Lightweight neural network development via knowledge distillation of deep neural network

Stanford CS224N Default Project

Naoki Yamamura
Department of Computer Science
Stanford University
yamamura@stanford.edu

Abstract

In this project we developed a lightweight neural network model that minimizes the trade-off between model size and performance. We mainly used a regularization technique, “knowledge distillation” to compress the learned representation of deep teacher models to a shallow student model over the SQuAD 2.0 to revisit the prediction power of shallow models. We extracted knowledge from the deep BERT\text{BASE} model and compressed it into a shallow BiDAF model, called “DistillBiDAF”. We restricted ourselves from any architectural change from its original implementation. Our proposed approach achieved an EM score of 65.50 and F1 score of 68.49 for single model and EM score of 67.68 and F1 score of 70.64 for ensemble model on the test set (ranked 1st on the non-PCE test leaderboard). Size and inference time of the model was kept from its original which is 71.8% less in number of parameters and 87.4% faster in inference time compared to the BERT\text{BASE} model.

1 Introduction

With the trend of the deep language representation models, such as BERT\textsuperscript{[1]}, out-performing in complex tasks, shallower previous generation neural network models had become somewhat obsolete. However, the state-of-art deeper models require a large amount of computational resources. Deployment of such large models in production could be problematic in resource-restricted systems (e.g. mobile application). Constraining model deployment on the server side may also not be ideal sometime (e.g. long real time prediction latency).

Model compression of deep teacher models by “knowledge distillation”\textsuperscript{[2]} to shallower student models is one popular approach to address such issues. Tang et al. (2019)\textsuperscript{[3]} had demonstrated knowledge distillation from teacher BERT to a single-layer student BiLSTM had consistently improved BiLSTM performance without any model architecture engineering, training data, or additional input features.

However, the architecture of models experimented and tasks addressed with knowledge distillation are still limited. To our knowledge, there is no prior research that focuses on Question-Answering on SQuAD 2.0\textsuperscript{[4]} with distilled shallow models. In this project we experimented with shallow BiDAF\textsuperscript{[5]} student models, which we call it as “DistillBiDAF”, to achieve higher performance against SQuAD 2.0 by knowledge distillation from deep BERT model while maintaining the student model size.

2 Related Work

Model compression was an idea originally introduced by Caruana et al. (2006)\textsuperscript{[6]} to circumvent the trade-off between size and performance. While compressing the knowledge from a trained model seemed very promising, it had not been actively investigated further as it seemed hard to alter the form but keep the same knowledge\textsuperscript{[2]}.

“Knowledge distillation” is a regularization technique introduced by Hinton et al. (2015)\textsuperscript{[2]} which popularized the research trend in teacher-student models. Cumbersome (i.e. deep and complex)
teacher models, such as BERT, are first trained to extract data representation. Then compressed into a single small student model to transfer teachers’ prediction power. Unlike transfer learning, distillation allows the student model to have completely different architecture from its teacher models.

DistillBERT is one of the recent distilled models introduced by Sanh et al. (2019)\[7\]. While most prior distilled models were task-specific, they focused on providing general purpose distilled models by distilling knowledge from BERT during the pre-training phase. DistillBERT compressed the model size by 40% from BERT but retained 97% of language understanding capabilities while being 60% faster. Question answering on SQuAD 1.1 was one of the downstream tasks they have evaluated. They showed that using previously SQuAD 1.1 fine-tuned BERT, DistillBERT performed within 3 points of the full BERT model.

3 Approach

3.1 Knowledge Distillation

Hinton et al. (2015)\[2\] suggested penalizing cross-entropy soft target loss against teacher’s softmax. Teacher models are trained to generalize well to new data. “Soft targets” is the class probabilities provided by the output of the final softmax layer and “hard targets” is a classifier often provided in the form of one-hot vectors. By training the student model to minimize the soft target loss against the teachers’, the generalization ability of the teacher models are transferred well. Tang et al. (2019)\[3\] suggested a simpler objective function to minimize the mean-squared-error (MSE) loss between the student’s logits (i.e. input to softmax) against the teachers’ logits. Because the paper stated logit based worked slightly better than softmax based, we used logit based objective function (1). \(z^B\) and \(z^S\) are respectively the teacher’s and student’s logits.

\[
L_{distill} = ||z^B - z^S||^2_2 \quad (1)
\]

We used SQuAD 2.0 training dataset as our transfer dataset. Hinton et al. (2015)\[2\] stated targeting to minimize loss against both soft and hard targets significantly improves the students’ performance. Therefore, to train student models, we calculated the loss as a weighted average of hard cross-entropy negative log-likelihood loss and soft distillation logit MSE loss (2). We experimented with different \(\alpha\) values to train the student model. Figure.1 provides an overview of our own teacher-student model training architecture.

\[
L_{total} = \alpha L_{distill} + (1 - \alpha) L_{nll} \quad where \ 0 \leq \alpha \leq 1 \quad (2)
\]

![Figure 1: Our teacher-student model training architecture (diagram adapted from Upadhyay\[8\)](image-url)
3.2 Baseline

We trained the baseline student and teacher model independently over SQuAD 2.0. Our baseline teacher is the open-source PyTorch implementations (https://github.com/huggingface/transformers) of BERT\textsubscript{BASE}. Our baseline student model is the original implementation of the BiDAF model with a character-level embedding layer proposed by Seo et al (2016)\cite{5}.

<table>
<thead>
<tr>
<th>Baseline model</th>
<th>Dev EM score</th>
<th>Dev F1 score</th>
</tr>
</thead>
<tbody>
<tr>
<td>BERT\textsubscript{BASE}</td>
<td>72.55</td>
<td>75.82</td>
</tr>
<tr>
<td>BiDAF</td>
<td>58.80</td>
<td>62.22</td>
</tr>
</tbody>
</table>

Table 1: Baseline models EM and F1 scores

3.3 Teacher Logit Data Transformation

While knowledge distillation seemed promising, the challenge we faced was transforming the teacher’s extracted knowledge (i.e. logits) as a useful training target. Unlike the spacy tokenizer our BiDAF models use, BERT\textsubscript{BASE} tokenizer converts OOV (Out Of Vocabulary) words to wordpiece tokens. Training features are from a sentence concatenated tokens of a question and a context. If the tokenized sentence pair exceeds the maximum sequence length, the sentence pair will span across multiple features. All such differences skew the logit mapping to the BiDAF model to train.

To our knowledge, existing publicly available distillation frameworks are not a great fit for our purpose. Hence, we explored different data transformation approaches to accurately map the teachers’ knowledge.

3.4 Student Model Performance Tuning

The main parameters we experimented were regularization parameters; dropout rate and distillation loss coefficient $\alpha$. Because teacher models are trained to generalize well in unseen data, the generalization ability is compressed to students well by overfitting to teacher models. Therefore, we experimented with model tuning by raising $\alpha$ with decreased dropout rate. We restricted the hidden layer size parameter to the default value of 100 due to our goal to build lightweight models. Only for this part, we experimented over BiDAF model without character-level embedding.

We trained several slightly different configurations of DistillBiDAF models by altering the RNN layer from bidirectional LSTM to GRU and unfreezing word-level embedding. We trained three different configurations and compared the performance and training time.

Additionally, we experimented with unsupervised training (i.e. $\alpha=1$ with no label) against small additional data over trained model for 40–50K steps. We used test/dev set that is not just overfitting to the evaluation data and adds impractical bias) an evaluation target (i.e. when Dev set used as evaluation, use Test set as unsupervised training target and vice versa) as an additional unsupervised training target.

In the final step, to unreveal the full prediction power of the DistillBiDAF model, we ensemble trained models and measured the performance against SQuAD 2.0. We simply took the weighted average of the logits of the trained models. Weights were experimented by directly feeding the aggregated logits into the softmax layer of the test script provided (i.e. test.py).

4 Experiments

4.1 Datasets

SQuAD 2.0\cite{4} is the latest version of the Stanford Question Answering Dataset. It combines previous versions of SQuAD data with over 50,000 unanswerable questions. It is widely used data to evaluate natural language understanding tasks for models. We have strictly experimented over the official SQuAD 2.0 train dataset (129941 examples) and custom dev (6078 examples) and test (5915 examples) dataset provided for the CS224N project.
4.2 Evaluation method

We measured the performance of our models by two metrics, EM (Exact Match rate against the ground truth answers) and F1 (the harmonic mean of precision and recall) scores. Because our other goal is to develop a “lightweight” model, we evaluated model size by megabyte and number of parameters as well.

While our primary goal is not to develop a faster model, one of the benefits of distilled models against large models is inference time efficiency. Therefore, we evaluated the inference time as a secondary metric. We measured it as the time needed to do a full validation over SQuAD 2.0 custom dev set with batch size of 8 on NVIDIA NV6 GPU (6 vcpus, 56 GiB memory).

4.3 Teacher Logit Data Transformation

4.3.1 BiDAF-BERT Logit Map

After we extracted logits from the teacher BERT models, we needed to understand how BiDAF and BERT features map. Generated BERT features contain the map of token id to the original-text (i.e. whitespace tokenized) to identify answer texts. Using this information and the respective tokenizers, we created the original-text-to-token map (i.e. reverse) for both BiDAF and BERT. Because BERT’s features are sentence pairs, mapping differs per example. BERT features also depend on max sequence length parameters. We created a per-model-examples spacy-BERT token id mapper. As an outcome of this step, we obtained per-model-examples many-to-many relationships dictionary between BiDAF and BERT tokens.

Next, we mapped the teacher’s logit values to a token of each BiDAF examples’ context tokens. The spacy transformer library (https://github.com/explosion/spacy-transformers) provides a mapper that maps single spacy word tokens to a list of BERT wordpiece tokens. While the mapper provides a good estimate, it often contains offset while tested mapping against the actual training tokenizers. We created a wrapper to fix the alignment that works perfectly for our case. Using this info, and many-to-many token dictionary from the previous step, we obtained a dictionary that maps each BiDAF example context token id to a list of logits from the BERT model. As a final step, we created a map that normalizes wordpiece tokens. Overview of the data transformation is in Figure.2 in next page.

4.3.2 Wordpiece Logit Normalization

To understand the effectiveness in different wordpiece logits normalization approaches (i.e. max/mean/min) and quality of the transformed logits data, we first experimented training BiDAF with fixed $\alpha$ (=0.5) for 30 epochs. Model hyperparameters also remained unchanged from default values. All normalized logits data improved the model performance around 2.5 points from student baseline in both EM and F1 score, but no notable differences were observed to evaluate among normalization approaches.

Thus, to objectively measure the difference, we fed in transformed logits directly to the softmax layer of the provided score evaluation method (i.e. test.py) and obtained an approximation of the EM and F1 score. We refer to the obtained value as the “transformed” performance of the model in comparison to the “actual” performance of the baseline BERT teacher model.

As it is clear from Table.2, max normalization had the best results. One fact observed from unnormalized transformed data is that wordpiece logits were most likely to have the highest score at the first token for start logits and last token for the end logits. This is logical because wordpieces split OOV words to smaller pieces but when questions are answered, it is most likely answered as a whole word instead of pieces of the word. From the metrics obtained, that aligns with the logical intuition, we decided to use max normalization for wordpiece logits.

<table>
<thead>
<tr>
<th>Normalization</th>
<th>Dev EM score</th>
<th>Dev F1 score</th>
</tr>
</thead>
<tbody>
<tr>
<td>max</td>
<td>70.73</td>
<td>73.86</td>
</tr>
<tr>
<td>mean</td>
<td>67.17</td>
<td>71.55</td>
</tr>
<tr>
<td>min</td>
<td>65.42</td>
<td>70.49</td>
</tr>
</tbody>
</table>

Table 2: Transformed BERT models performance
4.4 Regularization Parameters Tuning

We first experimented with distillation loss coefficient $\alpha$ for an untrained BiDAF model with no character-level embedding. Figure 3 shows F1 trend for first 5 epochs. Higher $\alpha$ contributed in avoiding initial drop caused by always predicting no-answer to reduce negative-log-likelihood loss. At the same time, $\alpha=0.5$ achieved highest performance at the end of epoch 5, after recovering from the initial drop. In aligning with what Hinton et al. (2015) [2] had stated, that hard targets would contribute to higher performance in shorter training time.

Similarly, to understand the relation of $\alpha$ and dropout rate at stable state, the same BiDAF model were retrained for another 5 epochs (Figure 4) from the point it achieved the best performance without distillation (at epoch 22, with $\alpha=0$, $dp=0.2$). During the initial training without distillation, we saw that the model already fitted to the hard targets and started reducing performance. On the other hand, with the teacher’s logit target and a lower dropout rate, it significantly improved performance. This indicates that overfitting to teacher’s logit is rather desired. Additionally, higher $\alpha$ accelerated the improvements.

Figure 3: F1 for untrained BiDAF model

Figure 4: F1 for retrained BiDAF model
From these results, we started training with moderate $\alpha$ (=0.5~0.6) then once improvements slow down, we trained with higher $\alpha$ (=0.7~1.0) while consistently keeping low dropout rate (=0.0~0.1).

5 Results

5.1 DistillBiDAF Model Performance

The DistillBiDAF single model had shown significant improvement in EM and F1 score of over 7.5 point from student baseline. We achieved 67.00 and 65.50 in EM and 69.95 and 68.49 in F1 score respectively on the Dev and Test set. No architectural change was made and no new inputs were used other than distillation target from the teachers. Interestingly, unsupervised training against small (~6k examples) unseen unlabeled datasets improved generalization ability of the models further. Minimizing MSE logit loss against a small new dataset raised 0.5 more points in EM and F1 scores. For word-embedding configuration used, there were no notable differences in the performance. But differences in training time were observed. The model with unfrozen word-level embedding were trained faster. It took 24.5% less training steps to reach the best performance with unfrozen word-embedding. Similarly models with GRU RNN layer took 30.1% less training steps compared to bidirectional LSTM.

As a last step, we experimented with the model ensemble taking a weighted average of logits from 3 trained DistillBiDAF models. The best performing ensemble model had raised 2 more points with 69.18 and 67.68 in EM and 71.73 and 70.64 in F1 score respectively on Dev and Test set.

<table>
<thead>
<tr>
<th>Type</th>
<th>Dev EM score</th>
<th>Dev F1 score</th>
<th>Test EM score</th>
<th>Test F1 score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single</td>
<td>66.48</td>
<td>69.44</td>
<td>64.73</td>
<td>67.81</td>
</tr>
<tr>
<td>Single + unsupervised training</td>
<td>67.00</td>
<td>69.95</td>
<td>65.50</td>
<td>68.49</td>
</tr>
<tr>
<td>Ensemble</td>
<td>69.18</td>
<td>71.73</td>
<td>67.68</td>
<td>70.64</td>
</tr>
</tbody>
</table>

Table 3: DistillBiDAF models best performances

5.2 DistillBiDAF Model Size and Inference Time

We also evaluated model size and inference time in comparison. By adding a character-level embedding, BiDAF model had a 10% increase in its size and inference time. But compared to BERT\textsubscript{BASE}, our single DistillBiDAF was 71.8% smaller in size and 87.4% faster in inference time while being 6 points away in F1 score.

<table>
<thead>
<tr>
<th>Model</th>
<th># of Parameters</th>
<th>Binary Size</th>
<th>Inference Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>DistillBiDAF</td>
<td>31M</td>
<td>124MB</td>
<td>24 sec</td>
</tr>
<tr>
<td>\textbf{BERT\textsubscript{BASE}}</td>
<td>110M (x3.55)</td>
<td>438MB (x3.53)</td>
<td>191 sec (x7.96)</td>
</tr>
</tbody>
</table>

Table 4: DistillBiDAF single model vs BERT\textsubscript{BASE} size and inference time

6 Analysis

6.1 Transformed Teacher Logit Data Quality

To validate the effectiveness of transformed logits, we first compared the difference in answers over transformed and actual BERT teacher models. In 93.5% of the cases answers were exact match, 5.84% were partial match, and 0.66% were entirely unmatched (Figure.5). We took a deeper dive into the answers that did not entirely match. Approximately 41.6% were in different score buckets from the actual BERT model’s prediction (Figure.6). For those in different buckets, transformed teacher model prediction had lower scores at a higher rate (35.4% vs. 6.20%). This caused the transformed data to have slightly lower (i.e. F1 of 75.82 vs. 73.86) prediction scores.

To clarify the gap in performance more in depth, we investigated different aspects. One finding was that there are some great gaps for contexts with large OOV word counts as shown in Figure.7. This isn’t logically surprising. Because while max logit in a wordpiece contains most critical information,
others contribute to the prediction as well. When OOV is less than 20, F1 scores are almost identical. 
While we understood one potential significant cause, given the limitation on data structure, and 
the fact that approximately 97% of the predictions were in the same prediction score bucket, we 
considered that it still represents sufficient prediction power to train student models.

![Figure 5: Teacher BERTs Answer Match](image1)

![Figure 6: Partial/unmatched Answer Scores](image2)

![Figure 7: Actual vs. Transformed BERT F1 score by OOV word counts](image3)

6.2 OOV Prediction Improvements

One of the main benefits of using BERT models’ logit as training targets is that OOV words are not 
considered as <UNK> tokens and have some trained predictions from wordpieces. Figure 8 maps 
the comparison of the F1 scores of the BiDAF models by context OOV word counts. In general 
DistillBiDAF has higher F1. However, when OOV count is greater than 15, the gap between two 
models tends to have some large gaps. Though some of it could potentially be a smaller sample 
size for larger OOV word count contexts, from this, we consider that distillation had contributed in 
enhancing the prediction power against contexts with OOV words.

![Figure 8: BiDAF model F1 scores by context OOV word counts](image4)
6.3 Teacher-Student Performance Relation

To understand the student DistillBiDAF model’s improvements, we mapped F1 score by number of tokens in each context. One finding was that the greater improvements in lower (<90) and higher ranges (>200) from the baseline BiDAF model. The baseline BiDAF model performed relatively well in the moderate length contexts. Its weak prediction ability on lower and larger ranges may be explained by logit distribution generated. For short contexts, logits are more peaked like one-hot values which makes prediction less robust. For longer contexts logits are more uniform. The BERT model seemed to handle these issues better by concatenating contexts with questions to make features and splitting above max sequence length sentence pairs into multiple features. By fitting into teacher’s logit distributions, knowledge to make more accurate predictions for such sentence pairs were well transferred.

Figure 9: Teacher vs. Student F1 score by context token counts

7 Conclusion

In this project, we significantly improved the performance of the BiDAF model without any architectural change. We experimented with different ways to improve performance by utilizing knowledge of deep teacher models. We showed that using a deep teacher models’ logit is a very effective training target to improve shallower models’ performance. Experiments with unsupervised training have shown a significance of training data that even very small size of unseen training targets would additionally increase performance. At last, an ensemble of distilled BiDAF models had built a robust prediction that placed at top on the test set in the leaderboard (March 17th, 2020). While we kept model size and inference time from its original by avoiding model architecturing, we managed to improve performance by 7.5% for a single model and 9.5% for an ensemble model.

8 Future Work

While we took a very data engineering heavy approach to bridge different tokens between two models, it could possibly be simplified by using the same tokenizer. With that, we could use more state-of-art models such as ALBERT [9]. With some quick tuning, ALBERTBASE model had surpassed BERTBASE model by more than 5 points in SQuAD 2.0 performance. Because effects of distillation are bounded by the ability of the teacher models, transferring knowledge from stronger teachers might result in better student model performance. On the other hand, using some other known Question-Answering well performing models such as QANet [10] may allow us to develop lightweight models with better performance. Additionally, we used MSE loss as an objective function. But there are several other well known distillation objective functions such as Softmax based [2] and cosine-similarity based [7] approach. These may possibly be more suitable for SQuAD 2.0 light model development with knowledge distillation. Lastly, there are some other inputs/roles we could extract from teacher models. For instance, some researches have shown effectiveness of using teacher’s attention maps [11] which may also be suitable for SQuAD 2.0 lightweight model development.
References


