Smoothing Multistage Fine-Tuning in Multi-Task NLP

Amir Ziai (amirziai@stanford.edu), Oleg Rudenko (orudenko@stanford.edu)

Motivation

A recent trend in many NLP applications is to fine-tune a network pre-trained on a language modeling task using models such as BERT[1] in multiple stages. In this work we have studied the effects of the **training schedule** for training models with a **shared** representation. We show that using a **bandit approach** for selecting training batches can results in achieving higher **validation F1** scores in a smaller number of epochs for a sentiment classification task, which is very important for training large networks.

Data

The data for this study comes from the **GLUE** benchmark[2].

Sentiment Classification (SST-2)
Each record in this dataset is a movie review with a corresponding human-annotated positive or negative label. Example of a negative record: `portray its literarily talented and notorious subject as anything much more than a dirty old man`

Paraphrasing news (MRPC)
This dataset consists of pairs of sentences automatically extracted from news and the label is whether one sentence is a paraphrasing of the other.

Stanford Question Answering (QNLI)
Data consists of pairs of questions and sentences. The label is whether the answer to the question is entailed in the sentence.

Features

For our baseline we used a binary bag-of-words representation for the top 10k tokens. For vanilla neural networks we used word embeddings trained from scratch. BERT uses token, segment, and position embeddings from WordPiece with a 30k token vocabulary.

Models

Evaluation metric
In all experiments we measured the validation F1 score after each epoch of training on SST-2.

Baseline
Our baseline is a logistic regression classifier which achieved a validation F1 score of 0.9.

Vanilla Neural Network
We used a single hidden-layer MLP with 256 units, 10k vocabulary, and Adam optimizer with lr=0.0003. This shared layer is connected to classification head for each task.

BERT
BERT is pre-trained on vast amounts of unlabeled text and leverages novel ideas such as transformers and bidirectional encoders. We used learning rate of 0.0003 and dropout of 0.1 with uncased base BERT.

Training Schedules

Round robin
We simply cycle between tasks for each epoch.

ε-greedy
ε determines how aggressively we explore new tasks. We chose ε=0.3, so 30% of the time we randomly picked a new task and 70% of the time we select the task with the highest average gain in F1 at time T relative to the previous epoch T-1:

$$\Delta F_1^T = F_1^T - F_1^{T-1}$$

Hypothetical ε-greedy run over 3 epochs for tasks A and B where $p \in [0,1]$ is generated randomly:

T	**0**	**1**	**2**	**3**
p	---	0.2	0.68	0.11
Decision	---	Explore	Exploit	Explore
Task	---	A	A	B
F_1	---	0.56	0.57	0.51
ΔF^A_1	0	0.56	0.285	0.285
ΔF^B_1	0	0	0	-0.06

Results

The following table shows the evolution of training at 500 and 2,500 epochs. After 500 epochs most algorithms had a chance to explore the tasks and the results show the early gains in learning. At 2,500 epochs most algorithms had stopped making significant improvements.

Training schedule	**Additional task**	**Train F1 score** N=67k, 3k, and 105k for SST-2, MRPC, and QQP		**Validation F1 score** **Baseline: 0.90** N=872	
		@ 500 epochs	@ 2500 epochs	@ 500 epochs	@ 2500 epochs
Round robin (vanilla)	MRPC	0.72	0.98	0.75	0.76
ε-greedy (vanilla)	MRPC	0.80	**0.99**	0.72	0.78
ε-greedy (vanilla)	QNLI	0.61	0.78	0.59	0.74
Round robin (BERT)	MRPC	0.84	0.92	0.79	0.81
ε-greedy (BERT)	MRPC	**0.87**	0.96	**0.83**	**0.93**

Discussion

The first observation is that ε-greedy can quickly focus on the more important task and feed more batches of that task to the model. This effect is more pronounced when using BERT since the model is pre-trained on a language modeling task and fine-tuning results in early gains in validation relative to a vanilla neural network. The vanilla network has the capacity to learn (training score of 0.99 achieved for MRPC), however, generalizing to an NLP task from scratch is harder in comparison to BERT. We also observe that using MRPC is more beneficial compared to QNLI, which is a larger dataset and contains more formal language that probably requires more training.

Future work

We will expand the list of tasks to include sentence similarity (STS-B), social QA questions (QQP), and coreference resolution (WNLI). We want to both train on an expanded set as well as evaluate on other tasks. Also, we want to explore different choices of network architecture, hyper-parameters, and pre-trained embeddings.

References

[1] Devlin, J., Chang, M.W., Lee, K. and Toutanova, K., 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.
[2] Wang, A., Singh, A., Michael, J., Hill, F., Levy, O. and Bowman, S.R., 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. arXiv preprint arXiv:1804.07461.

Video Link

https://www.youtube.com/watch?v=KRVn2OCBcBY