.01 (it was set to 1 for all models based off of a recommendation in [10]) reduced the performance drop. As the best performance on the race dataset came on the original baseline DistilBERT model, it is possible that the original main parameters from pre-trained DistilBERT were the global optimum.

In general, using a mixed task distribution created the best performance on the relation extraction and duorc datasets. While a task distribution consisting of probability proportional to dataset size (PPDS) did well (see Figure 7), a weighted expert model with a better classifier would have a considerably higher ceiling in terms of performance. This contrasts with the main result of [8], that the PPDS task distribution had the best task-specific performance.

Numerous hyperparameter searches were done, as shown in Figures 6 and 5. These indicated that significantly annealing β, the step size of the main parameter update, did not result in better performance.

7 Conclusion

This paper showed that a weighted expert model could strongly outperform a baseline model on low-resource datasets. Given more time, a wider hyperparameter search could yield more consistent results as the training process of any MAML / Reptile algorithm is known to be unstable [10].

References

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[2] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

[3] Ethan Alley, Grigory Khimula, Surojit Biswas, Mohammed Al-Quraishi, and George Church. Unified rational protein engineering with sequence-based deep representation learning. *Nature Methods*, pages 1315–1322, 2019.

[4] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *International Conference on Learning Representations (ICLR)*, 2015.

[5] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NeurIPS*, 2017.

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

[7] Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jian-feng Gao. Multi-task deep neural networks for natural language understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics)*, pages 4487—4496, 2019.

[8] Zi-Yi Dou, Keyi Yu, and Antonios Anastasopoulos. Investigating meta-learning algorithms for low-resource natural language understanding 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)*, pages 1192—1197, 2019.

[9] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *International Conference on Machine Learning (ICML)*, 2017.

[10] Antreas Antoniou, Harrison Edwards, and Amos Storkey. How to train your maml. *International Conference on Learning Representations (ICLR)*, 2019.

[11] David Kelley, Jasper Snoek, and John Rinn. Basset: learning the regulatory code of the accessible genome with deep convolutional neural networks. *Genome Res*, pages 990–999, 2016.

[12] Jiatao Gu, Yong Wang, Yun Chen, Victor Li, and Kyunghyun Cho. Meta-learning for low-resource neural machine translation. *EMNLP*, 2018.

[13] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *ArXiv*, 2018.

[14] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, 2016.

[15] Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for SQuAD. In *Association for Computational Linguistics (ACL)*, 2018.