Towards Meta-learning of Deep Architectures for Efficient Domain Adaptation

Abbas Raza Ali[1][0000−0003−0505−2638], Marcin Budka[1][0000−0003−0158−6309], and Bogdan Gabrys[2][0000−0002−0790−2846]

[1] Bournemouth University, Poole BH12 5BB, UK
{aali,mbudka}@bournemouth.ac.uk
[2] University Technology Sydney, Ultimo NSW 2007, Australia
bogdan.gabrys@uts.edu.au

Abstract. This paper proposes an efficient domain adaption approach using deep learning along with transfer and meta-level learning. The objective is to identify how many blocks (i.e. groups of consecutive layers) of a pre-trained image classification network need to be fine-tuned based on the characteristics of the new task. In order to investigate it, a number of experiments have been conducted using different pre-trained networks and image datasets. The networks were fine-tuned, starting from the blocks containing the output layers and progressively moving towards the input layer, on various tasks with characteristics different from the original task. The amount of fine-tuning of a pre-trained network (i.e. the number of top layers requiring adaptation) is usually dependent on the complexity, size, and domain similarity of the original and new tasks. Considering these characteristics, a question arises of how many blocks of the network need to be fine-tuned to get maximum possible accuracy? Which of a number of available pre-trained networks require fine-tuning of the minimum number of blocks to achieve this accuracy? The experiments, that involve three network architectures each divided into 10 blocks on average and five datasets, empirically confirm the intuition that there exists a relationship between the similarity of the original and new tasks and the depth of network needed to fine-tune in order to achieve accuracy comparable with that of a model trained from scratch. Further analysis shows that the fine-tuning of the final top blocks of the network, which represent the high-level features, is sufficient in most of the cases. Moreover, we have empirically verified that less similar tasks require fine-tuning of deeper portions of the network, which however is still better than training a network from scratch.

Keywords: Computer Vision · Convolutional Neural Networks · Deep Learning · Domain Adaption · Meta-Learning · Transfer Learning

1 Introduction

Deep learning has demonstrated tremendous success in various domains, particularly Computer Vision, and Speech and Language Processing (SLP) [28,

25], where it is consistently outperforming traditional machine learning approaches [2]. Among the key developments in the field of deep learning, Convolutional Neural Networks (CNNs) stand out as the workhorse of Computer Vision. Training a large CNN with millions of parameters is a computationally intensive task which also requires a significant amount of training data. However, several state-of-the-art image classification architectures trained on large image datasets are publicly available, including Visual Geometry Group Network (VGGNet) [20], Inception [22], Residual Networks (ResNet) [9] and Inception-ResNet [21]. These networks are trained on the ImageNet [18] dataset which consists of 1.2 million images and 1000 classes.

Training of these types of deep networks from scratch on a huge dataset is a computationally demanding task, e.g. training of models on each dataset used in this work usually takes a few days of processing time using an Nvidia GTX 1080 GPU. As a result, transfer learning, i.e. reusing parts of the pre-trained models either as-is or as a starting point within the training process, quickly became a de-facto standard in Computer Vision tasks. The general consensus seems to be that the more data one has, the more 'aggressive' the re-training process can be (e.g. re-training more final layers). Conversely, the more similar the new dataset is to the one used to train the original model, the fewer layers need to be fine-tuned. Despite the wide adoption of transfer learning in the context of CNNs, to the best of our knowledge, there is still no principled way of approaching this process. The number of layers to re-train or even the network architectures themselves are chosen in an ad-hoc manner and tested one after the other, which is a computationally inefficient procedure.

This paper proposes and investigates a new approach to adapt pre-trained CNNs to new domains using the Meta-level Learning paradigm. Meta-learning, also known as 'learning to learn', was introduced around three decades ago [24], and was initially limited to classification and clustering tasks [1, 12]. Recently, it has also been used in deep learning for the selection of hyper-parameters of a specific architecture. [15] proposed a comprehensive set of global and node level hyper-parameters which are critical to optimizing deep learning architectures through evolution. The use of Reinforcement learning to generate CNN and Recurrent Neural Network (RNN) architectures have been proposed by [3] and [29]. They have used Q-learning to produce new CNN architectures. [7] introduced a simple but powerful approach, model-agnostic meta-learning, which provides an optimal initialization of model parameters that lead to fast learning on new tasks.

Transfer learning has been positioned to effectively adapt pre-trained networks to a new domain by fine-tuning their final layers. Some studies, such as [26] and [19], propose re-training of only final fully-connected (FC) layers of the network which does not guarantee state-of-the-art accuracy, particularly on relatively dissimilar tasks. On the contrary, domain adaptation becomes beneficial by fine-tuning an increasing number of layers based on the complexity and relevance of the new task [27]. Therefore, a question arises as to how many blocks

need fine-tuning to adapt to a new domain based on the complexity, size and domain relevance.

The rest of the paper is organized as follows. Section 2 is devoted to elaborating the overall approach of this work. Section 3 outlines the methodology of this study. Section 4 reports the experimental results followed by their analysis. Finally, the paper is concluded in Section 5.

2 Methodology

In order to carry out the investigations, a platform has been implemented to conduct experiments with different combinations of pre-trained networks, their hyper-parameters, and image datasets. The experiments have been designed to investigate the relationships among these three key components while fine-tuning the pre-trained networks on new tasks. There are several characteristics which can be considered but the two most important features selected for this study are the size and similarity of the new task. The four transfer learning scenarios are based on these two features. A schematic view of transfer learning based is shown in Figure 1 where Task-A is representing the original problem and Task-B the new problem datasets.

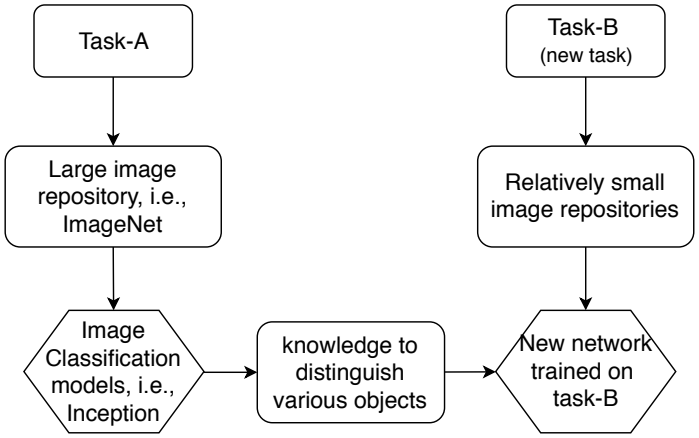


Fig. 1: Schematic diagram of transfer learning

Despite the popularity of transfer learning in computer vision, there is no principled way of finding the relation between characteristics of a dataset and depth of the network that needs to be re-trained. In this work, an effort has been made to find this relationship by identifying a pre-trained network where the minimum number of blocks need to be re-trained to achieve state-of-the-art accuracy. Moreover, instead of learning the general characteristics of the dataset which is usually practiced in shallow learning, e.g. feature statistics [1], a higher

level characteristics have been pursued, such as layer activations. The focus of the experiments was to investigate the following key scenarios:

1. If Task-B is small in size and similar to Task-A (e.g. both tasks are concerned with natural images), re-training of the entire network might lead to over-fitting. The higher-level features of the pre-trained network, Task-A, are usually relevant for Task-B. Hence, the re-training of a single or a few final layer(s) becomes very effective.
2. If Task-B is large and similar to Task-A, there is less possibility of over-fitting while fine-tuning more layers of the network.
3. If Task-B is small but less similar to Task-A, there is a possibility that Task-A does not contain relevant features for Task-B. In this case transfer learning might not be very useful, however, re-training of final layers might give reasonable results.
4. If Task-B is large and very different from Task-A, both the training of the network from scratch and initialization of the network with the weights of the pre-trained model would be beneficial.

Figure 2 is summarising the above four scenarios.

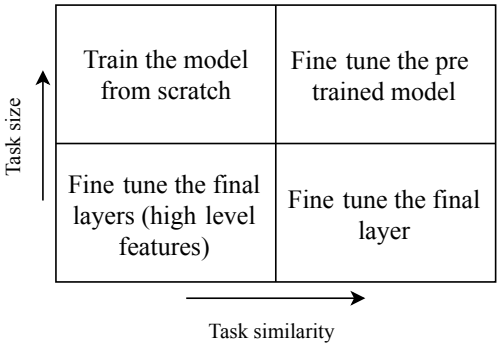


Fig. 2: Transfer learning scenarios

Datasets with appropriate characteristics have been gathered for the experiments to cover the above four scenarios. The network architectures used in this study vary greatly, hence we fine-tune groups of layers rather than individual layers. Please refer to Figure 3 for the details of how the layers of each architecture have been grouped into what we refer to as 'blocks'.

The pre-trained networks have been fine-tuned on each of the new tasks. The experimental approach was to fine-tune an iteratively increasing number of blocks of each network, starting from the final block, while the lower blocks of the network act as a fixed feature extractor for Task-B. The train and test accuracies have been recorded on every iteration. In some cases, where Task-B is similar, the re-training of only the final layer produces close to the state-of-the-art accuracy. On the contrary, it is hardly applicable when both tasks are

very different. In that case, more final layers need to be re-trained. In general, a network learns the hierarchy of features starting from generic ones, e.g., colors, edges, curves, etc., which can be reused for most of the tasks. Conversely, the later layers respond to more specific features of the original task which can only be reusable in case the new task is similar.

3 Experimentation Environment

To further investigate the questions raised in the previous section, a comprehensive experimentation environment has been setup. It comprises of five datasets of different characteristics and three state-of-the-art pre-trained image classification networks. The complexity of the experiments has been calculated as the number of datasets times the number of trainable blocks of all the networks. Therefore, computational power becomes a critical factor to perform these experiments in a reasonable time. There were five GPUs used to train around 200 models.

Table 1: Open-source image repositories

Dataset	Training-set	Testing-set	Classes	Avg. Class Size
ImageNet [18]	1.2 million	50,000	1,000	1,200
Food [4]	75,750	25,250	101	1,000
Caltech-101 [6]	6,144	2,096	101	82
Chest-Xray [5]	5,943	1,487	2	3,715
Flowers [16]	2,753	917	5	734
Coco-Animals [14]	800	200	8	125

3.1 Datasets

In this work, five publicly available datasets have been used with different domain and characteristics. They can be divided into two categories based on their size and number of classes; large and small as shown in Table 1. The pre-trained networks which are selected for this work are trained on ImageNet. The Food dataset, introduced by [4], is a challenging collection of 101 food categories and 101,000 instances. Likewise, Caltech dataset also has 101 categories with 82 images per category on average [6]. The images are not specific to any particular domain. Chest-Xray [5] is a relatively smaller dataset, originally published with 14 classes. The images were mostly tagged with multiple labels which are converted to a two-class problem where every image can be classified as either normal or nodule. This dataset is composed of frontal-view X-ray images of the screening and diagnosis of many lung-related diseases. Similarly, Flowers is another small dataset consisting of five different categories of flower species [16]. Microsoft has gathered a large dataset consisting of 91 categories, known as Common Objects in Context (Coco) [14]. Coco-Animals (Animals) is a subset of the original Coco dataset which is composed of 8 animal categories.

Table 2: Benchmarking of various pre-trained image classification models

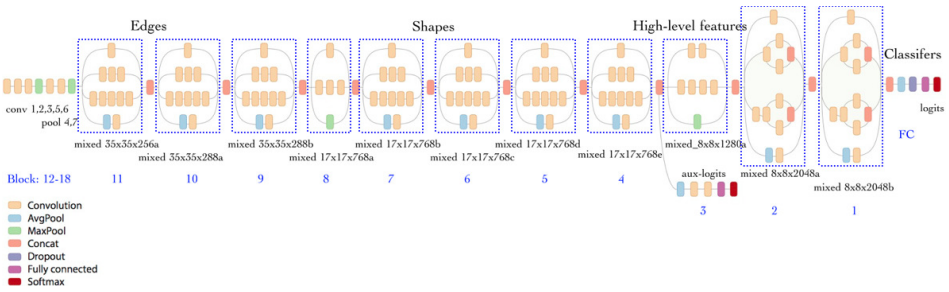
Network	Layers	Top-1 Accuracy	Top-5 Accuracy
Inception-v3 [23]	22	78.0%	93.9%
Inception-ResNet-v2	152	80.4%	95.3%
VGG-19 [20]	19	71.1%	89.8%

3.2 Pre-trained Image Classification Networks

Three pre-trained image classification and detection CNNs have been used in this work. These networks are trained on ImageNet dataset which consists of 1000 classes [18], however, their internal architecture, depth, and other aspects differ considerably. The first few layers of the networks capture low-level features of the image like edges, curves, etc. The subsequent layers learned shapes and more abstract features related to the problem domain. The final layers have learned more specific features corresponding to a particular category which is eventually used to classify the images. The pre-trained networks are listed in Table 2 along with the number of layers and accuracy in the ImageNet dataset.

Inception-ResNet-v2 Google released Inception-ResNet in 2016 and it became a state-of-the-art image classification network of ILSVRC-2016. Inception-ResNet-v2 is a deeper but simplified version of Inception-v3. The residual connections allow the model to be even deeper, leading to better performance. ResNet relies on micro-architecture modules which consist of building blocks.

A schematic view of different pre-trained architectures can be seen in Figure 3. The architectures are also labelled with the block numbers, in blue, that can be subject to fine-tuning.



VGG-19 VGG network was developed by Visual Geometry Group of Oxford University which secured first place in the ImageNet ILSVRC-2014. It has two versions which consist of 16 and 19 layers. The 19 layer version has been used

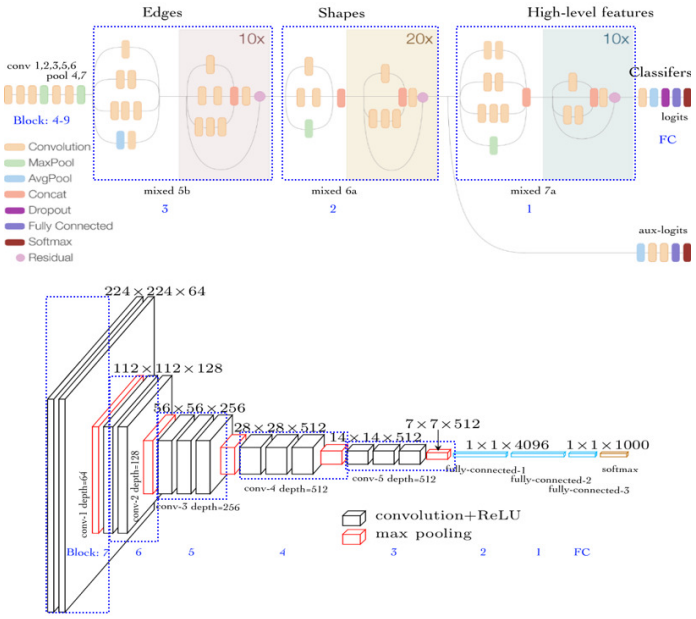


Fig. 3: Schematic view of Inception-v3, Inception-ResNet-v2 and VGG-19 networks where the blue colour is representing a re-trainable layer/block.

in our experiments. The VGG network uses 3x3 convolutions stacked on top of each other in increasing depth which makes it relatively simpler than AlexNet. The convolutional layers are followed by two FC layers, each one consisting of 4,096 neurons, and a Softmax classifier.

Inception-v3 Inception, or GoogLeNet, was developed by Google and was state-of-the-art for image classification and detection in the ILSVRC-2015. Inception-v3 is a 22 layers deep network but computationally inexpensive [22].

3.3 Transfer Learning

In transfer learning, three pre-trained networks are re-trained/fine-tuned sequentially on the same task. The training process fine-tunes a range of blocks per training iteration, starting from the final block. This process has been repeated for all the pre-trained networks and datasets. The hyper-parameters have been also updated layer-wise one by one where the learning rate initializes from a comparatively large number to iteratively smaller. Conversely, the number of training epochs parameter has been initialized from a smaller number which gets bigger as more layers need to re-train. The *rmsprop* optimizer [11] and layer dropout of 20-30% have been used while re-training the network. The learning rate and the number of training epochs are dependent on the nature of the tasks

and depth of the network. The training begins with the higher value of learning rate and lower number of epochs which gradually decreases and increases, respectively, as more layers of the network require fine-tuning. Their values are changed with a small factor upon the addition of a new layer for fine-tuning. The idea is to use the lower value of learning rate and a higher number of epochs for larger datasets. Table 3 is showing hyper-parameters that are used in our experiments.

Table 3: Hyper-parameters that are used for transfer Learning

Datasets	Learning rate	Epochs	Dropout
Food	$10^{-3} - 10^{-7}$	180-1000	20%
Caltech	$10^{-4} - 10^{-7}$	180-1000	20%
Chest-Xray	$10^{-3} - 10^{-6}$	120-800	20-30%
Flowers	$10^{-3} - 10^{-6}$	120-800	20%
Animals	$10^{-3} - 10^{-6}$	120-800	20%

4 Results and Analysis

An extensive set of experiments has been performed to analyze the relationship of size and similarity of a task with the depth of pre-trained network that needs to be fine-tuned. The depth of the pre-trained networks, which is fine-tuned, is varied from 7 to 18 blocks. The layer-wise training and validation accuracies have been reported in Table 4. The table shows accuracies of five datasets against three different architectures and the number of fine-tuned blocks. The top-performing numbers of blocks are in bold. The relationship between the validation accuracy and the number of blocks has been depicted in Figure 4.

The accuracy of a pre-trained network after fine-tuning every block, also know as block-wise result, is validated with dataset similarity analysis. The networks that are used in this work were originally trained on the ImageNet dataset. Therefore, the validation set of all the datasets have been inferred by the pre-trained networks to compute their similarity with ImageNet. As a result, the maximum of the probability mass function of an image over the 1000 classes, which is referred to as image similarity to ImageNet, and entropy have been calculated and averaged over the number of images N in the dataset. The similarity and entropy are calculated using Equations 1 and 2, respectively.

$$\text{similarity} = \frac{1}{N}\sum_{i=1}^{N} \max(\mathbf{f}(x_i)) \tag{1}$$

where $\mathbf{f}(x_i)$ is the probability mass function over classes conditioned on the input image x_i, typically the output of the Softmax layer.

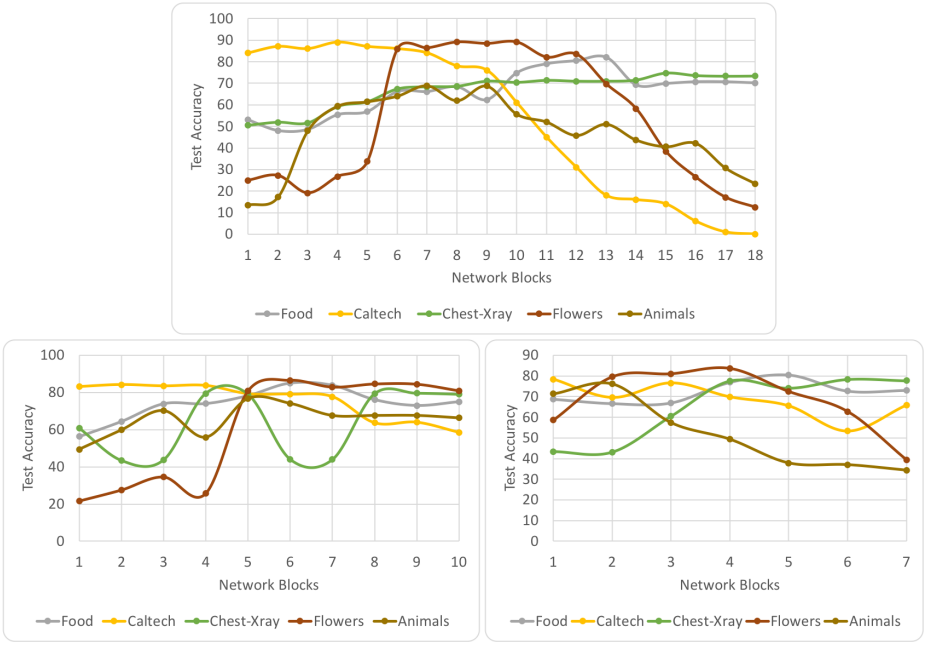


Fig. 4: Transfer learning accuracies of pre-trained networks; (a) Inception, (b) Inception-ResNet and (c) VGG-19 on ImageNet

Table 4: Transfer learning accuracies of the various datasets, classification architectures, and their layers

Network re-training accuracy (train %–test %) upon fine-tuning a range of blocks, one block per iteration															
Dataset	FC	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Inception-v3 network															
Food	85–53	84–48	84–49	72–55	70–57	73–66	73–66	76–68	79–62	84–75	82–79	85–80	**86–82**	79–69	80–70
Caltech	95–84	95–87	96–86	**96–89**	96–87	95–86	93–84	88–78	82–76	69–61	48–45	37–31	24–18	15–16	12–14
Chest-Xray	96–50	96–52	96–52	96–59	96–61	96–67	96–68	96–69	96–71	96–70	96–71	95–71	92–71	78–72	**87–75**
Flowers	84–25	84–27	87–19	88–27	91–34	95–86	96–86	**96–89**	96–88	95–89	92–82	86–83	78–70	56–58	41–38
Animals	54–14	47–17	58–48	81–59	88–61	90–64	**92–69**	91–62	90–69	88–56	81–52	69–46	58–51	47–44	38–41
Inception-ResNet-v2 network															
Food	86–56	86–64	86–74	86–74	87–78	**89–85**	86–84	80–76	71–73	78–75	–	–	–	–	–
Caltech	96–83	**96–84**	95–83	94–84	94–79	93–79	91–78	88–64	79–64	67–59	–	–	–	–	–
Chest-Xray	91–61	89–43	88–44	91–79	91–79	91–44	91–44	91–79	**91–80**	91–79	–	–	–	–	–
Flowers	89–22	89–28	91–35	92–26	93–81	**94–87**	95–83	96–85	96–84	94–81	–	–	–	–	–
Animals	65–49	71–60	77–70	77–56	**82–77**	85–74	86–68	88–68	88–68	85–66	–	–	–	–	–
VGG-19 network															
Food	85–69	85–67	85–67	90–77	**81–80**	77–73	77–73	–	–	–	–	–	–	–	–
Caltech	**79–78**	72–70	80–77	74–70	68–66	66–53	71–66	–	–	–	–	–	–	–	–
Chest-Xray	89–43	87–43	89–61	88–78	89–74	**89–78**	89–78	–	–	–	–	–	–	–	–
Flowers	83–59	81–80	83–81	**86–84**	79–72	79–63	90–39	–	–	–	–	–	–	–	–
Animals	78–71	**79–76**	70–57	74–49	72–38	73–37	79–34	–	–	–	–	–	–	–	–

$$\text{average entropy} = \frac{1}{N}\sum_{i=1}^{N}(-\sum_{j=1}^{1000}(\mathbf{f}_j(x_i) * \log_2(\mathbf{f}_j(x_i)))) \quad (2)$$

The similarity of an image from the new domain with the original domain is computed by feeding the image to the original pre-trained network and examining the output probability distribution over the classes. The dataset similarity scores have been recorded in Table 5. The similarity results are correlated with the number of blocks that are needed to fine-tune networks on new tasks. Figure 5(a-c) shows that for tasks where similarity is higher (and the entropy is lower), fewer blocks need to be fine-tuned. On the contrary, more blocks need to be fine-tuned where the datasets are less similar (having low similarity and higher entropy values). This supports our claim that transfer learning is effective for related tasks regardless of their size. However, transfer learning is also useful for dissimilar tasks, i.e., Chest-Xray and Food, but more blocks need to be re-trained to get good results. Moreover, similar tasks require fine-tuning of either only fully-connected layer(s) or high-level features block in some cases. Accordingly, less similar tasks require fine-tuning of more deeper layers, i.e., blocks representing shapes and edges blocks.

Table 5: The similarity and average entropy of different datasets

Dataset	Inception-v3	Inception-ResNet-v2	VGG-19
ImageNet	76.61% – 2.11	78.77% – 1.84	72.7% – 2.23
Food	53.40% – 3.52	59.23% – 3.47	51.24% – 3.83
Caltech	60.41% – 3.27	64.30% – 2.62	57.58% – 2.40
Chest-Xray	40.88% – 4.88	43.25% – 4.40	34.72% – 4.74
Flowers	52.25% – 4.01	60.19% – 3.15	49.72% – 3.12
Animals	54.88% – 3.68	64.87% – 2.68	53.08% – 2.79

Figure 5(d) shows that the size meta-feature has a good correlation with the depth of the network that is fine-tuned. The contribution of the similarity of a dataset dominates over its size when both datasets are similar. However, size becomes critical when both datasets have less similarity between them. It only supports the network to generalize while fine-tuning more deeper blocks of the network, e.g., Food and Chest-Xray dataset. The Food datasets consist of over 100,000 examples with over 100 classes whereas Chest-Xray has around 8,000 instances with only 2 classes. Based on the number of classes both datasets have a reasonable size to the class ratio which allow them to fine-tune more deeper networks.

The Food and Chest-Xray datasets' domains are different from ImageNet. Consequently, more deeper blocks have been fine-tuned. Transfer learning is more effective than training the model from scratch for these tasks. The maximum validation accuracy of fine-tuned Food and Chest-Xray is closer to the model which is trained from scratch. These accuracies as compared to the training of

Table 6: The state-of-the-art accuracy (training of the network from scratch) versus maximum accuracy from this work

Dataset	Accuracy of the network from scratch	Architecture	Reference	Accuracy from this work
Food	88.28%	InceptionV3	[8]	84.93%
Caltech	91.44%	SPP-Net	[10]	89.00%
Chest-Xray	84.11%	CheXNet	[17]	79.52%
Flowers	91.52%	CNN-SVM	[13]	89.06%
Animals	-	-	-	76.70%

the network from scratch, reported by various studies, are presented in Table 6. However, transfer learning requires much less effort and resources, in terms of parameter tuning and computation.

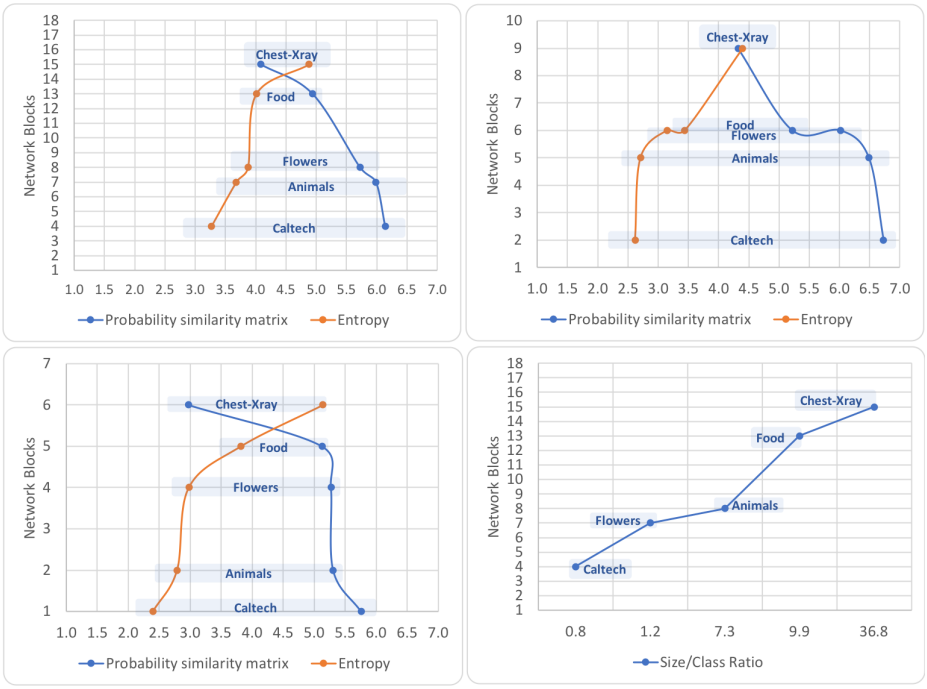


Fig. 5: Datasets similarity with ImageNet for: (a) Inception-v3, (b) Inception-ResNet-v2, (c) VGG-19. (d) Inception-v3 blocks vs dataset size/class ratio. The similarity is normalized to 1-10.

5 Conclusion

This paper presents an empirical study of the relationship between various characteristics describing the similarity of two datasets and based on that, the amount of fine-tuning required to achieve accuracy close to state-of-the-art. Even though the experiments were limited to only two characteristics, size and similarity with the original task, still as per some studies these are most important in this context. The datasets with both similar and different domains as well as different sizes have been used. Also, three state-of-the-art image classification networks trained on ImageNet were used in the experiments. Extensive experiments have been conducted on different combinations of pre-trained networks (and their blocks), datasets and hyper-parameters. The block-wise results are validated with dataset similarity analysis where the probability of match and entropy of the datasets are correlated with the fine-tuning of the number of blocks. The proposed approach first computes the similarity of the new task with the original one and combines it with the size of the new task to identify which section of the architecture needs fine-tuning.

The experiments were designed around two meta-features where only the datasets having different characteristics were considered. In general, transfer learning is found to be effective for tasks similar to the original one, regardless of the size, where mostly fine-tuning of the final blocks produces close to state-of-the-art accuracy. On the other hand, this work is handy for the tasks having less or no similarity with the original task with very few training examples, i.e., problems related to Medical Imaging [17]. It allows to find the minimum number of blocks a pre-trained network require fine-tuning to achieve the best possible accuracy based on the characteristics of two tasks. It also identifies the portion of the pre-trained network which can be reusable based on the similarity and size among two tasks. The key characteristic of transfer learning is that it saves significant computation and training time while achieving similar accuracy to the networks trained from scratch. This study preserves the key characteristics of transfer learning atleast for less similar tasks which verifies the intuition that one can more effectively reuse pre-trained network.

References

1. Ali, A., Gabrys, B., Budka, M.: Cross-domain meta-learning for time-series forecasting. Procedia Computer Science, Elsevier **126**, 9–18 (2018)
2. Alom, Z., Taha, T.M., Yakopcic, C., Westberg, S., et al.: The history began from alexnet: A comprehensive survey on deep learning approaches. Computing Research Repository (CoRR) **abs/1803.01164** (2018)
3. Baker, B., Gupta, O., Naik, N., Raskar, R.: Designing neural network architectures using reinforcement learning. Computing Research Repository (CoRR) **abs/1611.02167** (2016)
4. Bossard, L., Guillaumin, M., Gool, L.J.V.: Food-101 - mining discriminative components with random forests. In: ECCV (6). Lecture Notes in Computer Science, vol. 8694, pp. 446–461. Springer (2014)

5. Demner-Fushman, D., Kohli, M.D., Rosenman, M.B., Shooshan, S.E., et al.: Preparing a collection of radiology examinations for distribution and retrieval. Journal of the American Medical Informatics Association **23**(2) (2016)
6. Fei-Fei, L., Fergus, R., Perona, P.: Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. Comput. Vis. Image Underst. **106**(1), 59–70 (Apr 2007)
7. Finn, C., Abbeel, P., Levine, S.: Model-agnostic meta-learning for fast adaptation of deep networks. In: Proceedings of the 34th International Conference on Machine Learning. vol. 70, pp. 1126–1135. PMLR, International Convention Centre, Sydney, Australia (8 2017)
8. Hassannejad, H., Matrella, G., Ciampolini, P., De Munari, I., Mordonini, M., Cagnoni, S.: Food image recognition using very deep convolutional networks. In: Proceedings of the 2Nd International Workshop on Multimedia Assisted Dietary Management. pp. 41–49. MADiMa '16, ACM, New York, NY, USA (2016)
9. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 770–778 (2016)
10. Hem, K., Zhang, X., Ren, S., Sun, J.: Spatial pyramid pooling in deep convolutional networks for visual recognition. Computing Research Repository (CoRR) **abs/1406.4729** (2014)
11. Hinton, G., Srivastava, N., K., S.: Overview of mini-batch gradient descent lecture of neural networks for machine learning course (2014), http://www.cs.toronto.edu/ tijmen/csc321/slides/lecture_slides_lec6.pdf
12. Lemke, C., Budka, M., Gabrys, B.: Metalearning: a survey of trends and technologies. Artificial Intelligence Review **44** (7 2015)
13. Lin, K., Yang, H.F., Chen, C.S.: Flower classification with few training examples via recalling visual patterns from deep cnn. pp. 41–49. CVGIP (2015)
14. Lin, T., Maire, M., Belongie, S.J., Bourdev, L.D., et al.: Microsoft COCO: common objects in context. Computing Research Repository (CoRR) **abs/1405.0312** (2014)
15. Miikkulainen, R., Liang, J.Z., Meyerson, E., Rawal, A., et al.: Evolving deep neural networks. Computing Research Repository (CoRR) **abs/1703.00548** (2017)
16. Nilsback, M.E., Zisserman, A.: Automated flower classification over a large number of classes. In: 2008 Sixth Indian Conference on Computer Vision, Graphics Image Processing. pp. 722–729 (2008)
17. Rajpurkar, P., Irvin, J., Zhu, K., Yang, B., et al.: Chexnet: Radiologist-level pneumonia detection on chest x-rays with deep learning. Computing Research Repository (CoRR) **abs/1711.05225** (2017)
18. Russakovsky, O., Deng, J., Su, H., Krause, J., et al.: Imagenet large scale visual recognition challenge. International Journal Computer Vision **115**(3), 211–252 (Dec 2015)
19. Shin, H., Roberts, K., Lu, L., Demner-Fushman, D., Yao, J., Summers, R.M.: Learning to read chest x-rays: Recurrent neural cascade model for automated image annotation. Computing Research Repository (CoRR) **abs/1603.08486** (2016)
20. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. Computing Research Repository (CoRR) **abs/1409.1556** (2014)
21. Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.: Inception-v4, inception-resnet and the impact of residual connections on learning. In: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA. pp. 4278–4284 (2017)

22. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., et al.: Going deeper with convolutions. In: Computer Vision and Pattern Recognition (CVPR) (2015), http://arxiv.org/abs/1409.4842
23. Szegedy, C., Vanhoucke, V., Ioffe, S., et al.: Rethinking the inception architecture for computer vision. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016. pp. 2818–2826 (2016)
24. Vilalta, R., Drissi, Y.: A perspective view and survey of meta-learning. Artif. Intell. Rev. **18**(2), 77–95 (2002)
25. Wang, D., Zheng, T.F.: Transfer learning for speech and language processing. In: 2015 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA). pp. 1225–1237 (2015)
26. Wang, X., Peng, Y., Lu, L., Lu, Z., et al.: Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. Computing Research Repository (CoRR) **abs/1705.02315** (2017)
27. Yosinski, J., Clune, J., Bengio, Y., Lipson, H.: How transferable are features in deep neural networks? In: Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2. pp. 3320–3328. NIPS'14, MIT Press, Cambridge, MA, USA (2014)
28. Zhang, Z., Sun, Z., Liu, J., Chen, J., et al.: An experimental comparison of deep neural networks for end-to-end speech recognition. Computing Research Repository (CoRR) **abs/1611.07174** (2016)
29. Zoph, B., Le, Q.V.: Neural architecture search with reinforcement learning. Computing Research Repository (CoRR) **abs/1611.01578** (2016)