Multi-CLS BERT: An Efficient Alternative to Traditional Ensembling

Haw-Shiuan Chang* Ruei-Yao Sun* Kathryn Ricci* Andrew McCallum

amazon | science  UMassAmherst | Manning College of Information & Computer Sciences
BERT Classifier

- Problem
  - A small text classification task
  - Unstable BERT’s performance
- What About?
  - Ensembling
- But …
  - Costly

Classic
5 BERT Ensemble

Fine-tuning
5 Times

Inference Cost:
5 Times
Can We Make Ensembling Almost as Efficient as the Single Model?

Yes!
Sharing the BERT Encoder

Classic 5 BERT Ensemble

Inference Cost: 5 Times

Fine-tuning 5 Times

Sharing Parameters

Inference Cost: ~Once

Fine-tuning 5 Times?
Fine-tuning only Once!

Sharing Parameters

Fine-tuning 5 Times

Inference Cost: ~Once

Proposed Multi-CLS BERT

Fine-tuning ~Once

Inference Cost: ~Once

Standard BERT

VS
Goal and Challenge

• Our goal
  • Aggregate the contextualized word embeddings differently

• Challenge
  • CLS embeddings are often identical
    • After seeing the same training samples

Proposed Multi-CLS BERT

A man is lifting weights in a garage. This makes garage smell sweaty.

The heavy weights make the man look strong.

His lifting speed shows that he often does the exercise.
Pretraining Diversification

Input sentence: A man is lifting weights in a garage

Next sentence:
- This makes garage smell sweaty
- The heavy weights make the man look strong
- His lifting speed shows that he often does the exercise
Architecture Diversification

- Insert different linear layers for different CLS tokens
- The differences of CLS could be stored in the linear weights
- The parameter increase is relatively small

But what if they learn the same weights?
Fine-tuning Diversification

After fine-tuning using gradient descent

[Diagram showing BERT base Encoder + Linear Layers with 1-4 Transformer Layers and 9-12 Transformer Layers, with Minus symbols indicating removal of some layers after fine-tuning.]
Experiment Settings

- Our main baseline MTL

- By optimizing the pretraining and fine-tuning methods of a state-of-the-art BERT model (Aroca-Ouellette and Rudzicz, 2020)

- Repeat training 16 times

- Pretraining 4 times and fine-tuning 4 times

- Many previous work shows that random seeds are important in GLUE and SuperGLUE
### Natural Language Understanding

BERT Base could be better than BERT Large

<table>
<thead>
<tr>
<th>Configuration ↓</th>
<th>Model Name ↓</th>
<th>Model Size ↓</th>
<th>100</th>
<th>GLUE 1k</th>
<th>Full</th>
<th>SuperGLUE 100*</th>
<th>1k*</th>
<th>Full</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pretrained</td>
<td>109.5M</td>
<td></td>
<td>55.71 ± 0.62</td>
<td>71.67 ± 0.15</td>
<td>82.05 ± 0.98</td>
<td>57.18 ± 0.43</td>
<td>61.55 ± 0.37</td>
<td>65.04 ± 0.36</td>
</tr>
<tr>
<td>MTL</td>
<td>109.5M</td>
<td></td>
<td>59.29 ± 0.27</td>
<td>73.26 ± 0.13</td>
<td>83.30 [1] ± 0.97</td>
<td>57.50 ± 0.41</td>
<td>62.94 ± 0.36</td>
<td>66.33 ± 0.33</td>
</tr>
<tr>
<td>Ours (K=5, λ = 0)</td>
<td>111.3M</td>
<td></td>
<td>57.84 ± 0.32</td>
<td>74.14 ± 0.12</td>
<td>83.41 ± 0.97</td>
<td>57.31 ± 0.35</td>
<td>63.35 ± 0.18</td>
<td>66.29 ± 0.18</td>
</tr>
<tr>
<td>Ours (K=5, λ = 0.1)</td>
<td>118.4M</td>
<td></td>
<td>61.54 ± 0.32</td>
<td>74.10 ± 0.13</td>
<td>83.47 ± 0.05</td>
<td>58.29 ± 0.33</td>
<td>63.71 ± 0.26</td>
<td>66.74 ± 0.25</td>
</tr>
<tr>
<td>Ours (K=5, λ = 0.5)</td>
<td>118.4M</td>
<td></td>
<td>61.80 ± 0.35</td>
<td>74.02 ± 0.13</td>
<td>83.47 ± 0.05</td>
<td>58.20 ± 0.31</td>
<td>63.61 ± 0.27</td>
<td>66.74 ± 0.26</td>
</tr>
<tr>
<td>Ours (K=5, λ = 0.9)</td>
<td>118.4M</td>
<td></td>
<td>60.49 ± 0.35</td>
<td>74.02 ± 0.13</td>
<td>83.47 ± 0.05</td>
<td>58.41 ± 0.38</td>
<td>63.78 ± 0.25</td>
<td>66.80 ± 0.24</td>
</tr>
<tr>
<td>Ours (K=5, λ = 1)</td>
<td>118.4M</td>
<td></td>
<td>59.86 ± 0.34</td>
<td>73.93 ± 0.14</td>
<td>83.43 ± 0.07</td>
<td>57.84 ± 0.49</td>
<td>63.56 ± 0.22</td>
<td>66.39 ± 0.22</td>
</tr>
<tr>
<td>MTL</td>
<td>335.2M</td>
<td></td>
<td>61.39 ± 0.37</td>
<td>75.30 ± 0.27</td>
<td>84.13 ± 0.11</td>
<td>59.03 ± 0.54</td>
<td>65.21 ± 0.38</td>
<td>69.16 ± 0.37</td>
</tr>
<tr>
<td>Ours (K=1)</td>
<td>338.3M</td>
<td></td>
<td>59.19 ± 0.43</td>
<td>75.35 ± 0.21</td>
<td>84.59 ± 0.07</td>
<td>57.35 ± 0.42</td>
<td>64.67 ± 0.43</td>
<td>69.24 ± 0.41</td>
</tr>
<tr>
<td>Ours (K=5, λ = 0)</td>
<td>350.9M</td>
<td></td>
<td>63.19 ± 0.49</td>
<td>75.73 ± 0.26</td>
<td>84.51 ± 0.05</td>
<td>59.46 ± 0.44</td>
<td>65.43 ± 0.38</td>
<td>69.56 ± 0.31</td>
</tr>
<tr>
<td>Ours (K=5, λ = 0.1)</td>
<td>350.9M</td>
<td></td>
<td>64.24 ± 0.40</td>
<td>76.27 ± 0.12</td>
<td>84.61 ± 0.08</td>
<td>59.88 ± 0.43</td>
<td>65.58 ± 0.26</td>
<td>70.03 ± 0.25</td>
</tr>
<tr>
<td>Ours (K=5, λ = 0.5)</td>
<td>350.9M</td>
<td></td>
<td>63.02 ± 0.42</td>
<td>75.95 ± 0.10</td>
<td>84.49 ± 0.08</td>
<td>59.42 ± 0.34</td>
<td>65.84 ± 0.25</td>
<td>69.79 ± 0.25</td>
</tr>
<tr>
<td>Ours (K=5, λ = 1)</td>
<td>350.9M</td>
<td></td>
<td>62.07 ± 0.45</td>
<td>75.85 ± 0.17</td>
<td>84.61 ± 0.07</td>
<td>58.74 ± 0.50</td>
<td>65.00 ± 0.29</td>
<td>69.04 ± 0.27</td>
</tr>
</tbody>
</table>

---

The improvement of BERT Large is usually larger than the improvement of BERT Base.

<table>
<thead>
<tr>
<th>Configuration ↓</th>
<th>Model Name ↓</th>
<th>Model Size ↓</th>
<th>GLUE 100</th>
<th>GLUE 1k</th>
<th>GLUE Full</th>
<th>SuperGLUE 100*</th>
<th>SuperGLUE 1k*</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Pretrained</td>
<td>109.5M</td>
<td>± 0.62</td>
<td>± 0.15</td>
<td>± 0.08</td>
<td>± 0.43</td>
<td>± 0.37</td>
</tr>
<tr>
<td></td>
<td>MTL</td>
<td>109.5M</td>
<td>± 2.7</td>
<td>± 0.13</td>
<td>± 0.07</td>
<td>± 0.41</td>
<td>± 0.36</td>
</tr>
<tr>
<td></td>
<td>Ours (K=1)</td>
<td>111.3M</td>
<td>± 2.51</td>
<td>± 0.84</td>
<td>± 0.17</td>
<td>± 0.70</td>
<td>± 0.67</td>
</tr>
<tr>
<td>BERT Base</td>
<td>Ours (K=5, λ = 0)</td>
<td>118.4M</td>
<td>± 0.32</td>
<td>± 0.12</td>
<td>± 0.07</td>
<td>± 0.33</td>
<td>± 0.26</td>
</tr>
<tr>
<td></td>
<td>Ours (K=5, λ = 0.1)</td>
<td>118.4M</td>
<td>± 0.35</td>
<td>± 0.13</td>
<td>± 0.05</td>
<td>± 0.31</td>
<td>± 0.27</td>
</tr>
<tr>
<td></td>
<td>Ours (K=5, λ = 0.5)</td>
<td>118.4M</td>
<td>± 0.35</td>
<td>± 0.12</td>
<td>± 0.08</td>
<td>± 0.38</td>
<td>± 0.25</td>
</tr>
<tr>
<td></td>
<td>Ours (K=5, λ = 1)</td>
<td>118.4M</td>
<td>± 0.34</td>
<td>± 0.14</td>
<td>± 0.07</td>
<td>± 0.49</td>
<td>± 0.22</td>
</tr>
<tr>
<td></td>
<td>MTL</td>
<td>335.2M</td>
<td>± 0.37</td>
<td>± 0.27</td>
<td>± 0.11</td>
<td>± 0.54</td>
<td>± 0.38</td>
</tr>
<tr>
<td></td>
<td>Ours (K=1)</td>
<td>338.3M</td>
<td>± 2.85</td>
<td>± 0.97</td>
<td>± 0.48</td>
<td>± 0.85</td>
<td>± 0.37</td>
</tr>
<tr>
<td>BERT Large</td>
<td>Ours (K=5, λ = 0)</td>
<td>350.9M</td>
<td>± 0.49</td>
<td>± 0.26</td>
<td>± 0.05</td>
<td>± 0.44</td>
<td>± 0.38</td>
</tr>
<tr>
<td></td>
<td>Ours (K=5, λ = 0.1)</td>
<td>350.9M</td>
<td>± 0.40</td>
<td>± 0.12</td>
<td>± 0.08</td>
<td>± 0.43</td>
<td>± 0.26</td>
</tr>
<tr>
<td></td>
<td>Ours (K=5, λ = 0.5)</td>
<td>350.9M</td>
<td>± 0.42</td>
<td>± 0.10</td>
<td>± 0.08</td>
<td>± 0.42</td>
<td>± 0.26</td>
</tr>
<tr>
<td></td>
<td>Ours (K=5, λ = 1)</td>
<td>350.9M</td>
<td>± 0.67</td>
<td>± 0.17</td>
<td>± 0.07</td>
<td>± 0.50</td>
<td>± 0.29</td>
</tr>
</tbody>
</table>

Multi-CLS vs Ensembling

• In GLUE 100

57.76

+2.31

+4.86

+0.72

+ 3.27

+ Multi-CLS

Proposed Multi-CLS BERT

Classic 5 BERT Ensemble

([CLS]*5 + BERT)*5

Baseline Multi-CLS BERT Ensemble

([CLS] + BERT)*5

+ Classic Ensembling
Multi-CLS vs Ensembling

- In GLUE 100, Comparison of expected calibration errors (ECE).

<table>
<thead>
<tr>
<th>Model Type</th>
<th>Fine-tuning</th>
<th>Inference</th>
<th>ECE</th>
</tr>
</thead>
<tbody>
<tr>
<td>BERT</td>
<td>Once</td>
<td>Once</td>
<td>57.76</td>
</tr>
<tr>
<td>Multi-CLS BERT</td>
<td>Once</td>
<td>Once</td>
<td>25.22</td>
</tr>
<tr>
<td>5 BERT Ensemble</td>
<td>5 Times</td>
<td>5 Times</td>
<td>13.85</td>
</tr>
</tbody>
</table>

- Proposed Multi-CLS BERT: Fine-tuning: Once, Inference: Once (0.31 s), ECE: 15.46 s
- Classic 5 BERT Ensemble: Fine-tuning: 5 Times, Inference: 5 Times (1.46 s), ECE: 13.85
Conclusion

• Ensembling BERT almost without extra cost is achievable

• We need some tricks to diversify the multiple CLS hidden states

• Compared to standard ensembling
  • Improve more when the training dataset is small
  • Improve less otherwise
Our Other Work using Multiple Embeddings

H.-S. Chang*, Z. Yao*, A. Gon, H. Yu, and A. McCallum, “Revisiting the Architectures like Pointer Networks to Efficiently Improve the Next Word Distribution, Summarization Factuality, and Beyond” ACL Findings 2023
