Enhancing Transfer Learning Reliability via Block-wise Fine-tuning

Basel Barakat * Qiang Huang *
* School of Computer Science, University of Sunderland, Sunderland, UK.

2023 International Conference on Machine Learning and Applications (ICMLA) | 979-8-3503-4534-6/23/$31.00 ©2023 CRWN | DOI: 10.1109/ICMLA58977.2023.00064

***Abstract*—Fine-tuning can be used to tackle domain specific tasks by transferring knowledge learned from pre-trained models. However, previous studies on fine-tuning focused on adapting only the weights of a task-specific classifier or re-optimising all layers of the pre-trained model using the new task data. The first type of method cannot mitigate the mismatch between a pre-trained model and the new task data, and the second type of method easily causes over-fitting when processing tasks with limited data. To explore the effectiveness of fine-tuning, we propose a novel block-wise optimisation mechanism, which adapts the weights of a group of layers of a pre-trained model. This work presents a theoretical framework and empirical evaluation of block-wise fine-tuning to find a reliable transfer learning strategy. The proposed approach is evaluated on two datasets, Oxford Flowers and Caltech 101, using 15 commonly used state-of-the-art pre-trained base models.**

Results indicate that the proposed strategy consistently outperforms the baselines in terms of classification accuracy, although the specific block leading to optimal performance may vary across models. The investigation reveals that selecting a block from the fourth quarter of a base model generally yields improved performance compared to the baselines. Overall, the block-wise approach consistently outperforms the baselines and exhibits higher accuracy and reliability. This study provides valuable insights into the selection of salient blocks and highlights the effectiveness of block-wise fine-tuning in achieving improved classification accuracy in various models and datasets.

***Index Terms*—Fine-tuning, Transfer Learning, Block-wise, Explainable Performance, Pre-trained Model, Deep Learning**

I. Introduction

The rapid development of deep learning technologies has made it easy to construct and train complex neural networks [1]. The deep structure of neural networks has thus gained tremendous success. However one of their critical challenges is that it needs large amounts of data. Training a model for a specific task on a limited data can lead to poor generalization due to over-fitting. Although lots of data can be now collected online, data annotation is always an expensive and time-consuming task. Therefore, more often in practice, one would fine-tune existing networks by continuing training it on the new task data. This can benefit many applications not having sufficient data by transferring learned knowledge from multiple sources to a domain-specific task [2], [3].

Fine-tuning can process a pre-trained network in three different ways. The first is to freeze all the weights of the pre-trained network, but optimise only classifier layers using new task data. The second is to optimise the weights of all layers. The last one is done by adapting the weights of a subset layers of the pre-trained model as they would be more useful to learn dataset-specific features than other layers.

In our preliminary experiments, we noticed that using a pre-trained models for image classification tasks with large number of classes classification typically yield to low accuracy. We thus hypothesised that fine-tuning a group of layers may lead to some interesting results. Although fine-tuning a subset of layers seems to be more instinctively reasonable than the first two, it is not yet fully investigated how to determine which layers in the pre-trained network can have a greater contribution than other layers, when being tuned on new task data. We therefore propose a novel framework using block-wise fine-tuning in this paper and aim to explore an efficient way to find out the salient set of layers relevant to the features of new task data and improve fine-tuning reliability.

The block-wise mechanism is conducted by dividing a deep network into blocks, each of which consists of a group of layers. In this study, we conducted experiments using 15 widely adopted models that were pre-trained on the Imagenet dataset. These models were subsequently fine-tuned on two datasets, namely Oxford Flowers [4] and Caltech 101 [5]. To determine the most salient block, we tested the block-wise fine-tuning strategy on each of the model's individual blocks. By fine-tuning a subset of layers within each block, we aimed to identify the block that yields the highest accuracy performance. In the subsequent sections, we will present a detailed description of our proposed approaches, which outline the specific techniques and methodologies employed in our research.

The rest of this paper is organised as follows: Section 2 introduces the previous studies in relation to fine-tuning; the theoretical framework is presented in detail in Section 3; Section 4 describes the experimental set-up. The results and analysis of our experiments are presented in Section 5, and finally conclusions are drawn in the last section.

II. Related Work

Within the framework of transfer learning and relying on the architecture of a pre-trained model (PTM), fine-tuning can adapt the model parameters on the target data and has become one of the most promising deep learning techniques in different research fields, such as computer vision (CV), natural language processing (NLP), and speech processing.

1946-0759/23/$31.00 ©2023 Crown
DOI 10.1109/ICMLA58977.2023.00064

A. Fine-tuning in Computer vision

In computer vision community, the annual ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [6] provided multiple images sources and has resulted in a number of innovations in the architecture, such as VGG [7], MobileNetV2 [9], and ResNet50 [10]. By fine-tuning, these high-performing pre-trained models are now widely used in image generation [11], [14], image classification [15], [16], [17], [18], image caption [12], [13], anomaly detection [19], [20], [21], image retrieval [22], [23], etc. In these previous studies, the development of fine-tuning techniques in CV can be found.

The first aspect of fine-tuning is layer wise adaptation. Regarding the studies on the roles of hidden layers, Yosinski et al. [28] conducted empirical study to quantify the degree of generality and specificity of each layer in deep networks. The related studies [28], [29] further claimed that the low-level layers extract general features and the high-level layers extract task-specific features in a deep network. Since then, further works have been conducted to exploit the role of each layer. In [30], Tajbakhsh et al. showed that tuning only a few high-level layers is more effective than tuning all layers. Guo et al. [31] proposed an auxiliary policy network that decides whether to use the pre-trained weights or fine-tune them in layer-wise manner for each instance. In [18], Ro et al. proposed an algorithm that improves fine-tuning performance and reduces network complexity through layer-wise pruning and auto-tuning of layer-wise learning rates. To further reinforce auto-tuning of layer-wise learning rate, Tanvir et al. [32] proposed RL-Tune, a layer-wise fine-tuning framework for transfer learning which leverages reinforcement learning (RL) to adjust learning rates as a function of the target data shift.

The second aspect of fine-tuning is in relation to hyper-parameter optimisation. This is because the increasing complexity of deep learning architecture's slow training is partly caused by "vanishing gradients". In which, the gradients used by back-propagation are extremely large for weights connecting deep layers (layers near the output layer), and extremely small for shallow layers (near the input layer); this results in slow learning in the shallow layers[33]. So, Bharat et al. [33] proposed a method to allow larger learning rates to compensate for the small size of gradients in shallow layers. Since then, various approaches have been explored for better regularization of the transfer learning with effective hyper-parameter selection. In [34] Kornblith et al. proposed a grid-search based approach to search for better hyper-parameters, and Li et al. [3] provided an elaborate guideline of learning rates and other hyper-parameter selections. Furthermore, Parker et al. proposed provably efficient online hyperparameter optimization with population-based bandits, which is found to be effective in optimizing RL training [35] . To improve fine-tuning by using optimiser, Loshchilov et al. [40] designed two robust optimisers, SGDW and AdamW, by combining SGD [36] and Adam [37] with decoupled weight decay. The work in [38], [39] also explored the use of two optimisers, SGD and AdamW, on ImageNet like domains in terms of fine-tuning accuracy. Recently, [41] found that large gaps in performance between SGD and AdamW occur when the fine-tuning gradients in the first "embedding" layer are much larger than in the rest of the model, and claimed that freezing only the embedding layer can lead to SGD performing competitively with AdamW while using less memory.

The third aspect is neural architecture search (NAS), aiming to adapt the architecture of a pre-trained network to match to the characteristics of new task data. Liu et al. [24] used a sequential model-based optimization to guide the search through the architecture of the network. Pham et al. [25] proposed an efficient NAS (ENAS) with parameter sharing, which focuses on reducing the computational cost of NAS by reusing the trained weights of candidate architectures in subsequent evaluations. Lu et al. [26] proposed neural architecture transfer (NAT) to efficiently generate task-specific custom NNs across multiple objectives. Kim et al. [27] proposed to reduces the search cost using given architectural information and cuts NAS costs by early stopping to terminate the search process in advance. In [16], Tanveer et al. developed differentiable neural architecture search method by introducing a differentiable and continuous search space instead of a discrete search space and achieves remarkable efficiency, incurring a low search cost.

B. Fine-tuning in Natural Language Processing

Compared to CV, NLP models was typically more shallow and thus require different fine-tuning methods [42]. In NLP, Mikolov et al. [43] proposed a simple transfer technique by fine-tuning pre-trained word embeddings, a model's first layer, but has had a large impact in practice and is now used in most state-of-the-art models. To mitigate LMs' overfitting to small datasets, Jeremy et al. [42] proposed discriminative language model fine-tuning to retain previous knowledge and avoid catastrophic forgetting. In the last couple of years, large language models, such as GPT [44] and BLOOM [45], were developed by using mask learning on large amounts of text data. Given the size of these large language models, fine-tuning all the model parameters can be compute and memory intensive [46]. Some recent studies [47], [48] have proposed new parameter efficient fine-tuning methods that update only a subset of the model's parameters. As adversarial samples of new task are usually out-of-distribution, adversarial fine-tuning fails to memorize all the robust and generic linguistic features already learned during pre-training. To mitigate the impacts caused by this, Dong [49] et al. proposed to use mutual information to measure how well an objective model memorizes the useful features captured before. Furthermore, Mireshghallah et al. [50] empirically studied memorization of fine-tuning methods using membership inference and extraction attacks as large models have a high capacity for memorizing training samples during pre-training.

C. Fine-tuning in Speech Processing

For speech processing, fine-tuning can work not only for language model adaptation [51], [54], but also for tuning

acoustic models [52], [53], [55], [56]. Fine-tuning language models in speech processing is same as its use in NLP. Guillaume et al. [54] developed a method using a transformer architecture to tune a generic pre-trained representation model for phonemic recognition. For acoustic model adaptation, Violeta et al. [52] proposed an intermediate fine-tuning step that uses imperfect synthetic speech to close the domain shift gap between the pre-training and target data. Tsiamas et al. [53] proposed to use an efficient fine-tuning technique that trains only specific layers of our system, and explore the use of adapter modules for the non-trainable layers. Peng et al. [56] used fine-tuning to learn robust acoustic representation to alleviate the mismatch between a pre-trained model and new task data. The similar work can be also found in [57], where Haidar et al. employed Generative Adversarial Network (GAN) [58] to fine-tune a pre-trained model to match to the acoustic characteristics of new task data.

III. Theoretical Framework

Fine-tuning (**FT**) in this work is to adapt the weights (**W**) of a group of layers (**Ls**) of a pre-trained deep neural network (**DN**) given input data matrices $\mathbf{X} = \{X_1, X_2, \ldots, X_M\}$, where M is the number of training samples of a new task, and X_m represents the m-th sample matrix. The aim of the proposed approach is to find out the block $\mathbf{B}_{Ls}$ most relevant to the target data. This can be represented by:

$$\mathbf{B}_{Ls,W} = \arg\max_{\mathbf{Ls,W}} Accuray(\mathbf{DN}(\mathbf{X})) \tag{1}$$

To attain the aim, we designed block-wise fine-tuning. Fig. 1 shows the architectures of source model and three target models for fine-tuning. Fig. 1(a) is the pre-trained source model. Fig. 1(b) shows the target model (Baseline I) where only classifier layers marked blue are to be adapted and the weights of other layers will be freezed. Fig. 1(c) is the target model (Baseline II), where all layers are to be re-optimised. The two target models will be used as baseline models for a comparison in this paper. Fig. 1 (d) show the architectures of block-wise fine-tuning. The dash box means when a layer or a group layers are being tuned, the weights of other layers, except classifier layers, keep fixed.

Block-wise Fine-tuning

In a deep neural network, several adjacent layers with a similar function can be often grouped into a block. Tuning the weights of these layers in a block rather than all layers is an efficient way to match the learned knowledge to a specific task since only a relatively small part of parameters will be adapted. Fig. 1(d) shows the architecture of block-wise fine-tuning, starting from the input end of a deep network.

For block-wise fine-tuning, its critical step is to divide the structure of a deep neural network into blocks. In our work, the division can be done by *non-weighting layers*, such as the Maxpooling layer and Activation layer. In some typical deep neural networks, such as VGG16 [7] and ResNet50 [10], there include not only Convolutional layers, but also MaxPooling, Batch Normalization, and Activation layers. The convolutional layers in these models generally contain major parameters, and only a relatively small number of parameters are from other layers. This means Maxpooling layer, Batch Normalization layer, and even Activation layers could be viewed as a delimiter to segment a deep neural network into blocks.

Algorithm 1 Block-wise fine-tuning (**FT**)

```
Load the weights, W, of a pre-trained model (DN)
Preprocess input data, X, and select 70% data for training
    and 30% data for evaluation
N = # Blocks of DN
Initialise layer index, i = 1,
Initialise accuracy array, Acc = zeros(N)
Training:
  While i < N
      tune the weights of the layers in the i^th block, B_i,
      and classifier
      Acc[i] = FT(DN(B_i, C))
      i = i + 1
  EndWhile
Evaluation:
  Identify the most salient block, B_opt, using Acc
  Compare B_opt Acc to the Baselines Acc
```

Algorithm 1 shows the pseudo code of implementing block-wise fine-tuning, where a while loop is run over blocks.

IV. Experimental set-up

A. Data set

The algorithms were tested on two datasets: (1) Oxford Flowers (OXF) [4], (2) Caltech 101 (CAL) [5]. Both datasets have more than 100 classes with varying number of images per class. The images are color images resized to a resolution of 224×224 pixels, then we scaled RGB to 0-1 range. The datasets were split into training, validation, and testing with ratios of 70%, 20% and 10% respectively.

B. Classifiers

The training process for this experiment involves using the pre-trained models on the ImageNet weights, as a starting point, and then fine-tuning the models on the datasets. We used TensorFlow Keras library to build and train the models. We have evaluated the performance of the algorithms on 15 commonly used pre-trained base models shown in table I. The models had been chosen to be varying in layer numbers, and size.

For the classifier layers, we started with a Flattening layer, which flattens the dimensions of the output to prepare it for the use in Fully Connected (FC) layers [62]. Then a dense layer, with 512 units, followed by a dropout layer with a rate of 0.5 to reduce the probability of over-fitting [64]. Then a dense layers with 102 units respectively. The 512 units layer is activated by a 'relu' function, and the final dense layer has five units, activated by 'softmax' for the multi-class classification [63].

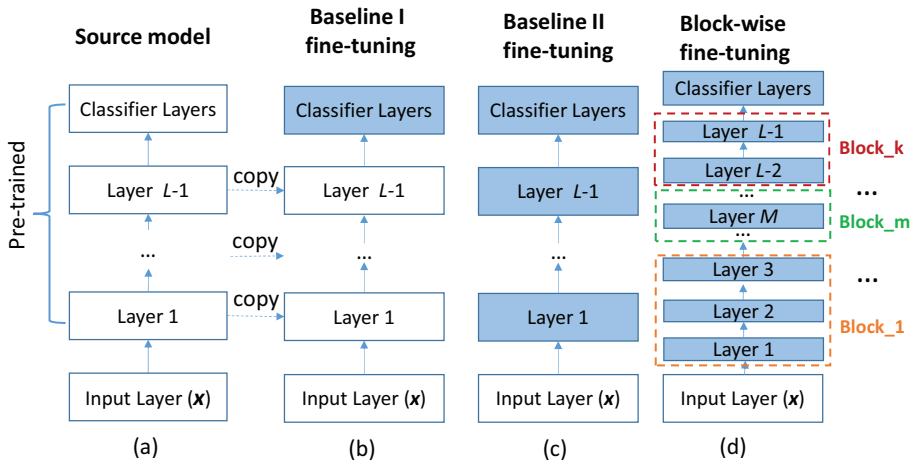

Fig. 1: Architectures of source model and three target models: (a) source model (b) fine-tuning only classifier layers of source model, (c) fine-tuning all layers of source model, (d) target model adapted by block-wise fine-tuning.

TABLE I: Tested Base Models

Base Models	Abbreviation	Ref.
VGG16 architecture	VGG16	[7]
MobileNetV1	MN1	[60]
MobileNetV2	MN2	[9]
MobileNetV3_small	MN3S	[61]
MobileNetV3_large	MN3L	[61]
ResNet50_V2	RestNet	[10]
Self_Regulated_Networks (2 giga flops)	RegNetX002	[65]
Self_Regulated_Networks (4 giga flops)	RegNetX004	[65]
EfficientNetV2B0 architecture	EffNetV2B0	[66]
EfficientNetV2B1 architecture	EffNetV2B1	[66]
EfficientnetV1B0 architecture	EffNetV1B0	[67]
EfficientnetV1B1 architecture	EffNetV1B1	[67]
Xception	XCP	[68]
NASNetMobile	NasNet	[69]
DenseNet121	DenNet	[70]

The model is then compiled with loss function "categorical cross-entropy", an optimizer "Adam" and metrics as validation "accuracy". The model is then trained on the training data with batch size of 24, 10 epochs and validation data and callbacks.

V. Results and Analysis

Our investigation commenced by evaluating the performance of three different models: Baseline I (BL1), Baseline II (BL2), and individual blocks of the base models. BL1 only fine-tunes the classifier layers, BL2 fine-tunes both the base model layers and the classifier layers, while the individual block models fine-tune a specific block and the classifier layers while keeping the remaining parts of the model frozen.

To assess the performance of these models, we conducted experiments on two datasets: OXF dataset and CAL dataset. The accuracy results obtained on the OXF dataset are presented in Figure 2, while the accuracy results on the CAL dataset are illustrated in Figure 3.

Upon examining the figures, it is apparent that the optimal block (B_{opt}) consistently outperforms both baselines across a majority of the models for both datasets. This observation demonstrates the superior performance of our proposed approach compared to the baselines, underscoring the effectiveness of selectively fine-tuning specific blocks in achieving improved accuracy results.

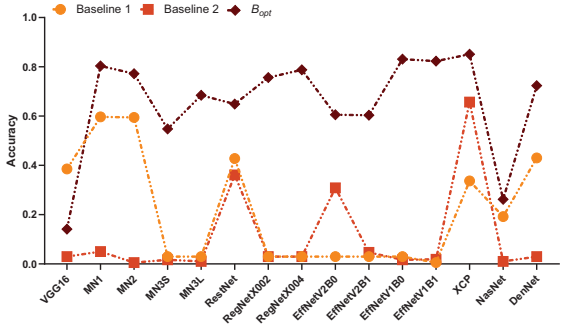

Fig. 2: B_{opt} and baselines classification accuracy on the OXF dataset.

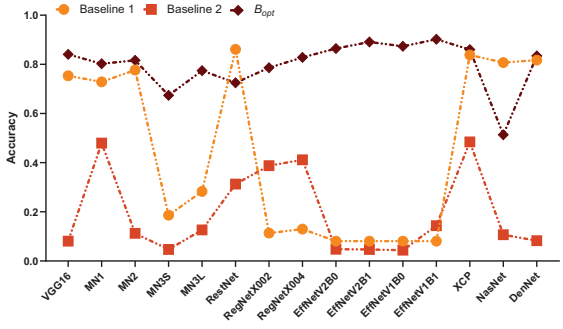

Fig. 3: B_{opt} and baselines classification accuracy on the CAL dataset.

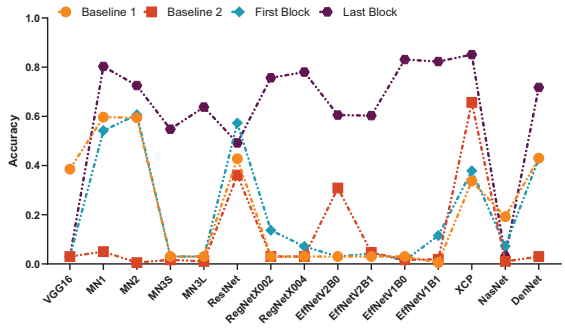

Fig. 4: Classification accuracy of OXF dataset showing the first and last blocks.

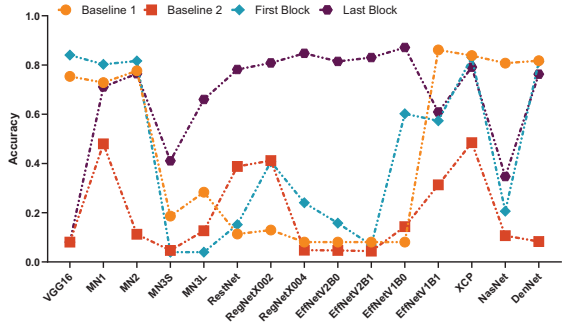

Fig. 5: Classification accuracy of CAL dataset showing the first and last blocks.

Subsequently, we further explored the performance of all the blocks to gain insights into a method for identifying the most salient block. The results of this investigation are presented in Appendix A. Upon analyzing the figure, it becomes apparent that pinpointing the exact block that would yield optimal performance is uncertain. However, there is a discernible pattern that emerges from the results.

To gain a deeper understanding of the performance of the blocks, we compared the performance of the baselines with the first and last blocks for all the models. The results of this comparison are presented in Figure 4 for the OXF dataset and Figure 5 for the CAL dataset. From the figures, it becomes evident that the last block consistently outperforms both the baselines and the first block. This indicates the significance of the final block in achieving improved performance. However, it is important to note that the last block does not necessarily correspond to the optimized block (B_{opt}) shown in Figures 2 and 3. The comparison highlights the complexity of identifying the most salient block, as the optimized block may not always be the last block in the model architecture.

Based on our findings, we can draw a general conclusion that selecting the blocks closer to the classifier layers of almost any base model will result in improved performance compared to the baselines, leading to higher classification accuracy.

To further illustrate this, we present box plots in Figure 6 for

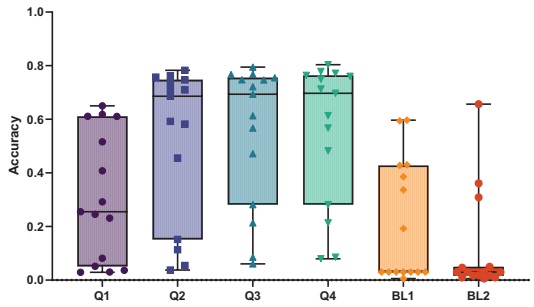

Fig. 6: Box plot representing the classification accuracy of all base models on the OXF dataset. The plot showcases both the baseline accuracies and the results of block-wise fine-tuning for each quarter of the blocks.

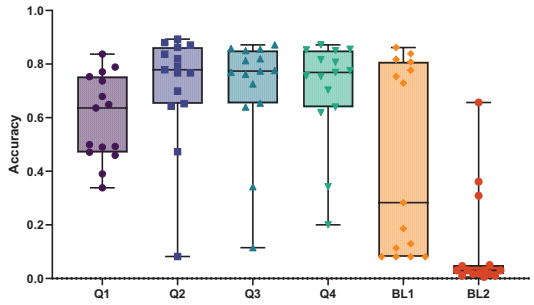

Fig. 7: Box plot representing the classification accuracy of all base models on the CAL dataset. The plot showcases both the baseline accuracies and the results of block-wise fine-tuning for each quarter of the blocks.

the OXF dataset and Figure 7 for the CAL dataset. These box plots display the performance of all the models per quarter, including the baselines. The blocks here are equally divided into four quarters. Q1,2,3,4 covers the first 25% blocks, the middle two 25% layers, and the last 25% layers close to the target layer, respectively.

By examining these figures, we observe that the blocks from the fourth quarter consistently outperform the baselines, as evidenced by the higher accuracy values. Furthermore we can observe that the inter quarterly rang for the block-wise strategy is much smaller than BL1. This further supports our conclusion that selecting a block from the fourth quarter (the last 25% of the blocks) yields better performance and increased classification accuracy compared to the baselines. Furthermore the performance would be much more reliable and consistent using the block-wise strategy.

VI. Conclusion and Future Work

In this study, we proposed and evaluated a block-wise fine-tuning approach for optimizing the performance of pre-trained deep neural networks on new tasks. The experimental evaluation was conducted on two datasets, Oxford Flowers

(OXF) and Caltech 101 (CAL), using 15 commonly used pre-trained base models.

The results demonstrated that the block-wise approach consistently outperformed the baselines, Baseline I (tuning only the classifier layers) and Baseline II (tuning all layers), in terms of classification accuracy. The optimal block, although varying across models, consistently yielded better performance than the baselines. Furthermore, analysis of the first and last blocks revealed the superiority of the last block, indicating its importance in achieving improved performance. However, it should be noted that the optimized block does not necessarily correspond to the last block, emphasizing the complexity of identifying the most salient block within a base model.

Moreover, the investigation revealed a pattern where selecting a block from the fourth quarter of a base model generally resulted in improved performance compared to the baselines. This finding provides valuable insights for selecting the most relevant block in the block-wise fine-tuning process.

In conclusion, the block-wise fine-tuning approach demonstrated higher accuracy and reliability compared to the baselines. It offers a practical and effective method for adapting pre-trained base models to new tasks by selectively tuning specific blocks rather than the entire network.

Future research directions include exploring the effectiveness of block-wise fine-tuning on different types of models, investigating additional performance metrics beyond classification accuracy, developing advanced algorithms for automated selection of the optimal block, and extending the evaluation to other datasets and tasks. By pursuing these directions, can further enhance our understanding of block-wise fine-tuning and develop more efficient and accurate strategies for adapting pre-trained models to new tasks.

APPENDIX

The classification accuracy for each block for both datasets are presented in Figure 8

REFERENCES

[1] Rupesh Kumar Srivastava, Klaus Greff, Jurgen Schmidhuber, "Training Very Deep Networks", in *Proceedings of the 28th International Conference on Neural Information Processing Systems*, pp. 2377–2385, 2015.

[2] Bansal, M. A., Sharma, D. R., and Kathuria, D. M. A, "Systematic review on data scarcity problem in deep learning: Solution and applications", in *ACM Computing Surveys(CSUR)*, 2020.

[3] Li, H., Chaudhari, P., Yang, H., Lam, M., Ravichandran, A., Bhotika, R., and Soatto, S., "Rethinking the hyperparameters for fine-tuning", in *International Conference on Learning Representation (ICLR)*, 2020.

[4] M. -E. Nilsback and A. Zisserman, "Automated Flower Classification over a Large Number of Classes," 2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing, Bhubaneswar, India, 2008, pp. 722-729, doi: 10.1109/ICVGIP.2008.47.

[5] Li, F.-F., Andreeto, M., Ranzato, M., and Perona, P. (2022). Caltech 101 (1.0) [Data set]. CaltechDATA. https://doi.org/10.22002/D1.20086

[6] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg & Li Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge", in *International Journal of Computer Vision*, volume 115, pp. 211–252, 2015.

[7] Karen Simonyan and Andrew Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition", in *International Conference on Learning Representations*, 2015.

[8] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna, "Rethinking the inception architecture for computer vision", In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826, 2016.

[9] Andrew G. Howard and Menglong Zhu and Bo Chen and Dmitry Kalenichenko and Weijun Wang and Tobias Weyand and Marco Andreetto and Hartwig Adam, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications", *ArXiv, abs/1704.04861*, 2017.

[10] He, K., Zhang, X., Ren, S. and Sun, J., 2016. Identity mappings in deep residual networks. In Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14 (pp. 630-645). Springer International Publishing.

[11] Yaxing Wang, Chenshen Wu, Luis Herranz, Joost van de Weijer, Abel Gonzalez-Garcia, Bogdan Raducanu, "Transferring GANs: generating images from limited data", in *European Conference on Computer Vision*, pp. 220–236, 2018

[12] Jiamei Sun, Sebastian Lapuschkin, Wojciech Samek, Alexander Binder, "Explain and Improve: LRP-Inference Fine-Tuning for Image Captioning Models", in *Information Fusion*, vol.77, pp. 233–246, 2022.

[13] Jaemin Cho, Seunghyun Yoon, Ajinkya Kale, Franck Dernoncourt, Trung Bui, Mohit Bansal, "Findings of the Association for Computational Linguistics", in NAACL, pp. 517–527, 2022.

[14] Zongwei Zhou, Jae Shin, Lei Zhang, Suryakanth Gurudu, Michael Gotway, and Jianming Liang, "Fine-tuning Convolutional Neural Networks for Biomedical Image Analysis: Actively and Incrementally?", in *Computer Vision and Pattern Recognition*, pp. 7340–7351, 2017

[15] Simon Kornblith, Jonathon Shlens, and Quoc V. Le., "Do better imagenet models transfer better?", in *Computer Vision and Pattern Recognition (CVPR)*, pp. 2661–2671, 2019.

[16] M. Tanveer, M. Karim Khan and C. Kyung, "Fine-Tuning DARTS for Image Classification", in *International Conference on Pattern Recognition (ICPR)*, pp. 4789–4796, Milan, Italy, 2021.

[17] Heechul Jung Sihaeng Lee Junho Yim Sunjeong Park Junmo Kim, "Joint Fine-Tuning in Deep Neural Networks for Facial Expression Recognition", in *International Conference on Computer Vision*, pp. 2983-2991, 2015.

[18] Younmgin Ro1, Jin Young Choi, "AutoLR: Layer-wise Pruning and Auto-tuning of Learning Rates in Fine-tuning of Deep Networks", in *AAAI Conference on Artificial Intelligence*, pp. 2486–2494, 2021

[19] Hamdi, S., Snoussi, H., Abid, M., "Fine-Tuning a Pre-trained CAE for Deep One Class Anomaly Detection in Video Footage", in *Pattern Recognition and Artificial Intelligence*, Communications in Computer and Information Science, vol 1322. Springer, 2020.

[20] Rippel, O., Chavan, A., Lei, C., and Merhof, D., "Transfer Learning Gaussian Anomaly Detection by Fine-Tuning Representations", in *ArXiv, abs/2108.04116.*, 2021.

[21] Jhih-Ciang Wu, Ding-Jie Chen, Chiou-Shann Fuh, and Tyng-Luh Liu, "Learning Unsupervised Metaformer for Anomaly Detection", in *International Conference on Computer Vision*, pp. 4369–4378, 2021.

[22] Jian Xu, Cunzhao Shi, Chengzuo Qi, Chunheng Wang, Baihua Xiao, "Unsupervised Part-Based Weighting Aggregation of Deep Convolutional Features for Image Retrieval", in *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 7436–7443, vol.32, no.1, 2018.

[23] F. Radenović, G. Tolias and O. Chum, "Fine-Tuning CNN Image Retrieval with No Human Annotation", in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 7, pp. 1655–1668, 2019.

[24] C. Liu, B. Zoph, M. Neumann, J. Shlens, W. Hua, L.-J. Li, L. Fei-Fei, A. Yuille, J. Huang, and K. Murphy, "Progressive neural architecture search", in *Lecture Notes in Computer Science*, pp. 1–16, 2018.

[25] H. Pham, M. Y. Guan, B. Zoph, Q. V. Le, J. Dean, "Efficient neural architecture search via parameter sharing", in *Proc. of the International Conference on Machine Learning*, pp. 4092–4101, 2018

[26] Z. Lu, G. Sreekumar, E. D. Goodman, W. Banzhaf, K. Deb, V. N. Boddeti, "Neural architecture transfer", in *IEEE transactions on pattern analysis and machine intelligence*, vol.43, no.9, pp. 2971–2989, 2021.

[27] Youngkee Kim, Won Joon Yun, Youn Kyu Lee, Soyi Jung, Joongheon Kim, "Two-stage architectural fine-tuning with neural architecture search using early-stopping in image classification", in *https://doi.org/10.48550/arXiv.2202.08604*, 2022.

[28] Yosinski, J., Clune, J., Bengio, Y., and Lipson, H., "How transferable are features in deep neural networks?", in *Advances in neural information processing systems*, pp. 3320–3328, 2014.

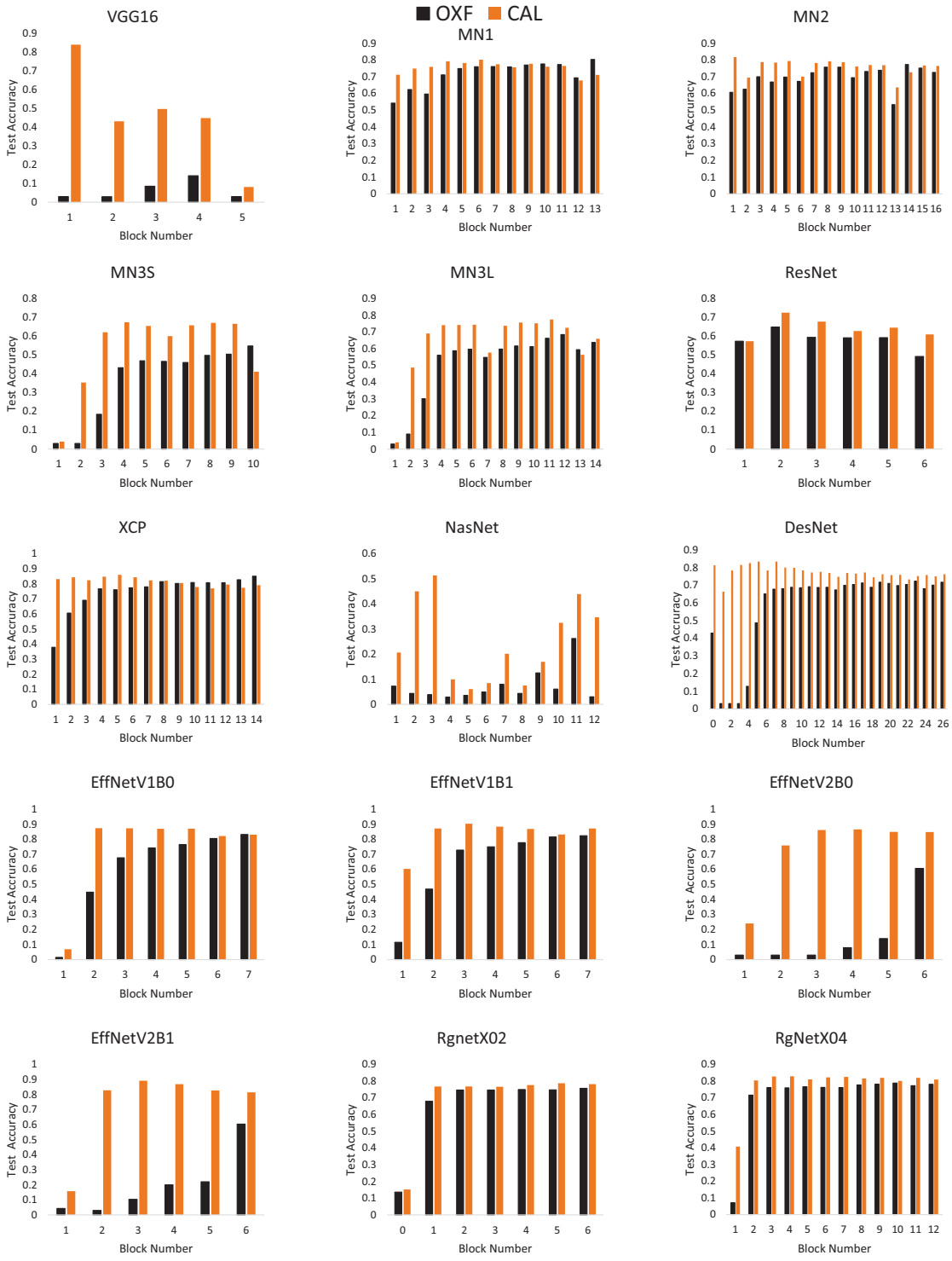

Fig. 8: Classification accuracy per block number for both datasets.

[29] Zeiler, M. D., Fergus, R., "Visualizing and understanding convolutional networks", in *The European Conference on Computer Vision (ECCV)*, pp. 818–833, 2014.

[30] Tajbakhsh, N., Shin, J. Y., Gurudu, S. R., Hurst, R. T., Kendall, C. B., Gotway, M. B., and Liang, J., "Convolutional neural networks for medical image analysis: Full training or fine tuning?", in *IEEE transactions on medical imaging vol.35(5)*, pp. 1299–1312, 2016.

[31] Guo, Y., Shi, H., Kumar, A., Grauman, K., Rosing, T., and Feris, R., "SpotTune: transfer learning through adaptive fine-tuning", in *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pp. 4805–4814, 2019.

[32] Tanvir Mahmud and Natalia Frumkin and Diana Marculescu, "RL-Tune: A Deep Reinforcement Learning Assisted Layer-wise Fine-Tuning Approach for Transfer Learning", in *First Workshop on Pre-training: Perspectives, Pitfalls, and Paths Forward at ICML*, 2022.

[33] Bharat Singh and Soham De and Yangmuzi Zhang and Thomas A. Goldstein and Gavin Taylor, "Layer-Specific Adaptive Learning Rates for Deep Networks", in *IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pp. 364–368, 2015

[34] Kornblith, S., Shlens, J., and Le, Q. V., "Do better imagenet models transfer better?", in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2661–2671, 2019.

[35] Parker-Holder, J., Nguyen, V., and Roberts, S. J., "Provably efficient online hyperparameter optimization with population-based bandits", in *Advances in Neural Information Processing Systems*, vol.33, pp. 17200–17211, 2020.

[36] Bottou, Léon, Bousquet, Olivier, "The Tradeoffs of Large Scale Learning", In *Optimization for Machine Learning*, Cambridge, MIT Press. pp. 351–368, 2012.

[37] Kingma, D. P., Ba, L. J., "Adam: A method for Stochastic Optimization", in *International Conference on Learning Representations (ICLR)*, 2015.

[38] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszko- reit, and Neil Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale", in *International Conference on Learning Representations (ICLR)*, 2021.

[39] Ananya Kumar, Aditi Raghunathan, Robbie Jones, Tengyu Ma, and Percy Liang, "Fine-tuning can distort pretrained features and underperform out-of-distribution", in *International Conference on Learning Representations (ICLR)*, 2022

[40] Ilya Loshchilov and Frank Hutter, "Decoupled weight decay regularization", in *International Conference on Learning Representations*, 2019.

[41] Kumar, Ananya and Shen, Ruoqi and Bubeck, Sébastien and Gunasekar, Suriya, "How to Fine-Tune Vision Models with SGD", in *https://arxiv.org/abs/2211.09359*, 2022

[42] Jeremy Howard and Sebastian Ruder, "Universal Language Model Fine-tuning for Text Classifier", in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pp. 328–339, 2018.

[43] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean, "Distributed Representations of Words and Phrases and their Compositionality", in *Advances in Neural Information Processing Systems*, pp. 3111–3119, 2013.

[44] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al., "Language models are few-shot learners", in *Advances in Neural Information Processing Systems*, 2020.

[45] Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilic et al., "BLOOM: A 176B-Parameter Open-Access Multilingual Language Model", in *https://doi.org/10.48550/arXiv.2211.05100*, 2022.

[46] Fedus, W., Zoph, B., and Shazeer, N., "Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity", in *Journal of Machine Learning Research*, vol.23, pp. 1–39, 2022.

[47] Li, X. L. and Liang, P., " Prefix-tuning: Optimizing continuous prompts for generation", in *59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pp. 4582–4597, 2021.

[48] He, J., Zhou, C., Ma, X., Berg-Kirkpatrick, T., and Neubig, G., "Towards a unified view of parameter-efficient transfer learning", in *International Conference on Learning Representations (ICLR)*, 2022.

[49] Xinhsuai Dong, Luu Anh Tuan, Min Lin, Shuicheng Yan, Hanwang Zhang, "How Should Pre-Trained Language Models Be Fine-Tuned Towards Adversarial Robustness?", in *35th Conference on Neural Information Processing System*, 2021.

[50] Fatemehsadat Mireshghallah, Archit Uniyal, Tianhao Wang, David Evans, Taylor Berg-Kirkpatrick, "Memorization in NLP Fine-tuning Methods", in *https://doi.org/10.48550/arXiv.2205.12506*, 2022.

[51] Wen-Chin Huang and Chia-Hua Wu and Shang-Bao Luo and Kuan-Yu Chen and Hsin-Min Wang and Tomoki Toda, "Speech Recognition by Simply Fine-Tuning Bert", in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7343–7347, 2021.

[52] Violeta, L., Ma, D., Huang, Wen-Chin and Toda, T., "Intermediate Fine-Tuning Using Imperfect Synthetic Speech for Improving Electrolaryngeal Speech Recognition", in *10.48550/arXiv.2211.01079*, 2022

[53] Ioannis Tsiamas, Gerard I. Gállego, Carlos Escolano, José A. R. Fonollosa, Marta R. Costa-jussà, "Pretrained Speech Encoders and Efficient Fine-tuning Methods for Speech Translation: UPC at IWSLT 2022", in *Proceedings of the 19th International Conference on Spoken Language Translation (IWSLT 2022)*, pp. 265–276, 2022.

[54] Séverine Guillaume, Guillaume Wisniewski, Cécile Macaire, "Fine-tuning pre-trained models for Automatic Speech Recognition: experiments on a fieldwork corpus of Japhug", in *Proceedings of the Fifth Workshop on the Use of Computational Methods in the Study of Endangered Languages*, pp. 170–178, 2022.

[55] Jan Vanek, Josef Michálek and Josef Psutka, "Tuning of Acoustic Modeling and Adaptation Technique for a Real Speech Recognition Task", in *International Conference on Statistical Language and Speech Processing*, pp. 235–245, 2019.

[56] Linkai Peng, Kaiqi Fu, Binghuai Lin, Dengfeng Ke, Jinsong Zhang, "A study on fine-tuning wav2vec2.0 Model for the task of Mispronunciation Detection and Diagnosis", in *InterSpeech*, pp. 4448–4451, 2021.

[57] Haidar, M. A. and Rezagholizadeh, M., "Fine-Tuning of Pre-Trained End-to-End Speech Recognition with Generative Adversarial Networks", in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6204–6208, 2021.

[58] Goodfellow, Ian etc., "Generative Adversarial Nets", in *Proceedings of the International Conference on Neural Information Processing Systems*, pp. 2672–2680, 2014.

[59] Saini, M., Susan, S. Bag-of-Visual-Words codebook generation using deep features for effective classification of imbalanced multi-class image datasets. Multimed Tools Appl 80, 20821–20847 (2021). https://doi.org/10.1007/s11042-021-10612-w

[60] Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., ; Adam, H. (2017). MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. https://doi.org/10.48550/arxiv.1704.04861

[61] A. Howard et al., "Searching for MobileNetV3," 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea (South), 2019, pp. 1314-1324, doi: 10.1109/ICCV.2019.00140.

[62] Chollet, F., 2021. Deep learning with Python. Simon and Schuster.

[63] Goodfellow, I., Bengio, Y. and Courville, A., 2016. 'Deep learning'. MIT press.

[64] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. and Salakhutdinov, R., 2014. Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research, 15(1), pp.1929-1958.

[65] J. Xu, Y. Pan, X. Pan, S. Hoi, Z. Yi and Z. Xu, "RegNet: Self-Regulated Network for Image Classification," in IEEE Transactions on Neural Networks and Learning Systems, doi: 10.1109/TNNLS.2022.3158966.

[66] Tan, M. and Le, Q., 2021, July. Efficientnetv2: Smaller models and faster training. In International conference on machine learning (pp. 10096-10106). PMLR.

[67] Tan, M. and Le, Q., 2019, May. Efficientnet: Rethinking model scaling for convolutional neural networks. In International conference on machine learning (pp. 6105-6114). PMLR.

[68] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions," in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 2017 pp. 1800-1807.

[69] Zoph, B., Vasudevan, V., Shlens, J. and Le, Q.V., 2018. Learning transferable architectures for scalable image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 8697-8710).

[70] G. Huang, Z. Liu, L. Van Der Maaten and K. Weinberger, "Densely Connected Convolutional Networks," in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 2017 pp. 2261-2269.