BitFit: Simple Parameter-efficient Fine-tuning for Transformer-based Masked Language-models

Elad Ben-Zaken[1] **Shauli Ravfogel**[1,2] **Yoav Goldberg**[1,2]
[1]Computer Science Department, Bar Ilan University
[2]Allen Institute for Artificial Intelligence
{benzakenelad, shauli.ravfogel, yoav.goldberg}@gmail.com

Abstract

We show that fine-tuning only the bias terms (or a subset of the bias terms) of pre-trained BERT models is competitive with (and sometimes better than) fine-tuning the entire model.

Besides their practical utility, these findings are relevant for the question of understanding the commonly-used process of finetuning: they support the hypothesis that finetuning is mainly about exposing knowledge induced by language-modeling training, rather than learning new task-specific linguistic knowledge.

1 Introduction

Large pre-trained transformer based language models, and in particular bidirectional masked language models from the BERT family (Devlin et al., 2018; Liu et al., 2019; Joshi et al., 2019), are responsible for significant gains in many NLP tasks. Under the common paradigm, the model is pre-trained on large, annotated corpora with the LM objective, and then *finetuned* on task-specific supervised data. The large size of these models make them expensive to train and, more importantly, expensive to deploy. There is also a large theoretical question regarding the fine-tuning process itself, and what is being learned in it. These led researchers to consider fine-tuning variants where one identifies a small subset of the model parameters which need to be changed for good performance in end-tasks, while keeping most of the parameters intact (§2).

In this work we present a simple and effective approach to fine tuning (§3), which has the following benefits:

1. Changing very few parameters per fine-tuned task.

2. Changing the same set of parameters for every tasks (task-invariance).

3. The changed parameters are both isolated and localized across the entire parameter space.

4. Changing only these parameters reaches the same task accuracy as full fine-tuning, and sometimes even improves results.

Specifically, we show that freezing most of the network and **fine-tuning only the bias-terms** is surprisingly effective. Moreover, if we allow the tasks to suffer a small degradation in performance, we can fine-tune only two bias components (the "query" and "middle-of-MLP" bias terms), amounting to half of the bias parameters in the model, and only 0.04% of all model parameters.

This result has a large practical utility in deploying multi-task fine-tuned models in memory-constrained environments, as well as opens the way to trainable hardware implementations in which most of the parameters are fixed.

It also open up a set of research directions regarding the role of bias terms in pre-trained networks, and the dynamics of the fine-tuning process.

2 Background: fine-tuning and parameter-efficient fine-tuning

In transfer-learning via model fine-tuning, a pre-trained encoder network takes the input and produces contextualized representations. Then, a task-specific classification layer (here we consider linear classifiers) is added on top of the encoder, and the entire network (encoder+task specific classifiers) is trained end-to-end to minimze the task loss.

Desired properties. While fine-tuning per-task is very effective, it also results in a unique, large model for each pre-trained task, making it hard reason about (what was changed in the fine-tuning process?) as well as hard to deploy, especially as the number of tasks increases. Ideally, one would want a fine-tuning method that:

(i) matches the results of a fully fine-tuned model; (ii) changes only a small portion of the model's parameters; and (iii) enables tasks to arrive in a stream, instead of requiring simultaneous access to all datasets. For efficient hardware based deployments, it is further preferred that (iv): set of parameters that change values will be consistent across different tasks.

Learning vs. Exposing. The feasibility of fulfilling the above requirements depends on a fundamental question regarding the nature of the fine-tuning process of large pre-trained LMs: to what extent does the fine-tuning process induces the *learning of new capabilities*, vs. the *exposing of existing capabilities*, which were learned during the pre-training process. If fine-tuning can be cast as exposure of existing capabilities, this can allow for more efficient fine-tuning and deployment, by building on the frozen, pre-trained model, and constraining the fine-tuning to a "small", task-specific modification, rather than unconstrained fine tuning over the entire parameter space.

Existing approaches. Two recent works have demonstrated that adaptation to various end-tasks can in fact be achieved by changing only a small subset of parameters. The first work, by Houlsby et al. (2019) ("Adapters"), achieves this goal by injecting small, trainable task-specific "adapter" modules between the layers of the pre-trained model, where the original parameters are shared between tasks. The second work, by Guo et al. (2020) ("Diff-Pruning"), achieves the same goal by adding a sparse, task-specific difference-vector to the original parameters, which remain fixed and are shared between tasks. The difference-vector is regularized to be sparse.

Both methods allow adding only a small number of trainable parameters per-task (criteria ii), and each task can be added without revisiting previous ones (criteria iii). They also partially fulfill criteria (i), suffering only a small drop in performance compared to full fine-tuning. This supports, to some extent, the "fine-tuning-as-exposing" hypothesis. Additionally, the Adapter method, but not the Diff-Pruning method, also supports criteria (iv). However, Diff-Pruning is more parameter efficient than the Adapter model, and also achieves better task scores. We compare against Diff-Pruning in the experiments section, and show that we perform better, while also satisfying criteria (iv).

3 Bias-terms Fine-tuning (BitFit)

We propose a method we call BitFit (BIas-Term FIne-Tuning), in which freeze most of the transformer-encoder parameters, and train only the bias-terms and the task-specific classification layer.

The approach is parameter-efficient: each new task requires storing only the bias terms parameter vectors (which amount to less than 0.1% of the total number of parameters), and the task-specific final linear classifier layer.

Concretely, the BERT encoder is composed of L layers, where each layer ℓ starts with M self-attention heads, where a self attention head (m, ℓ) has *key*, *query* and *value* encoders, each taking the form of a linear network:

$$\begin{aligned}\text{query}^{m,l}(\mathbf{x}) &= \mathbf{W}^{Q,m,l}\mathbf{x} + \mathbf{b}^{Q,m,l}\\ \text{key}^{m,l}(\mathbf{x}) &= \mathbf{W}^{K,m,l}\mathbf{x} + \mathbf{b}^{K,m,l}\\ \text{value}^{m,l}(\mathbf{x}) &= \mathbf{W}^{V,m,l}\mathbf{x} + \mathbf{b}^{V,m,l}\end{aligned}$$

The attention computation based on these are then concatenated and fed into a 3-layer MLP, where layer i in the MLP takes the form

$$\text{MLP}^{i,\ell}(\mathbf{x}) = \max(0, (\mathbf{W}^{M_i,\ell}\mathbf{x} + \mathbf{b}^{M_i,\ell}))$$

Within the MLP there are also 2 LayerNorm operation:

$$LN^{\ell,i}(\mathbf{x}) = \mathbf{g}^{LN_i,\ell} \odot \frac{\mathbf{x} - \mu}{\sigma} + \mathbf{b}^{LN_i,\ell}$$

The collection of all matrices $\mathbf{W}^{(\cdot)}$ and vectors $\mathbf{g}^{(\cdot)}$ and $\mathbf{b}^{(\cdot)}$ are the network's *parameters* Θ, where the subset of parameters corresponding to vectors $\mathbf{b}^{(\cdot)}$ are the *bias terms*.

The bias terms are additive, and correspond to a very small fraction of the network: In BERT$_{\text{BASE}}$ there are 102k bias parameters and in BERT$_{\text{LARGE}}$ there are 271k bias parameters which make up 0.09% and 0.08% of the total number of parameters in each model, respectively.

We show that by freezing all the parameters $\mathbf{W}^{(\cdot)}$ and $\mathbf{g}^{(\cdot)}$ and fine-tuning only the additive bias terms $\mathbf{b}^{(\cdot)}$, we achieve transfer learning performance which is comparable (and sometimes better!) than fine-tuning of the entire network.

We also show that we can fine-tune only a subset of the bias parameters, namely those associated with the *query* and the *second MLP layer* (only $\mathbf{b}^{Q,(\cdot),(\cdot)}$ and $\mathbf{b}^{M_2,(\cdot)}$), and still achieve accuracies that rival full-model fine-tuning.

	%Params	QNLI	SST2	MNLI$_{\text{m}}$	MNLI$_{\text{mm}}$	CoLA	MRPC	STSB	RTE	QQP
Full-FT†	100%	93.5	94.1	86.5	87.1	62.8	91.9	89.8	71.8	87.6
Full-FT	100%	89.7	93.4	—	—	60.1	89.1	—	71.7	—
Diff-Prune†	0.1%	92.7	93.3	85.6	85.9	58	87.4	86.3	68.6	85.2
BitFit	0.08%	91.1	93.3	—	—	62.9	91.5	90	75.1	87.6

Table 1: BERT$_{\text{LARGE}}$ model performance on the GLUE benchmark validation set. Lines with † indicate results taken from (Guo et al., 2020). Cells with — were not ready on time for the anonymity period.

This approach to parameter-efficient fine-tuning is substantially simpler than previous works, while being highly efficient in terms of number of changed parameters, localized in where the changes happen and performing better on the GLUE benchmark.

4 Experiments and Results

Datasets. We evaluate the bias-terms fine-tuning on the GLUE benchmark (Wang et al., 2018). Consistent with previous work (Houlsby et al., 2019; Devlin et al., 2018) we evaluate on all GLUE tasks except for WNLI, on which BERT models do not outperform the majority baseline. The remaining tasks are: Linguistic Acceptability (CoLA), Sentiment Prediction (SST-2), Paraphrase-identification (MRPC), Quora Question Pairs Classification (QQP), Semantic Textual Similarity Benchmark (STS-B), Multi-Genre Natural Language Inference (MNLI), Question Answering NLI (QNLI), and Recognizing Textual Entailment (RTE). For each task, we evaluate on the validation set, report the metric used in the GLUE submission website.

Models and Optimization We use the publicly available pre-trained BERT-base, BERT-large (Devlin et al., 2018) and RoBERTA (Liu et al., 2019) models, using the HuggingFace interface and implementation.

To perform classification with BERT, we follow the approach of Devlin et al. (2018), and attach a linear layer to the contextual embedding of the `CLS` token to predict the label. The GLUE tasks are fed into BERT using the standard procedures.

We optimize using Adam (Kingma and Ba, 2015), with batch sizes of 8. For full fine-tuning, we used the default initial learning rate of 3e-5, and for the bias-only experiments we used initial learning rates between 1e-3 and 1e-4, as the 3e-5 rate took a very long time to converge on some of the tasks. With the larger learning rates, the bias-only fine-tuning consistently converged in 7 or fewer iterations, on all tasks. We did not perform hyperparameter optimization beyond the minimal search over 3 learning rates.

4.1 Comparison to Diff-Pruning (Table 1)

In the first experiment, we compare BitFit to the Diff-Pruning method, when using a comparable number of parameters. Table 1 reports the accuracy compared to the Diff-Pruning numbers reported by Guo et al. (2020), on their least-parameters setting. This experiment used the BERT-large model.

The BitFit (bias-only) results outperform the Diff-Pruning ones on 5 out of 7 tasks, tying in 1 task, and underperform in 1,[1] while using fewer trainable parameters.

4.2 Different Base-models (Table 2)

We repeat the BERT-large results on different base-models (the smaller BERT-base and the better performing RoBERTA-base). The result in Table 2 shows that the trends remain consistent.

4.3 Fewer bias parameters (Table 3)

Can we fine-tune on only a subset of the bias-parameter?

We define the amount of change in a bias vector $\mathbf{b}$ to be $\frac{1}{dim(\mathbf{b})}|\mathbf{b}_0 - \mathbf{b}_F|_1$, that is, the average absolute change, across its dimensions, between the initial LM values $\mathbf{b}_0$ and its fine-tuned values $\mathbf{b}_F$. We rank the different bias terms according to this metric, and find that the bias terms that change the most during bias-only fine-tuning are those associated with the query ($\mathbf{b}^{Q,\ell,m}$) and with the intermediate layer in the MLP ($\mathbf{b}^{M_2,\ell}$, the layer which takes the input from 768 dimensions to 3072). These amount to about half of the bias parameters in the model. Table 3 reports the results on fine-tuning only these bias-terms, for the BERT-base model. Results are only very marginally lower than when tuning all bias parameters.

[1] Two numbers are missing, as they were not ready in time for the anonymity period. They will be updated soon.

	% Params	QNLI	SST2	MNLI$_{\text{m}}$	MNLI$_{\text{mm}}$	CoLA	MRPC	STSB	RTE	QQP
BERTBASE										
Full-FT	100%	90.6	92.5	—	—	53.4	89.9	—	71.4	85.4
BitFit	0.09%	90.5	92.7	—	—	56.0	91.7	—	72.7	82.9
BERTLARGE										
Full-FT	100%	89.7	93.4	—	—	60.1	89.1	—	71.7	—
BitFit	0.08%	91.1	93.3	—	—	62.9	91.5	90	75.1	87.6
RoBERTABASE										
Full-FT	100%	91.9	93.6	—	—	—	92.5	—	78.7	—
BitFit	0.09%	91.8	93.7	—	—	62.0	92.7	—	81.5	—

Table 2: Results for different base models. Cells with — indicate missing values, which were not available on time.

	% Params	QNLI	SST2	MNLI$_{\text{m}}$	MNLI$_{\text{mm}}$	CoLA	MRPC	STSB	RTE	QQP
Full-FT	100%	90.6	92.5	—	—	53.4	89.9	—	71.4	85.4
BitFit	0.09%	90.5	92.7	—	—	56.0	91.7	—	72.7	82.9
BitFit$-\partial$	0.04%	90.2	92.3	—	—	57.2	—	—	72.7	—

Table 3: Fine-tuning using a subset of the bias parameters. Reported results are for the BERT$_{\text{BASE}}$ model.

4.4 Generalization gap

When considering the generalization gap (different between train-time and test-time performance), we see that it is substantially smaller for the BitFit models: while for full fine-tuning the train set accuracy reaches nearly 100%, in the bias-only fine-tuned models the difference between the train and test set performance is often less than 2% points.

5 Related Work

The problem of identifying the minimal set of parameters that need to be fine-tuned to achieve good performance in end-tasks relates both to practical questions of model compression, and also to more fundamental question on the nature of the pre-training and finetuning process, the "linguistic knowledge" induced by each of them, and the extent to which it generalizes to different tasks.

5.1 Over-parameterization

A large body of work has demonstrated that large LM models are *over-parameterized* and have a large degree of redundancy, which can be effectively reduced without significantly affecting performance. Those methods can largely be partitioned to pruning-based and distillation-based compression. In distillation (Buciluǎ et al., 2006; Hinton et al., 2015; Urban et al., 2017), one trains a smaller model from scratch to mimic the behavior of the larger model. In pruning (Karnin, 1990; Reed, 1993; Augasta and Kathirvalavakumar, 2013; Liu et al., 2014; Han et al., 2015; Molchanov et al., 2017), one identifies the parts in the network whose removal is less damaging to performance, and removes them. Both methods were shown to be effective for transformer-based language models (Abadeer (2020); Michel et al. (2019), and many others). The remarkable success of those works have sparked interest the lottery-ticket hypothesis (Frankle and Carbin, 2019; Chen et al., 2020; Prasanna et al., 2020): the conjecture that large models are needed in pretraining only to induce (in high probability) the existing of sub-networks initialized with the correct inductive bias for learning, and the findings that those sparse networks often transfer well to different tasks.

Over-parameterization and fine-tuning. The majority of works dealing with model compression focused on retraining the model's original performance on the task it was trained on. Gordon et al. (2020) have focused on the influence of pruning on transfer learning in transformer-based LMs. They have shown that medium pruning level do not harm end-task performance, and that pruning the pretrained model once and then finetuning it to different task is not worse then pruning each model after the finetuning on a specific task. This suggests the necessary "core" parts of the pre-trained model—the parts left after pruning—are largely shared between different NLP tasks. Furthermore, the fact pruning does not damage end-task performance suggests that only a subset of parameters are inherently needed for good performance. Aghajanyan et al. (2020) have attempted to explain the empirical effectiveness of transfer from pruned networks by the notion of intrinsic-dimensionality (Li et al., 2018): the minimum dimension that is needed to optimize a given objective to some precision. They argue that LM pretraining implicitly induces representations that compress well various NLP tasks, and show that the intrinsic dimensionality of those representations with respect to end-tasks is low.

5.2 Bias terms

Bias terms and their importance are rarely discussed in the literature. Indeed, the equations in the paper introducing the Transformer model (Vaswani et al., 2017) do not include bias terms at all, and their existence in the BERT models might as well be a fortunate mistake. Zhao et al. (2020) describe a fine-tuning method based on masking, and explicitly mention ignoring the bias terms, as handling them "did not observe a positive effect on performance".

An exception is the work of Wang et al. (2019) who analyzed bias terms from the perspective of attribution method. They have demonstrated that the values of the bias in the last layer are responsible for the predicted class, and propose a way to back-propagate their importance. For piecewise linear models, they point out to the ability to represent the action of the entire model on an example as a linear model (a possibly different model for each example), whose bias term is a function of *all* bias terms in the model, as well as the other weights. This decomposition suggests that when finetuning the bias terms—as we do here—we implicitly change only the bias term of the equivalent linear model for each example, leaving the other weights intact: the parameters that interact with the input or activations are the same in the original pre-trained model and in the fine-tuned model.

Our work empirically shows the importance and power of the bias parameters to substantially change the networks' behavior, calling for further analysis and attention on the bias terms.

6 Conclusions

We have proposed a novel method for localized, fast fine-tuning of pre-trained transformers for end-tasks. The method focuses the finetuning on a specific fraction of the model parameters—the biases—and maintains good performance in all GLUE tasks we evaluated on. The ability to focus on the same small group of parameters eases deployment, as the vast majority of the parameters of the model are shared between various NLP tasks. It also allows for efficient hardware implementations that hard-wire most of the network computation with the pre-trained weights, while only allowing few changeable parts for inference time.

Besides its empirical utility, the remarkable effectiveness of the bias-only fine-tuning raises intriguing questions on the fine-tuning dynamics of pre-trained transformers, and the relation between the bias terms and transfer between LM and new tasks. We aim to study those questions in a future work.

References

Macarious Abadeer. 2020. Assessment of distilbert performance on named entity recognition task for the detection of protected health information and medical concepts. In *Proceedings of the 3rd Clinical Natural Language Processing Workshop, ClinicalNLP@EMNLP 2020, Online, November 19, 2020*, pages 158–167. Association for Computational Linguistics.

Armen Aghajanyan, Luke Zettlemoyer, and Sonal Gupta. 2020. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. *arXiv preprint arXiv:2012.13255*.

M. Gethsiyal Augasta and T. Kathirvalavakumar. 2013. Pruning algorithms of neural networks - a comparative study. *Central Eur. J. Comput. Sci.*, 3(3):105–115.

Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 535–541.

Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Zhangyang Wang, and Michael Carbin. 2020. The lottery ticket hypothesis for pre-trained BERT networks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Jonathan Frankle and Michael Carbin. 2019. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Mitchell A. Gordon, Kevin Duh, and Nicholas Andrews. 2020. Compressing BERT: studying the effects of weight pruning on transfer learning. *CoRR*, abs/2002.08307.

Demi Guo, Alexander M. Rush, and Yoon Kim. 2020. Parameter-efficient transfer learning with diff pruning.

Song Han, Jeff Pool, John Tran, and William Dally. 2015. Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28:1135–1143.

Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531.

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. *CoRR*, abs/1902.00751.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2019. Spanbert: Improving pre-training by representing and predicting spans. *CoRR*, abs/1907.10529.

Ehud D. Karnin. 1990. A simple procedure for pruning back-propagation trained neural networks. *IEEE Trans. Neural Networks*, 1(2):239–242.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Chunyuan Li, Heerad Farkhoor, Rosanne Liu, and Jason Yosinski. 2018. Measuring the intrinsic dimension of objective landscapes. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Chao Liu, Zhiyong Zhang, and Dong Wang. 2014. Pruning deep neural networks by optimal brain damage. In *INTERSPEECH 2014, 15th Annual Conference of the International Speech Communication Association, Singapore, September 14-18, 2014*, pages 1092–1095. ISCA.

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. *CoRR*, abs/1907.11692.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 14014–14024.

Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. 2017. Pruning convolutional neural networks for resource efficient inference. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Sai Prasanna, Anna Rogers, and Anna Rumshisky. 2020. When BERT plays the lottery, all tickets are winning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 3208–3229. Association for Computational Linguistics.

Russell Reed. 1993. Pruning algorithms-a survey. *IEEE Trans. Neural Networks*, 4(5):740–747.

Gregor Urban, Krzysztof J. Geras, Samira Ebrahimi Kahou, Özlem Aslan, Shengjie Wang, Abdelrahman Mohamed, Matthai Philipose, Matthew Richardson, and Rich Caruana. 2017. Do deep convolutional nets really need to be deep and convolutional? In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. *CoRR*, abs/1804.07461.

Shengjie Wang, Tianyi Zhou, and Jeff A. Bilmes. 2019. Bias also matters: Bias attribution for deep neural network explanation. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 6659–6667. PMLR.

Mengjie Zhao, Tao Lin, Fei Mi, Martin Jaggi, and Hinrich Schütze. 2020. Masking as an efficient alternative to finetuning for pretrained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2226–2241, Online. Association for Computational Linguistics.