\( p \)-Laplacian Adaptation for Generative Pre-trained Vision-Language Models

Haoyuan Wu*, Xinyun Zhang*, Peng Xu, Peiyu Liao, Xufeng Yao, Bei Yu

Department of Computer Science and Engineering
The Chinese University of Hong Kong

Feb. 08, 2024
Introduction
By leveraging massive amounts of unlabeled data during training, pre-trained vision-language models can learn highly performant and generalizable representations, leading to improvements on various downstream tasks.

As model sizes continue to grow rapidly, fine-tuning is increasingly affected by the parameter-efficiency issue. To address this challenge, researchers proposed parameter-efficient fine-tuning to achieve high parameter efficiency and demonstrated promising results on various downstream tasks.
Given query $Q \in \mathbb{R}^{N_1 \times d_k}$, key $K \in \mathbb{R}^{N_2 \times d_k}$ and value $V \in \mathbb{R}^{N_2 \times d_v}$, attention aggregates the features by:

$$\text{Attn}(Q, K, V) = MV, \quad (1)$$

where

$$M = \text{softmax} \left( \frac{QK^\top}{\sqrt{d_k}} \right) \quad (2)$$

represents the attention weights, $N_1$ and $N_2$ are the number of the query and key/value features, respectively.
An adapter is a small learnable module containing two matrices $W_{\text{down}} \in \mathbb{R}^{l_1 \times l_2}$, $W_{\text{up}} \in \mathbb{R}^{l_2 \times l_1}$ and a non-linear function $\sigma(\cdot)$, where $l_1$ and $l_2$ are the feature dimensions in pre-trained models and the hidden dimension in adapter (usually $l_2 < l_1$). Given a feature $U \in \mathbb{R}^{N \times l_1}$ in the pre-trained model, the adapter encoding process can be represented as:

$$U' = \sigma(UW_{\text{down}})W_{\text{up}} + U.$$  

(3)
From Equation (3) and Equation (1), we can formulate the features sequentially encoded by attention and adapter as:

$$U' = \sigma(MVW_v W_o W_{down})W_{up} + MVW_v W_o,$$

where $M \in \mathbb{R}^{N_1 \times N_2}$ is the attention matrix computed by the transformed query $QW_q$ and key $KW_k$ using Equation (2).
Modeling adapter as graph message passing

We define the augmented value feature \( \tilde{V} \) which concatenates the transformed query and value and the augmented attention matrix \( \tilde{M} \) as

\[
\tilde{V} = \begin{bmatrix} QW_q \\ VW_v \end{bmatrix}, \quad \tilde{M} = \begin{bmatrix} 0 \\ M^\top \\ 0 \end{bmatrix}.
\]  

(5)
Modeling adapter as graph message passing

Illustration of the generation of the bipartite attention graph $G_{attn}$.

Defining the projected augmented value feature $\hat{V} = \tilde{V}W_o$, with the augmented attention mechanism, we can further define the augmented adapter encoding process by:

$$\tilde{U}' = \sigma(\tilde{M}\hat{V}W_{down})W_{up} + \tilde{M}\hat{V}. \tag{6}$$

Comparing Equation (4) and Equation (6), we indicate that the adapter encoding process and the augmented one are equal. Since $\tilde{M}$ is a square and symmetric matrix, we can regard it as the adjacency matrix of the attention graph $G_{attn}$. 
The t-SNE\(^2\) visualization of the features in the projected query and value space for self- and cross-attention. The VLM is BLIP\(_\text{CapFilt-L}\)^3 and data come from COCO Captions\(^4\).


\(^3\)Junnan Li et al. (2022). “Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation”. In: Proc. ICML.

Method
For $p$-adapter, we take the attention matrix $M$ and the projected augmented value feature $\hat{V}$, as the output of attention. Note that this transformation does not alter any learned parameters in attention. Then, we augment the attention matrix to $\tilde{M}$, as shown in Equation (5). Following $p$-Laplacian message passing, we normalize the augmented attention matrix by:

$$\bar{M}_{i,j} = \frac{\tilde{M}_{i,j}}{\tilde{D}_{i,i}} \hat{V}_{i;:} - \frac{\tilde{M}_{i,j}}{\tilde{D}_{j,j}} \hat{V}_{j;:}$$

(7)

where $\tilde{D}$ is the degree matrix of $\tilde{M}$. Further, we can aggregate the features using the calibrated attention matrix $\bar{M}$ by:

$$\bar{U} = \bar{\alpha} \tilde{D}^{-1/2} \tilde{M} \tilde{D}^{-1/2} \hat{V} + \bar{\beta} \hat{V},$$

(8)

where $\bar{\alpha}$ and $\bar{\beta}$ are calculated according to the algorithm in $p$-Laplacian message passing. With the aggregated feature $\bar{U}$, we encode it with the learnable adapter weights by:

$$\bar{U}' = \sigma(\bar{U}W_{\text{down}})W_{\text{up}} + \bar{U}.$$  

(9)
Overall architecture of $p$-adapter
Experiments
For VQA, we consider it as an answer generation problem. We test our model on VQA2.0\textsuperscript{5} with the widely-used Karpathy split and VizWizVQA\textsuperscript{6}.

For VE, we adopt SNLI-VE\textsuperscript{7} as the evaluation benchmark.

For image captioning, we conduct extensive experiments on three benchmarks, i.e., COCO Captions\textsuperscript{8} with Karpathy split\textsuperscript{9}, TextCaps\textsuperscript{10}, and VizWizCaps\textsuperscript{11}.


Implementation details

1. Our experiments are implemented in PyTorch\textsuperscript{12} and conducted on 8 Nvidia 3090 GPUs.

2. We validate our method on two generative pre-trained VLMs, BLIP\textsubscript{CapFilt-L}\textsuperscript{13} and mPLUG\textsubscript{ViT-B}\textsuperscript{14}.

---


\textsuperscript{13} Junnan Li et al. (2022). “Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation”. In: \textit{Proc. ICML}.

### Method Comparison

<table>
<thead>
<tr>
<th>Method</th>
<th>Updated Params (%)</th>
<th>VQA2.0 Acc. (%)</th>
<th>VizWizVQA Test-dev Acc. (%)</th>
<th>SNLI_VE Test-P Acc. (%)</th>
<th>COCOCaps Karpathy Test BLEU@4</th>
<th>CIDEr</th>
<th>TextCaps Test-dev BLEU@4</th>
<th>CIDEr</th>
<th>VizWizCaps Test-dev BLEU@4</th>
<th>CIDEr</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full fine-tuning</td>
<td>100.00</td>
<td>70.56</td>
<td>36.52</td>
<td>78.35</td>
<td>39.1</td>
<td>128.7</td>
<td>27.1</td>
<td>91.6</td>
<td>45.7</td>
<td>170.0</td>
<td>76.40</td>
</tr>
<tr>
<td>Prefix tuning</td>
<td>0.71</td>
<td>60.49</td>
<td>22.45</td>
<td>71.82</td>
<td>39.4</td>
<td>127.7</td>
<td>24.8</td>
<td>80.0</td>
<td>40.6</td>
<td>153.3</td>
<td>68.95</td>
</tr>
<tr>
<td>LoRA</td>
<td>0.71</td>
<td>66.57</td>
<td>33.39</td>
<td>77.36</td>
<td>38.3</td>
<td>128.3</td>
<td>24.6</td>
<td>82.2</td>
<td>41.3</td>
<td>154.3</td>
<td>71.81</td>
</tr>
<tr>
<td>Adapter</td>
<td>6.39</td>
<td>69.53</td>
<td>35.37</td>
<td>78.85</td>
<td>38.9</td>
<td>128.8</td>
<td>25.4</td>
<td>86.7</td>
<td>43.3</td>
<td>160.5</td>
<td>74.15</td>
</tr>
<tr>
<td>$p$-Adapter (Ours)</td>
<td>6.39</td>
<td>70.39</td>
<td>37.16</td>
<td>79.40</td>
<td>40.4</td>
<td>130.9</td>
<td>26.1</td>
<td>87.0</td>
<td>44.5</td>
<td>164.1</td>
<td>75.54</td>
</tr>
</tbody>
</table>

**Table:** The main results on various datasets for full fine-tuning, adapter\(^{15}\), prefix tuning\(^{16}\), LoRA\(^{17}\), and our proposed $p$-adapter. We bold the scores for full fine-tuning and the highest scores separately for approaches with PETL methods.


\(^{17}\)Edward J Hu et al. (2022). “Lora: Low-rank adaptation of large language models”. In: Proc. ICLR.
**Comparison with transfer learning methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Updated Params (%)</th>
<th>VQA2.0 Acc. (%)</th>
<th>VizWizVQA Acc. (%)</th>
<th>SNLI_VE Acc. (%)</th>
<th>COCOCaps Acc. (%)</th>
<th>TextCaps Acc. (%)</th>
<th>VizWizCaps Acc. (%)</th>
<th>BLEU@4</th>
<th>CIDEr</th>
<th>BLEU@4</th>
<th>CIDEr</th>
<th>BLEU@4</th>
<th>CIDEr</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full fine-tuning</td>
<td>100.00</td>
<td>70.91</td>
<td>59.79</td>
<td>78.72</td>
<td>40.4</td>
<td>134.8</td>
<td>23.6</td>
<td>74.0</td>
<td>42.1</td>
<td>157.5</td>
<td>75.76</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Prefix tuning</td>
<td>0.71</td>
<td>60.95</td>
<td>47.42</td>
<td>72.11</td>
<td>39.8</td>
<td>133.5</td>
<td>18.8</td>
<td>51.9</td>
<td>35.5</td>
<td>135.6</td>
<td>66.18</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LoRA</td>
<td>0.71</td>
<td>66.67</td>
<td>52.49</td>
<td>75.29</td>
<td>39.4</td>
<td>129.4</td>
<td>21.0</td>
<td>64.4</td>
<td>39.5</td>
<td>146.0</td>
<td>70.46</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Adapter</td>
<td>6.39</td>
<td>70.65</td>
<td>56.50</td>
<td>78.56</td>
<td>40.3</td>
<td>134.7</td>
<td>22.9</td>
<td>71.5</td>
<td>41.9</td>
<td>155.6</td>
<td>74.73</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(p)-Adapter (Ours)</td>
<td>6.39</td>
<td>71.36</td>
<td>58.08</td>
<td>79.26</td>
<td>40.4</td>
<td>135.3</td>
<td>23.2</td>
<td>73.3</td>
<td>43.1</td>
<td>160.1</td>
<td>76.01</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>mplUG_{\text{Vit-B}}</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full fine-tuning</td>
</tr>
<tr>
<td>Prefix tuning</td>
</tr>
<tr>
<td>LoRA</td>
</tr>
<tr>
<td>Adapter</td>
</tr>
<tr>
<td>(p)-Adapter (Ours)</td>
</tr>
</tbody>
</table>

**Table:** The main results on various datasets for full fine-tuning, adapter\(^{18}\), prefix tuning\(^{19}\), LoRA\(^{20}\), and our proposed \(p\)-adapter. We bold the scores for full fine-tuning and the highest scores separately for approaches with PETL methods.


\(^{19}\)Xiang Lisa Li and Percy Liang (2021). “Prefix-Tuning: Optimizing Continuous Prompts for Generation”. In: [*Proc. ACL*].

\(^{20}\)Edward J Hu et al. (2022). “Lora: Low-rank adaptation of large language models”. In: [*Proc. ICLR*].
## Ablation studies

<table>
<thead>
<tr>
<th>GNN</th>
<th>VQA2.0 Acc. (%)</th>
<th>SNLI_VE Acc. (%)</th>
<th>COCOCaps BLEU@4</th>
<th>COCOCaps CIDEr</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>GCN</td>
<td>69.53</td>
<td>78.85</td>
<td>38.9</td>
<td>128.8</td>
<td>79.02</td>
</tr>
<tr>
<td>APPNP</td>
<td>70.22</td>
<td>79.03</td>
<td>39.4</td>
<td>129.1</td>
<td>79.44</td>
</tr>
<tr>
<td>GCNII</td>
<td>70.13</td>
<td>79.12</td>
<td>39.7</td>
<td>129.7</td>
<td>79.66</td>
</tr>
<tr>
<td>pGNN</td>
<td><strong>70.39</strong></td>
<td><strong>79.40</strong></td>
<td><strong>40.4</strong></td>
<td><strong>130.9</strong></td>
<td><strong>80.27</strong></td>
</tr>
</tbody>
</table>

**Table:** Ablation study on the graph neural networks.
Visualization

To validate the effectiveness of $p$-adapter, we visualize\(^{21}\) the cross-attention weights at the last transformer layer on some VQA\(^{22}\) data.

We take the [CLS] token as the query since it represents the whole question and plot the attention weights on the image features in the key/value space.


Conclusion
1. We first propose a new modeling framework for adapter tuning\textsuperscript{23} after attention modules in pre-trained VLMs. Within this framework, we can identify the heterophilic nature of the attention graphs, posing challenges for vanilla adapter tuning\textsuperscript{24}.

2. To mitigate this issue, we propose a new adapter architecture, $p$-adapter, appended after the attention modules. Inspired by $p$-Laplacian message passing\textsuperscript{25}, $p$-adapters re-normalize the attention weights using node features and aggregate the features with the calibrated attention matrix.

3. Extensive experimental results validate our method’s significant superiority over other PETL methods on various VL tasks.


\textsuperscript{25}Guoji Fu, Peilin Zhao, and Yatao Bian (2022). “$p$-Laplacian Based Graph Neural Networks”. In: Proc. ICML.
THANK YOU!