Appendix for "Fine-Tuning Pre-Trained Language Models Effectively by Optimizing Subnetworks Adaptively"

Haojie Zhang¹, Ge Li¹; Jia Li¹, Zhongjin Zhang¹, Yuqi Zhu¹, Zhi Jin¹
¹Key Laboratory of High Confidence Software Technologies (Peking University), Ministry of Education; Institute of Software, EECS, Peking University, Beijing, China
zhanghaojie@stu.pku.edu.cn, lige@pku.edu.cn
lijia@stu.pku.edu.cn, zjjz123@stu.pku.edu.cn, zhuyuqi197@gmail.com, zhijin@pku.edu.cn

A Appendix A. Case Study

In Sec.3.3, we have experimentally verified that DPS outperforms various fine-tuning methods. To understand what type of cases DPS predicts more accurately and justify the effectiveness of DPS from another perspective, we conduct case study. Specifically, we fine-tune BERT_LARGE on RTE with 10 random restarts and count the overall proportion of easy cases (cases with more than 5 accurate predictions out of 10) and hard cases (cases with more than 5 predictions incorrect out of 10). Figure 1 summarizes the statistics. Compared with various baselines, DPS has the largest proportion of easy cases and the smallest proportion of hard cases. This demonstrates that compared with vanilla fine-tuning, Mixout, and CHILD-TUNING, DPS can better maintain general contextual representation, explore the potential value of data and models, and thus solve difficult cases without affecting the ability to identify easy cases, which is ultimately reflected in the improvement of metrics.

![Figure 1: Subfigure.(a) summarizes the percentage of easy cases on various methods; Subfigure.(b) summarizes the percentage of hard cases on various methods.](image)

B Appendix B. GLUE Benchmark Datasets

we conduct experiments on 8 datasets in GLUE benchmark [Wang et al., 2019], the dataset statistics for each task are illustrated in Table 1, we also provide a brief description for each dataset:

- **RTE**: Binary entailment classification task [Bentivogli et al., 2009]
- **MRPC**: Semantic similarity [Dolan and Brockett, 2005]

- **STS-B**: Semantic textual similarity [Cer et al. 2017]
- **CoLA**: Acceptability classification [Warstadt et al. 2019]
- **SST-2**: Binary sentiment classification [Socher et al. 2013]
- **QNLI**: Binary question inference classification [Rajpurkar et al. 2016]
- **QQP**: Binary semantically equivalent classification [Shankar Iyer and Csernai 2017]
- **MNLI**: Textual entailment classification [Williams et al. 2018]

<table>
<thead>
<tr>
<th>Dataset</th>
<th>RTE</th>
<th>MRPC</th>
<th>STS-B</th>
<th>CoLA</th>
<th>SST-2</th>
<th>QNLI</th>
<th>QQP</th>
<th>MNLI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train Examples</td>
<td>2.5k</td>
<td>3.7k</td>
<td>5.7k</td>
<td>8.5k</td>
<td>6.7k</td>
<td>105k</td>
<td>364k</td>
<td>393k</td>
</tr>
<tr>
<td>Dev Examples</td>
<td>277</td>
<td>408</td>
<td>1.5k</td>
<td>1.0k</td>
<td>872</td>
<td>5.5k</td>
<td>40k</td>
<td>4.8k</td>
</tr>
<tr>
<td>Metrics</td>
<td>Acc</td>
<td>F1</td>
<td>SCC</td>
<td>MCC</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
</tr>
</tbody>
</table>

Table 1: Eight datasets used in this paper form GLUE benchmark. Acc stands for Accuracy, SCC stands for Spearman Correlation Coefficient and MCC stands for Matthews Correlation Coefficient.

## C Appendix C. Hyper-parameters and Experimental Details of Different Pre-trained Language Models

In this paper, we investigate the performance of DPS on five distinctive and widely used large-scale pre-trained language models, namely BERT [Devlin et al. 2018], RoBERTa [Liu et al. 2019], ELECTRA [Clark et al. 2020], BART [Lewis et al. 2019], and DeBERTa [He et al. 2021]. BERT is the first Transformer [Vaswani et al. 2017] encoder based pre-trained language model with Mask Language Modeling and Next Sentence Prediction pre-training tasks. RoBERTa is similar to BERT in terms of model architecture but is only pre-trained on the Mask Language Modeling task only, but for longer and on more data. ELECTRA is a BERT-like model trained to distinguish tokens generated by masked language model from tokens drawn from the natural distribution. BART is a sequence-to-sequence model trained as a denoising autoencoder. DeBERTa improves Transformer-based pre-trained model with disentangled attention mechanism and enhanced mask decoder. Table 2 summarizes the hyper-parameters of each model for each dataset. We use AdamW [Loshchilov and Hutter 2019] optimizer, clip the gradients with a maximum norm of 1, and the maximum sequence length is set as 128. We use mixed precision training to speed up the experimental process. We conduct all the experiments on a single Tesla-V100 GPU (32G).

<table>
<thead>
<tr>
<th>Model</th>
<th>Datasets</th>
<th>Batch Size</th>
<th>Learning Rate</th>
<th>Training Epochs/Steps</th>
<th>Warmup Ratio/Steps</th>
<th>LLRD</th>
</tr>
</thead>
<tbody>
<tr>
<td>BERT</td>
<td>all</td>
<td>16</td>
<td>2e-5</td>
<td>3 epochs</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RoBERTa</td>
<td>RTE</td>
<td>16</td>
<td>2e-5</td>
<td>2036 steps</td>
<td>122 steps</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>MRPC</td>
<td>16</td>
<td>1e-5</td>
<td>2296 steps</td>
<td>137 steps</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>STS-B</td>
<td>16</td>
<td>2e-5</td>
<td>3598 steps</td>
<td>214 steps</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>CoLA</td>
<td>16</td>
<td>1e-5</td>
<td>5336 steps</td>
<td>320 steps</td>
<td>-</td>
</tr>
<tr>
<td>ELECTRA</td>
<td>RTE</td>
<td>32</td>
<td>5e-5</td>
<td>10 epochs</td>
<td>10%</td>
<td>0.9</td>
</tr>
<tr>
<td></td>
<td>MRPC</td>
<td>32</td>
<td>5e-5</td>
<td>3 epochs</td>
<td>10%</td>
<td>0.9</td>
</tr>
<tr>
<td></td>
<td>STS-B</td>
<td>32</td>
<td>5e-5</td>
<td>10 epochs</td>
<td>10%</td>
<td>0.9</td>
</tr>
<tr>
<td></td>
<td>CoLA</td>
<td>32</td>
<td>5e-5</td>
<td>3 epochs</td>
<td>10%</td>
<td>0.9</td>
</tr>
<tr>
<td>BART</td>
<td>RTE</td>
<td>32</td>
<td>1e-5</td>
<td>3 epochs</td>
<td>10%</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>MRPC</td>
<td>64</td>
<td>2e-5</td>
<td>3 epochs</td>
<td>10%</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>STS-B</td>
<td>32</td>
<td>2e-5</td>
<td>3 epochs</td>
<td>10%</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>CoLA</td>
<td>64</td>
<td>2e-5</td>
<td>3 epochs</td>
<td>10%</td>
<td>-</td>
</tr>
<tr>
<td>BEBERTA</td>
<td>RTE</td>
<td>32</td>
<td>1e-5</td>
<td>6 epochs</td>
<td>50 steps</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>MRPC</td>
<td>32</td>
<td>1e-5</td>
<td>6 epochs</td>
<td>50 steps</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>STS-B</td>
<td>32</td>
<td>7e-6</td>
<td>4 epochs</td>
<td>100 steps</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>CoLA</td>
<td>32</td>
<td>7e-6</td>
<td>6 epochs</td>
<td>100 steps</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 2: Fine-tuning hyper-parameters of BERT and its variants as reported in the official repository of each model for best practice. Note that Layer-wise Learning Rate Decay (LLRD) [Howard and Ruder 2018] is a method that applies higher learning rates for top layers and lower learning rates for bottom layers. This method is applied by ELECTRA when fine-tuning downstream tasks.
D Appendix D. Experimental Details for Different Fine-tuning Methods

The following is our hyperparameter search space for different fine-tuning regularization methods:

- **Mixout**: We grid search Mixout probability \( p \in \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8\} \).
- **R3F**: We grid search Noise Types \( \in \{N, U\} \), \( \sigma \in \{1e-5\} \), \( \lambda \in \{0.1, 0.5, 1.0, 5.0\} \).
- **Re-init**: We grid search \( L \in \{1, 2, 3, 4, 5, 6, 7\} \).
- **CHILD-TUNING\(_D\)**: We grid search \( \text{CHILD-TUNING}\(_D\) \( \in \{0.1, 0.2, 0.3\} \), and learning rate \( \text{lr} \in \{2e-5, 4e-5, 6e-5, 8e-5, 1e-4\} \).
- **R-Dropout**: We grid search Dropout probability \( p \in \{0.1\} \) and \( \alpha \in \{0.1, 0.5, 1, 3, 5\} \).
- **DPS Dense**: We grid search reserved probability \( p \in \{0.1, 0.2, 0.3, 0.4, 0.5\} \) and update ratio \( \text{ur} \in \{0.05, 0.1, 0.2\} \).
- **DPS Mix**: We grid search reserved probability \( p \in \{0.1, 0.2, 0.3, 0.4, 0.5\} \), and update ratio \( \text{ur} \in \{0.05, 0.1, 0.2\} \).

E Appendix E. Mapping Strategy

We train DPS on SNLI and MNLI datasets respectively and evaluate several different target NLI datasets. The SNLI and MNLI datasets contain three labels: entailment, neutral, and contradiction, however, some datasets have only two labels (SciTail, RTE). The SciTail dataset contains two labels: entailment, neutral, and we map the predicted labels neutral and contradiction to neutral, following Mahabadi et al. [2021]. We conduct the same process for RTE.

F Appendix F. Memory Consumption

In addition to time usage, we further analyze memory consumption of various approaches when fine-tuning BERT\(_\text{LARGE}\). The table below shows the results. DPS Dense requires extra memory because it needs to store Gradient Accumulation Matrix (GAM), DPS Mix requires more memory than DPS Dense because it needs to store Frequency Accumulation Matrix (FAM) and GAM. It is worth noting that since the size of GAM and FAM is fixed, the additional memory consumption introduced by DPS does not increase as the batch size increases. Therefore, as the batch size increases, the percentage of additional memory consumption decreases for DPS compared to vanilla fine-tuning. Overall, DPS does introduce additional memory overhead, but we believe that the memory consumption for DPS is acceptable compared to other regularization methods.

<table>
<thead>
<tr>
<th>Batch Size</th>
<th>Vanilla</th>
<th>Mixout</th>
<th>R3F</th>
<th>R-Dropout</th>
<th>CHILD-TUNING(_D)</th>
<th>DPS Dense</th>
<th>DPS Mix</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>10.6G</td>
<td>12.9G</td>
<td>17.7G</td>
<td>17.7G</td>
<td>10.9G</td>
<td>16.3G</td>
<td>21.1G</td>
</tr>
<tr>
<td>32</td>
<td>14.3G</td>
<td>16.3G</td>
<td>28.3G</td>
<td>28.3G</td>
<td>14.7G</td>
<td>20.0G</td>
<td>24.7G</td>
</tr>
</tbody>
</table>

Table 3: Memory consumption for DPS and multiple fine-tuning regularization methods.

References


Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In EMNLP, 2013.


