AdaKron: an Adapter-based Parameter Efficient Model Tuning with Kronecker Product

Marco Braga, Alessandro Raganato, Gabriella Pasi
University of Milano-Bicocca
Department of Informatics, Systems and Communication
m.braga@campus.unimib.it,
alessandro.raganato@unimib.it, gabriella.pasi@unimib.it

Abstract

The fine-tuning paradigm has been widely adopted to train neural models tailored for specific tasks. However, the recent upsurge of Large Language Models (LLMs), characterized by billions of parameters, has introduced profound computational challenges to the fine-tuning process. This has fueled intensive research on Parameter-Efficient Fine-Tuning (PEFT) techniques, usually involving the training of a selective subset of the original model parameters. One of the most used approaches is Adapters, which add trainable lightweight layers to the existing pretrained weights. Within this context, we propose AdaKron, an Adapter-based fine-tuning with the Kronecker product. In particular, we leverage the Kronecker product to combine the output of two small networks, resulting in a final vector whose dimension is the product of the dimensions of the individual outputs, allowing us to train only 0.55% of the model's original parameters. We evaluate AdaKron performing a series of experiments on the General Language Understanding Evaluation (GLUE) benchmark, achieving results in the same ballpark as recent state-of-the-art PEFT methods, despite training fewer parameters.

Keywords: Adapters, Kronecker Product, Parameter Efficient Tuning

1. Introduction

The conventional approach to fine-tuning for downstream tasks requires the training of all parameters of a neural model (Devlin et al., 2019; Liu et al., 2019; Raffel et al., 2020). However, with the recent increase of Large Pretrained Language Models (PLMs) reaching billions of parameters (Min et al., 2023), the traditional fine-tuning process has become challenging due to large memory requirements. In response to this resource bottleneck, Parameter-efficient Fine-Tuning (PEFT) techniques have emerged as a new paradigm: these methods have been designed with the explicit goal of training only a fraction of the original model parameters while fine-tuning the model to a downstream task, and keeping performance levels comparable to traditional fine-tuning. These PEFT methods can be divided into two categories: i) those that add a new small set of parameters to be trained on, e.g. Adapter-based methods (Bapna et al., 2019; Houlsby et al., 2019; Pfeiffer et al., 2020a,b, 2021; Wang et al., 2022; Li et al., 2023), LoRA-based techniques (Hu et al., 2022; Zhang et al., 2023; Xu et al., 2023), Diff-Pruning (Guo et al., 2021) or UNIPELT (Mao et al., 2022), and ii) those that train a small subset of the original model parameters, e.g. BitFit (Ben Zaken et al., 2022).

In this work, we focus on the first category, specifically on Adapter-based methods. Adapters have been widely used in different tasks, such as Natural Language Understanding and Inference (Houlsby et al., 2019; Edalati et al., 2022; Pfeiffer et al., 2021), Text Generation (Wang et al., 2022; Xu et al., 2022), Named Entity Recognition (Pfeiffer et al., 2020b; Ansell et al., 2021), Retrieval-based systems (Braga et al., 2023; Kasela et al., 2024a) and Cross-Lingual Transfer (Pfeiffer et al., 2020b), to name a few.

In general, Adapters consist of two fully connected layers: the first one is a down projection of the input vector into an intermediate dimension, which is followed by a non-linear activation function and by an up projection to the hidden dimension of the model. The capability of an Adapter depends on its intermediate dimension, and recent empirical studies (Chen et al., 2022) suggest that low-dimensional Adapter modules can give better performances than high ones. Inspired by these studies, we define a new PEFT technique called AdaKron, where the Kronecker product is applied to the output vectors of two small Feed-Forward Networks (FFNs), which results in the creation of a new vector whose dimensionality is the product of the dimensions of the two individual FFNs. The Kronecker product has been recently used to improve PEFT techniques to reduce the number of parameters as well as floating point operations (Jiang and Zheng, 2023; Hameed et al., 2021; Edalati et al., 2022; Karimi Mahabadi et al., 2021). These approaches parameterize weights matrices as Kronecker products of low-dimensional matrices, e.g. Kronecker decomposition is used for BERT (Tahaei et al., 2021) and GPT-2 compression (Edalati et al., 2021). In contrast, we use the Kronecker product at a vector level, not at weight matrix one. The use of the Kronecker product al-

LREC-COLING 2024, pages 350–357

lows us to train fewer parameters in the down projection layer compared to a single FFN.

Overall, we define a new PEFT method, AdaKron, which combines the Adapter modules and Kronecker product, showing that by training only 0.55% of parameters, we reach performance on par with recent state-of-the-art PEFT methods that require more parameters to train. We make our code publicly available. [1]

2. Methodology

In this section, we first briefly present Adapters and the Kronecker product (Section 2.1). Next, in Section 2.2, we describe in detail our proposed method, AdaKron.

2.1. Preliminaries

Adapters (Houlsby et al., 2019; Pfeiffer et al., 2020a) usually consist of a down projection Feed-Forward layer $\mathbf{W}_{down} \in \mathbb{R}^{r \times d}$ to project the input d-dimensional vector to an intermediate dimension $r \ll d$, followed with a non-linear activation function and an up Feed-Forward projection $\mathbf{W}_{up} \in \mathbb{R}^{d \times r}$ to the original dimension d, coupled with a residual connection. Adapters have been studied (Chen et al., 2022) and successfully used in different NLP tasks, such as Text Generation (Wang et al., 2022; Xu et al., 2022) or Named Entity Recognition (NER) (Ansell et al., 2021).

The Kronecker product (Henderson et al., 1983), denoted as $\otimes$, is an operation between two matrices, $A \in \mathbb{R}^{m \times n}$ and $B \in \mathbb{R}^{p \times q}$, whose result is a block matrix of dimension $m \cdot p \times n \cdot q$, where each block is the product between an element of A and the entire matrix B:

$$A \otimes B := \begin{bmatrix} a_{11}B & a_{12}B & \dots & a_{1n}B \\ a_{21}B & a_{22}B & \dots & a_{2n}B \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1}B & a_{m2}B & \dots & a_{mn}B \end{bmatrix} \in \mathbb{R}^{m \cdot p \times n \cdot q}$$

When applied between vectors, $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^m$, the Kronecker product yields $y \otimes x = vec(xy^T)$ where vec is a mathematical operation that stacks the columns of the matrix into a vector. Consequently, the Kronecker product between two vectors is represented as:

$$y \otimes x = vec(xy^T) = \begin{bmatrix} y_1 x & \dots & y_m x \end{bmatrix} \in \mathbb{R}^{n \cdot m}. \quad (1)$$

The Kronecker product has recently been used for parameter-efficient training in Adapter layers or attention matrices (Jiang and Zheng, 2023; Karimi Mahabadi et al., 2021; Tahaei et al., 2021; Edalati et al., 2021), usually adopted as a method to decompose FFN weight matrices into smaller low-dimensional ones, which are then multiplied using the Kronecker product.

[1] https://github.com/DetectiveMB/AdaKron

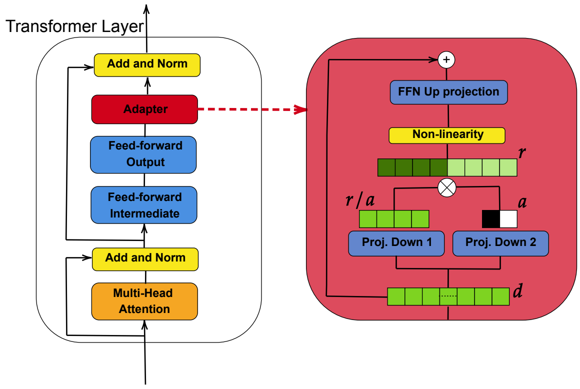

Figure 1: Architecture of a Transformer layer with the integration of AdaKron. **Left**: We add the Adapter module to a Transformer layer following Pfeiffer et al. (2021). **Right**: Each AdaKron module consists of two small Feed-Forward Networks (Proj. Down 1 and Proj. Down 2) and one Feed-Forward up projection matrix. The input $d-$dimensional vector is fed into both the down projection layers, which have two different output dimensions, $\frac{r}{a}$ and a. The outputs of the two small Feed-Forward down projections are multiplied using the Kronecker product to obtain a $r-$dimensional vector.

2.2. AdaKron

Our approach introduces the Kronecker product in an Adapter module (Pfeiffer et al., 2020a), as shown in Figure 1. But, in contrast to Karimi Mahabadi et al. (2021), where the Kronecker product has been used to decompose the down and up projection weight matrices of the Adapter, our method employs the Kronecker product between the output vectors of two FFNs, which compose the down projection of the Adapter, while the up projection remains a unique FFN layer. The final output of the down projection, as shown in Equation 1, is a weighted concatenation of one of the two vectors.

Let $y = \mathbf{W}x + b$ be the down projection Feed-Forward layer, where $x \in \mathbb{R}^d$, $y \in \mathbb{R}^r$, $\mathbf{W} \in \mathbb{R}^{r \times d}$ and $b \in \mathbb{R}^r$. We define two different down projection layers, whose final outputs are

$$y_i = \mathbf{W}_i x + b_i, \; i = 1, 2$$

where $\mathbf{W}_i \in \mathbb{R}^{r_i \times d}$, $b_i \in \mathbb{R}^{r_i}$, $y_i \in \mathbb{R}^{r_i}$ and $r_1, r_2 \ll d$. Next, we apply the Kronecker product and the GELU function (Hendrycks and Gimpel, 2016) to obtain

$$output = GELU(y_2 \otimes y_1) \in \mathbb{R}^{r_1 \cdot r_2}$$

Model	# Params (M)	MNLI Acc	QNLI Acc	SST2 Acc	QQP F1	MRPC F1	CoLa Mcc	RTE Acc	STS-B Pearson	Avg.
Fine-Tuning	110	83.2	90.0	91.6	87.4	90.9	62.1	66.4	89.8	82.7
Houlsby Adapter†	0.9	83.1	90.6	91.9	86.8	89.9	61.5	71.8	88.6	83.0
BitFit$^{\diamondsuit}$	0.1	81.4	90.2	92.1	84.0	90.4	58.8	72.3	89.2	82.3
Prefix-tuning†	0.2	81.2	90.4	90.9	83.3	91.3	55.4	76.9	87.2	82.1
LoRA†	0.3	82.5	89.9	91.5	86.0	90.0	60.5	71.5	85.7	82.2
UNIPELT (AP)†	1.1	83.4	90.8	91.9	86.7	90.3	61.2	71.8	88.9	83.1
UNIPELT (APL)†	1.4	83.9	90.5	91.5	85.5	90.2	58.6	73.7	88.9	83.5
AdaMix Adapter$^{\triangle}$	0.9*	84.7	91.5	92.4	87.6	92.4	62.9	74.7	89.9	84.5
Pfeiffer Adapter$_{48}$	0.9	83.3	91.1	92.0	87.5	90.7	60.3	67.6	89.6	82.7
Pfeiffer Adapter$_{32}$	0.6	83.2	90.9	92.1	87.4	90.1	60.5	69.1	89.4	82.8
AdaKron$_{48}$	0.6	83.5	91.1	92.0	87.1	90.8	61.1	73.8	89.4	83.6
AdaKron$_{32}$	0.4	83.7	90.9	92.2	87.1	89.5	60.7	74.1	89.5	83.5

Table 1: Main results on the GLUE development set with BERT-base. †, $^{\diamondsuit}$ and $^{\triangle}$ denote that the reported results are taken from Mao et al. (2022), Ben Zaken et al. (2022) and Wang et al. (2022) respectively. *# Params (M)* refers to the number of updated parameters (in Millions). * denotes that AdaMix requires training twice the number of parameters because, during each training phase, each batch is processed two times. Acc, Mcc and Pearson refer to Accuracy, Matthews correlation coefficient and Pearson correlation, respectively.

that is then fed to the up projection layer. Thus, the final intermediate dimension of our Adapter is $r_1 \cdot r_2 \ll d$.

This design choice is informed by recent empirical studies (Chen et al., 2022), which suggest that lower-dimensional adapter modules can give better performances compared to higher-dimensional ones.

The use of the Kronecker product allows us to train fewer parameters in the down projection layer compared to a single FFN layer. Specifically, instead of a down projection layer with an output dimension of r, we have two FFNs with output dimensions equal to $r_1 = \frac{r}{a}$ and $r_2 = a$, where $a \leq r$ is a reduction factor that corresponds to the number of weighted repetition of the $\frac{r}{a}$-dimensional vector. Since $r_1 \in \mathbb{N}$, a must be a factor of r. Finally, given d the input dimensionality of the network, the number of parameters is reduced from $r \cdot d$ to $(\frac{r}{a} + a) \cdot d$.

3. Experiments

In this section, we first describe the datasets and baselines of our experimental evaluation (Section 3.1). Then, in Section 3.2 we present and discuss the results of AdaKron, followed by an ablation study (Section 3.3).

3.1. Experimental Setup

Benchmark We perform experiments on the General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2018), which involves eight types of Natural Language Understanding tasks including Linguistic Acceptability (CoLA (Warstadt et al., 2019)), Sentiment Analysis (SST-2 (Socher et al., 2013)), Similarity and Paraphrase tasks (MRPC (Dolan and Brockett, 2005), STS-B (Cer et al., 2017), QQP (Wang et al., 2018)), and Natural Language Inference (MNLI (Williams et al., 2018), QNLI (Rajpurkar et al., 2016), RTE (Dagan et al., 2006; Bar-Haim et al., 2006; Giampiccolo et al., 2007; Bentivogli et al., 2009)). Following prior studies (Houlsby et al., 2019; Devlin et al., 2019), we do not include the WNLI dataset (Levesque et al., 2012).[2]

Baselines We compare AdaKron to full model Fine-Tuning and several PEFT methods, all applied to BERT-base (Devlin et al., 2019) (12-layer, 768-hidden, 12-heads, 110M parameters)[3] and RoBERTa-large (Liu et al., 2019) (24-layer, 1024-hidden, 16-heads, 355M parameters)[4]. We compare our model against **Houlsby Adapter** (Houlsby et al., 2019); **AdaMix**, which incorporates the Mixture of Experts paradigm (Fedus et al., 2022) in an adapter layer (Wang et al., 2022); **BitFit** (Ben Zaken et al., 2022), which trains only a subset of bias-terms; **Prefix-Tuning** (Li and Liang, 2021), which prepends several task-specific vectors to the input of multi-head attention; **LoRA** (Hu et al., 2022), which introduces low-rank matrices and combines them with the original matrices in the multi-head attention layer; **UNIPELT** (Mao et al., 2022), which incorporates existing methods (Adapters, LoRa and Prefix-Tuning) as submod-

[2] See (12) in gluebenchmark.com/faq.
[3] bert-base-uncased
[4] roberta-large

Model	# Params (M)	MNLI Acc	QNLI Acc	SST2 Acc	QQP Acc	MRPC Acc	CoLa Mcc	RTE Acc	STS-B Pearson	Avg.
Fine-Tuning	355	90.2	94.7	96.4	92.2	90.9	68	86.6	92.4	88.9
Houlsby Adapter†	6	89.9	94.7	96.2	92.1	88.7	66.5	83.4	91	87.8
Houlsby Adapter†	0.8	90.3	94.7	96.3	91.5	87.7	66.3	72.9	91.5	86.4
Pfeiffer Adapter†	3	90.2	94.8	96.1	91.9	90.2	68.3	83.8	92.1	88.4
Pfeiffer Adapter†	0.8	90.5	94.8	96.6	91.7	89.7	67.8	80.1	91.9	87.9
LoRA†	0.8	90.6	94.8	96.2	91.6	90.2	68.2	85.2	92.3	88.6
AdaMix Adapter$^\triangle$	0.8*	90.9	95.4	97.1	89.8	94.1	70.2	89.2	92.4	89.9
AdaKron$_{16}$	0.6	90.2	94.8	96.9	91	90.9	69.2	87.4	92.1	89.1
AdaKron$_{32}$	1.0	90.2	94.4	96.1	87.8	93.1	69.9	86.1	92.2	88.7

Table 2: Main results on the GLUE development set with RoBERTa-Large. † and $^\triangle$ denote that the reported results are taken from Hu et al. (2022) and Wang et al. (2022) respectively. *# Params (M)* refers to the number of updated parameters (in Millions). * denotes that AdaMix requires training twice the number of parameters because, during each training phase, each batch is processed two times. Acc, Mcc and Pearson refer to Accuracy, Matthews correlation coefficient and Pearson correlation, respectively.

ules and automatically learn to activate, through a gate mechanism, the appropriate submodules for a given task. Moreover, we include two Adapter-based baselines, i.e. **Pfeiffer Adapter** (Pfeiffer et al., 2021), using our intermediate dimensions, i.e. 48 and 32, but without the application of Kronecker product in the down projection layer.

AdaKron hyperparameters We implement AdaKron in Pytorch (Paszke et al., 2019), using an RTX 8000 GPU for our experiments, following the hyperparameter configuration in Wang et al. (2022). We use Adam with weight decay (Loshchilov and Hutter, 2019) to optimize our models. Adakron uses intermediate adapter dimensions of 48 and 32, the dimensions of two down projections layer are $r_1 = 12$, $a = 4$ and $r_1 = 8$, $a = 4$ respectively. The number of FFNs in the down projection layers is set to 2 for all the tasks and experiments.

3.2. Results and Discussion

Table 1 shows the performance attained by our models and several recent PEFT models, using BERT-base as a PLM, on the GLUE development dataset. Our direct competitors, i.e., Pfeiffer Adapter$_{48}$ and Pfeiffer Adapter$_{32}$, score almost one point lower than Adakron on average. This comparison shows the effectiveness of our proposed PEFT method despite having a reduced number of trainable parameters. Across the board, most of the PEFT models achieve similar results. AdaKron shows on average better performance compared to the full Fine-Tuning, and Houlsby Adapter, achieving improvements of 1.0, and 0.7 average score, respectively. Moreover, AdaKron achieves an average one point improvement over smaller PEFT methods like BitFit, Prefix-tuning, and LoRA. Interestingly, our approach also achieves better performance than UNIPELT, which uses twice the amount of parameters compared to AdaKron. It is worth noting that even though Adamix achieves the best average result compared to all PEFT methods, it adds 0.9 million parameters to the original PLM during the inference stage only, however, it updates a total of 1.8 million parameters during training BERT. Nonetheless, AdaKron closely matches AdaMix's performance levels on specific tasks such as QQP, STS-B and QNLI. On average, AdaKron is one point lower than AdaMix, but training only one-third of AdaMix's parameters.

Additionally, in Table 2 we report the average performance on the GLUE development set using RoBERTa-large as a PLM.In this case, to keep the same number of trainable parameters as in the BERT-base evaluation, we use 16 as our intermediate dimension, i.e. $r_1 = 4$, and $a = 4$. Overall, once again, AdaKron proves to achieve consistent competitive performance, showing to be a competitive alternative compared to its counterparts, despite its reduced parameter numbers.

3.3. Ablation Study

We perform an ablation study on the RTE dataset to assess the impact of different intermediate dimensions of the Adapter layers and the output dimensions of the two FFNs comprising the down projections. The intermediate size and the reduction factor a, defined in Section 2.2, play a pivotal role in controlling the parameter efficiency of an Adapter: smaller intermediate sizes result in a reduced parameter count, but they may potentially impact performance negatively, while a smaller reduction factor introduces more parameters. To explore this trade-off, we explore a range of different intermediate sizes and reduction factors. Follow-

Model	# Params (M)	Output dims. $a, \frac{r}{a}$	RTE Acc
AdaKron$_8$	0.1	2,4	71.1
AdaKron$_{16}$	0.2	2,8	71.8
AdaKron$_{32}$	0.5	2,16	<u>74.4</u>
AdaKron$_{48}$	0.7	2,24	**74.7**
AdaKron$_{64}$	0.9	2,32	72.6
AdaKron$_{256}$	3.6	2,128	73.6
AdaKron$_8$	0.1	4,2	70.8
AdaKron$_{16}$	0.2	4,4	70.4
AdaKron$_{32}$	0.4	4,8	<u>73.3</u>
AdaKron$_{48}$	0.6	4,12	**75.6**

Table 3: Ablation study on RTE development set with BERT-base. *Output dims.* refers to the different output dimensions of the two FFNs in the down projection layers. Best model in **bold**, <u>underlined</u> is the second-best one.

ing previous works (Houlsby et al., 2019; Wang et al., 2022), we evaluate different intermediate sizes, i.e. 8, 16, 32, 48, and 256, coupled with two reduction factors, i.e. 2 and 4. Results are reported in Table 3. The intermediate dimension of 48 achieves the best result overall, regardless of the reduction factors. We note that for intermediate dimensions of 8 or 16, an output dimension of 2 outperforms 4, while maintaining an identical parameter count. Consequently, we opt for the optimal configuration, featuring an intermediate dimension of 48 and a reduction factor of 4. Furthermore, this setting involves training fewer parameters, as shown by column *# Params (M)* in Table 3.

4. Conclusions and Future Works

In this paper we present AdaKron, a Parameter-Efficient Fine-Tuning approach implemented within an Adapter-based framework, augmented with Kronecker product operations on output vectors. By leveraging this technique, we manage to fine-tune only 0.55% of the original BERT parameters, while consistently achieving competitive performance results comparable to other state-of-the-art PEFT methods, even with larger parameter counts. As future work, we plan to improve our approach by incorporating it within a Mixture of Experts framework (Fedus et al., 2022; Kasela et al., 2024b), extending our evaluation to different datasets and tasks, in multiple languages.

5. Acknowledgments

We acknowledge the CINECA award under the ISCRA initiative, for the availability of high-performance computing resources and support. The authors also wish to acknowledge CSC – IT Center for Science, Finland, for computational resources. This work was partially supported by the MUR under the grant “Dipartimenti di Eccellenza 2023-2027” of the Department of Informatics, Systems and Communication of the University of Milano-Bicocca, Italy.

6. Ethics Statement and Limitations

Ethics Statement Our method belongs to the general category of Parameter Efficient Fine-Tuning, and our focus is to reduce the number of trained parameters. Our approach can help reduce the carbon footprint of the model training: the model is fine-tuned from a pretrained model, thus can be adapted to a downstream task quickly updating only a small set of parameters while reaching satisfying performances. The usage of our model would be the same as previous methods, i.e., the practical deployment for some Natural Language Processing applications. Our method shares the same possibilities as most of previous efficient fine-tuning approaches, such as misusage, containing data bias, and suffering from adversarial attacks. We conclude that our work will not likely have a negative ethical impact.

Limitations We focus on the GLUE dataset only, with the following consequent limitations: (1) we work in a monolingual setting and only on English data; (2) we lack a more comprehensive evaluation using different NLP benchmarks and tasks, e.g. SuperGLUE (Wang et al., 2019) for Text Classification and E2E (Novikova et al., 2017) for Text Generation.

7. Bibliographical References

Reinald Kim Amplayo, Kang Min Yoo, and Sang-Woo Lee. 2022. Attribute injection for pretrained language models: A new benchmark and an efficient method. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 1051–1064, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Alan Ansell, Edoardo Maria Ponti, Jonas Pfeiffer, Sebastian Ruder, Goran Glavaš, Ivan Vulić, and Anna Korhonen. 2021. Mad-g: Multilingual adapter generation for efficient cross-lingual transfer. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4762–4781.

Ankur Bapna, N. Arivazhagan, and Orhan Firat. 2019. Simple, scalable adaptation for neural machine translation. In *Conference on Empirical Methods in Natural Language Processing*.

Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. 2022. BitFit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1–9, Dublin, Ireland. Association for Computational Linguistics.

Marco Braga, Alessandro Raganato, Gabriella Pasi, et al. 2023. Personalization in bert with adapter modules and topic modelling. In *Proceedings of the 13th Italian Information Retrieval Workshop (IIR 2023). Pisa, Italy*, pages 24–29.

Guanzheng Chen, Fangyu Liu, Zaiqiao Meng, and Shangsong Liang. 2022. Revisiting parameter-efficient tuning: Are we really there yet? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2612–2626, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Jae Won Cho, Dawit Mureja Argaw, Youngtaek Oh, Dong-Jin Kim, and In So Kweon. 2023. Empirical study on using adapters for debiased visual question answering. *Computer Vision and Image Understanding*, 237:103842.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Yang Zonghan, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, Jing Yi, Weilin Zhao, Xiaozhi Wang, Zhiyuan Liu, Hai-Tao Zheng, Jianfei Chen, Yang Liu, Jie Tang, Juanzi Li, and Maosong Sun. 2023. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 5:1–16.

Ali Edalati, Marzieh Tahaei, Ivan Kobyzev, Vahid Partovi Nia, James J Clark, and Mehdi Rezagholizadeh. 2022. Krona: Parameter efficient tuning with kronecker adapter. *arXiv preprint arXiv:2212.10650*.

Ali Edalati, Marzieh S. Tahaei, Ahmad Rashid, V. Nia, James J. Clark, and Mehdi Rezagholizadeh. 2021. Kronecker decomposition for gpt compression. In *Annual Meeting of the Association for Computational Linguistics*.

William Fedus, Jeff Dean, and Barret Zoph. 2022. A review of sparse expert models in deep learning.

Demi Guo, Alexander Rush, and Yoon Kim. 2021. Parameter-efficient transfer learning with diff pruning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4884–4896, Online. Association for Computational Linguistics.

Marawan Gamal Abdel Hameed, Marzieh S. Tahaei, Ali Mosleh, and Vahid Partovi Nia. 2021. Convolutional neural network compression through generalized kronecker product decomposition. *CoRR*, abs/2109.14710.

Harold V. Henderson, Friedrich Pukelsheim, and Shayle R. Searle. 1983. On the history of the kronecker product. *Linear & Multilinear Algebra*, 14:113–120.

Dan Hendrycks and Kevin Gimpel. 2016. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Jingjing Jiang and Nanning Zheng. 2023. Mixphm: Redundancy-aware parameter-efficient tuning for low-resource visual question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24203–24213.

Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. 2021. Compacter: Efficient low-rank hypercomplex adapter layers. *Advances in Neural Information Processing Systems*, 34:1022–1035.

Pranav Kasela, Marco Braga, Gabriella Pasi, and Raffaele Perego. 2024a. Se-pqa: Personalized community question answering. In *The Web Conference 2024 (WWW '24)*.

Pranav Kasela, Gabriella Pasi, Raffaele Perego, and Nicola Tonellotto. 2024b. Desire-me: Domain-enhanced supervised information retrieval using mixture-of-experts. In *Advances in Information Retrieval*, pages 111–125, Cham. Springer Nature Switzerland.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Yingting Li, Ambuj Mehrish, Rishabh Bhardwaj, Navonil Majumder, Bo Cheng, Shuai Zhao, Amir Zadeh, Rada Mihalcea, and Soujanya Poria. 2023. Evaluating parameter-efficient transfer learning approaches on sure benchmark for speech understanding. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Yuning Mao, Lambert Mathias, Rui Hou, Amjad Almahairi, Hao Ma, Jiawei Han, Scott Yih, and Madian Khabsa. 2022. UniPELT: A unified framework for parameter-efficient language model tuning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6253–6264, Dublin, Ireland. Association for Computational Linguistics.

Bonan Min, Hayley Ross, Elior Sulem, Amir Pouran Ben Veyseh, Thien Huu Nguyen, Oscar Sainz, Eneko Agirre, Ilana Heintz, and Dan Roth. 2023. Recent advances in natural language processing via large pre-trained language models: A survey. *ACM Comput. Surv.*, 56(2).

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2021. AdapterFusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 487–503, Online. Association for Computational Linguistics.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020a. Adapterhub: A framework for adapting transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP 2020): Systems Demonstrations*, pages 46–54, Online. Association for Computational Linguistics.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. 2020b. MAD-X: An Adapter-Based Framework for Multi-Task Cross-Lingual Transfer. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7654–7673, Online. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Marzieh S Tahaei, Ella Charlaix, Vahid Partovi Nia, Ali Ghodsi, and Mehdi Rezagholizadeh. 2021. Kroneckerbert: Learning kronecker decomposition for pre-trained language models via knowledge distillation. *arXiv preprint arXiv:2109.06243*.

Yaqing Wang, Sahaj Agarwal, Subhabrata Mukherjee, Xiaodong Liu, Jing Gao, Ahmed Hassan Awadallah, and Jianfeng Gao. 2022. AdaMix: Mixture-of-adaptations for parameter-efficient model tuning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5744–5760, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Junde Wu, Rao Fu, Huihui Fang, Yuanpei Liu, Zhaowei Wang, Yanwu Xu, Yueming Jin, and Tal Arbel. 2023. Medical sam adapter: Adapting segment anything model for medical image segmentation. *arXiv preprint arXiv:2304.12620*.

Xilie Xu, Jingfeng Zhang, and Mohan Kankanhalli. 2023. Autolora: A parameter-free automated robust fine-tuning framework. *arXiv preprint arXiv:2310.01818*.

Yan Xu, Etsuko Ishii, Samuel Cahyawijaya, Zihan Liu, Genta Indra Winata, Andrea Madotto, Dan Su, and Pascale Fung. 2022. Retrieval-free knowledge-grounded dialogue response generation with adapters. In *Proceedings of the Second DialDoc Workshop on Document-grounded Dialogue and Conversational Question Answering*, pages 93–107, Dublin, Ireland. Association for Computational Linguistics.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. 2023. Adaptive budget allocation for parameter-efficient fine-tuning. In *The Eleventh International Conference on Learning Representations*.

8. Language Resource References

Bar-Haim, Roy and Dagan, Ido and Dolan, Bill and Ferro, Lisa and Giampiccolo, Danilo. 2006. *The second PASCAL recognising textual entailment challenge*.

Luisa Bentivogli and Bernardo Magnini and Ido Dagan and Hoa Trang Dang and Danilo Giampiccolo. 2009. *The Fifth PASCAL Recognizing Textual Entailment Challenge*. NIST.

Cer, Daniel and Diab, Mona and Agirre, Eneko and Lopez-Gazpio, Iñigo and Specia, Lucia. 2017. *SemEval-2017 Task 1: Semantic Textual Similarity Multilingual and Crosslingual Focused Evaluation*. Association for Computational Linguistics.

Dagan, Ido and Glickman, Oren and Magnini, Bernardo. 2006. *The PASCAL Recognising Textual Entailment Challenge*. Springer Berlin Heidelberg.

Dolan, William B. and Brockett, Chris. 2005. *Automatically Constructing a Corpus of Sentential Paraphrases*.

Giampiccolo, Danilo and Magnini, Bernardo and Dagan, Ido and Dolan, Bill. 2007. *The Third PASCAL Recognizing Textual Entailment Challenge*. Association for Computational Linguistics.

Levesque, Hector and Davis, Ernest and Morgenstern, Leora. 2012. *The winograd schema challenge*.

Novikova, Jekaterina and Dušek, Ondřej and Rieser, Verena. 2017. *The E2E dataset: New challenges for end-to-end generation*.

Rajpurkar, Pranav and Zhang, Jian and Lopyrev, Konstantin and Liang, Percy. 2016. *SQuAD: 100,000+ Questions for Machine Comprehension of Text*. Association for Computational Linguistics.

Socher, Richard and Perelygin, Alex and Wu, Jean and Chuang, Jason and Manning, Christopher D. and Ng, Andrew and Potts, Christopher. 2013. *Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank*. Association for Computational Linguistics.

Wang, Alex and Pruksachatkun, Yada and Nangia, Nikita and Singh, Amanpreet and Michael, Julian and Hill, Felix and Levy, Omer and Bowman, Samuel R. 2019. *SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems*. Curran Associates Inc.

Wang, Alex and Singh, Amanpreet and Michael, Julian and Hill, Felix and Levy, Omer and Bowman, Samuel. 2018. *GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding*. Association for Computational Linguistics.

Warstadt, Alex and Singh, Amanpreet and Bowman, Samuel R. 2019. *Neural Network Acceptability Judgments*. MIT Press.

Williams, Adina and Nangia, Nikita and Bowman, Samuel. 2018. *A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference*. Association for Computational Linguistics.