Efficient Transfer Learning Towards Constrained Environments

Tom Pégeot

université PARIS-SACLAY

Efficient Transfer Learning Towards Constrained Environments

Apprentissage par Transfert Efficient pour des Environnements Contraints

Thèse de doctorat de l'université Paris-Saclay

École doctorale n°580: Sciences et technologies de l'information et de la communication (STIC)
Spécialité de doctorat: Informatique
Graduate School: Informatique et sciences du numérique
Référent: Faculté des sciences d'Orsay

Thèse préparée à l'Institut LIST (Université Paris-Saclay, CEA), sous la direction de **Bertrand DELEZOIDE**, professeur associé, HDR, le co-encadrement de **Inna KUCHER**, Ingénieur-chercheur, et le co-encadrement de **Adrian POPESCU**, Ingénieur-chercheur, HDR

Thèse soutenue à Paris-Saclay, le 16 décembre 2024, par

Tom PEGEOT

Composition du jury

Membres du jury avec voix délibérative

Wassila OUERDANE Professeure, HDR, CentraleSupelec, Université Paris-Saclay	Présidente
Bertrand GRANADO Professeur, HDR, Sorbonne Université	Rapporteur & Examinateur
Olivier SENTIEYS Professeur, HDR, Université de Rennes	Rapporteur & Examinateur
Hedi TABIA Professeur, HDR, Université Paris-Saclay	Examinateur
Maria A. ZULUAGA Maîtresse de conférences, HDR, EURECOM	Examinatrice

THESE DE DOCTORAT
NNT : 2024UPASG101

université PARIS-SACLAY

ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Titre: Apprentissage par Transfert Efficient pour des Environnements Contraints
Mots clés: Apprentissage automatique, Apprentissage par transfert, Réseaux de neurones artificiels

Résumé: Cette thèse explore des techniques d'apprentissage par transfert efficientes pour des environnements contraints, où la réduction du nombre de paramètres ajustables devient centrale.

Dans un premier temps, nous examinons l'impact de la quantité de données de pré-entraînement ainsi que du nombre de classes associées. Lors du transfert, nous étudions également la méthode de transfert employée ainsi que le nombre d'exemples par classe dans la tâche cible. Nos résultats montrent que, durant le pré-entraînement, les performances saturent après une certaine quantité de données, et qu'une fois ce seuil atteint, le nombre de classes a peu d'influence, même s'il est élevé et que la précision du modèle sur la tâche de pré-entraînement diminue. Nous avons également observé que, bien que l'apprentissage d'une couche de classification linéaire soit généralement moins performant qu'un fine-tuning complet, il peut être plus efficace lorsque la quantité de données est faible et que la tâche cible est similaire à la tâche source.

Nous étudions ensuite l'impact du temps sur les extracteurs de caractéristiques basés sur des modèles neuronaux profonds. Pour cela, nous introduisons un nouveau jeu de données qui met en évidence le manque de robustesse des modèles pré-entraînés face aux décalages temporels. Nous évaluons la capacité de plusieurs stratégies de pré-entraînement et méthodes d'adaptation à mitiger ce problème. Nos résultats soulignent l'importance de mettre régulièrement à jour les modèles pour s'adapter aux changements dans le temps des distributions des classes visuelles, même lorsque le modèle est fortement pré-entraîné. De plus, la vitesse de ce changement varie selon les classes.

Enfin, nous proposons une nouvelle méthode d'apprentissage par transfert optimisée en termes de paramètres ajustables. Cette méthode se base sur notre découverte d'un biais dans l'approximation de la métrique de sensibilité permettant de déterminer les paramètres importants à ajuster. Ce biais, favorisant les couches ayant une grande variance dans les valeurs des poids, a pu être éliminé grâce à une nouvelle approximation de la sensibilité. La méthode offre également une alternative au fine-tuning non structuré grâce à des adaptations dont le rang est déterminé avant le transfert, en fonction de la sensibilité des paramètres du réseau. Les gains apportés par cette méthode ont été mis en évidence sur un ensemble de 19 datasets.

Ainsi, nos travaux contribuent à l'amélioration de l'apprentissage par transfert dans des environnements contraints en optimisant l'utilisation des données de pré-entraînement, en renforçant la robustesse des modèles face aux décalages temporels, et en proposant une nouvelle méthode de fine-tuning efficiente en termes de paramètres.

université PARIS-SACLAY

ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Title: Efficient Transfer Learning Towards Constrained Environments
Keywords: Machine Learning, Transfer Learning, Artificial Neural Networks

Abstract: This thesis explores efficient transfer learning techniques for constrained environments, where reducing the number of adjustable parameters becomes central.

Firstly, we examine the impact of pre-training data volume and the number of associated classes. During transfer, we also investigate the transfer method used and the number of examples per class in the target task. Our results show that, during pre-training, performance saturates after a certain data threshold. Beyond this point, the number of classes has little influence even when high and despite a decrease in model accuracy on the pre-training task. We also observe that, while freezing the feature extractor generally performs worse than full fine-tuning, it can be more effective when the data are scarce and the target and source tasks are similar.

Next, we study the impact of temporal shifts on feature extractors based on deep neural network models. To this end, we introduce a new dataset highlighting the lack of robustness of pre-trained models against temporal shifts. We evaluate the effectiveness of several pre-training strategies and adaptation methods in mitigating this issue. Our findings underscore the importance of regularly updating models to adapt to temporal changes in the distribution of visual classes, even when the model is heavily pre-trained. Moreover, the rate of this change varies across classes.

Finally, we propose a new transfer learning method optimized to minimize the number of adjustable parameters. This method is based on our discovery of a bias in sensitivity metric approximation currently used to identify the important parameters for adjustment. This bias, favoring layers with high variance in weight values, was eliminated through a new sensitivity formulation. The method also offers an alternative to unstructured fine-tuning through adaptations, with a ranks determined before transfer and based on the sensitivity of network parameters. The benefits of this method were established on a set of 19 datasets.

Our work contributes to advancing transfer learning in constrained environments by optimizing the use of pre-training data, enhancing model robustness against temporal shifts, and proposing an new parameter-efficient fine-tuning method.

Remerciements

Dans un premier temps, je tiens à exprimer ma gratitude envers Inna Kucher, Adrian Popescu ainsi que Bertrand Delezoide pour m'avoir encadré avec bienveillance tout en m'accordant une grande liberté. Les conseils avisés qu'ils ont pu me prodiguer tout au long de ma thèse ont été d'une aide précieuse. Je suis particulièrement reconnaissant envers Inna pour avoir repris l'encadrement de ma thèse ainsi que pour son optimisme constant, qui a été une véritable source d'encouragement.

Je souhaite aussi remercier chaleureusement les rapporteurs et membres du jury qui ont su donner de leur temps pour examiner mes travaux et me prodiguer des retours.

Je remercie également mes collègues et amis du LIAE avec qui j'ai eu la chance de partager de nombreux moments de convivialité, tant à l'intérieur qu'à l'extérieur du laboratoire. L'amitié que nous avons tissée au cours de ces trois années a une grande importance pour moi. Les souvenirs de nos voyages, discussions et réflexions resteront gravés dans ma mémoire.

Enfin, merci à ma famille et à Ludivine pour leur soutien inconditionnel et la confiance qu'ils ont eue en moi. Vous avez toujours cru en moi, et pour cela, je vous en suis infiniment reconnaissant.

Table of Contents

Chapter 1

Introduction

In recent years, deep learning has revolutionized the field of computer vision, with applications spanning from satellite imagery analysis and facial recognition to healthcare diagnostics. The growing adoption of this technique has been driven primarily by the development of very large models, the collection of large datasets, as well as the deployment of powerful computing infrastructure. The features provided by these deep neural networks have demonstrated transferability to specific tasks, contributing to their wide adoption, even in cases where data is scarce. However, while the utility of transfer learning in such contexts is well-recognized, many challenges remain, especially regarding its application in embedded environments, where hardware accelerators and neural architectures create new constraints, such as limited memory, processing power, and energy efficiency.

This thesis contributes to the growing field of research on transfer learning, specifically focusing on the capabilities of transfer learning when the models are constrained in terms of trainable parameters. In this introductory section, we outline the broader context of this thesis, emphasizing the key factors that motivated the research. We explore the challenges and research questions that define the objectives of this work. Finally, we will present an overview of the contributions made in this thesis and the structure of the document.

1.1 Motivation

The size and complexity of machine learning models, particularly neural networks, have grown exponentially. In terms of computations, models have become more expensive to train and operate. Traditional general-purpose processors, such as CPUs, struggle to manage the demands of this increasingly large workload, particularly regarding operational phases like training or inference in real-time. Therefore Graphics Processing Units (GPUs) are used to accelerate these processes, offering significant improvements in parallel processing capabilities. Building on this, more specialized AI accelerators, such as Google's Tensor Processing Unit, announced in 2016, have been developed, enabling much higher energy efficiency and inference speed compared to what is achieved with traditional CPUs or even GPUs.

Owing to their effectiveness, the range of applications of deep models has expanded. This includes embedded environments where memory and energy constraints are more stringent, as well as applications where latency is critical, such as autonomous driving and edge AI. In this context, even more specialized accelerators have emerged, allowing inference to be performed with almost no costly memory accesses [204, 79]. However, this has come at a cost. The supported neural network is unique, both in terms of architecture and parameters, which are fixed and cannot be updated once the circuit is manufactured.

The viability of such a circuit mainly depends on its ability to adapt to changes or variations in tasks. The adaptation of a feature extractor to a new task is called transfer learning, and it has already been widely studied and proven effective. It is, therefore, important to understand how and to what extent the network can be adapted. However, a problem arises: current studies on transfer learning [29, 32, 49, 90, 224] primarily focus on very large models and mostly recommend adapting the network's weights. Neither of these is feasible here.

The most common alternative to fine-tuning network weights is training a new linear classifier to directly reuse the features produced by the pretrained neural network. This approach, known as linear probing, is particularly useful in embedded environments. Indeed, it can be used in hardware like the one mentiened earlier, but even more importantly, it is considerably less computationally expensive, as it requires tuning far fewer parameters. However they are not a

panacea, transfer performance are usually very limited compared to a full fine-tuned [91].

Parameter-efficient transfer learning has gained traction [57], and it focuses on modifying a small number of network parameters to adapt feature extractors to target tasks. However, they are not directly usable, particularly because the set of learned parameters varies or lacks structure. Similarly, numerous studies examine essential elements in the pre-training of feature extractors, such as dataset composition, but these studies focus on networks whose compactness is far from what is required.

The development of these accelerators is one of the many factors that make transfer learning an essential tool in deep learning and motivate research, such as this work, aimed at improving transfer learning methods and understanding their limitations. Beyond the transfer method and embedded constraints, more general challenges are also important for transfer learning. As is often the case in deep learning, data play a significant role, both during pre-training, where the choice of data can enhance the model's transfer capabilities but may also make pre-training costly if the data volume is large, and during transfer, where the low quantity of task-specific data and its divergence from the source data one can impact performance. Hence, the selection of upstream data and consideration of the downstream data represent multiple challenges that need to be addressed.

The next section will describe in more detail the challenges and questions explored in this thesis.

1.2 Challenges and Objectives

The challenges addressed in this thesis revolve around three very general questions, motivated in the previous section, and to which this thesis attempts to provide insights.

- How can we effectively pre-train a feature extractor for transfer learning?
- What are the limitations of a frozen network?
- How can we transfer a pre-trained network while minimizing the number of parameters to be fine-tuned?

These three general questions, too broad to be studied directly, can be subdivided into more specific research questions. We identified them by examining the extensive research corpus on these topics. We will, therefore, describe the research questions identified and linked to each of these three questions.

The first point is particularly complex due to the number of factors that impact a model's ability to generalize to other tasks. The literature shows, for instance, that the architecture, the data used during pre-training, and the training procedure are important elements to consider. The concept of transferability is also difficult to measure, as a model may generalize well to some tasks but not to others. The research corpus on this subject is quite extensive [91, 224, 113, 33] but several challenges remain open. The first concerns the size of the dataset to be used for pre-training, whose benefits appear to plateau according to certain findings. However, these studies focus on large models and overlook the potential benefit of adding more classes as scaling occurs. This is important because it is unclear whether the plateau is due to overly represented classes or model capacity. The research questions to which this thesis provides insights are: *Can the saturation of the gains from increasing the dataset size be pushed further by adding more classes, and thus more diversity, during training? And, what is the influence of downstream dataset size, number of classes, and transfer method on the transferability of compact neural networks?*

Transfering with a frozen feature extractor allows for efficient training due to the small number of parameters to tune and can be used in accelerators mentioned in the motivations [204, 79]. When considering this type of transfer, called linear probing, an important limitation to study is the robustness to temporal shifts in the input data. This limitation is crucial because hardware accelerators relying on a model that cannot be modified would see their viability greatly diminished if temporal shifts in the input data render the network obsolete. Moreover, while studies on certain domain shifts, such as representation style [65, 140], are quite comprehensive, temporal shifts have been very minimally explored. In this thesis, we will therefore address the following research questions: *Are pre-trained models robust to temporal shifts? If not, how can the issue be mitigated? Finally, in the context of object recognition, are all types of objects equally affected?*

Finally, to answer the third general question, it seems natural to turn to Parameter-Efficient

Fine-Tuning (PEFT) methods, whose aim is to balance transfer learning effectiveness and efficiency by only modifying a small set of relevant network parameters(see Section 2.4). These methods have recently sought to estimate the importance of each weight in order to improve efficiency. Following a review of existing methods, which we will describe in the next chapter, the research questions addressed are as follows: *Can we improve the parameter's importance estimation in order to better allocate the budget of trainable parameters?* And, *can unstructured fine-tuning be replaced by an alternative to avoid the widely distributed fixed weights, which are not leveraged in the hardware architecture mentioned in the motivation section[204, 78]?*

1.3 Contributions overview

After reviewing related work in Chapter 2 to ground the research questions and justify the approaches chosen to answer them, we present the three main contributions in the following chapters:

- Chapter 3 investigates how scaling pretraining datasets impacts the performance of deep learning models, particularly compact ones. Multiple factors are considered, such as the transfer strategy, the amount of data, the number of classes in the pretraining dataset, and the number of samples per class in the target datasets. The findings indicate that the saturation in the benefits provided by scaling up the pretraining dataset cannot be pushed further by adding more classes at the same time. Even more interestingly, the number of classes has almost no influence on downstream accuracy once saturation is reached, even when it causes the network's accuracy to drop significantly on the upstream dataset. We also show that linear probing can outperform full fine-tuning in few-shot scenarios when the final and source tasks are similar. The work presented in this chapter has resulted in the following publication: Pégeot, T., Kucher, I., Popescu, A., & Delezoide, B. (2023). A comprehensive study of transfer learning under constraints. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) workshops* (pp. 1148-1157).

- Chapter 4 introduces a new dataset, VCT-107, designed to explore the effect of temporal shifts on visual classification models. We investigate how these changes affect model performance and show that different categories of objects are impacted to varying extents. We also examine various methods to mitigate these effects in pre-trained models and analyze factors impacting robustness, such as data augmentation and pretraining strategies. Our results highlight the lack of robustness of pre-trained models and, therefore, the need to adapt them regularly. They also demonstrates that it can be done without updating the feature extractor using linear probing with stored data or continual learning algorithm depending on the memory constraints. The work, results, and dataset presented in this chapter have been accepted for the WACV 2025 conference.

- Chapter 5 analyses current approximations made to estimate the importance of fine-tuning specific weights. We show that these approximations introduce a bias towards layers whose weights have a high standard deviation. This causes significant inconsistency in the layers considered important, which can become even more critical when scaling factors, such as those found in quantization, are used. Therefore, we propose a new formulation for the importance estimation to mitigate this bias. While current methods leverage this importance metric for both structured and unstructured PEFT, our analyses reveal that both the existing and new approximations are too noisy to effectively guide unstructured fine-tuning. Consequently, we propose an alternative by adapting the size of LoRA modules based on sensitivity, which is also easier to leverage in the context of specific hardware design. Using all these elements, we propose a new PEFT method. We were able to measure the gains provided by our method and propose a new adversarial attack to highlight the additional benefits in the case of scaling factors, such as those found in quantization methods. The two innovations of this new method (the unbiased importance metric and the alternative to unstructured fine-tuning) were also independently tested on a wide range of 19 datasets to provide an ablation study of their individual contributions. The work presented in this section, along with the new PEFT method, has been accepted at the *International Conference on Neural Information Processing 2024* (ICONIP), which will take place in

December.

Finally, in the last chapter, we put the results into perspective and discuss their impact in the context of fixed networks. In particular, we will discuss the key elements to consider, such as the pre-training dataset, the limitations of the transferability of fixed neural networks, and the necessity of regularly adapting the network once in production. We will also discuss methods for transfer and adaptation. We will present the limitations of this work and explore future research directions that could improve the performance of predominantly fixed networks, as well as their utility in the context of specific hardware accelerators.

Chapter 2

Background and Related Works

2.1 Image Classification via Supervised Learning

The classification tasks regroup all tasks in which we predict the category a sample belongs to. In the case of computer vision, the samples are often images, and the categories represent the type of object represented in the image. While this common case of classification, known as object recognition, is very common, other classification tasks exist. For instance, counting the number of elements in the CLEVER dataset [84] or predicting the azimuth of the object represented in the small NORB dataset [101]. These classification tasks can thus be found in fields as diverse as the medical field, with the classification of retinopathy from fundus images [34] or the or the classification of trafics sign for autonomous vehicles [167, 168]. In this document, we will focus on cases where a sample belongs to only one category.

Formally, given a dataset $\mathcal{D} = (x_i, y_i)_{i=1}^{N}$, where each x_i represents a sample from the input space $\mathcal{X}$ and each y_i is a label from the set of possible classes $\mathcal{Y} = \{1, 2, \ldots, K\}$, the classification tasks consist in learning a function $f : \mathcal{X} \to \mathcal{Y}$ predicting the class label y_i associated with x_i. In practice the output of f is a vector of size K. Each position in this vector correspond to one element of $\mathcal{Y}$. A softmax function can be applied to the vector to normalize it into a probability distribution, and the predicted class is the index of the maximum element in the vector. However, the probability after a softmax aren't easily usable as the neural network tend to be overconfident

as pointed out by [176].

Supervised learning is a method of machine learning, where the parameters θ of the model f are optimized using a set of labeled examples. These labels represent the ground truth for the model's predictions. The set of examples used for training the model is called the training set. The goal is to generalize the model's performance to unseen data, allowing it to make predictions on new samples.

To learn the parameter θ, the difference between the current prediction of f on the training set and the ground truth is measured. This measurement, called the loss, is often expressed using the cross-entropy loss. This loss takes as input the prediction $\hat{y}$ of f as a vector as discussed previously. This prediction is compared to the one-hot vector representing the ground truth label, that is the vector y with a 1 only on the index corresponding to the classes associated to the input. The following formulation already includes the softmax transformation of $\hat{y}$.

$$\mathcal{L}_{CE}(\hat{y}, y) = -\sum_{i=1}^{K} y_i \cdot \log\left(\frac{\exp(\hat{y}_i)}{\sum_{j=1}^{K}\exp(\hat{y}_j)}\right) \tag{2.1}$$

The model parameters can be optimized by minimizing this loss computed on the training dataset $\mathcal{D}_t$. In this work the gradient descent will be performed using Stochastic Gradient Descent. The stochastic gradient descent (SGD) consists of iteratively sampling a small batch of samples $\mathcal{B} \subset \mathcal{D}_t$ on which the gradient is computed. The weights are then updated using this gradient and a factor called learning rate, which usually decreases during the training.

The loss can be regularized using, for exemple, weight decay [94]. The other most common method to perform gradient descent is Adam [88]. Both have a variant in which the weight decay regularization is applied decoupled from the gradient-based update, allowing for better control over the regularization during optimization. those are called AdamW and SGDW [110]. This supervised learning method is opposed to the self-supervised methods that can train a part or the totality of the model f without requiring labeled data [15, 80]. These methods can be particularly useful when training on very large datasets.

The model structure can be divided into two parts. The first, called feature extractor or backbone, takes the image and outputs a vector called embedding. This embedding is a

numerical representation of the input data, usually in a lower-dimensional space, that is easier to analyze and process for tasks such as classification. In this work, those embeddings will also be used for clustering and similarity comparison in Section 4. The second part of the network call classification head performs the actual classification based on this embedding. It typically consists of one or more fully connected (dense) layers, which are simple matrix multiplication converting the embedding into a vector containing scores for each class. In practice, both parts can be learned simultaneously, but some research has shown that embeddings produced by the feature extractor can be reused through different tasks [161]. This transferability of the representation is the foundation of the transfer-learning methods we discuss in the next sections.

2.2 Transfer learning

2.2.1 Definition

Network-based Transfer learning is a machine learning technique where we reuse the optimized model weights from one task to solve a different, but related, task. These weights are often interpreted as containing useful information or patterns learned from the source task. This approach removes the need to train a model entirely from scratch for the new task. It also allows the model to benefit from the larger dataset used in the source task, which can help compensate for limited data in the target task.

In 2018, the survey by Tan et al. [179] categorizes methods that leverage an auxiliary data source to improve performance on a target task. This broader definition of transfer learning include the following types of approaches: (1) Instance-based transfer refers to selecting samples from the source dataset and adding them to the dataset used to train the target task. (2) Mapping-based transfer consists of mapping both source and target data into a space where they share the same domain, making it possible to use the union of both data sources. (3) Adversarial-based transfer learning consists of constructing a feature space that is suitable for both source and target data. Unlike mapping-based transfer, the common representation and the task are learned simultaneously. Ensuring that the produced representation is indistinguishable between

the two datasets becomes an additional objective when optimizing the weights. (4) Network-based transfer learning is the method we described earlier, based on reusing part or all of the weights and features produced. This type of transfer is very common, as pre-training on the upstream dataset can be reused multiple times, eliminating the need to share the pre-training data and reducing the cost of training on the target task. In an older survey, this type of transfer is also called feature transfer [134], although this term overlooks the possibility of fine-tuning the pre-trained weights.

In this document, "Transfer Learning" will refer to Network-based transfer learning. The terms "upstream dataset" and "downstream dataset" will refer to the source and target datasets, respectively, reflecting their sequential roles in the transfer learning process.

2.2.2 Motivation

Transfer learning is motivated by its ability to solve multiple challenges of deep learning. First, transfer learning helps to improve the accuracy and reliability of deep neural networks. As an example, in 2014, Sharif et al reused the features produced by a model trained to solve the ILSVRC 2013 challenge [155] to surpass the state-of-the-art methods in various tasks such as object recognition and image retrieval. In the same year, two other works demonstrated empirically that using a pre-trained model can improve the prediction quality of neural network [219, 28]. It can also make the model more robust [66].

Another motivation for Transfer learning is the important cost of training. Deep learning models require long optimization processes, which are computationally and financially costly. Even when transfer learning does not improve performance, it helps shorten the training procedure [63]. This is a critical challenge in a field where training costs keep growing [185].

In addition to improving accuracy and reducing costs, transfer learning allows the application of deep learning methods to tasks with limited available data. Deep learning models typically require large datasets for optimal performance, and a lack of data directly impacts model quality [68]. However, transfer learning is particularly effective in such scenarios, as it leverages pre-trained models from larger datasets [16, 230]. Medical imaging is a good example,

as collecting large amounts of data for tasks such as diagnosing rare diseases is often difficult. A recent research paper benchmarked pre-trained models and acknowledged their effectiveness in few-shot learning tasks, such as diagnosing diabetic retinopathy and classifying breast cancer from mammograms [205]. The paper also highlights the need for more research and datasets to improve the pre-training of foundation models. Statistical analysis of transferring the representation for few-shots has also been realized in 2021 [31]. We refer to the process of learning from downstream tasks with very few data as 'few-shot' learning. In this document, we will use 'low-shot' to describe cases where the number of samples is limited but not small enough to qualify as 'few-shot.' Typically, 'few-shot' refers to scenarios with 1 to 5 samples per class, while 'low-shot' applies when there are between 5 and 100 samples per class.

Finally, transfer learning also allows for efficient hardware design. Indeed, memory accesses are costly. Several research works have developed hardware architectures that do not require such access, resulting in low power consumption and latency [204, 78]. However, the drawback is that the feature extractor need to be fixed. Therefore, the applicability of these results relies on the transferability of the representations produced by the pre-trained network. In other words, transfer learning enables these hardware implementations to adapt, at least partially, to domain changes or new tasks.

Motivated by this challenge, this document will explore the transfer performance of fixed feature extractors and the methods that allow the largest possible portion of the network to remain static while maintaining strong performance in diverse tasks.

2.2.3 Challenges and Current methods

As explained previously, transfer learning is a valuable method in various ways. However, transfer learning also have drawbacks and can be challenging. Here, we will present open problems and current works aiming at solving each one of them.

Adapting or Freezing the feature extractor

A challenge when selecting a transfer learning method is finding the right tradeoff between adapting the representation or directly reusing it. In that sense, the two most common transfer techniques are also the more extreme. In Linear Probing (LP), the representation is used directly as produced after the pretraining. Only the final classification layer is learned on the downstream dataset. This method is sometimes called representation learning. The other one, called Full Fine-Tuning, also updates the weights of the feature extractor during the gradient descent. Those two methods have been compared multiple times, and Full Fine-Tuning has shown superior performances [91, 224]. Those results align with the previous research pointing out that reusing features can lead to weak co-adaptation and a drop in performance due to representation specificity [219]. However, Full Fine-Tuning also has drawbacks. Indeed, during the fine-tuning process, the features produced by the pretrained models can be distorted and underperform out of distribution (OOD) [96]. A metric for robustness to out-of-distribution data has been proposed in [182], by comparing the accuracy in and out of distribution. This metric was used to measure the evolution of the robustness during the fine-tuning, confirming that the fine-tuning process affects OOD performances.

This issue is also crucial in incremental learning, as it involves transferring multiple times while preserving the performances on the upstream domains. The distortion of features can, therefore, lead to a performance loss in the upstream domains called catastrophic forgetting [39].

Le choix est également essential dans le cadre du hardware design car les backbones figer peuvent être leveraged to reduce the number of memory read as we will see in Section 2.3.

Recent research has shown that the limited accuracy provided by Linear Probing can be partially mitigated by using intermediate representations in the features extractor. Indeed, depending on the downstream tasks, using the representation produced by the last layer of the feature extractor can be sub-optimal. The Head2toe [35] transfer methods use the Group Lasso [220] to select intermediate representations and aggregate them to perform the classification on an augmented representation. This leads to better accuracy without modifying the weights of the feature extractor. However, this is still limited as the choice of features is specific

to each downstream dataset and requires access to the intermediate representation. Similarly, for transformers, VQT [193] leverages the intermediate representations by introducing a new token.

More nuanced methods try to find a tradeoff between fine-tuning and linear probing. For instance, BitFit [222] only updates the bias terms in each layer, which account for about 0.1% of the model weights. Such methods that change only a small part of the model's parameters are called Parameter Efficient Fine Tuning (PEFT). We will describe in more detail the State of the art of those methods in Subsection 2.4.

Bias in pretrained models

Another challenge that needs to be mitigated is the presence of bias in the pretrained models. Bias are inherent to pre-trained models and often originate from the upstream datasets, which are known to be biased [190]. Therefore this is partially a dataset-related issue. We will explore in delve deeper into dataset-bias and solutions involving the construction of a dataset in the Subsection 2.5.

Despite significant efforts to improve the diversity of datasets, most available pre-trained models are developed using widely used datasets. This often introduces inherent biases. In the domain of computer vision, a particularly common dataset is the ILSVRC [27], on which many pre-trained models are based. However, these models exhibit a bias towards texture recognition, often failing to sufficiently account for shapes and other critical features within the image [43]. Similarly, pre-trained language models often suffer from stereotypical bias, and datasets as been produced to measure it [126]. Several advanced transfer learning techniques have been proposed and developed to mitigate this issue. In the paper [43], the proposed solution relies on constructing an ImageNet dataset variant that is added at the pretraining stage. And is not applicable after pretraining, especially if the upstream dataset is not provided with the model. But in [200], the researchers analysed bias as spurious correlations and underrepresentation. They offered a solution to mitigate them at the fine-tuning stage by artificially modifying the downstream dataset. However, that conclusion was made only for spurious correlations and underrepresentation and, more importantly, did not consider that the pre-trained model can be frozen. Finally, recent transfer methods assume that pretrained models are inherently biased

and shift away from removing the bias. Rather than focusing on eliminating this bias, they select different representations produced by models and aggregate them to solve the downstream task [47, 203]. The more the pretraining datasets and strategies are diverse, the more the ensemble will be performant on downstream tasks [47]. However, these approaches are limited by the multiple inferences required and call for the creation of more datasets and pre-training methods.

Pretraining dataset selection

As we just saw, using an ensemble of feature extractors pre-trained on diverse upstream datasets can improve downstream performance. But when only one feature extractor is considered, the choice of the models and it upstream dataset becomes essential. Indeed, the upstream dataset has a strong influence on downstream performance [219]. For instance, research in the field of medical imaging points out the benefits of using an upstream dataset composed of medical images when transferring to medical tasks [205]. This aligns with early observation, suggesting that the specificity of an extractor of a feature in a domain different from the one used downstream leads to the worst performance due to negative co-adaptation [219].

To avoid the issue of being specialized in tasks related to the datasets, the ensemble approach [203] can be reused. Some approaches have been found to fuse the ensemble of models into a single model [45]. Unlike the entire ensemble, this single model can be exported to embedded targets like the one described in Subsection 2.3. However, this still requires the costly development of multiple pre-trained models, and merging those can be considered part of the pretraining. When considering only a single pre-trained model, a solution is to use intermediate datasets to train different modules on top of the unique pre-trained model [38]. Similarly to the ensemble, it makes it possible to transfer to diverse tasks further from the source while removing the necessity of fully training multiple foundation models. However, it still requires the creation of multiple datasets.

Security concern with pretrained models

Finally, other challenges arise when the pretraining and the final user on the downstream task are different actors.

The actor who realizes the pretraining may refuse to share the pretraining data. Sharing the pre-trained model for transfer might also be considered a leak of this private property. Indeed, once shared, pre-trained data might be extracted and the model could be reused for undesired purposes. Therefore some methods have been developed to remove the pretraining knowledge after transfer, making it impossible to share the transferred model without risk [89, 11]. However, this supposes that the actor, realizing the pretraining, can realise the transfer himself and that the final user can share data from the downstream tasks with the upstream user.

Another issue when using a pretrained model from another actor is the lack of confidence the user can have in the model. Indeed, the model can contain a backdoor that will lead to malicious behavior [104, 206]. This backdoor can be activated by some trigger placed in the input sample. The issue is that some backdoor techniques aim to keep their capabilities after transfer learning even if the extractor of feature is updated [103, 217, 202]. Hopefully, some transfer learning techniques have been invented to allow prevent this [173, 3]. But they might not be enough as some triggers aren't activated at transfer time but even after the moment of the quantization [111].

Those security issues may limit the range of pre-train models available for transfer.

Our usage.

In this document, we primarily focus on two of these challenges. The first challenge is bias in pretrained models, specifically time bias. Indeed, in Chapter 4, we propose a new dataset that allows for the evaluation of transfer capabilities to images from different time periods. Analysing, in the same time, the robustness of models to time shift. The Chapter 5, on the other hand, is closely related to the challenge of finding a trade-off between linear probing and fine-tuning as we propose new PEFT method which represent a new alternative.

2.2.4 Evaluate Transferability

A thorough evaluation of a pre-trained model's transferability or the transfer learning technique's effectiveness can be challenging. Indeed, as [1] pointed out, evaluation requires multiple and diversified downstream datasets. As explained previously, bias and similarity between upstream and downstream datasets can have a significant impact on downstream performance. This justifies the multiplicity and diversity of downstream datasets used for evaluation.

Results also vary depending on the transfer method (e.g., linear probing vs. full fine-tuning). Different techniques may highlight different aspects of transferability performance. Typically, the common linear probing method is tested alongside the opposite fine-tuning approach [91]. We will follow these methods in Chapter 3. Some key settings, such as the number of samples in the downstream tasks or the model architecuture, are also important and can influence the perceived performance of various pretraining or transfer methods. The variation in the number of samples per class often comes with the different datasets selected for evaluation.

This high number of settings, such as number of classes, number of samples or type of tasks, makes the evaluation challenging. But many downstream vision datasets can be used. Some datasets have been created to provide representations of the same classes, but a domain shift like a different styles, [65, 140]. It allows testing transfer to the same task but in a different domain. Multiple other datasets are commonly used to test for transfer. Some contain natural images of diverse objects like CIFAR-100 [93], others more specialized like GTSRB with the traffic signs [168] or SVHN with street number [127]. Other have specificly structured images like texture [21] or generated images [101, 84]. Some are contain diverse. or have been concived to be run in a few shot settings like Flower.

All this diversity of downstream dataset is necessary to evaluate transfer learning performance but have a major drawback. They make the results from different research harder to compare, as the set of chosen datasets is not always the same. Therefore, some standard groups of datasets has been proposed and can serve as a benchmark. For instance, the Meta-Dataset [192], Visual Decathlon [147], both of which regroups 10 datasets, or the common benchmark is the Visual Task Adaptation Benchmark [224].

Some advantages of VTAB over visual decathlon and Meta-Dataset are: (1) It contains multiple types of classification tasks, such as orientation detection with SmallNorb [101], depth prediction and counting [42, 84] whereas visual decathlon only contains object recognition. (2) Contain more tasks with 19 tasks for VTAB and 10 for Visual Adaptation Benchmark (3) Contain a variant with only 1000 training samples, called VTAB-1K, which is harder to solve and can help evaluate low-shot transfer capabilities [224]. (4) regroup different kinds of images: Natural images with datasets like CIFAR [93] or Oxford-III pets [138], specialized images like retina scan [24] and generated images with datasets like dSprite [117], SmallNorb [101] or CLEVR [84].

This benchmark is not always adopted when tackling questions like dataset scaling but is commonly used for testing parameter-efficient fine-tuning in computer vision [82, 61].

Usage. Based on those works, we evaluate the transfer performance of the models on diverse downstream datasets in Chapter 3 and 5. In Chapter 3, we also follow the recommendations of [33] when fine-tuning on downscaled versions of the datasets. Additionally, in Chapter 4, we also propose a new dataset specifically designed to evaluate the temporal robustness of the pre-trained models, which had not been given much attention until now.

2.3 Frozen Backbone for efficient hardware design

2.3.1 Motivation

Deep neural networks (DNNs) continue to scale in complexity and size [211], necessitating breakthroughs in computational efficiency, energy consumption, and cost. This challenge has driven the development of domain-specific hardware like Google's TPU, which accelerates the execution of neural networks by specializing in tensor operations [85]. Specialized hardware is not only about performance but also focuses on optimizing energy and resource efficiency, crucial for real-time applications and power-constrained environments, such as Mobile devices [18].

A significant portion of energy and latency costs in DNN execution arises from memory access, especially when relying on external memory, such as DRAM [175]. Memory operations contribute more to power consumption than the actual computations [71], making the design of

memory-efficient architectures a key priority for both large-scale cloud systems and low-power edge devices.

Emerging hardware architectures, such as the Eyeriss accelerator, aim to tackle these challenges by reducing memory access and optimizing dataflow, leading to more energy-efficient implementations [19].

2.3.2 Related Works

To reduce memory access and, consequently, improve performance, some hardware architectures have been designed with the neural network weights hard-coded into the circuitry itself. One of the earliest examples of this is FixyNN [204]. In this work, the researchers proposed a hardware architecture designed to run inference with a MobileNetV1 model. In this architecture, the first part of the model is frozen, with the weights hard-coded into the circuit. This allows the first part of the inference to run without relying on DRAM. The results presented in the paper show that as the number of fixed layers increased, the die size also increased, but the number of operations per second and energy efficiency improved significantly. By freezing 11 out of the 13 layers of the MobileNetV1, FixyNN managed to stay below the 4 mm^2 and provide an increase of approximately five times in the operations per watt compared to the same model on Nvidia's NVDLA with the same area.

Some improvements have been made to enhance this approach in SemiFreddoNet [78]. In this architecture, the weights are not simply frozen in the first part, but all layers are partially frozen. The results showed that having some trainable weights helps improve performance for both classification tasks tested: facial recognition and object recognition.

A circuit of the same type as FixyNN has been fabricated and tested in real-world conditions. The results, which will be presented in November 2024 at the Asia Pacific Conference on Circuits and Systems, confirm the findings [79]. In this implementation, the process node used is larger than the one used in FixyNN simulations. Furthermore, the entire MobileNet feature extractor [74] is frozen, but it does not contain residual connections. These elements, combined with the fact that the results presented in the FixyNN paper were obtained solely through simulation,

make this work an important new contribution to the field.

2.3.3 Limitations

While promising, this specific hardware suffers from some limitations. First, these architectures were designed for CNNs, but limited studies have been conducted on the applicability of this approach to more recent architectures, such as Transformers. Secondly, as discussed in subsection 2.2.3, the frozen feature extractor offers limited generalization capabilities.

In this document, we will present results on the generalization capabilities of small networks within the context of frozen feature extractors (Linear Probing) 3. More importantly, in Chapter 4, we will examine the temporal robustness of feature extractors, specifically the degradation in performance of pre-trained models over time. This aspect is crucial for such hardware, as temporal shifts could cause this frozen architecture to become obsolete. Finally, in Chapter 5, we analyze sensitivity approximation and propose a modification to a state-of-the-art PEFT method to improve transferability while maintaining a very low number of trainable parameters. Although this method may not be directly applicable to fixed hardware like FixyNN [204], it is a step toward maximizing the ratio of fixed parameters and, consequently, enabling better hardware performance.

2.4 Parameter-Efficient Fine-Tuning

2.4.1 Definition and Motivation

As we discussed in the subsection 2.2.3, fine-tuning usually leads to better performance but can harm the generalization capabilities [96] or negative transfer [219]. On the other hand, using the feature produced by the pre-trained model (Linear Probing) while keeping the extractor of features constant is a very efficient solution as the number of trainable parameters is low and can lead to better hardware design [204] but usually underperform compared to a regular fine-tuning [91]. Parameter-Efficient Fine-Tuning (PEFT) methods aim to maximize transfer performance while training only a small subset of the model parameters. These methods, therefore, represent a

tradeoff between these two extremes.

Those methods are increasingly popular, partially because of the very important cost of training a full network, but also because it allows fine-tuning on consumer-level hardware. It also represents a great solution for few-shot learning [107]. And can even be used on very limited computational resources [222].

2.4.2 Categories of methods

There are numerous methods to fine-tune a network by only modifying a small set of parameters. We will therefore categorize these methods into several groups and present the most common methods within each one. Three categories presented are based on the 2024 surveys [57] and [210].

The first category, called Additive PEFT in [57] and Addition-based Tuning in [210]. The common point between these methods is that they add additional parameters or modules to the architecture. The Visual Prompt Tuning (VPT) method [82] involves adding a small number of new tokens (additional parameters) to the model, which, consequently, slightly increases the computational load. Therefore, it is an additive PEFT method. Similary, Side Tuning [225] consist in training a second (smaller) network and aggregate the results of extractor with a sum. The most common additive PEFT are the Adapters [72]. Adapters were initially proposed for natural language processing (NLP) models and consisted of adding two small modules in the transformers blocks. Each of them is placed right before the addition and normalization of the residual connection that surrounds the Multi-Head Attention and Feed-Forward layers. The modules contain a downward projection $W_1 \in \mathbb{R}^{r \times d}$, $r \ll d$ and an upward projection $W_2 \in \mathbb{R}^{d \times r}$ separated by activation function (non-linear). After a bias is added to each projection. The module is also surrounded by a simple residual connection. In total the adapter module adds $2 \cdot d \cdot r + d + m$ trainable parameters which is far less than training a linear layer from the MLP. Several variants of Adapters have also been proposed, for example, Adapter-Fusion [143], or more simply, Adapters placed in parallel [62].

The second category of PEFT is "Reparametrization PEFT [57] which is included in Partial-

Based Tuning in the second survey [210]. In this method the fine-tuning includes new weights during the training but they can be merged for inference. Therefore this method can be used to make inference with a fine-tuned model without any overhead, or if not merged, the number of extra parameters will be the same as in the Additive PEFT. This second category looks more promising for fixed-weights hardware design such as [78, 204]. Most of reparametrization PEFT are rely on a Low-Rank modification of the weights called LoRA [76]. LoRA modules are placed in parallel to fully connected layer (matrix $W \in \mathbb{R}^{d_{out} \times d_{in}}$) and consist in a downward projection $W_1 \in \mathbb{R}^{r \times d_{in}}$, $r \ll \min(d_{in}, d_{out})$, an upward projection $W_2 \in \mathbb{R}^{d_{out} \times r}$, and a scaling by a factor $\frac{\alpha}{r} \in \mathbb{R}$. The output of the LoRA module is then added to the one of the fully connected layer that is modified. Unlike the Adapter module, LoRA does not contain any non-linearity, which allows it to be merged into the weights of the adapted layer. To do this, the matrix $\Delta W = \frac{a}{r} \cdot W_1 \cdot W_2$ is added to the pre-trained weights matrix. Several variants of this methods have been proposed, for example, adapted learning rates [58] or a Bayesian approach [212].

Finaly the last category of PEFT methods is called Selective PEFT [57] is also included in Partial-Based Tuning [210]. Those methods directly update a small set of weights inside the network. This set can be the lasts layers, target specific types of layers like attention layers [44] or bias [222]. The set of modified weights can also be unstructured, that is select specific weights inside a tensor [105, 172]. However unstructured modification can make it hard to take advantage of the frozen weights, even if unlike LoRA, the number of impacted weights in a layer is smaller.

However all those methods have limits. Some, like VPT, are very specific to the transformer architecture. Adapters create a permanent overhead and have lower performance compared to LoRA. LoRA modules can count to trainable parameters. Finally unstructured methods can be hard to take advantage in specific hardware like the one described in Section 2.3. To improve performance, in terms of accuracy, recent works attempted to combine multiple categories of PEFT in a single method [229, 61], but have also developed methods to select more efficiently the weights or layer on which the PEFT methods are applied.

2.4.3 Layer selection in PEFT methods

To reduce the number of trainable parameters even futher while maintaining high transfer performance, researchers propose methods to select the layers on which to apply PEFT techniques. A basic selection was already proposed in the original Adapters paper. However, this selection of affected layers was empirical and limited to a single contiguous group of layers [72]. Finding the best set of parameters is also a concern in the design of frozen-network-based hardware, as it is important for their efficiency [78, 204]. However, SemiFreddoNet only considers a few freezing schemes, and FixyNN focuses solely on freezing the first layers. Therefore in the next subsection we will look at methods used to select the layers affected by the PEFT techniques, especially the promising use of the sensitivity metric.

Several approaches have been proposed to determine an efficient allocation of trainable parameters. Among these approaches are those based on Neural Architecture Search (NAS). For example, a variant of BitFit, which originally optimizes all the biases, has been optimized using a NAS approach. Similarly, the Neural prOmpt seArcH (NOAH) method [229] uses this approach to simultaneously place different PEFT modules previously described : LoRA, Adapters, and VPT. It was observed on this occasion, as well as in other analyses [51, 54], that the effectiveness of a method and the choice of adapted parameters depended heavily on the target task. This research is therefore often conducted for a given target dataset [229, 51, 54]. To effectively determine the importance of fine-tuning a weight, some methods have started using a metric called Sensitivity.

2.4.4 Sensitivity based PEFT

Sensitivity is a metric used to determine the importance of modifying a given parameter. Before being applied to PEFT, this metric was also used for pruning [123] and mixed quantization [207]. For a given parameter, it is defined as the improvement in the loss after modifying that weight. Formally, this notion of sensitivity is expressed as follows:

$$S_i = L(D_t, w) - L(D_t, w^*) \tag{2.2}$$

$$w_n^* = \begin{cases} n = i & w_n + \Delta w_n \\ n \neq i & w_n \end{cases} \tag{2.3}$$

Where L the loss function, w_i is the i^{th} parameter and Δ_{w_i} the variation of w_i after modification. That is, after fine-tuning or prunning depending on the application.

However, whether it is for pruning, mixed quantization, or selecting weights to fine-tune for PEFT, this notation is not directly usable in practice. The second term would need to be evaluated for each parameter in the network. Therefore, a Taylor expansion is performed.

Two PEFT methods based on sensitivity to determine the allocation of trainable parameters have been proposed. The first, called AdaLoRA [227], progressively reduces the size of LoRA using pruning based on sensitivity values. Since the pruned weights are discarded, the value of Δw_i is known: $\Delta w_i = -w_i$. This method primarily target the field of natural language processing but can also be used in vision tasks thanks to the ViT architecture. The main limitation is the reliance on LoRA pruning, which implies that the fine-tuning architecture is not fixed, cannot be determined before training, and does not satisfy the budget in terms of the number of trainable parameters from the start.

The second method, Sensitivity-Aware Visual Parameter Efficient Tuning (SPT) [61], on the other hand, directly evaluates the sensitivity of the network weights. This is highly advantageous because the fine-tuning architecture can be known before starting to update the weights, meaning the budget in terms of the number of parameters is set and respected from the beginning. However, since weight modification does not simply involve their removal, the value of Δw_n is no longer known. A new approximation is therefore performed in the form of a One-Step Gradient Descent. In this method, sensitivity values are used to place LoRA modules when sensitivity is high for many parameters but also to rely on unstructured PEFT when appropriate.

2.4.5 Limitations

As explained previously, AdaLoRA represents a great way of distributing the trainable parameter, but relies on pruning LoRA, which implies that the fine-tuning architecture is not fixed and does not satisfy the budget in terms of the number of trainable parameters from the start. The second sensitivity-based PEFT method, SPT, provides a great solution to those issues by evaluating the sensitivity directly on the pre-trained model weights. However, the unstructured aspect of the methods can limit the efficiency of the methods. Moreover the lack of analysis on the additional One-Step-Gradient-Descent approximation can also refrain from using it. Other approximations, such as using mini-batch, were on the other analysed [228]. Finally, the fact that these methods specifically adapt to a given target dataset can be limiting, particularly when we aim to use them for hardware design, as stated in [222]. It would therefore be interesting to measure sensitivity across a set of datasets in order to determine a less dataset-specific set of trainable parameters.

Our usage and contribution. This new use of sensitivity for PEFT without pruning is promising. To push back the limits associated with this approach, in Chapter 5, we provide a new analysis of the second approximation, which makes it possible to use sensitivity outside of the pruning context. We identify and address a bias it causes, which can be particularly important when quantization is involved. Finally, this also allows us to eliminate the limitation of relying on unstructured PEFT by using resized LoRA modules instead of fixed-size ones.

2.5 Pretraining Dataset

2.5.1 Importance of the pretraining dataset

Deep Learning models are known to be data hungry [115]. The use of increasingly large datasets allows for the training of ever more efficient and potentially larger models [68, 86]. This observation explains the creation of many large-scale datasets, such as YFCC100M [184] with 100 Million images or LAION-5B [159] with more that 5 billions text-images pairs. Fields specific pretraining dataset, similar to the target task, also allow for increased performances [25, 205]

increasing even more the necessity of pretraining dataset.

The importance of these datasets has pushed the community to study various aspects, such as biases or the possibility of reducing their size, thereby limiting the significant costs associated with data collection and training on such large datasets. In this section, we will explore several research directions, along with their limitations, that have influenced the work presented in this document.

2.5.2 Bias in pretraining dataset

Bias are inherent to pretraining datasets and can have a strong impact on trained model even after transfer [200]. In the field of computer vision, dataset biases have been grouped into three categories [36].

The first category, called Selection Bias, refers to the non-representativity of the data and the resulting incorrect correlations caused by the method used to collect the data. This bias can be concerning, as it can lead to undesired behavior in the final models [174] Some datasets have been collected in a way that removes one of the selection biases. For example, DollarStreet [151] was collected to eliminate geographical bias, which causes datasets to predominantly represent more developed, higher-income countries. Similarly, FairFace [87] was collected to ensure parity across ethnicities, ages, and genders. Those dataset also allow researsher to study the biais impact and mitigation methods

The second category, called framing bias, refers to the association caused by the way the element are represented in the images. For example, the ILSVRC dataset [27] only contains natural images of objects. This allows convolutional neural networks (CNNs) to solely focus on textures [13]. Including images that represent the same objects in different styles, such as drawings, allows the models to learn patterns that are more representative of these objects [65, 140].

The last category, called label bias, includes defects or errors in the labels assigned to dataset samples or in the way the labels are defined. An example of a dataset affected by this bias is ImageNet-21k [27]. This dataset consists of 21,841 classes and, thus, the same number of

possible labels for each sample. However, these labels are organized according to the WordNet dictionary [121], where some names are hyponyms or hypernyms. That is, one label semantically contains others. This is the case with *Animal* and *Dog*, for instance. An image labeled 'Dog' can fall under both categories. These biases are often mitigated during training. For example, [150] proposes a new loss function for pre-training on the ImageNet-21k dataset. Labeling errors have also led to the development of several techniques to reduce their impact [208].

Limits and usage. Research on biases in datasets and on the production of unbiased datasets still has certain limitations. Datasets specifically produced to avoid bias are generally too small to train large models, and are thus often merged with others [40]. Nevertheless, they remain useful for the analysis and consideration of new biases. Some biases are also underexplored in the literature, such as the temporal bias, which will be explore in Chapter 4. Since no dataset can be entirely free of bias, the datasets presented are not sufficient to fully replace the post-hoc mitigation methods described in Section 2.2.3.

2.5.3 Effect of dataset scaling

The size of the dataset is also a widely researched topic. Dataset size is important for two main reasons. First, there is a strong relationship between dataset size and the performance achieved on target tasks after transfer learning [170]. Second, datasets are expensive to collect, and pre-training is often computationally costly, so it may be beneficial to avoid using an unnecessarily large or complex pre-training dataset [33].

Past examination of pre-training tend to show that increasing the size of the upstream dataset has a positive effect on downstream accuracy [113, 170, 223]. These observations align with two related findings when considered together. Increasing the number of training samples in a dataset leads to improved performance, measured by test loss, on that dataset [68]. Meanwhile, as accuracy on the pre-training dataset increases, the accuracy on the target dataset, once the model has been transferred, also improves [91].

However, there appear to be limits to the gains that can be achieved by increasing the number of images in pre-training datasets. For example, [33] shows that, if the number of iterations in

gradient descent is kept constant, using 10% of ILSVRC can yield similar transfer performance to training on the full version of the dataset. This is consistent with the findings of [1], which show that the performance gains from improving accuracy on the source dataset tend to plateau toward an upper bound.

Limits and usage. While very interesting, these works present certain limitations. First, they do not consider compact deep learning architectures. Additionally, they do not always consider the case where the feature extractor is frozen, which is important when the chosen transfer learning method is linear probing. Finally, these studies do not account for a possible change in the number of classes when the pre-training dataset is expanded. This is important because it is unclear whether the saturation is due to over-representation of the classes or because the network's capacity to store information has been maximized. In Chapter 3, we will study, among other things, the influence of the pre-training dataset in order to reduce these limitations.

2.5.4 Alternative data collection

Collecting and annotating datasets can be relatively costly, even when considering the case of unsupervised pre-training, which does not require annotation. Consequently, several other data sources have been explored.

Diffusion models, such as Stable Diffusion [152], are capable of generating high-resolution images. The images generated by these neural networks can therefore serve as a data source for training. [158] used Stable Diffusion to create a synthetic version of ILSVRC, training on this alternative version resulted in a model with generalization capabilities similar to those trained on the original ILSVRC. Without going as far as creating a purely synthetic dataset, diffusion models represent an effective method of data augmentation[37].

Textual data is another modality that can often be collected alongside visual data, as is the case with Wikipedia images [166]. When paired with corresponding visual data, these additional textual datasets, combined with Contrastive pre-training, have allowed for the development of highly effective feature extractors [144]. However, textual data can also be used even if samples are not directly paired with visual data. For example, they allow training vision models on tasks

without the any domain-specific data [50].

Limits and usage. While synthetic datasets created using Stable Diffusion are more accessible, they require more data to achieve comparable results to the original versions [158]. The trade-off for the low production cost and easy access to these datasets is the increased computational cost due to the need for pre-training on larger datasets. Additionally, although multimodal models can benefit from textual data, they still require a certain amount of images to train the visual component. In this document, we will not train or fine-tune models on textual data. However, given the popularity and strong performance of multimodal models, we will also evaluate their capabilities when evaluating the temporal robustness of vision models in Chapter 4.

2.6 Neural Network Architectures

This last section of the chapter will give a short description of the architectures used in this document.

2.6.1 Convolutional Neural Networks (CNN)

Convolution operation. Convolutional Neural Network (CNN) architectures are types of neural networks that learn features, which are transformed through a succession of convolutions. One of the first convolutional neural networks is LeNet, developed by Yann LeCun [100].

The discrete convolution operation applied to a feature $I \in \mathbb{R}^{\text{width}\times\text{height}}$ and a kernel $K \in \mathbb{R}^{N_w \times N_h}, (N_w, N_h) \in \mathbb{N}^2$ can be written as:

$$(Input * K)_{w,h} = \sum_{i=1}^{N_w} \sum_{j=1}^{N_h} K_{i,j} \cdot I_{w+i,h+j}$$

In practice, the input feature map often contains multiple channels, typically corresponding to different color channels in an image. As the network deepens, the number of channels usually increases, a pattern commonly observed in many architectures such as LeNet and ResNet [100, 64]. This operation applied to multiple channels can be written as :

$$\forall c \in [1, C_{out}],\ (Input * K)_{c,w,h} = \sum_{l=1}^{C_{in}} \sum_{i=1}^{N_w} \sum_{j=1}^{N_h} K_{l,i,j} \cdot I_{l,w+i,h+j}$$

With C_{in} the number of channels in the input and C_{out} the number of channels in the outputs. I now have a new dimension for the channel and therefore belong to $\mathbb{R}^{C_{in} \times \text{height} \times \text{width}}$, meanwhile the kernel has a dimension for the input channel and a dimension for the output channel $K \in \mathbb{R}^{C_{out} \times C_{in} \times N_w \times N_h}$. Additionally, a bias $b \in \mathbb{R}^{C_{out}}$ can be added, and the k^{th} value of this vector is added to all output of the channel k.

Activation functions

In feed-forward neural networks, non-linearities are essential and allow the model to approximate complex functions [70]. Those non-linear functions are often placed between each layer of the network and are called the activation functions. Those functions are often applied element-wise. One of the most common activation function is ReLU, and is defined as:

$$\forall x \in \mathbb{R},\ \text{ReLU}(x) = \max(0, x)$$

This activation function is used in most CNNs discussed in this document. However, in the ViT architecture, which will be described later in the document, the ReLU is usually replaced by the Gaussian Error Linear Unit (GELU) function, defined as:

$$\forall x \in \mathbb{R},\ \text{GELU}(x) = x \cdot \Phi(x)$$

Where Φ is the Cumulative Distribution Function for Gaussian Distribution.

2.6.2 ResNet

One of the major issues with the feed-forward architectures was the vanishing gradient problem. This problem is that the magnitude of the gradient tends to decrease when the number of layers, also called depths, of an architecture increases. Therefore, the training of deeper neural networks was difficult, preventing the use of such architecture to solve complex tasks. The ResNet architecture was introduced in 2015 by He et al [64]. In the paper "Deep Residual Learning for

Image Recognition", the authors introduced the deep residual learning technique to mitigate the vanishing gradient problem. The idea is to avoid learning the directly the mapping of a layer $H(x)$ but learn the difference between the input and output $F(x) = H(x) - x$. In practice, this desired output $H(x) = F(x) + x$ is produced with shortcut connections in feedforward neural networks.If the dimension does not match a linear projection is made to match the dimensions.

With this method, the authors successfully train very deep CNN networks with up to 152 layers. The version of the architecture with 50 or more the shortcut connections are placed around a stack of three layers. The first is a 1×1 convolutions responsible of reducing the number of channels, the second one is the actual 3×3 convolution that is performed on the reduced number of channels and the last 1×1 convolutions restore the number of channels before summing the output with the shortcut connection. This block, called a "bottleneck", helps reduce the training time and is also used in other architecture proposed in latter research.

2.6.3 MobileNet

The MobileNetV2 [157] is designed to work in resource-constrained environments and mobile devices. Indeed, while the previous architectures, such as ResNet, performed accurate prediction, they require high computational resources. According to [7], MobileNetV2 requires more than four times less floating point operations than the smallest ResNet architecture while having better performances. It also have three times less trainable parameters. To increase its efficiency, this architecture relies on multiple key components.

First, MobilenetV2 reuses the depthwise separable convolutions introduced in MobileNetV1 [74]. Which replace convolutions by two operation. A depth-wise convolution which combine a convolution with only one filter per output channel, and a 1×1 standard convolution called point-wise convolution.

As discussed in Subsection 2.6.1, the standard convolution takes as input a tensor of dimensions width $\times$ height $\times$ $\text{channel}_{\text{in}}$ and use a kernel of size $\text{channel}_{\text{in}} \times \text{channel}_{\text{out}} \times \text{N}_{\text{w}} \times \text{N}_{\text{h}}$. Hence, This computational complexity is width $\times$ height $\times$ $\text{channel}_{\text{in}} \times \text{channel}_{\text{out}} \times \text{N}_{\text{w}} \times \text{N}_{\text{h}}$. Meanwhile with the depthwise separable convolutions the cost is reduced to width $\times$ height $\times$ $\text{channel}_{\text{in}} \times$

$(\text{channel}_{\text{out}} + N_w \times N_h)$.

They also use inverted bottleneck blocks instead of the "bottleneck" of the ResNet architecture.

That optimization makes it suitable for a constrained environment and explains its use in many embedded applications. This is also the architecture selected for the design of several accelerators described in Section 2.3 [204, 79].

2.6.4 Transformers

Transformer [196] are a recent architecture initially developed for Natural Language Processing (NLP) tasks. The core structure of a transformer is based on a repeated stack of blocks, each composed of a multi-head self-attention layer followed by a series of fully connected layers (linear transformations). These layers involve matrix multiplications, in contrast to the convolution used in CNNs. These operations allows transformers to analyze relationships across the entire input, however it also increases computational costs significantly. The computation of the attention mechanism causes the complexity to scale quadratically relative to the number of tokens in the input.

Transformers have been adapted to computer vision with the Vision Transformers (ViT) [29], in which the image is segmented into 16x16 patches, each one being interpreted as a token in NLP. The ViT model then processes these tokens with the transformer architecture to capture spatial relationships across the image. ViT has demonstrated superior performance over traditional CNNs on various vision benchmarks [29], highlighting the architecture's potential for complex visual tasks and in transfer learning [133, 144]. However, this improved performance comes at high computational cost, as transformers generally require a substantial number of parameters and operations. For instance, the results from the original ViT paper show that the base version of this architecture requires approximately the same number of floating-point operations to train as a ResNet-50, which is significantly more than MobileNetV2, as discussed previously. This architecture also has 86 million parameters, which is more than twenty times the 3.4 million parameters of a MobileNetV2 [157].

Chapter 3

Pretraining Dataset for Compact Models

3.1 Introduction

Deep neural networks are known to be data hungry [115], particularly when they include a large number of parameters. Transfer learning alleviates this problem by pre-training an upstream dataset to improve performance in downstream task, or accelerate the training process [161, 35]. Pre-training is also useful when the target domain data are not sufficient to learn an effective model from scratch, and the gain obtained from the upstream model is larger than the loss of representativeness due to domain shift [134]. The importance of pre-training grew with the advent of deep neural networks, whose learned representations are transferable [161]. Recent studies of transfer learning [29, 32, 49, 90, 224] focused on pre-training models with increasingly large number of parameters and amounts of data. They conclude that increasing the size of models and of data improves the performance in target tasks, at least until saturation is reached [1].

While interesting, these studies disregard the fact that transfer learning is often useful when training and inference capacity are limited [77]. In this work, we investigate transferability under constraints by analyzing the effects of core factors which drive this process. During pre-training, we notably test the influence of: (1) pre-training for compact deep architectures, which

are likely to be used in transfer learning for constrained environments [211]; (2) deep neural network architectures since they are known to influence both the upstream and downstream performance [83]; (3) the amount of available training data, as well the ratio between number of classes and samples per class for a fixed-size upstream dataset, since downstream accuracy saturation was already analyzed for large deep architectures [1, 33, 91], but not for compact ones. During inference, we analyze the influence of: (1) the type of downstream training strategy, with the deployment of linear probing and full fine tuning, since the depth of the fine tuning process leads determines the degree to which features are adapted to the downstream task or preserved from the upstream model [224]; (2) the number of images per class in the target datasets to assess transferability in four few-shot settings [32] and full dataset availability scenario since they are all important in practice.

We run experiments using different subsets of ImageNet [27] as upstream dataset, four deep architectures, and with six downstream datasets designed for diverse visual tasks. The empirical study reported here concludes that:

- Downstream performance saturation is reached much faster with the compact deep architectures compared to the large architectures analyzed in previous studies [1, 91]. Increasing the number of classes as the amount of upstream data grows does not delay saturation. This finding indicates that very large pre-training datasets are not needed to obtain good downstream performance with compact deep architectures.

- The structure of the upstream dataset (number of classes, samples per class) has a small influence on downstream accuracy once there are enough data in it.

- The type of architecture makes a difference, particularly when linear probing is used for downstream training. In this setting, architectures with higher-dimensional output features are clearly a better choice.

- The performance of the full fine tuning and of linear probing depends on the downstream configuration. The latter strategy is competitive when the domain shift between upstream and downstream tasks is small and/or in case of low-shot settings. Since linear probing training is much simpler, it should be considered for deployment in these cases.

As a whole, the reported results give a comprehensive view of transfer learning under constraints. They provide a sound baseline for future work performed by both researchers and engineers.

3.2 Related Work

Transfer learning is important for practical applications of deep learning, and is the subject of a large number of existing studies. We discuss the most relevant studies for transfer learning under constraints, which is in focus here. Prior works are further put into perspective when analyzing the results of the different experiments.

Past examination of pre-training tend to show that increasing the size of the upstream dataset has a positive effect on downstream accuracy [113, 170, 223]. However, recent studies, such as [1, 33], find that the improvement tends to saturate, and this phenomenon occurs faster for self-supervised pre-training. The works cited above focus on the total size of the dataset in terms of samples, and they give less importance to the structure of the dataset in terms of the number of classes and of samples per class. The importance of the dataset structure was highlighted for domain adaptive transfer learning [128]. The authors of this study conclude that adding more data, including more classes, can have a deleterious effect on downstream performance. In this study, the pre-training has the prior knowledge about the target task. In contrast, we pre-train models without any assumption regarding the content of downstream tasks in order to avoid meta-overfitting [224].

The strategy used for downstream training has a strong influence on performance. Past studies [91, 224] tested pre-training for full downstream datasets, but also in few-shot learning settings. They showed that full fine tuning of downstream models is better than linear probing, which consists in retraining only the final fully-connected layer of the model. This finding seems intuitive since fine tuning adapts the features of downstream models to the characteristics of the downstream tasks. A nuance was brought by [96], a study which shows that linear probing is actually better than fine tuning when testing with out-of-distribution data for downstream tasks. However, past results were reported for the pre-training with large deep models. It is interesting to study whether they hold for smaller models, which are in focus here. Importantly,

we run a more systematic study of few-shot settings compared to [91, 224] in order to have a fine-grained analysis of the merits and limitations of the two strategies. We note that there exist more refined transfer strategies. Image-level adaptation of the strategy is proposed in [54], adaptive fine tuning is explored in [53], while a combination of features from different layers is used in [35]. While interesting, they are out of the immediate scope of this work, which focuses on two opposite strategies.

Previous works focused on transfer learning for computationally-constrained devices showed the benefits of freezing part of the networks [204, 209]. However, they focused on hardware optimization [204] in order to reduce the overall energetic footprint of the implemented deep models, or architecture quantization [209] to reduce their parametric footprint. Here, we take a complementary approach and pay more attention to the upstream and downstream data, and use network scaling to preserve the precision of downstream representations.

3.3 Study Setup

3.3.1 Datasets

Pre-training datasets. Following the common practice [35, 91, 224], we transfer data from a single upstream dataset to all downstream tasks in order to assess the generalization capacity of the upstream model. The authors of [224] underline the importance of mitigating meta-overfitting when transferring knowledge. They advise to create the upstream model independently of any knowledge about downstream data. Therefore, we generate different versions of pre-training datasets by sampling ImageNet21k [27]. The classes included in these datasets are selected randomly from the set of leaves classes that have enough samples per classes. A first series of tests use a variable number of classes from 100 to 6000 and fixes the number of 500 samples per class. These subsets are used to assess if downstream performance continues to increase or saturates when adding new classes. A second series of tests simultaneously vary the number of classes and samples to keep the total number of samples in the dataset constant. The size of the dataset is 1M images and the number of classes varies from 1000 to 6000. This experiment

could not be carried out with fewer classes since ImageNet does not contain enough richly-represented leaf classes to reach the target dataset size. This setting corresponds to an upstream training on a fixed budget. These subsets are used to assess whether class diversity or individual class representations are more important. Note that there is no assumption made regarding the similarity between the upstream dataset and the downstream ones. This is important in order to simulate a situation in which pre-training is done without knowledge of the downstream tasks, and thus ensure the generalization of the proposed transfer scheme.

Dataset	Oxford-IIIT Pet [138]	DTD [21]	GTSRB [168]	SVHN [127]	FGVC [114]	Cifar100 [93]
# classes	37	47	43	10	100	100
# training/class	99.432	39.979	619.512	7325.7	33.34	500
stdev training/class	1.534	0.144	457.377	2800.661	0.474	0
# test/class	99.135	39.979	293.721	2603.2	33.33	100

Table 3.1: Downstream datasets statistics.

Downstream datasets. A thorough evaluation of the usefulness of pre-training requires the use of multiple and diversified downstream datasets [1]. We follow this observation and transfer upstream models toward six downstream tasks: Oxford-IIIT Pet [138] is designed for pet race recognition, Describable Textures Dataset (DTD) [21] provides different types of textures as perceived by humans, GTSRB [168], Street View House Numbers (SVHN) [127] includes house number images, FGVC-Aircraft (FGVC) [114] is designed for aircraft model recognition and Cifar100 [93] includes commonsense-level classes [153]. These datasets cover a wide range of visual tasks, and the conclusions drawn from a study of pre-training involving all of them are robust. Their main statistics are presented in Table 3.1. Images are resized to match the input size used during pre-training which is 224x224. Standard data augmentation [93, 64] which includes random cropping and random horizontal flipping is applied for four datasets out of six. Horizontal flipping is deactivated for GTSRB and SVHN because they mainly represent classes that depends on the orientation.

3.3.2 Downstream data availability

It is important to study the influence of the amount of data available for downstream tasks since pre-training is most needed when downstream data are scarce. We first run experiments with the

full datasets, and then test with four few-shot learning regimes. For this we limit the number of samples per class in the downstream task to 1, 5, 10 and 25. This is a finer-grained investigation of the influence of data availability compared to [224], where a single few-shot learning setting were used. To mitigate data selection bias, we follow a standard procedure in few-shot learning and sample training images five times for each regime.

3.3.3 Downstream training strategies.

Following [224, 96], we run experiments with fine tuning and linear probing, two opposite strategies. Fine tuning retrains all the layers of downstream models, while linear probing only retrains the final layer. Fine tuning is usually preferred [91, 96, 224] since the full retraining of the downstream model adapts it to the domain of the downstream task. Linear probing [161] is less adaptive since it exploits pre-trained features as such. The latter can be interesting if the computational capacity of the device is limited [204] and/or when the amount of available training data is insufficient to learn a full model in an efficient manner. We note the existence of the other downstream training strategies [53, 54, 35], but their usage is out of the immediate scope here.

3.3.4 Training details

Training parametrization is done using a procedure which is inspired by the lightweight sweep mode proposed in [224]. Fixed values are used for most hyperparameters across network architectures and tasks. While not fully optimized for each task, this mode allows a fair comparison in a constrained environment.

The resolution of the input images used for training was classically set to 224x224 [64].

Upstream training. To make the results comparable we used the same hyper-parameters for each training. All of them were trained during 110 epochs using the "1cycle" learning rate scheduler [164]. We chose this scheduler because it allows fast training [164], which was important in order to reduce the time needed to pre-train all the networks on all the different subsets of ImageNet21K. The batch-size is set to 128, and the maximum learning rate for the OneCycleLR is

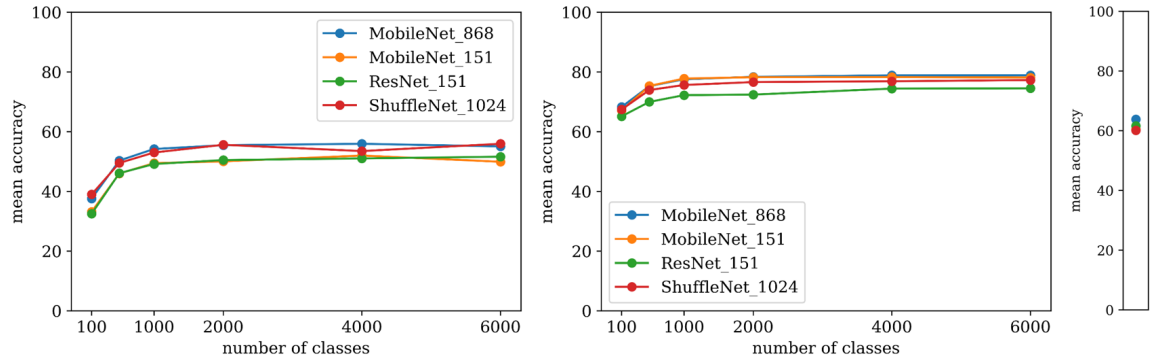
set to 0.005. Also even if recent works demonstrate that increasing the weight-decay on the head of the network lead to better downstream performances [223], we use a constant weight-decay of 5e-4 for all the layers to avoid any side effects.

Downstream training for full dataset. Two common training strategies for transfer are considered here. Linear probing is deployed because it is well adapted for constrained environments [204]. We use a fully-connected layer for classification, which receives the features provided by the upstream model. This final layer is trained for 100 epochs, with a ReduceLrOnPlateau[1] learning rate scheduler based on the loss metrics with a patience of 5. This allow us to stop the training if the learning rate reaches 10^{-8}. The initial learning rate is set to 0.01. The weight decay is set to a constant 5e-4 over the whole network for the same reason as the upstream training.

Full fine tuning adapts all the layers of the architecture during downstream training, and past studies indicate that it outperforms linear probing, even in few-shot learning scenarios [91, 224]. While it requires more computational power than linear probing, it can be implemented on edge devices after optimization [204]. During this training we used the same parameters as for linear probing except for the initial learning rate which is set to 0.001 to avoid damaging the pre-trained features in the first steps.

Downstream training in a few-shot setting. A recent work pointed out that training for a large number of epochs can be beneficial if downstream datasets are small [33]. However, overfitting sometimes occurs if this process is run until its end. To accommodate these two observations, we fine-tune for a large number of epochs (2500 for single shot), but stop the process if the learning rate value is too low (10^{-7}). Following [33], when increasing the number of samples per class we divide the number of epochs by the number of samples per class to keep the same number of updates during the different training. The learning rate is again reduced on plateau.

[1] https://pytorch.org/docs/stable/generated/torch.optim.lr_scheduler.ReduceLROnPlateau.html



(a) Linear probing. (b) Fine tuning. (c) w/o pretrain.

Figure 3.1: Mean accuracy on the downstream tasks as a function of the number of classes, and using 500 images per class for all tested deep architectures.

3.3.5 Deep Network Architectures

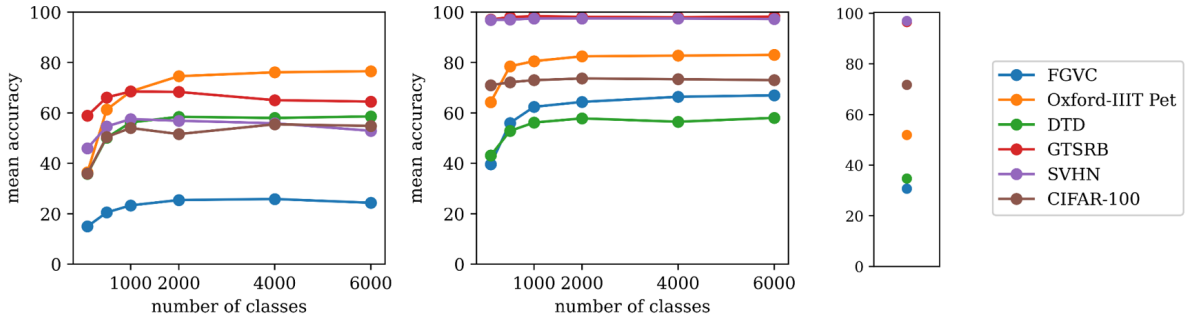
We choose MobileNetv2 [157], ShuffleNetv2 [112], Resnet18 [64] architectures for our experiments. To make the results obtained with these architectures comparable they are all downscaled to reach a size of 1M parameters. When scaling strategies are presented in the original papers, as it is the case for MobileNetv2 [157] and ShuffleNetv2 [112], we follow them here. We use a similar strategy for ResNet18, and also create a second version of MobileNetv2 to study the effect of embedding sizes.

MobileNetv2. We selected MobileNetv2 [157] because it was designed for computationally constrained environments. We test two version of the model scaling. The first version, MobileNet_{868}, is scaled using the scaling method from the original paper [157, 74] with a width multiplier of 0.678. The number of channels in the output of the inverted residual blocks are 11, 16, 22, 43, 65, 108 and 217 and the size of the vector in output of the feature extractor is 868. The second version, MobileNet_{151}, is downscaled by fixing the embedding size of the extractor to 151 and before using the strategy from [157] to adapt the rest of the network. We created MobileNet_{151} to match the embedding size of ResNet_{151}, and also test the influence of the embedding size against MobileNet_{868}.

ShuffleNetv2. We used the scaling method proposed in the original paper [112], to downsize this architecture to 1M parameters. The output channels of each stage are multiplied by a subunit factor (0.866), while leaving the first and the last convolution unchanged. Since the output of

the extractor is 1024, we will refer to the downsized architecture as ShuffleNet$_{1024}$.

ResNet18. ResNet18 [64] is a generic architecture which is often used in literature. It has over 11M parameters in its full version and we downscale it to reach 1M parameters. The number of channels in each residual block is reduced uniformly, using a 0.295 width multiplication factor.



(a) Linear probing. (b) Fine tuning. (c) w/o pretrain.

Figure 3.2: Detailed accuracy for each downstream task with MobileNet$_{868}$.

3.4 Experiments

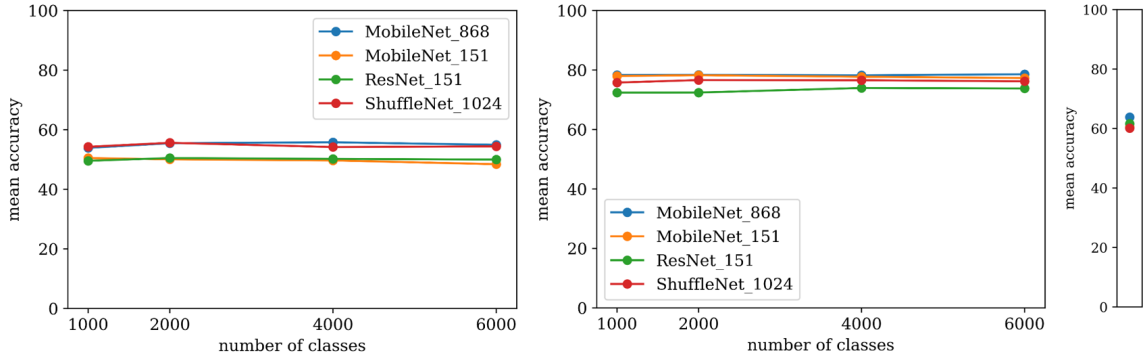
3.4.1 Effect of a larger pre-training dataset

Past studies of pre-training [1, 49, 113] showed that larger upstream datasets translated into higher downstream performance. However, it was noted that saturation occurs beyond a certain point, and adding supplementary data is not useful anymore [1]. Given that past studies were focused on large deep neural networks, it is interesting to analyze the behavior of smaller models with respect to the number of upstream classes. We keep the number of images per class constant at 500, regardless of the total number of classes included in the pre-training dataset.

We present the results obtained with different architectures in Figures 3.1a and 3.1b with a linear probing and full fine tuning of downstream tasks, respectively. Performance increases a lot when the total number of classes used for pre-training is small. An important gain is observed between 100 and 500 classes, particularly for linear probing. The relative gain starts to decline between 500 and 1000 classes, and even more between 1000 and 2000 classes. Then, performance starts to saturate beyond 2000 classes. Some performance variability is observed in the 1000 to

6000 classes range for all tested architectures and both training strategies when increasing the number of classes, but they do not exceed 3 accuracy points between the lowest and highest points. This finding is important insofar it indicates that increasing the number of classes is not useful for deep architectures designed for constrained environments. Performance saturation occurs for much smaller volumes of data compared to previous studies [1, 49, 113], which focused on larger deep architectures and tested much larger upstream datasets. The conclusion is that pre-training of compact deep architectures is effective with an upstream dataset which includes approximately 1M diversified images.

An interesting observation is that fine-tuning-based training is clearly better than a direct use of features learned upstream via linear probing. The accuracy gain when using the first strategy is over 15 points for all tested numbers of classes of the upstream dataset, and all backbone architectures. A similar finding was already reported in literature [91, 96] for larger deep architectures, and is confirmed here for compact architectures, which are adapted for computationally-constrained environments. We also note that the gain offered by fine tuning over linear probing is larger when upstream training is done with a low number of classes (up to 1000). This is explained by the stronger sensitivity of linear probing to the quality of the upstream features, due to the direct use of features versus an adaptation of them for downstream tasks during fine tuning.



(a) Linear probing. (b) Fine tuning. (c) w/o pretrain.

Figure 3.3: Mean accuracy on the downstream tasks when the total size of the dataset is constant (1M images) and the number of images per class decreases when the number of classes increases. The minimum number of classes is 1000 because ImageNet does not contain enough leaf classes with enough images to run experiments with 100 and 500 classes.

The performance obtained with the four tested architectures varies for both downstream training strategies (linear probing in Figure 3.1a, fine tuning in Figure 3.1b). Globally, MobileNetv2 and ShuffleNetv2 behave better than ResNet after scaling to 1M parameters. This is somewhat expected since the first two types of architectures were designed purposely for computationally-constrained environments. Interestingly, the difference between MobileNet_{868} and MobileNet_{151} is much smaller when the upstream models are fine tuned (Figure 3.1b) compared to linear probing (Figure 3.1a). This indicates that models which have a wider output are more adequate for linear probing if the overall number of parameters is equivalent. An explanation resides in the higher dimensionality of the frozen features produced by MobileNet_{868}, which favors the separability of classes downstream. This finding is in line with the well-known result reported for wide residual networks [221].

We propose a per-dataset view of results obtained with linear probing and with fine tuning in Figure 3.2. These results are reported with a MobileNet_{868}, which provides the best overall results in Figures 3.1a and 3.1b. Fine tuning is better than linear probing for five datasets out of six and number of classes included in the upstream datasets. The differences are much stronger for downstream tasks whose domain shift compared to the upstream task is larger. This is the case of FGVC, SVHN and GTSRB, three datasets focused on aircrafts, house number plates and street signs. The domains are not well represented in ImageNet and the retraining of all weights during fine tuning is clearly needed. Linear probing is better than fine tuning only for DTD . This result might be explained by the low total size of this dataset, which includes only 1600 training images, combined with the large domain shift between ImageNet and this texture-focused dataset.

We also report performance with downstream tasks without pre-training on average and per dataset to assess the overall effect of pre-training. The difference between the best and worst of the four tested architectures is 4 points in Figure 3.1c, but there are strong differences between individual datasets (Figure 3.2c). The global comparison shows that the use of pre-training for fine tuning brings a significant improvement compared to training from scratch. The dataset-level analysis (Figure 3.2) gives more insight into the merits and limitations of pre-training with the two downstream training strategies. Linear probing is effective for small domain shifts between

upstream and downstream tasks (Oxford-IIIT Pet) or when the dataset size is small (DTD), but provides lower performance in the other cases. This is expected since the features are not adapted to each task. Fine tuning provides similar performance to that of training from scratch for easy tasks, such as GTSRB and SVHN, and brings important improvements for FGVC, Oxford-IIIT Pet and DTD.

3.4.2 Effect of pre-training with a constant-size dataset

We complement the analysis from Subsection 3.4.1 with experiments run with a dataset which total size is kept fixed at 1M images. Here, the number of images per class decreases when the number of selected classes increases. The dataset is balanced, meaning the samples are distributed evenly between classes. This corresponds to an upstream training with a fixed sample budget.

The figures 3.3a and 3.3b show the mean accuracy on the six downstream task with linear probing and full fine-tuning. The global trends are similar to those observed in Figure 3.1, as is the accuracy obtained with linear probing and fine tuning in different configurations.

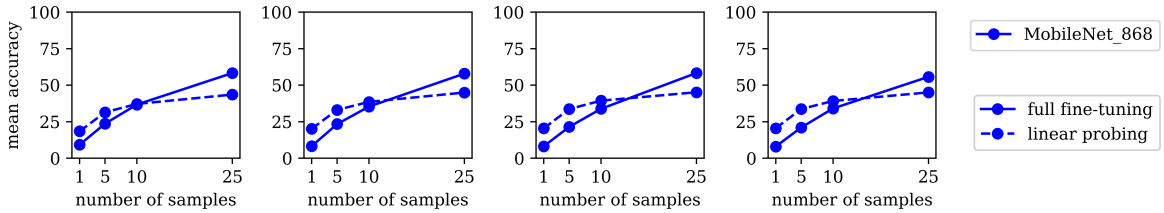


Figure 3.4: Mean accuracy on the downstream tasks in low-shots four settings. The number of image per class is constant (500) in the upstream dataset. It includes 1000, 2000, 4000, 6000 classes, from left to right.

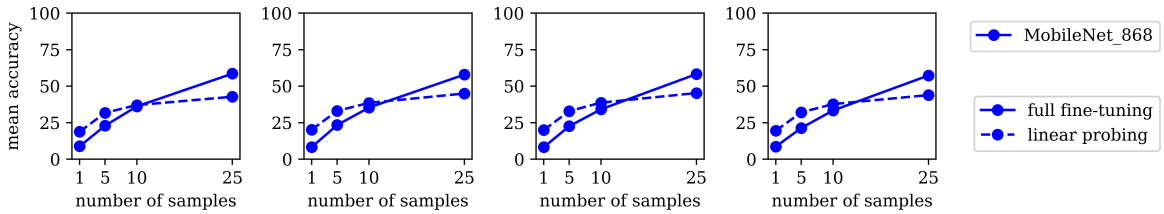


Figure 3.5: Mean accuracy on the downstream tasks in low-shots four settings. The total size of the upstream dataset stays constant (1M images). It includes 1000, 2000, 4000, 6000 classes, from left to right.

We observe a performance gain of up to two points when the number of classes in the dataset changes from 1000 to 2000. Beyond 2000 classes, performance seems to oscillate, and is even slightly decreasing for linear probing (Figure 3.3a). For this strategy, the obtained accuracy decreases in all tested configurations except one when the number of classes increases from 4000 to 6000. This result can be explained by 2 opposite phenomena. While a more diversified pre-training dataset is likely to lead to a better representation, a larger number of classes also makes the upstream task more difficult. Our finding is consistent with the saturation of downstream performance beyond a certain point, even when upstream performance is improved [1]. Here, the improvement of the representation brought by the addition of new classes is degraded by the growth of the complexity of the task, and by the scarcer representation of each class when the total number of classes increases.

The comparison of the results for 4000 and 6000 classes from Figures 3.1 and 3.3 is interesting because the total size of the upstream dataset is smaller in the latter configuration. There are 2M and 3M images for 4000 and 6000 classes in Figure 3.1, but only 1M in Figure 3.3. This finding shows that a representation of upstream classes with fewer images does not have a significant impact on downstream performance.

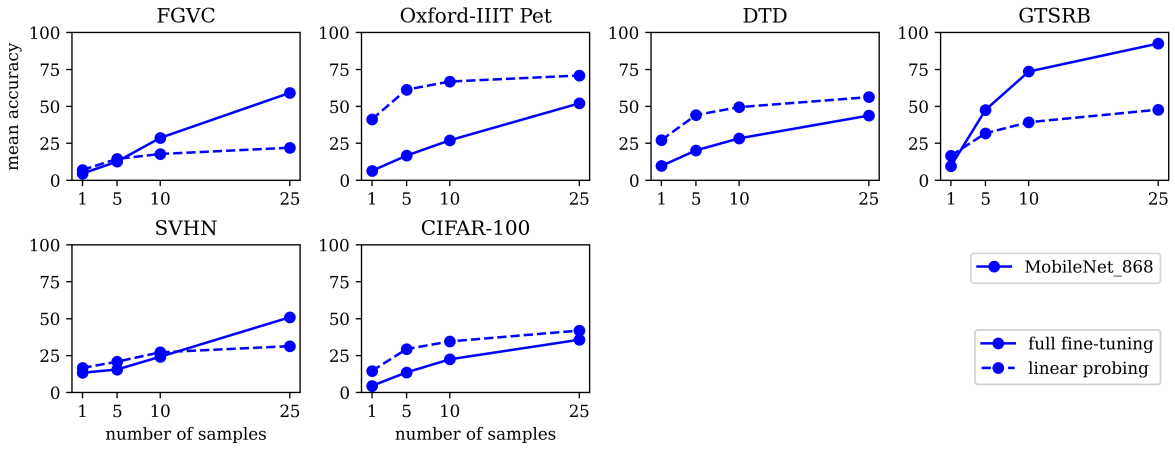


Figure 3.6: Downstream accuracy for each downstream task with 6000 classes and 500 images per classes for the pre-training.

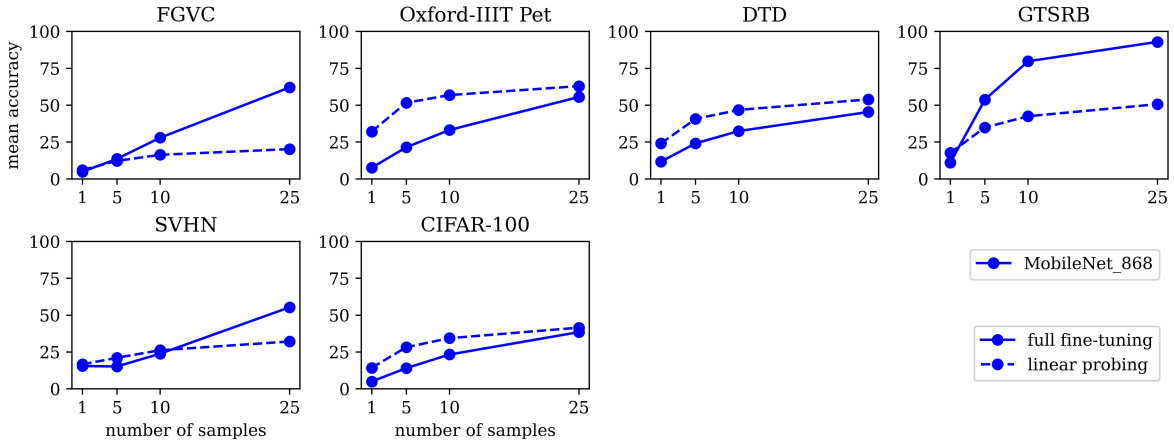


Figure 3.7: Downstream accuracy for each downstream task with 1000 classes and 500 images per classes for the pre-training.

3.4.3 Effect of pre-training in few-shot scenarios

Pre-training is particularly useful when only few samples are available per class since deep neural networks are data hungry [32]. Performance is reported for pre-training with MobileNet_{868} for 1M parameters, the configuration which works best for downstream training with all data. We plot accuracy for pre-training with 1000 and 6000 classes for each model to also assess the influence of this parameter. We investigate the performance on downstream tasks for different few-shot learning regimes. We again report results with two upstream pre-training strategies: the number of images per class is constant in Figure 3.4 and the total size of the dataset is constant 3.5. The observation that performance is very similar in the two upstream dataset configurations remains valid for all tested few-shot settings. Interestingly, the obtained results indicate that linear probing is significantly better than full fine tuning up to 10 samples per class. The full training of the downstream models is difficult with very few samples due to the occurrence of overfitting [215]. The gap between the two training strategies narrows when the number of samples increases. Fine tuning becomes better than linear probing when 25 samples are available. Naturally, this tendency is even clearer when all samples are available for downstream tasks, as we discussed in Subsection 3.4.1. Our results are at odds with those reported previously [224], regarding the superiority of fine tuning over linear probing even in a few-shot setting. The main difference comes from the scale of the networks, with much larger architectures being tested

in [224]. The observations made here show that the downstream training strategy should be adapted depending on the quantity of data available for target datasets.

Performance of few-shot learning is similar for upstream dataset variants with a constant number of images per class (Figure 3.4) and with a constant size dataset (Figure 3.5). This echoes the results obtained when using the full downstream datasets. The difference between pre-training with 1000 and 6000 datasets is larger for linear probing compared to fine tuning. The accuracy gain between these two variants of the upstream dataset reaches approximately 2, 2 and 1.5 accuracy points for 1-shot, 5-shots and 25-shots settings, respectively. This is expected since linear probing makes direct usage of upstream features, and thus benefits more from strong pre-trained representations.

The global comparison of linear probing and fine tuning in few shot scenarios, presented in Figures 3.4 and 3.5, is refined with a presentation of the accuracy per dataset. Figures 3.6 and 3.7 illustrate results for MobileNet_{868} pre-trained with 6000 and 1000 classes, respectively. Linear probing is clearly better than fine tuning for datasets which are semantically related to the content on the pre-trained models, such as Oxford-IIIT Pet and CIFAR-100. ImageNet [27], the dataset used for pre-training, includes a large number of classes which describe the natural world, which are also well represented in the three downstream datasets for which probing has good results in few-shot scenarios. Linear probing accuracy is also better for DTD, a texture dataset which does not benefit from fine tuning even when all its images are available (Figure 3.2). Fine tuning is the better option for FGVC, SVHN, and GTSRB, the three datasets with a larger domain shift compared to ImageNet.

3.5 Conclusion

We investigate transfer learning in image recognition under constraints through a comprehensive empirical study, which analyzes the roles of the dataset used for upstream training, the performance of different deep architectures, the results obtained with two opposite training strategies. Our experiments confirm findings reported in previous studies regarding performance saturation for large deep architectures [1, 49, 113]. It shows that the phenomenon appears faster in terms

of scale of the upstream dataset, due to the compactness of tested architectures. As a result, the conclusion to use increasingly larger pre-training datasets to improve performance [1, 49, 113] does not seem justified for compact deep architectures. In contrast to past studies [91, 224] which assert that full fine tuning is preferable to linear probing, our result shows which strategy is better depending on number of images available. Linear probing is a good strategy when the domain shift is small and/or when the available number of samples per class in the downstream task is low. It is also interesting due to its lower computational complexity, which is important in constrained environments [204].

We used variants of a fully-supervised dataset for pre-training. It would be useful to extend it by testing weakly-supervised and unsupervised pre-training. It would be equally interesting to explore ways to predict an adapted downstream training strategy based on an analysis of the domain shift between the upstream and downstream tasks.

Chapter 4

Impact of Time on Classification Accuracy

4.1 Introduction

Models trained on large-scale datasets provide robust representations [133, 145, 150]. Certain hardware implementations leverage these robust representations to create highly efficient systems [204, 78, 79] using models with fixed weights. However, as discussed in the previous chapter, fine-tuning only the linear classifier on a new task does not achieve the same transfer performance as fully fine-tuning the entire network. Therefore, freezing the entire feature extractor to create a more efficient implementation would limit performance on tasks that differ from the original training task, resulting in limited reusability of the hardware.

However, even if the task remains the same, some other limitations are worth exploring. Since networks are trained on data collected at a particular point in time, understanding how robust pre-trained models are to temporal data shifts is critical. This analysis is crucial as such shifts could make hardware relying on frozen weights obsolete over time.

In this chapter, we will explore several key questions. First, are recent deep learning models robust to temporal shifts in data? What factors contribute to this phenomenon, and how can the problem of temporal shifts be mitigated? Are all object types equally affected by temporal shifts?

Finally, are there any metrics applicable to unlabeled data that could help predict whether a class and/or a dataset is prone to temporal shifts that could negatively impact the pre-trained model?

To tackle the challenge of temporal shift and address these research questions, we introduced a new dataset called VCT-107. This dataset, consisting of 107 classes and 951,176 images sourced from Flickr, was necessary because the majority of existing datasets lack temporal information. One of the few exceptions, YFCC100M [184], also includes upload timestamps, but over too short a period and distributed across too many classes, which limits the range of experiments that can be performed.

In this chapter, we outline the steps taken to construct the VCT-107 dataset. We describe the clustering method employed to reduce the annotation workload, as well as preprocessing techniques, such as deduplication, used to eliminate highly similar images. This dataset allows us to observe performance degradation when the test period does not correspond to the training period. This performance loss increases as the gap between these two periods widens.

Next, using these new data, we present comparisons between different models. This allows us to examine the influence of various architectures but, more importantly, different pre-training strategies on robustness to temporal shifts. In this analysis, we also include CLIP models[145] to determine whether models trained with textual data are more resilient. We further test factors such as the number of samples from the original period used during training.

We experimented with several methods for adapting the network to mitigate the impact of temporal shifts. These include updating the linear classifier using some or all of the data retained from earlier periods. We also tested several domain adaptation methods [116, 118, 48], as these offer solutions that require less storage, which can be an important consideration in some embedded applications.

Finally, we analyze temporal shifts to determine which classes are the most affected. We analyze embeddings using high-dimensional distribution distance to determine whether certain distances can be used to determine classes impacted by temporal shifts. In practice this can be useful in assessing if the model's performance has degraded and need adaptation for some classes.

Dataset	Yearbook [46]	FMoW [20]	AgeDB [125]	AmsterTime [218]	Core50 [109]	**VCT-107**
Input	Yearbook photos	Satellite images	Faces in the wild	Landmarks	Video frames	**Web images**
Prediction	Gender	Land use	Face identification, Age, Gender	Visual place recognition	Object recognition	Object recognition
Time range	1930-2013	2002-2017	∼1890-2017	∼1850-2020	/	**2007-2020**
#domains	-	-	-	2	8 train + 3 test	**5 periods**
#classes	2	63	568	1,231	50	**107**
#samples	37,189	118,886	16,488	2,462	164,866	**951,176**

Table 4.1: Comparison of visual datasets containing temporal information.

Through this chapter's experiments, we show that all models are sensitive to temporal shifts to different degrees. Although strong pre-training helps mitigate the effects, temporal shifts remain a significant challenge. Fortunately, updating the classifier alone is sufficient to prevent substantial performance degradation, avoiding the need to adjust all network weights. In scenarios with limited storage capacity, domain adaptation algorithms provide an efficient way to address the problem with minimal overhead.

Additionally, our experiments reveal that certain classes and class groups are more vulnerable to temporal shifts. However, some high-dimensional distribution distances can help identify these classes and indicate if the network needs to be adapted.

4.2 Related work

Visual classification datasets. Recent advances in visual classification learning have been fostered by the publication of visual datasets [190]. Classification models are commonly evaluated on CIFAR-100 [93], ImageNet-1000 [156] or on domain-specific, fine-grained datasets such as Food-101[10], Stanford Cars [92] or Oxford Flower-102 [131]. However, these datasets are not designed to challenge the robustness of models against distribution shifts.

Specific visual datasets have been proposed for the task of domain adaptation, including MNIST with diverse backgrounds [41], Office-Home [197], Citiscapes[22] and DomainNet [140]. Datasets like ImageNet-R [65] or ImageNet-D [154] are designed to benchmark the robustness of ImageNet-trained models against domain shifts. The CORE50 dataset [109] comprises 50 objects filmed in 11 settings and is built specifically for continual domain adaptation. In addition to disparate backgrounds and image styles, distribution shifts may arise from different geogra-

phies [5, 14, 151], weather conditions [73, 122], or ethnicities [106, 69]. Another line of work proposes to use synthetic images. For example, Visda [141] focuses on the simulation-to-reality shift, and SHIFT [171] uses a generative model to control shifts in scene elements for autonomous driving.

Works on temporal shifts. One of the rare vision datasets with a distribution shift directly caused by time is AmsterTime [218], a collection of 2,500 images matching a street view in Amsterdam (using Mapillary navigation platform) to a historical archival image from the same scene. AgeDB [125] is a dataset for face verification in the wild and contains temporal information about a person's age in each image. The Wild-Time benchmark [216] focuses on temporal shift and covers two visual tasks: gender prediction with the Yearbook dataset [46] and prediction of land use with satellite images from FMoW [20]. We compare our proposed VCT-107 with these datasets in Table 4.1. Natural language processing works study temporal changes in lexical semantics and propose methods to detect such changes, e.g., [95, 8]. The authors of [56] distinguish between semantic shifts that are more cultural or more linguistic. We refer to the survey of [97] for more details on semantic shifts in word embeddings. In our experiments, we use models trained on visual data only and vision-language models.

Biases and generalization. Datasets partially represent the visual world and are inherently biased [190]. The authors of [36] identify three main types of biases, arising from (1) selecting a subset of items that differ from the general population, (2) framing the object to convey a specific message via the image composition, or (3) assigning different labels or wrong semantic categories. Biases lead to distributional shifts between the data used to train a model and the data encountered during its operational phase, challenging the model on unseen data.

Several lines of work aim to increase a model's ability to generalize to new domains or tasks. Generalization is favored by the quantity, quality, and diversity of its training data [90, 130, 133]. Pre-training with large corpora [191] and multiple data augmentation techniques [169, 137] is now common practice. Multimodal language vision models such as CLIP [144] and DALL-E [146] show strong transferability without per-sample labels. Their self-supervised training uses up to billions of image-text pairs. Diversifying representations using features from intermediate layers of a pre-trained model [35] or combining multiple encoders [178] can also improve transferability.

Finally, transfer learning and domain adaptation focus on reusing knowledge gained for solving a source task in a different but related problem [12]. We refer to the surveys of [188, 163] for a detailed review of transfer learning and domain adaptation algorithms.

Continual learning. Continual Learning (CL) builds models that can adapt to their environment and incrementally develop more complex skills and knowledge [187, 6]. Domain-Incremental Learning (DIL) [194] is a CL scenario that learns a classification model sequentially, with each step in the sequence introducing data from a new domain. The set of target classes remains the same throughout the process, but class distributions change. Thus, the challenge is to recognize classes in an increasing number of domains without storing all previous data, a challenge addressed in different ways. The approach of [98] does not require task boundaries but relies on a costly clustering step. The work of [186] leverages self-supervised learning. An adapter method [142] is applied in [135] to adapt a pre-trained model on the initial subset efficiently and then incrementally train a classifier based on a linear discriminant analysis layer. Similarly, RanPAC [118] combines a Parameter-Efficient Transfer Learning (PETL) procedure with a random layer that projects samples in a higher dimensional space to improve discrimination. FeCAM [48] also uses a fixed feature extractor and focuses on incrementally updating a classifier based on the Mahalanobis distance.

4.3 Constitution of VCT-107

We describe the VCT-107 collection, processing, and labeling process. Then, we analyze the resulting dataset.

Data collection. We downloaded images from the Flickr platform because its content covers diverse visual concepts over a long interval, and its API facilitates the collection of images using predefined temporal intervals. We collected images for five distinct periods: 2007-2008, 2010-2011, 2013-2014, 2016-2017, and 2019-2020, denoted as 07/08, ... 19/20. Grouping images in two-year intervals ensures enough training and test images for all classes. The one-year gap between intervals facilitates the analysis by better separating the data subsets. We initially collected data for the 22/23 period but dropped it because the number of images was insufficient

for most classes.

To ensure diversity in the dataset, we prompt ChatGPT-4o to provide class names and definitions from the following nine topics: plants, animals, food, buildings, vehicles, household objects, electronic devices, sporting equipment, and apparel: *Please provide a list of 50 popular [TOPIC_NAME] types using a JSON format for the output.* Since the LLM answers sometimes include less than 50 items, the initial class count is 439. We verify the correctness of the proposed class names and descriptions to filter out hallucinations. We then collect up to 3000 Flickr images and associated metadata for each target year using Flickr's internal search engine ranking. This initial uncurated dataset includes nearly 11 million images.

Image rights and safety. Following [184], we collected only freely redistributable images, but this approach did not provide enough samples per period for most classes. Therefore, we broadened the search and collected Flickr images with all licenses. This change has practical implications for the distribution of copyrighted content. We follow recent practice in sharing visual datasets [160] and provide the URLs rather than the image files themselves.

Concerned about image safety issues [183], we instructed the annotators to remove any image that could be considered "not safe for work" and to flag any image that might have been taken without the subject's consent. We provided them with clear textual safety guidelines and interacted with them when in doubt. If an image from a cluster was flagged, we removed the entire cluster.

Dataset preprocessing. We preprocess the dataset to minimize the labeling effort. We compute the embeddings of all the collected images using a ViT-B/14 pre-trained using DinoV2 [133]. We remove near-duplicates using a 0.9 cosine similarity threshold between each pair of images uploaded in the same year. We cluster images using K-means [139] with 50 clusters per year. We keep only clusters involving at least two Flickr users to ensure a minimal social consensus on the class's visual representation. We use these clusters to accelerate the annotation process.

Content annotation. We implement a dedicated labeling interface (illustrated in the appendix). Each row of images represents the visual summary of a cluster and contains at most ten images. These images are sampled uniformly based on their L2 distance to the cluster centroid

and shown in increasing order of distances from left to right in the interface. This sampling relies on the hypothesis that there is a correlation between the distance to the centroid and the representativeness of an image for a given class. We provide annotators with textual instructions illustrated by examples. The instructions require them to annotate the rightmost image of each row, including a depiction of the visual class according to the LLM's definition. They state that the object may be located in any image region and that other objects can be visible. Three participants contributed to the annotation task, and one participant annotated each cluster. To reduce the annotation effort, the participants first label the image subset from 2020 because it contained the fewest images. Then, we rank the classes according to the number of relevant images labeled for 2020 and keep the 125 most populated classes. Finally, we ask participants to label the images from the remaining nine collection years for these 125 classes. This step provides a fast labeling of the images, but some noise might subsist. Next, we check the annotations of the test subset to ensure a reliable evaluation.

Candidate images for the test set are sampled uniformly from the selected clusters and labeled by the other annotators. They are included in the final test subset if the three annotators agree on their relevance. The specific annotation of test images also validates the clustering-based annotation. We find that the three participants agree on the relevance of over 98% of the images sampled from the clustering-based annotations. We keep a class only if it has at least 40 valid test images and 100 training images per year.

VCT-107 summary and illustration. The dataset includes 107 classes from 9 topics, ranging from 31 animal classes to 2 types of electronic devices. The dataset includes between 2881 and 21237 samples per class, with at least 483 images per period. The class names and sample distribution are detailed in the appendix. The images were uploaded by over 248772 Flickr users, who each contributed an average of 4.4 images. The minimum, mean, and maximum user counts per class are 1106, 4289, and 11593, respectively. These numbers ensure that VCT-107 class representations benefit from social consensus. Nevertheless, a selection bias occurs, as with any visual dataset [36].

Figure 4.1 illustrates the impact of time on visual classes. Due to space restrictions, we sample three images of four classes taken during the earliest and most recent VCT-107 peri-

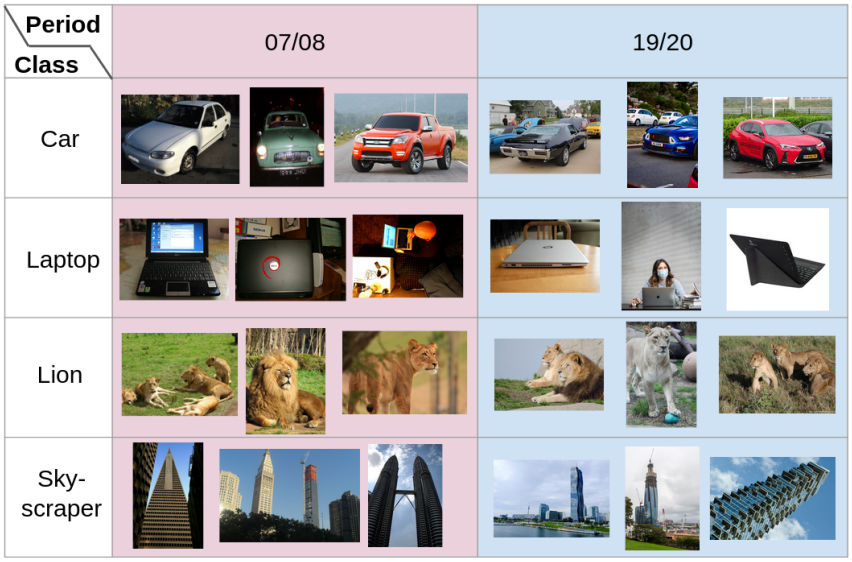


Figure 4.1: Samples representing four VCT-107 classes during the 2007-2008 and 2019-2020 periods. The car, laptop, and skyscraper classes illustrate the appearance changes of human-made objects whose design changes over time, shifting the representations learned for these classes. Lion has a stable appearance, and the representation shift is much smaller in this case.

ods. Changes over time in the representations of human-made objects are mainly determined by the lifespan of these objects [162], itself determined by technological advancements, visual design trends, regulation, and brand strategies [198]. Vehicles illustrate the complex interaction between these factors with a continually evolving technological and visual design. For instance, the shift toward electric batteries changes the appearance of cars to match technical requirements [4] but also to highlight their difference from fossil-fuel-based vehicles and increase their appeal [124]. Similar considerations apply to mass-consumption electronic devices, such as laptops and smartphones [2]. Their usage and representations depend on technical advancement and their functions for users of different ages, incomes, and world regions. Interestingly, the visual representations of human-made objects mix the old and the new, highlighting users' fascination for the past [149]. Figure 4.1 shows that users upload vintage cars during both periods. Visual representation changes are also observed in architecture, with increasing stylistic diversity and the availability of new building materials and techniques [55].

The impact of time is reduced for natural classes such as lions because their appearance does not significantly change. However, trends also appear, particularly for classes closely associated with humans, such as pets. For instance, the popularity of dog breeds evolves [67], influencing

the class' visual representation. Equally important, framing biases [36] might still affect their depictions regarding how they are photographed and in which contexts.

While we focus on the impact of time, multiple factors influence visual class depictions. VCT-107 classes are subject to a selection bias [36, 190]. This bias is amplified in operational conditions due to the long-tailed nature of visual datasets [214]. Another important bias comes from the demographic characteristics of the users of Flickr, with variations of social status, ethnicity, gender, and location across time [132]. In particular, some regions of the world tend to be more represented than others in visual datasets [151]. This leads to an imbalanced depiction of visual concepts, particularly for classes such as buildings. The cameras used to take the photos influence image quality and can affect the representations learned. Finally, disparities due to lighting conditions or image colorimetry also occur [180]. Together, these factors create temporal shifts in visual classes. We quantify their effect on image classification in Section 4.4 and provide an embedding-based analysis in Section 4.5.

4.4 Experiments

4.4.1 Experimental setup

We split VCT-107 into five temporal periods, as described in Section 4.3. We run experiments with the entire training set and in low-shot scenarios by sampling 200, 100, 50, or 20 images per period. To assess the models' generalization ability, we train them on each period and measure their test accuracy on the other periods. In some experiments, we also accumulate training samples over time to evaluate the effect of retraining from scratch. The test set of each period is fixed across experiments and contains 80 images per class.

We use SGD with a momentum set to 0.9, a weight decay set to $4 \cdot 10^{-5}$, and a cosine learning rate scheduler initialized at 0.1. We train for 100 epochs for full training and 20 epochs for linear probing (LP). This transfer learning method freezes all parameters except the classification layer [91]. Unless otherwise stated, data preprocessing is the same and consists of rescaling the images to $256 \cdot 256$ pixels, then randomly cropping to $224 \cdot 224$ pixels and normalizing

using ILSVRC [156] statistics. We will justify this choice of data augmentation with empirical experiments in the next subsection.

4.4.2 Data Augmentation Selection

Throughout the rest of Section 4.5, we employ standard data augmentation techniques. Below, we present the results of preliminary tests that guided our choice of these specific augmentations.

To do so, we used ResNet18, as it is the easiest model to train from scratch with a "small" dataset such as VCT-107. We selected three sets of data augmentations. (i) The first does not include any data augmentation. (ii) The second uses the most common data augmentation operation, which consists of randomly cropping the images and then flipping them horizontally with a probability of 0.5. (iii) Finally, the third set of data augmentations includes additional transformations, such as random adjustments to luminosity, saturation, contrast, and hue, randomly rotating the images, random cropping, and finally randomly flipping the images horizontally. The factors for luminosity, saturation, and contrast are picked from the range [0.6, 1.4], the hue factor is chosen from the range [-0.4, 0.4], and the rotation is uniformly selected from the range [-20°, +20°].

In Figure 4.2, we can see that the model is affected by the change in the data collection period, regardless of the data augmentations chosen. However, we note that not performing any data augmentation generally reduces the model's accuracy.

In Figure 4.3, we observe that in the case of linear probing with a pre-trained model, applying either more data augmentation (option (iii)) or no data augmentation at all (option (i)) leads to worse performance. This can be explained by the fact that the feature extractor was pre-trained using only the data augmentations corresponding to the intermediate data augmentation set (option (ii)).

In conclusion of these experiments, we decided to only use the standard data augmentations that correspond to those used in the pre-training of the backbones. We also maintain these values for tests with non-pre-trained networks because the results from Figure 1 show that we do not gain any improvement by using additional data augmentations.

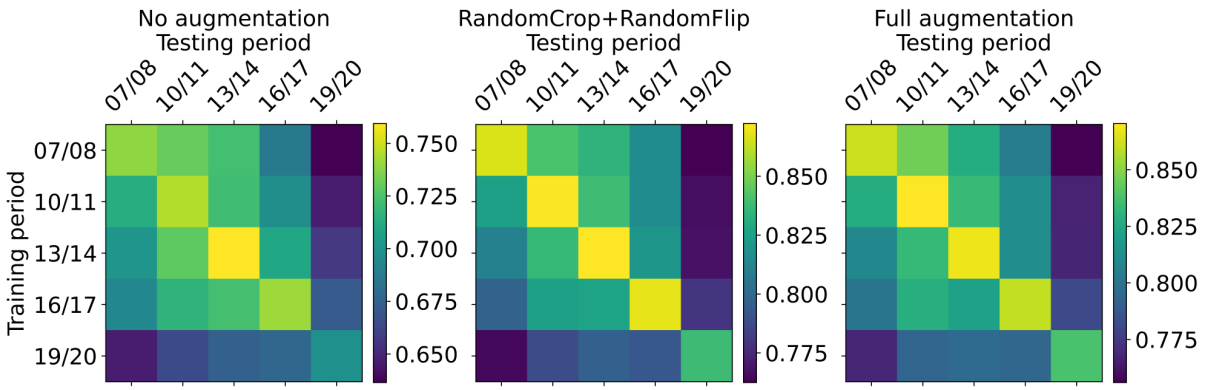


Figure 4.2: Accuracy when training a ResNet18 from scratch on one period and testing on the others. The experiment was done with three sets of data augmentation.

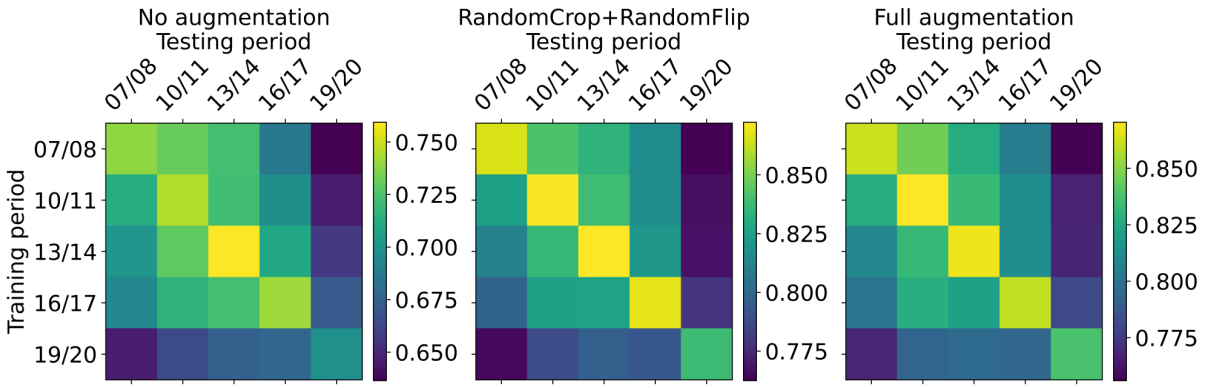


Figure 4.3: Accuracy when training a linear probes on a ResNet18 on one period and testing on the others. The pretraining was done with ILSVRC. The experiment was done with three sets of data augmentation.

4.4.3 Impact of the training strategies

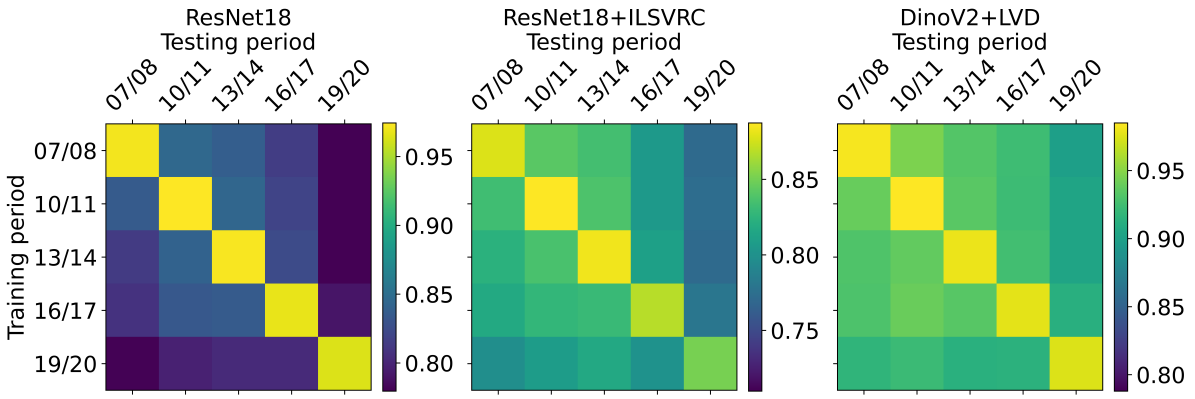


Figure 4.4: Accuracy across temporal periods when training with the entire VCT-107 dataset using three different backbone models. To facilitate comparison, the range of values is displayed from 80% of the maximum accuracy value of each backbone.

We evaluate the capacity of pre-trained and fully-trained models to mitigate the temporal shift. Full training involves the entire VCT-107 dataset because it requires more samples. We use a smaller ResNet18 instead of a ViT as it requires less sample to train. Therefore in Figure 4.4 we experiment with: (1) a ResNet-18 [64] trained from scratch, (2) a ResNet-18 pre-trained on ILSVRC [156] (3) a DinoV2 ViT-B/14 pre-trained on the LVD for easier comparison with subsequent experiments. The primary objective of these experiments is to assess the accuracy stability over time, not to compare the accuracies obtained with each backbone. Figure 4.4 shows that the backbone trained from scratch exhibits the largest performance variation. This highlights the importance of pre-training for mitigating temporal shifts. The pre-trained ResNet-18 comes second, with the ViT-B/14 network pre-trained using DinoV2 achieving the highest stability across time. The results from Figure 4.4 confirm that combining strong pre-training and linear probing constitutes a good baseline for mitigating temporal shift.

However, due to significant architectural differences, the model trained using DinoV2 is not directly comparable to the ResNet-18 model. A comparison with more similar architectures is necessary to assess whether all pre-training methods yield the same robustness to temporal shifts. To isolate the effect of DinoV2's unsupervised pre-training, we compare its generalization performance against a ViT-B/16 model pre-trained in a supervised manner on ILSVRC. Additionally,

we include a ViT-B/16 variant pre-trained on the full ImageNet-21k to evaluate the impact of pre-training dataset size. We evaluate those by training linear probes with 200 samples per class and period. This removes the issue of having an imbalance in the training data, which may slightly alter our results. We provide the results for all combinations of training and testing periods in Figure 4.5.

With 200 samples per period, DinoV2 achieves a higher average accuracy and improves generalization over time. DinoV2 experiences a maximum accuracy drop of 7.2%, whereas the ViT-B model pre-trained on ILSVRC sees a loss of up to 9.0%. Although the ViT model pre-trained on ImageNet-21k performs slightly worse than DinoV2, it also exhibits a maximum accuracy loss of 6.7%. These results suggest that the primary limitation of the ILSVRC pre-training method lies in the quantity of data rather than its supervised nature.

Finally, we also consider the increasingly popular Contrastive Language-Image Pre-training (CLIP)[145], as its multimodal approach could offer greater robustness. To maintain consistency with our previous experiments, we experimented with two models: the standard ViT-L/14 and a ViT-B/16. For linear probing, we attach the linear classifier after the projection to the shared latent space, retaining only the vision component of the model. This method follows the original approach described by Radford et al.[145]. Figure 4.5 indicates that temporal shifts also affect multimodal models. However, the ViT-B/16-based CLIP experiences a maximum accuracy loss of only 5.5%, which is lower than that of the other ViT-B models, suggesting that CLIP training provides increased robustness.

The next subsection provides zero-shot classification scores for each period to illustrate their relative classification difficulty.

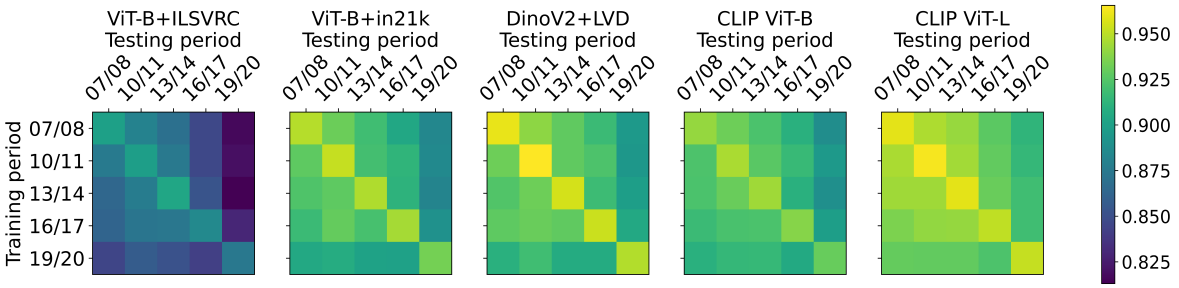


Figure 4.5: Accuracy across time for cross-period training and testing. All models use linear probing with 200 samples per class and period.

2007-08	2010-11	2013-14	2016-17	2019-20
86.6%	87.4%	86.7%	85.3%	84.0%

Table 4.2: Zero-shot accuracy of ViT-B/16 CLIP model on each period

4.4.4 Zero-shot classification with CLIP

In the previous experiment of Section 4.4, we noticed that the accuracy on the last period is lower than the others. The temporal shift does not seem to be the only reason for these results. Indeed, even when the training and testing periods remain identical, the accuracy in the 2019-2020 period is slightly lower. This leads us to hypothesize that the relative difficulty of each period varies to some extent.

To investigate this, we measure the zero-shot accuracy achieved with CLIP models, which provide a baseline accuracy for each period without any VCT-107 training data. This approach offers an unbiased assessment unaffected by the training period. Additionally, this experiment helps confirm that, despite the overfitting observed when training on a specific period, training a linear classifier remains more effective than relying on CLIP's zero-shot capability.

To realize this experiment, we used the same ViT-B/16 CLIP as we used in Subsection 4.4.3. To measure the zero-shot accuracy, we follow the original approach described by Radford et al [145]. For the classification, we used the CLIP scores. Those are computed using the cosine similarity between the two modalities in the common embedding space. We used the names of the classes used to create the dataset as input for the textual part of the model.

The results, presented in Table 4.2, indicate that the final period (2019-2020) is indeed slightly more challenging than the others. Nevertheless, the zero-shot accuracy is consistently lower than the accuracy obtained by training a linear classifier, as shown in the previous subsections. This confirms the advantage of using a trained linear classifier.

4.4.5 Impact of the training set size

The size of the training set strongly influences the generalization ability in static datasets [190, 130]. Following the findings from Subsection 4.4.3, we use DinoV2 with linear probing. We ex-

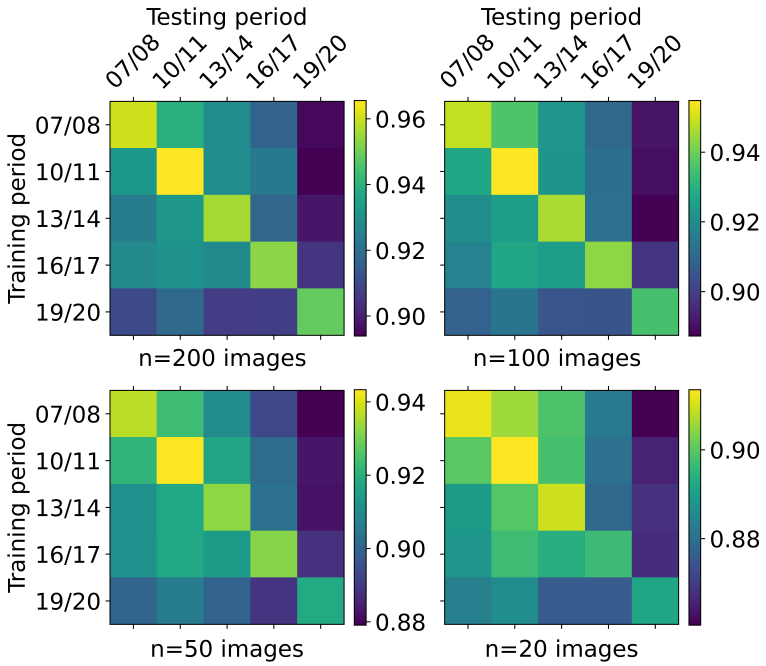


Figure 4.6: Accuracy over time for DinoV2 ViT-B/14 and linear probing for $n = \{200, 100, 50, 20\}$ samples per class and period.

periment with $n \in \{200, 100, 50, 20\}$ training samples per class and period to assess the influence of time in low-shot settings. We repeat each experiment 4 times for each low-shot scenario using four random seeds for sampling and report average results in Figure 4.6.

Reducing the number of images per class harms the overall performance since individual class representations progressively weaken. Figure 4.6 highlights that when n decreases, the accuracy on periods other than the training period decreases slower than on the same training period. The average accuracy obtained when testing on the same period as training drops by 5.2% when n goes from 200 to 20. Meanwhile, the average accuracy for the other periods only drops by 3.3%. We also observe that when testing on periods other than the training period, the *relative* accuracy loss decreases slowly as n decreases. In this case, the average accuracy loss is 3.8% and 2.0% for 200 and 20 training images per period, respectively. This result highlights the ability of strong pre-training to handle temporal shifts. This is important in practice, as many real-life datasets include limited training data per class [165].

Algorithm	DinoV2 ViT-B14 LVD-142m [133]	ViT-B16 Aug-Reg IN21k [191]	#Stored params
NCM	92.6	90.7	$82 \cdot 10^3$
FeCAM-1	92.7	92.6	$672 \cdot 10^3$
FeCAM-n	<u>94.3</u>	92.9	$63 \cdot 10^6$
RanPAC	**94.8**	**94.6**	$108 \cdot 10^6$
Replay20	93.4	92.5	$1.6 \cdot 10^9$
Accumulate	94.1	<u>93.0</u>	$16 \cdot 10^9$

Table 4.3: Average accuracy for six algorithms and two pre-trained backbones. The algorithms are ordered by the number of parameters added to the backbone. The storage requirements are computed for images of size 3×224×224. The best results are shown in bold, and the second-best are underlined.

4.4.6 Domain-incremental learning

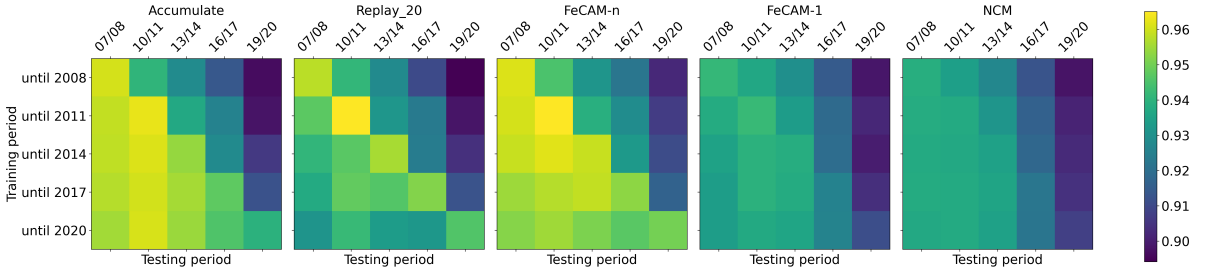


Figure 4.7: Accuracy comparison when accumulating samples ("Accumulate" and "Replay_20") and using linear probing vs. updating the model using incremental learning ("FeCAM-1", "FeCAM-n" "NCM"). Experiments with a pre-trained DinoV2 ViT-B/14 network.

The experiments in Subsections 4.4.3 and 4.4.5 do not include any mitigation strategy other than using a strongly pre-trained backbone. Here, we test the effectiveness of continual learning (CL) [136] algorithms against temporal shifts. Domain-incremental learning (DIL) [194] is a sequential learning process where each step corresponds to a new domain. Here, a domain is a data collection period, e.g., 2007-2008, 2010-2011, etc. The set of classes to recognize remains the same, but their distribution changes. Each step of the process aims to obtain a model that can recognize all classes, regardless of the data collection period. We follow the DIL protocol from [118] and include all $T = 5$ periods in the test set. The average accuracy is computed as the mean value of the test accuracy across the T training steps: $A = \frac{1}{T}\sum_{t=1}^{T} Acc(M_t, \bigcup_{i=1}^{T} D_i)$, where M_t is the model trained at step s_t on data collected at time t and D_i is the test dataset

corresponding to period i.

We experiment with several competitive CL algorithms using a fixed encoder. The Nearest Class Mean classifier (NCM)[81] updates a running mean embedding vector for each class and predicts the class using the cosine similarity to class prototypes. FeCAM [48] also stores a mean vector for each class and computes a shared feature covariance matrix ("FeCAM-1") or one feature covariance matrix per class ("FeCAM-n"), used to compute the Mahalanobis distance between the embedding of a test sample and the mean class vectors. RanPAC [118] combines a PETL step with a random projection from dimension 768 to 10,000 to better separate classes. At inference, distances to class means are computed using the Gram matrix. These algorithms do not store past images, which is useful when storage or privacy issues must be considered. Still with a fixed encoder, we also consider linear probing with a cumulative replay buffer of 20 images per period ("Replay-20") and a cumulative replay buffer containing all the training images seen so far ("Accumulate").

We report the average DIL accuracy in Table 4.3. The results show that DIL algorithms match or outperform naive replay and accumulation strategies while requiring at least 250 times less additional memory. RanPAC and FeCAM-n, the two algorithms that perform the most refined modeling of past knowledge, obtain the best accuracy. Figure 4.7 indicates the DIL algorithms reduce the accuracy losses for test data from past periods but are ineffective for future data. The results confirm the effectiveness of CL algorithms in mitigating the effects of domain shifts when combined with a pre-trained model. However, higher accuracies tend to be obtained with higher memory requirements.

4.5 Temporal shifts analysis

We investigate the importance of temporal shift in VCT-107 by analyzing the embedding space and the performance variations per general topics over time.

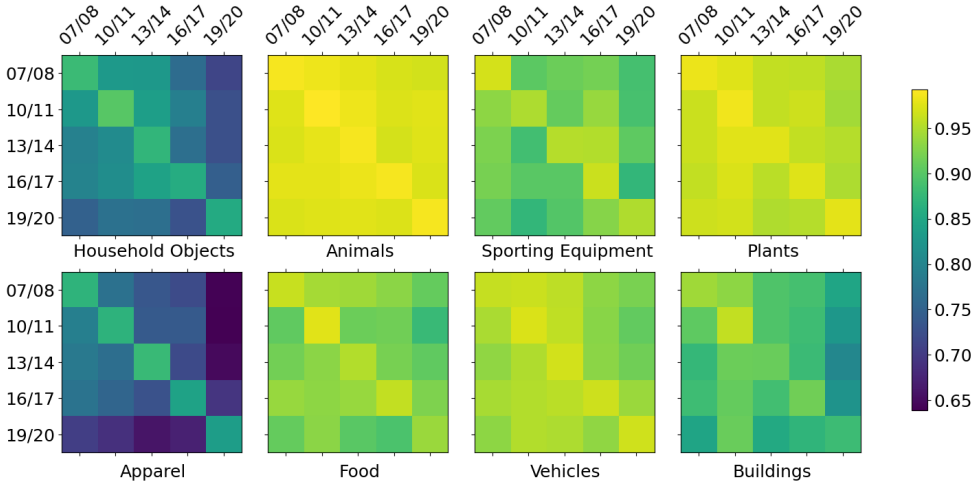


Figure 4.8: Accuracy across temporal periods for the general topics included in VCT-107. The results are obtained using a DinoV2 backbone with linear probing and 200 training images per class. We exclude *Electronic Devices* because this topic has only two classes.

4.5.1 Topic-based analysis of temporal shifts

We discuss the effect of temporal shifts for eight VCT-107 general topics by refining the analysis of results from Subsection 4.4.5 obtained with 200 images per class and period. Figure 4.8 shows that the intra-period accuracy varies significantly depending on the topic. *Household Objects* and *Apparel* are the most challenging topics, while *Animals* and *Plants* are the easiest ones. The effect of temporal shifts is also more significant for human-made classes than natural ones. The fact that time has variable effects for different topics is important in practice since it indicates that temporal adaptation could be tailored at the class level.

4.5.2 Embedding-based analysis of temporal shifts

After observing the effect of the time shift on VCT classes, we aim to find a measure of the shift that can be used to predict its impact. Measuring the impact would otherwise require training on new data, which implies continuous labeling of incoming data. This is important, as the metric could indicate when and on which classes the model should be adapted.

We compare the following distances: (1) the $L2$ distance between the centroids of two distributions, (2) the Fréchet Inception Distance [30] (FID) used by [120] to measure the gap

between two distributions when studying generalization, (3) the energy distance [52, 177] that tests for equal distribution in high dimensions without distributional assumption [177] and (4) the Sinkhorn distance for optimal transport, a popular approximation of the Wasserstein distance [26, 119, 201].

We measure the distance of each class's mean DinoV2 embedding distributions for each pair of temporal periods in which the training and test periods are distinct. We compare these distances to the loss of accuracy for the same pairs of periods. Let $A_{c,origin}$ and $A_{c,target}$ be the test accuracy on the class c for training and the target periods. The relative loss in accuracy is given by: $(A_{c,target} - A_{c,origin})/A_{c,origin}$. For readability, in Figure 4.9, we average the distances and the accuracy losses by general topic for every pair of a train and a test period. In this figure, the values are displayed for each distance D, topic T, and the pair of periods p_{origin} and p_{eval}. $c \in T$ represents the classes of the topic T. The values displayed are :

$$D(T, p_{origin}, p_{eval}) = \frac{1}{card(T)} \sum_{c \in T} D(S_{c,p_{origin}}, S_{c,p_{eval}})$$

$$Accuracyloss(T, p_{origin}, p_{eval}) = \frac{1}{card(T)} \sum_{c \in T} \frac{A_{c,p_{eval}} - A_{c,p_{origin}}}{A_{c,p_{origin}}}$$

With $S_{c,p}$ the sample of the class c and the period p.

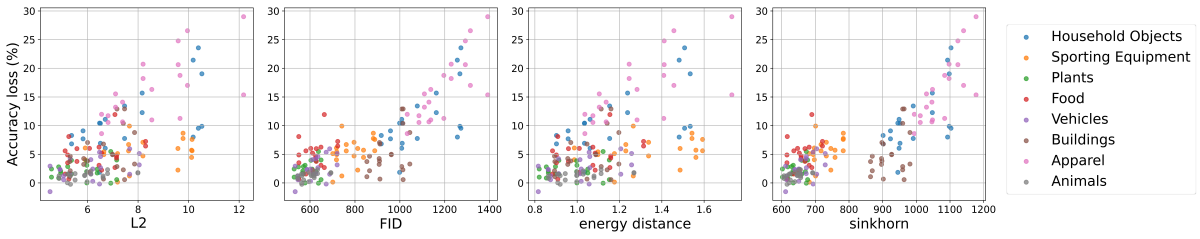


Figure 4.9: Relative accuracy loss over time for the classes of the general VCT-107 topics as a function of distribution shift measured with four metrics. Results aggregate distances and accuracies for individual classes for the assessed training-test period pairs.

We observe that for each considered metric, the average distances generally grow with the accuracy loss. FID and Sinkhorn's algorithms successfully assigned higher values to the two most affected topics. They could be used in practice to decide whether to update the visual representation of a topic (or even an individual class).

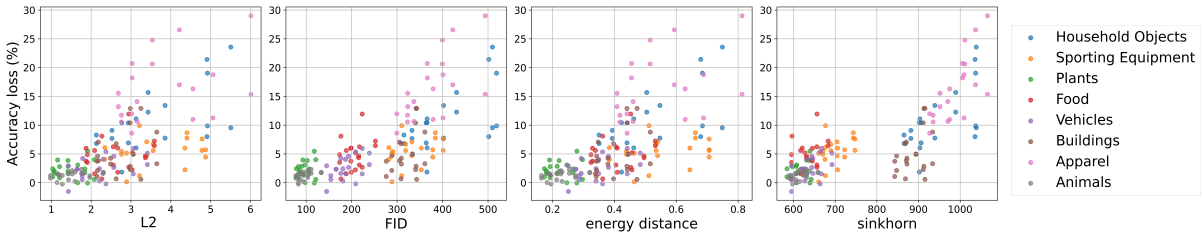


Figure 4.10: Relative accuracy loss over time for the classes of the general VCT-107 topics as a function of distribution shift measured with four metrics. In this figure, all the samples belonging to the *topic* are considered as samples from a single distribution. This differs from the results given in Figure 4.9 for which the distances were computed between class distributions instead of entire topics.

However, while this measure provides a good indication of the distances that can be used to predict performance losses due to temporal shifts and identify the affected classes, these measures are not always practical. Indeed, we consider a class at a given period as a distribution, but the data may not necessarily be separated by class. Therefore, we also test the case where the distributions between which we measure the distances contain multiple classes. In this case, we group them by topics. Intuitively, the distributions will have a greater chance of being multimodal.

The distance represented on the x-axis thus becomes:

$$D(T, p_{origin}, p_{eval}) = D(S_{\bigcup_{c \in T} c, p_{origin}}, S_{\bigcup_{c \in T} c, p_{eval}})$$

Intuitively, the distribution will have a greater chance of being multimodal. This can be important for FID as it supposes multivariate normality [30]. However, this dataset doesn't count enough samples to realize a reliable multivariate normality test. Therefore, we could not test this hypothesis. In Figure 4.10, we can see that in this case, FID fails to assign smaller values to *Sporting Equipment* than *Household Objects*. This would have been the expected result as the topic *Sporting equipment* suffers from less loss in accuracy when generalizing to other periods.

Meanwhile, the results for Sinkhorn stay very similar to those observed in Figure 4.9. We would, therefore, consider Sinkhorn as a better alternative when measuring the temporal shift for multiple classes at once.

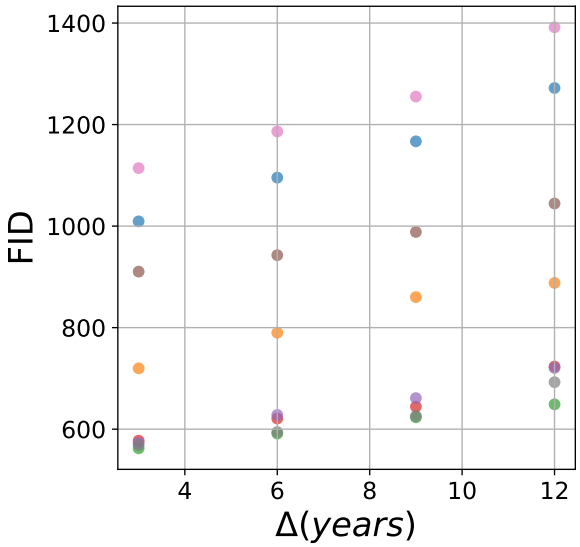


Figure 4.11: VCT-107 topic distributions shift measured with the FID distance as a function of the temporal interval between training and test periods. We use the same colors for topics as Figure 4.9.

Finally, we check if the magnitude of the shift increases with the time difference between the two distributions by the FID metric. We average the distances corresponding to each topic based on the temporal interval between the target period and the others. Figure 4.11 confirms that the average distance grows for all topics as the interval increases. This confirms the temporal aspect of the domain shift, which differs from other types of domain shifts that can be evaluated using specific datasets like ImageNet-D [154].

4.6 Discussion and conclusion

We introduce VCT-107 to analyze the impact of time on visual classification models. We experiment in several settings and observed an accuracy drop when training and testing during different periods. The performance loss generally grows with the temporal distance between the training and testing periods. We also observe that the classification accuracy loss depends on the type of classes.

Practical guidelines. Based on these results, we propose the following recommendations

for improving classification performance under temporal shifts:

- *Use self-supervised pre-training* with linear probing to reduce the performance variability over time. The results confirm the improved generalization ability of pre-trained models [133] in a temporal context. The relative pre-training performance depends on the implemented type of learning, the dataset size, and the dataset diversity, but the relation is not always straightforward. In particular, our experiments indicate that self-supervised visual learning outperforms multimodal training in an image classification task despite visual models having a smaller parametric footprint and using a smaller training set.
- *Implement continual learning algorithms* to further mitigate performance loss on past data if retraining with all historical data is not an option. CL algorithms require the storage of samples or statistical information but make the training process much more efficient. They benefit costly learning processes, such as training foundation models with huge datasets [23].
- *Consider the type of visual classes* when learning over time. Our experiments confirm the intuition that human-made objects are more impacted by temporal shifts. However, there are important differences between the different types of human-made objects. The analysis of class embeddings indicates that using an appropriate distance can predict the need to update the training set. Adapting the update rate for different classes is particularly interesting when training foundation models, whose updates are needed to keep pace with novelty but are also costly.

Limitations and future work. We discuss limitations and suggest future work directions to mitigate them.

- The dataset is sourced from Flickr. Adding supplementary sources would increase the generality of the findings, but access to photos with temporal metadata over such a long period is not straightforward. We can only hope that social platforms will facilitate researchers' access to data, but we observe an inverse trend in practice.
- The reliance on third-party data when building large datasets is needed, which induces redistribution limitations. Acknowledging the potential reproducibility limitations, we follow recent practices [160] and provide the image URLs to respect image rights.
- We tested pre-training and continual learning to mitigate temporal shifts. Other techniques can

be considered to counter this shift, including (1) PETL methods [72] with adapters designed for temporal shifts, (2) domain adaptation methods [163] to better preserve past knowledge through time, and (3) imbalanced learning methods [60, 148] to rebalance performance when the number of samples per class varies within a period or between them.

- VCT-107 covers several general topics, enabling their analysis over time. However, the dataset would benefit from including additional topics and enriching existing ones to broaden the analysis. It would also be interesting to analyze the effects of time for finer-grained visual classes. These developments are left for future work, building on the proposed dataset creation pipeline.
- The images included in VCT-107 are labeled for a single class, following a protocol commonly used in image classification [27, 93, 195]. It would be interesting to add multi-label annotations to all dataset images to test the effect of class co-occurrences during classification.
- We fixed classes over time to facilitate comparisons across periods. An enriched version of the dataset could include classes that appear over time. This enrichment would be beneficial for fine-grained datasets.
- The dataset measures the effect of time at the year scale. Refining the temporal scale to enable stream learning would be interesting, as proposed in [17].

We hope this work will stimulate the community's interest in considering the temporal dimension of image classification. This research topic can increase the robustness of deep models, especially for classes whose visual representations change frequently over time.

Chapter 5

Parameter-Efficient Fine-Tuning

5.1 Introduction

Deep models pretrained with large amounts of data [133, 145, 150] provide reusable visual representations in downstream tasks via transfer learning. As we have seen in Chapter 2 freezing the model's topology and parameters allow the creation of very efficient hardware [204, 78]. However, in Chapter 3, we also observed that when transferring to a new task, only training the final classifier (linear probing) often underperforms compared to the full fine-tuning. This limited ability to adapt to new tasks, also observed in other works [35, 224], would reduce the reusability of the created hardware in the case of the network topology and its weights fixed into the integrated circuit. The second most common method, full fine-tuning, adapts the model's representations by updating all parameters. However, this approach does not allow for hardware optimization. It also requires costly retraining [224] and proves suboptimal in few-shot learning scenarios due to insufficient training data, as discussed in Chapter 3. Therefore, a balance must be found between modifying a larger portion of the network (compared to linear probing) to enhance adaptability and freezing a greater part (compared to full fine-tuning) to reduce data requirements and training costs and at the same time to allow for hardware optimization[204, 78]. These reasons motivate us to explore other fine-tuning methods in which most of the parameters are fixed.

Parameter-efficient fine-tuning (PEFT) methods are developed to update only a small subset of the network parameters and balance adaptability and efficiency well. PEFT is particularly relevant in computation- and memory-constrained environments, allowing only a fixed parametric budget. For resource-constrained on-device learning one must implement fine-tuning directly on the hardware [199]. PEFT is implemented with semi-structured [61] or structured [75, 72] modifications of the pretrained models. Semi-structured also adjusts individual parameters, such as specific weights or biases, rather than larger components, allowing for more granular updates to the model.

Recent methods [61, 227] estimate weight importance to select updatable parameters. They use sensitivity [102], a measure encoding the loss variation for each parameter, to drive the PEFT process. The exact sensitivity formulation provides a robust estimation of weight importance. However, its computation is impossible in practice since it would require recomputing the metric on the training set for each new possible parameter value. Instead, a Taylor-based approximation was first introduced for pruning [102] to enable computational tractability. A further approximation uses a one-step gradient descent computation for PEFT [61]. In this chapter, we will study those methods.

Analyzing the impact of the second approximation leads us to two findings. Firstly, the approximation leads to inconsistent behavior that favors fine-tuning layers with a small weight standard deviation. We highlighted this inconsistency and undesirability by constructing counterexamples through a specifically designed adversarial attack. We propose an alternative metrics that improves the robustness of the approximated sensitivity. Secondly, we observed that the second approximation is highly noisy. This led us to question the relevance of using this approximation for an unstructured selection of weights [61]. Therefore, we proposed an alternative method that only relies on structured modification. In this approach, we used the sensitivity to resize the LoRA layers instead of using fixed sizes. This way, layers with fewer sensitive weights are assigned smaller LoRA layers, avoiding the need for unstructured fine-tuning. This change also brings the methods closer to being adaptable for AI accelerators that leverage frozen weights, as these specific hardware systems only utilize structured fine-tuning [204, 78, 79]. Our method also respects the parametric budget throughout the fine-tuning process, unlike existing

methods [227] that start with a larger LoRA and progressively prune them.

5.2 Related Works

Transfer learning [232] enables the reuse of pretrained representations on different target tasks. The availability of models pretrained on massive datasets can significantly improve the downstream performance [133, 1, 145]. Finding a good balance between effectiveness and efficiency is challenging when transferring representations. Linear probing [91, 161] represents an extreme case of transfer learning. It freezes the pretrained model and only trains a linear classification layer. This approach maximizes efficiency at the expense of effectiveness when the gap between the pretraining and the target datasets is large [129]. At the other end of the spectrum, full fine-tuning updates the entire network and facilitates the bridging of larger domain gaps [224]. However, it also entails a significant computational cost that might be prohibitive, especially for hardware implementations of transfer learning [78, 204]. Equally important, full fine-tuning is inefficient in few-shot settings [189] often encountered in real-life applications.

Parameter-efficient fine-tuning aims to maximize transfer performance with few trainable parameters. PEFT methods belong to two main categories. Addition-based approaches accommodate the target task by adding parameters and operations to the initial network. For instance, adapters insert, in series, two linear layers and a non-linearity into transformer models [72]. Another approach requires supplementary operations in parallel without the possibility of merging them [181]. Addition-based approaches are not adapted in constrained contexts because they entail a supplementary computational cost during inference. The second category, parametrization-based PEFT, seeks to avoid this overhead by working under a predefined tuning budget constraint. A first stream of works adjusts network weights [219] or biases [222]. Another work direction adds operations during training and merges with the base model during deployment. LoRA modules [75] insert two fully-connected layers with a small intermediate dimension alongside the existing fully-connected layers. These parallel modules are merged in the network before inference. LoRAs constitute the basis of numerous subsequent methods [59, 108, 226]. We build on this approach and seek a way to adapt the size of LoRAs while respecting a predefined

parametric budget during the PEFT process.

Sensitivity aims to select the most relevant parameters for modification during fine-tuning. Different tasks rely on adapted variants of this metric. Sensitivity predicts the influence of changing a given weight on model performance in pruning methods [102, 213]. It selects the number of bits allocated to different network parts in mixed quantization task [207]. Sensitivity is used for structured PEFT to determine which layers should be modified [99, 227, 61]. The AdaLoRA method [227] uses it in NLP to reduce the size of LoRA modules as training progresses. This functioning is problematic in PEFT on a budget because it only respects the predefined parametric budget at the end of the process but not during fine-tuning. The authors of [61] recently employed sensitivity for semi-structured PEFT in the visual domain [61]. First, the structured part of the method attributes parameters to LoRAs with a predefined size to layers whose estimated sensitivity is sufficiently large to accommodate a LoRA module. Given that the module size is predefined, only a part of the budget is usable. Then, the unstructured part assigns the remaining parameters to the most sensitive individual weights from other layers. This semi-structured approach is appealing for tuning under constraints since it enables a fine-grained allocation of trainable parameters at the start of the training process. Therefore, our work will be based on the latter approach and only mention the methods that apply sensitivity to prune LoRA modules to explain the different approximations. However, our sensitivity analysis indicates that the approximations reduce the metric's usefulness.

5.3 Analyzing bias in sensitivity approximation

PEFT [75, 72] aims to fine-tune a small number of parameters whose update is relevant for downstream tasks. Recent semi-structured [61] and structured [227] PEFT methods use sensitivity to allocate the parametric budget. Sensitivity encodes the loss variation once the parameters are fine-tuned in PEFT or removed in network pruning. It is defined as:

$$S_i = L(D_t, w) - L(D_t, w^*) \tag{5.1}$$

$$w_n^* = \begin{cases} n = i & w_i + \Delta w_i \\ n \neq i & w_n \end{cases} \tag{5.2}$$

Where w_i is the i^{th} parameter and Δ_{w_i} the variation of w_i after modification. That is, after fine-tuning or prunning depending on the application.

The exact computation of this metric would require training network weights individually to determine which ones are most impactful during PEFT. This computation is unfeasible, and a Taylor approximation was introduced in pruning tasks [102] to define sensitivity as:

$$S_i^p = -\frac{\delta L(D_t)}{\delta w_i}\Delta_{w_i} \tag{5.3}$$

This formulation is usable for pruning weights or prunning LoRA because Δ_{w_i} is known and equal to the opposite of the weight value[227]. This information is unknown if the sensitivity is computed on the model weights before fine-tuning them. In such a case, it cannot be considered as pruning and training the weights would be necessary to know their final values. Therefore, a further approximation is needed. Its implementation uses a one-step gradient descent[61], with $\Delta_{w_i} \approx -lr \cdot \frac{\delta L(D_t)}{\Delta_{w_i}}$. The sensitivity definition becomes:

$$S_i^t = lr \cdot \left(\frac{\delta L(D_t)}{\delta w_i}\right)^2 \tag{5.4}$$

Semi-structured PEFT [61] uses the sensitivity approximation from Equation 5.4 at two granularity levels. Sensitivity is aggregated at the layer level to place LoRA modules of a predefined size in the structured part. It is used at the individual weight level when a LoRA module cannot be associated with a layer [61]. The approximation is assumed to preserve the properties of the exact definition from Equation 5.1. In particular, any change in the layer that does not impact a weight's ability to improve empirical risk (loss) should not alter its measure of importance. This

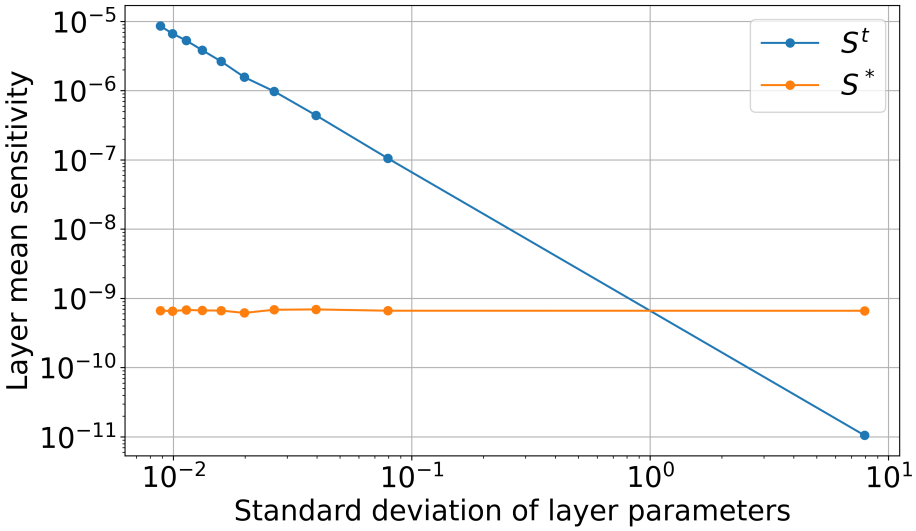


Figure 5.1: Illustration of the lack of robustness of the current definition of sensitivity. We perform a simple adversarial attack on a layer that only changes standard deviation locally without affecting the global fine-tuning process. This attack should not change the sensitivity estimation, but it does. We modify the definition of sensitivity to counter the attacks and improve robustness. See Section 5.3 for details.

assumption is valid for Equation 5.1, as it measures this very capacity, but the approximate sensitivity (Equation 5.4) does not satisfy this property. The approximation leads to inconsistency, as a weight can take very different sensitivity values despite generating the same improvement in the empirical risk.

We highlight this lack of robustness by introducing a simple adversarial attack. We focus this attack on two successive layers, allowing the modification of the activation amplitude. Note that these layers should not be separated by any activation other than a ReLU. Given a pretrained ViT-Base [29], we modify the trainable weights of the first fully connected layer of the MLP in the sixth encoder and those of the linear normalization layer that precedes it. We exemplify the attack with a single layer, but it applies to any similar architecture part. We multiply the activations of the fully connected layer by a positive value α and those of the normalization layer by the inverse value $\frac{1}{\alpha}$. The induced perturbation has a local effect but theoretically does not modify the network as a whole. However, it might impact the optimization of the network hyperparameters and thus change the obtained solution in practice. In particular, a variation of α modifies the standard deviation of the attacked fully connected layer without affecting the

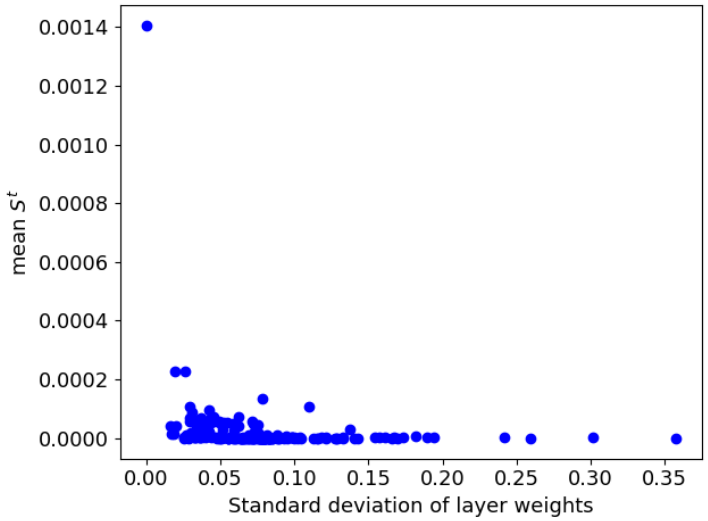


Figure 5.2: Illustration of relation between mean sensitivity value of the layers of a ViT-Base model computed using Equation 5.4 and the standard deviation of layer weights.

sensitivity of the layer weights following Equation 5.1. Changing α induces a modification of the approximate sensitivity computed with Equation 5.4. This variation is highlighted in Figure 5.1 and underlines the lack of robustness of the measure. We propose a modified version of the sensitivity formula (Equation 5.5) to address this issue and maintain a constant sensitivity value during the proposed attack. This adjustment follows directly from the chain rule of derivatives. The approximate sensitivity definition becomes:

$$S_i^* = lr \cdot \sigma_i^2 \cdot \left(\frac{\delta L(D_t)}{\delta w_i} \right)^2 \tag{5.5}$$

Where σ_i is the standard deviation of the layer containing the weight i. The approximation introduced in Equation 5.5 is not affected by the attack, as illustrated in Figure 5.1. This is the expected behavior as modifing α does not change the behavior of the network, only the amplitude of the activatation between the two layers. It enables a more robust estimation of the importance of weight.

We complement the analysis of sensitivity robustness with a second experiment that measures the relation between each layer's average sensitivity and standard deviation. The analysis uses a pretrained ViT-Base [29] without modification. We present the results obtained for all network

layers in Figure 5.2. These results show high sensitivity values for layers with a low standard deviation. This trend is similar to the one obtained after an attack (Figure 5.1).

The above analyses indicate that the sensitivity approximation based on one-step gradient descent lacks robustness. Its use induces a biased estimation of weight importance. The experimental results reported in Section 5.5 further support this finding.

5.4 Toward Adaptive Structured PEFT on a Budget

The authors of SPT [61] use sensitivity to select the individual weights fine-tuned in the unstructured part of their method. We argue that the approximation given in Equation 5.4 is only partially suited for selecting specific weights for fine-tuning. We support this with an experiment assessing the relation between the empirical value Δ_{w_i} measured after fine-tuning and the estimated value of Δ_{w_i} found with the one-step gradient descent. The experiment applies SPT to a pretrained ViT-Base [29] for the CIFAR100 dataset [93]. A good sensitivity estimation would result in a strong linear correlation between the two measures. The weights that have significantly changed during fine-tuning should have a high gradient value before training in Figure 5.3, but this is not systematically the case. Added to the bias analyzed in Section 5.3, this finding further questions the suitability of semi-structured PEFT based on sensitivity estimation.

We observe a weak correlation, with more weights placed in the bottom-left and top-right quarters of Figure 5.3. Consequently, we hypothesize that the layer-level sensitivity estimation can support fully structured PEFT under constraints. Here, the challenge is to respect a predefined parametric budget throughout the fine-tuning process, unlike the AdaLoRA approach that starts with larger modules and reduces them progressively [227]. The module size for a given layer, reflected in the LoRA rank, depends on the share of sensitive weights of that layer and the size of the layer. Assuming that the PEFT budget is n, we select the n weights having the highest sensitivities in the network, as estimated by Equations 5.4 or 5.5. We define the LoRA rank for the j^{th} layer as:

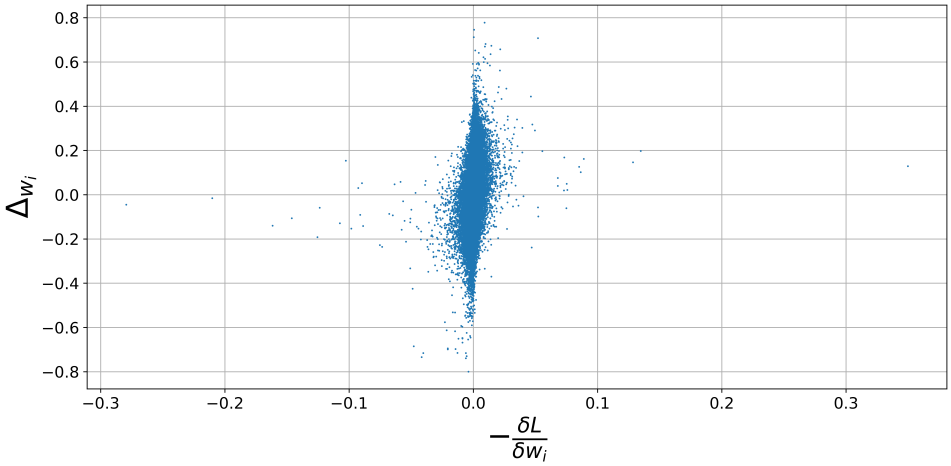


Figure 5.3: The actual weights shift when fine-tuning 100.000 random weights in a network compared with the one-step gradient used to approximate sensitivity in Equation 5.4. Results for a ViT-Base architecture fine-tuned on a few-shot variant of CIFAR100 dataset [93].

$$r_j = \frac{k_j}{(d_j^{in} + d_j^{out})} \tag{5.6}$$

With: k_j - the number of sensitive weights on the j^{th} layer; d_j^{in} and d_j^{out} - the size of the input and output vectors for the j^{th} layer.

Equation 5.6 attaches LoRAs to all layers with $r_j \geq 1$. It correlates LoRA size and sensitivity and optimizes the parametric budget allocation from the start of the PEFT process.

5.5 Experiments

We adapt the methodology proposed in [61] for the evaluation. We report results using the top-1 accuracy (%) globally and per dataset. The main modifications come from a different choice of PEFT budgets to make the evaluation more realistic and from the exclusive use of the pretrained model with the best performance in [61].

5.5.1 Methodology

Datasets. We evaluate the proposed contribution using the VTAB-1k benchmark [224], designed for few-shot transfer learning. It includes 19 visual classification tasks belonging to three groups:

(i) Natural tasks with natural images, (ii) Specialized tasks including images collected with specialized devices, such as medical equipment, and (iii) Structured tasks comprising images sampled from synthetic environments. Datasets contain 800 train and 200 validation images and a variable number of test samples. Unlike [61], we do not use the validation images because their usage reduces the comparability, as explained below.

PEFT parametric budgets. We experiment with budgets representing only a tiny proportion of the total ViT-Base/16 size. We report results with $1 \cdot 10^5$, $2 \cdot 10^5$, $4 \cdot 10^5$ and $6 \cdot 10^5$ trainable parameters. Predefined budgets make the evaluation more realistic and reproducible than the procedure introduced in [61]. The authors use the validation sets to select the best budget per target dataset for each method and then average results. This choice reduces comparability since the parametric budget is method-specific. Equally, scores are artificially boosted by running multiple training sessions per configuration and reporting only the best outcome.

Pre-trained backbone. We use a standard ViT-Base/16 vision transformer pretrained in a supervised manner on ImageNet21k [27]. This model has 12 encoder blocks and approximately 86 million parameters.

Compared methods. We compare our method to the recently introduced SPT method [61] and ablated versions of it. We use the SPT-LORA version because it performs best in the original evaluation from [61] and does not add parameters to the final architecture. In particular, SPT works better than the original LoRA [75], complete fine-tuning and linear probing, and we do not reuse these baselines here. We test three ablated versions of SPT to assess the method components' contributions thoroughly. We compute sensitivity using the existing definition from Equations 5.4 to ensure comparability with the full version of the method. SPT^{uns}_{layer} is a fully unstructured method that assigns parameters to layers based on the aggregated layer sensitivity and distributes tunable parameters randomly within the layer. This version uses sensitivity but ablates the structured part of SPT. SPT^{semi}_{rand} is a semi-structured method that preserves the SPT distribution of the parametric budget between LoRAs and individual weights and the LoRA ranks. Both components are placed randomly in suitable parts of the backbone. $\text{SPT}^{semi}_{layer}$ is semi-structured and uses layer sensitivity to place LoRAs and individual weights. However, it assigns weights randomly within each layer, thus ablating the individual weight assignment from

SPT. These baselines are non-deterministic, and we present results averaged over five runs. We run experiments with a single parametric budget for the ablated versions of SPT.

We name the proposed adaptation of LoRA for a restricted parametric budget LoRA_{ar} and test two versions based on the existing and proposed approximate sensitivity definition from Equations 5.4 and 5.5. These variants are named LoRA_{ar}^{t} and LoRA_{ar}^{*}, following the notations from the corresponding equations.

Robustness to attack. We complete the evaluation with a robustness-oriented experiment. We deploy the attack described in Section 5.3 on the first fully connected MLP layers of each ViT/Base/16 architecture to compare the robustness of sensitivity estimations defined in Equations 5.4 and 5.5. We report results for $\alpha = 0.01$, a value that alters the amplitude of the weights in the attacked layers.

parameters	SPT_{layer}^{uns}	SPT_{rand}^{semi}	$\text{SPT}_{layer}^{semi}$	SPT [61]	LoRA_{ar}^{t}	LoRA_{ar}^{*}
$1 \cdot 10^5$	-	-	-	70.62	71.21	70.78
$2 \cdot 10^5$	69.81	70.28	70.74	70.99	72.06	71.91
$4 \cdot 10^5$	-	-	-	72.40	72.79	72.84
$6 \cdot 10^5$	-	-	-	72.25	72.57	72.92

Table 5.1: Average accuracy of the different PEFT methods tested with four predefined parametric budgets for the 19 VTAB-1k tasks. SPT_{layer}^{uns} and SPT_{rand}^{semi} are non-deterministic, and we average results for five runs corresponding to different random selections of tunable weights. These runs have similar accuracies, with differences between the best and worst runs being lower than 0.05 points.

5.5.2 Results

Aggregated results. We present the accuracy obtained with the compared methods in Table 5.1. The proposed LoRA_{ar}^{t} and LoRA_{ar}^{*} outperform SPT for all parametric budgets, with maximum gains for $2 \cdot 10^5$. The obtained result confirms the hypothesis that the parametric budget is better spent in a structured way than a semi-structured one. Structured PEFT is better suited for implementation under constraints, which is the scenario of interest here. This observation adds a further advantage to the proposed contribution. We provide results with the versions of the proposed method using the two approximations of sensitivity defined in Equa-

	Natural							Specialized				Structured								Category		
datasets	Cifar	Clatech	DTD	Flowers	Pets	SVHN	SUN397	Camelyon	EuroSAT	Resisc45	Diabetic Retino	CLEVR (count)	CLEVR (dist)	DMLab	KITTI	DSprite (loc)	DSprite (ori)	SmallNorb (azi)	SmallNorb (ele)	Natural	Specialized	Structured
SPT	**72.7**	91.2	70.0	**99.2**	91.3	84.2	54.1	83.0	94.6	83.2	68.9	79.6	66.2	47.5	79.6	76.1	45.3	26.0	36.0	80.4	82.4	57.0
LoRA$^t_{ar}$	71.8	91.1	71.4	99.1	**91.4**	**85.9**	54.6	84.5	95.3	82.9	70.6	**82.2**	**66.8**	**49.3**	**81.0**	78.6	46.5	28.0	**38.1**	80.8	83.3	**58.8**
LoRA$^*_{ar}$	72.5	**91.8**	**71.9**	99.1	91.1	84.7	**55.4**	**85.2**	**95.8**	**84.2**	**70.7**	80.2	62.9	48.9	79.8	**78.9**	**48.2**	**28.7**	36.3	**80.9**	**84.0**	58.0

Table 5.2: Results for individual datasets and categories of datasets when fine-tuning $2 \cdot 10^5$ parameters for each method.

tion 5.4 and Equation 5.5. The LoRA$^t_{ar}$ variant that uses the existing sensitivity measure works better for $1 \cdot 10^5$ parameters. The two methods provide similar results for the intermediate budgets. LoRA$^*_{ar}$ optimized with the proposed sensitivity estimation from Equation 5.5 becomes better for $1 \cdot 10^5$. We remind the reader that this sensitivity formulation mainly aims to improve robustness. However, it also provides performance that is similar to the existing definition. Finally, performance saturates for all methods when increasing the parametric budget. However, LoRA$^*_{ar}$ saturates later compared with SPT and LoRA$^t_{ar}$. This behavior enables performance improvement for larger available budgets.

SPT ablation. We hypothesize that the unstructured parameter allocation is suboptimal to motivate the move from semi-structured to structured PEFT. We ablate SPT to gain insight into the roles of its components. SPT and SPT$^{semi}_{layer}$ results are close, indicating only a minor role of the sensitivity-driven individual weight selection within a layer. The gain is in the majority due to assigning more weights to more sensitive layers, as proven by the accuracy drop observed for SPT$^{semi}_{rand}$, which allocates LoRAs and individual weights randomly, compared with the sensitivity-based allocation in SPT. Finally, the fully unsupervised sensitivity-based allocation of individual weights (SPT$^{uns}_{rand}$)gives the lowest performance. These findings support the proposed shift toward fully structured PEFT.

Detailed results. We present results per dataset and category in Table 5.2. They confirm the global tendency presented in Table 5.1 and highlight interesting differences between datasets and categories of datasets. The proposed variants of structured PEFT outperform SPT for 17 datasets out of 19. CIFAR and Flowers are the two exceptions. CIFAR is well-represented in the pretraining dataset, and transfer is easier. Flowers accuracy is nearly equivalent and also

	Natural							Specialized				Structured								
	Cifar	Clatech	DTD	Flowers	Pets	SVHN	SUN397	Camelyon	EuroSAT	Resisc45	Diabetic Retino	CLEVR (count)	CLEVR (dist)	DMLab	KITTI	DSprite (loc)	DSprite (ori)	SmallNorb (azi)	SmallNorb (ele)	Average
S^t	17.5	38.8	44.5	75.5	38.5	52.8	5.0	78.9	92.2	59.5	73.6	77.6	59.1	40.4	69.5	68.3	28.3	5.4	29.8	50.3
S^*	44.1	89.9	66.4	98.7	88.0	79.0	51.5	83.7	93.1	78.0	70.2	78.0	61.3	49.1	80.0	77.5	47.1	33.2	35.8	68.7

Table 5.3: Accuracy of SPT when using Equation 5.4 and 5.5 (our) on a pretrained ViT-Base with standard deviation imbalance created with the protocol described in section 5.3.

saturated. While the methods based on resized LoRA obtain better scores on all Specialized and Structured datasets, the results are closer for the Natural category. These findings indicate that adapting LoRA position and size improves transfer when the domain shifts between the pretraining dataset and downstream tasks.

Results under attack. To test the robustness of the two sensitivity approximations, we attack the first fully connected MLP layers of the ViT-Base [29]. The attack with $\alpha = 0.01$ is sufficiently strong to change the distribution of LoRA modules in the network if the sensitivity estimation lacks robustness. In Table 5.3, we observe a 20.7 points accuracy gap when attacking the existing definition of sensitivity (Equation 5.4). The corresponding performance drop is only 2.2 points for the corrected version from Equation 5.5. Moreover, the difference comes from one dataset, while the attack affects $S*$ only marginally for the other datasets. We explain the performance variation for the modified version by the interplay between the local changes brought by the attack and the optimization of the network hyperparameters. Even if the equivalent network could be obtainable theoretically, convergence cannot be guaranteed.

5.6 Conclusion

We analyze the existing sensitivity approximation and question its suitability for the unstructured part of semi-structured PEFT approaches. We highlight its inconsistency via a simple attack and propose an alternative definition of the metric that mitigates the effect of the attack. We also show that using sensitivity to select individual weight is suboptimal. Based on these findings, we

propose a shift toward fully structured PEFT. The proposed approach adapts the size of LoRAs based on the aggregate sensitivity of each network layer. This contribution is a step toward adapting PEFT methods to constrained environments [199, 204, 78].

5.7 Limitations and future directions

In this section, we discuss the main limitations of our contribution and the future research directions aimed at addressing them. First, while the proposed approach improves PEFT accuracy, it remains dataset-dependent. The long-term objective is to propose methods adapted for fixed fine-tuning for multiple tasks with layer-level sensitivity estimation. Having a fixed transfer architecture for all datasets would facilitate the design of hardware that leverages fixed weights more effectively. In our case, the choice of tunable and fixed weights is different depending on the datasets. This may not be an issue when the task remain constant (as in the case studied in Chapter 4), but it could limit the reusability of the hardware for other tasks, as the optimal choice of tunable and fixed weights would differ across tasks.

Another limitation is using standard LoRA, which targets models with mainly fully connected layers such as ViT [29]. While these architectures are becoming increasingly popular, CNNs are still commonly used in constrained environments [204, 78]. Therefore, it could be useful to explore how this methods performs in the context of the recent adaptation of LoRA for convolution layers [231].

Finally, we observed that under our proposed attack, the new sensitivity metric also loses accuracy in a few datasets. Further work could be done to analyze the fine-tuning process in more detail and improve robustness even more without compromising accuracy.

Chapter 6

Conclusion

This final chapter presents a synthesis of the findings and contributions of this PhD work. It is also an opportunity to take a step back and discuss the main limitations of this work, as well as its potential impact. We then address future work and research directions that would be interesting to explore.

6.1 Summary of key findings

Throughout this thesis, we have obtained results that contribute to answering the general questions raised in the introduction section. motivated by the emergence of hardware designs specifically tailored to leverage fixed deep neural networks, we raised the following questions:

- How can we effectively pre-train a feature extractor for transfer learning?
- What are the limitations of a frozen network?
- How can we transfer a pre-trained network while minimizing the number of parameters to be fine-tuned?

First, in chapter 3, we found that when their size is on the order of millions of parameters, feature extractors can be pre-trained on a reduced subset of data, confirming previous assumptions. The emphasis, therefore, should be placed on the choice of this data. Moreover, we also

demonstrated that increasing diversity by increasing the number of classes does not prevent saturation. Interestingly, the number of classes has little impact on downstream performance, even if the model accuracy on the upstream task drops due to a large number of classes. We also confirmed that freezing the network leads to a significant loss of performance, except in cases where the amount of data on the target task is very small. In the context of this thesis, these initial results show that a network embedded in a fully frozen architecture will be difficult to transfer. Moreover, training it with even more data is not a viable solution, contrary to what is sometimes observed with very large networks. This significantly limits the reusability of the circuit, leading us to conclude that frozen hardware should support more advanced transfer methods than simple linear probing.

Second, we analyzed the effect of time on image classification performance. Unlike in the previous chapter, we do not impose constraints on model size in Chapter 4, and we consider large models trained in a self-supervised manner as well as with multimodal data (e.g., CLIP). Although we empirically demonstrated the sensitivity of pre-trained models to temporal shifts, we also showed that adapting the classification head is sufficient to mitigate the problem. Furthermore, methods from incremental learning can also be used when resources are limited. These potential adaptations suggest that hardware based on permanently frozen models will not become obsolete. Our results also highlight that the shift is not as significant for all classes and that distance measures between high-dimensional embeddings can be used to estimate which ones are most likely to experience a drop in performance and therefore need to be updated. The dataset we provided will allow future research to improve adaptation methods or test the robustness of future architectures.

Finally, as we showed in Chapter 3, more significant adaptation than a simple update of the classification head is necessary when switching tasks. Advances in the field of parameter-efficient transfer learning, therefore, seem essential for hardware architectures relying on frozen networks. In Chapter 5, we analyzed the importance metric used to determine which parameters to fine-tune in recent PEFT methods. This analysis revealed a bias in the metric, which favors the selection of weights in layers with a high standard deviation in weight values. This leads to inconsistencies that can be amplified in the presence of scaling factors, such as those

found in quantization processes. Consequently, we proposed an adjustment to the sensitivity approximation to mitigate this bias. Moreover, current methods leverage importance metric for both structured and unstructured PEFT, but our analyses reveal that both existing and new approximations are too noisy to effectively select individual weights for fine-tuning. This finding questions the usefulness of it for unstructured fine-tuning. Hence, we propose an alternative based on low-rank adaptation, whose ranks are determined before transfer based on the sensitivity of network parameters. This approach is particularly interesting because previous hardware with frozen models [204, 78] does not leverage unstructured freezing. Therefore, our new method and the updated, more robust importance metrics represent a step toward adapting state-of-the-art parameter-efficient fine-tuning for frozen-network-based hardware.

Overall, these results have broadened the understanding of transfer learning and even proposed a new method. While hardware accelerator design was one of the motivations for this work, the vast majority of the results presented here will be more generally useful for engineers and researchers seeking to pre-train networks or transfer them to new tasks.

6.2 Limitations

The findings, methods, and resources proposed in this thesis contribute toward adapting transfer learning to constrained environments. Below, we discuss the main limitations of each contribution.

A first series of limitations arises from the rapid development of deep learning models observed the moment when the work was carried and today. This development impacts the proposed results at different levels. In chapter 3, we focus on supervised learning to analyze the impact of the pre-training dataset size and number of classes, setting aside the self-supervised methods. Although self-supervision does not use image labels, varying the number of classes would allow us to observe the impact of a form of diversity on transfer performance. In Chapter 4, we extend our analysis to recent self-supervised [133] and multimodal models such as the CLIP model [144]. However, large vision models (LVMs) are becoming increasingly prevalent [9]. Therefore, it would be interesting to extend the work on temporal shift robustness to these models. On the other

hand, the opposite phenomenon can also be observed. Chapter 5 contributes to the improvement of recent parameter-efficient fine-tuning methods, which are mainly tested on transformers. For ease of comparison and reproducibility we did the same with our work on bias in the sensitivity metrics and the new PEFT method. But research described in section 2.3 focuses on older, more compact models. It would be interesting to extend the implementations in Chapter 5 to older CNN-type models using the version of LoRA proposed in [231].

Although it represents a new analytical tool in a domain that lacks such resources, the new VCT-107 dataset also has limitations. As with any dataset, it has biases that mainly come from the single data source, Flickr, that induces a selection bias and the limited number of classes. Moreover, the study does not describe the influence of certain phenomena, such as variations in class imbalance, which could be caused by trends or fads, for example. We hope our work will inspire further development of this under-researched topic and help overcome this limitation. A second limitation is that the classes used are generic and we do not study the effect of time for fine-grained visual classes. The restriction of its use for research purposes due to the copyright of the images; producing a commercially usable dataset would, unfortunately, be very expensive.

The newly proposed sensitivity-based parameter-efficient fine-tuning (PEFT) method remains highly dependent on the dataset. The method should be adapted to generalize better across datasets, as for some applications, such as those mentioned in the subsection 2.3, a dataset-dependent method is not practical. However, we believe that future work can mitigate this issue, as we will discuss in the next subsection.

6.3 Future research directions

We discuss below future research directions that stem from our work and might mitigate current limitations.

First, as explained in the limitations, the analyses given in Chapter 3 can be extended to cover self-supervised pretraining, in which the number of classes is still important as it brings more diversity, in this case without the negative aspect of making the pretraining task harder. Moreover, we saw that all the CNNs tested, once downscaled to one million parameters, saturate

at the same point. Therefore, it would be interesting to see if the number of samples in the upstream dataset before saturation of the transfer performance can be estimated using the number of parameters in the model.

Following the Chapter 4 results, an exciting research direction would be to try making the models more robust to future temporal shifts based on already observed temporal shifts. Also, as stated in the limitations, it would be interesting to extend the analysis to VLM, especially to see if the temporal shift can be mitigated using only textual data. This seems possible as current research shows that vision models can recognize objects and characters using only textual descriptions [50].

It would also be useful to improve the VCT dataset by using other data sources to reduce various selection biases in it and add finer-grain classes. The use of distance metrics to detect temporal shifts could also benefit from this as, for example, the number of samples is currently insufficient to test the normality of the distribution of a class or the dataset in the embedding space.

Improving PEFT methods and addressing their dataset-specific nature, which we described as a limitation to their application in hardware discussed in section 2.3, seems to be a logical continuation of the current work. An exciting approach to explore would be a PEFT-aware pre-training. In this scenario, we would aim to force certain weights to concentrate the adaptability capacity, making them the ones to modify regardless of the target task. For instance, by including an importance metric for transfer, such as the one we propose in Chapter 5, into the loss function during pre-training.

At the hardware level, it would be interesting to explore the implementation of parameter-efficient fine-tuning methods and measure their impact on the latency and energy requirements of ASICs, such as [79]. This would help improve reusability and guide future improvements in PEFT methods.

Bibliography

[1] Samira Abnar, Mostafa Dehghani, Behnam Neyshabur, and Hanie Sedghi. Exploring the limits of large scale pre-training. In International Conference on Learning Representations, 2021.

[2] Jon Agar. Constant touch: A global history of the mobile phone. Icon Books Ltd, 2013.

[3] Sabbir Ahmed, Abdullah Al Arafat, Mamshad Nayeem Rizve, Rahim Hossain, Zhishan Guo, and Adnan Siraj Rakin. Ssda: Secure source-free domain adaptation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 19180–19190, 2023.

[4] Ferruh Altun, Sezai Alper Tekin, Seyfettin Gürel, and Mihai Cernat. Design and optimization of electric cars. a review of technological advances. In 2019 8th International Conference on Renewable Energy Research and Applications (ICRERA), pages 645–650. IEEE, 2019.

[5] Sara Beery, Guanhang Wu, Trevor Edwards, Filip Pavetic, Bo Majewski, Shreyasee Mukherjee, Stanley Chan, John Morgan, Vivek Rathod, and Jonathan Huang. The auto arborist dataset: a large-scale benchmark for multiview urban forest monitoring under domain shift. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 21294–21307, 2022.

[6] Eden Belouadah, Adrian Popescu, and Ioannis Kanellos. A comprehensive study of class incremental learning algorithms for visual tasks. Neural Networks, 135:38–54, 2021.

[7] Simone Bianco, Remi Cadene, Luigi Celona, and Paolo Napoletano. Benchmark analysis of representative deep neural network architectures. IEEE access, 6:64270–64277, 2018.

[8] Yuri Bizzoni, Stefania Degaetano-Ortlieb, Peter Fankhauser, and Elke Teich. Linguistic variation and change in 250 years of english scientific writing: A data-driven approach. Frontiers in Artificial Intelligence, 3:73, 2020.

[9] Florian Bordes, Richard Yuanzhe Pang, Anurag Ajay, Alexander C Li, Adrien Bardes, Suzanne Petryk, Oscar Mañas, Zhiqiu Lin, Anas Mahmoud, Bargav Jayaraman, et al. An introduction to vision-language modeling. arXiv preprint arXiv:2405.17247, 2024.

[10] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – mining discriminative components with random forests. In European Conference on Computer Vision, 2014.

[11] Lucas Bourtoule, Varun Chandrasekaran, Christopher A Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. In 2021 IEEE Symposium on Security and Privacy (SP), pages 141–159. IEEE, 2021.

[12] Stevo Bozinovski. Reminder of the first paper on transfer learning in neural networks, 1976. Informatica, 44(3), 2020.

[13] Wieland Brendel and Matthias Bethge. Approximating CNNs with bag-of-local-features models works surprisingly well on imagenet. In International Conference on Learning Representations, 2019.

[14] Kyle Buettner, Sina Malakouti, Xiang Lorraine Li, and Adriana Kovashka. Incorporating geo-diverse knowledge into prompting for increased geographical robustness in object recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 13515–13524, 2024.

[15] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In Proceedings of the IEEE/CVF international conference on computer vision, pages 9650–9660, 2021.

[16] Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Frank Wang, and Jia-Bin Huang. A closer look at few-shot classification. In International Conference on Learning Representations, 2018.

[17] Xinlei Chen, Abhinav Shrivastava, and Abhinav Gupta. Neil: Extracting visual knowledge from web data. In Proceedings of the IEEE international conference on computer vision, pages 1409–1416, 2013.

[18] Yanjiao Chen, Baolin Zheng, Zihan Zhang, Qian Wang, Chao Shen, and Qian Zhang. Deep learning on mobile and embedded devices: State-of-the-art, challenges, and future directions. ACM Computing Surveys (CSUR), 53(4):1–37, 2020.

[19] Yu-Hsin Chen, Tushar Krishna, Joel S Emer, and Vivienne Sze. Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks. IEEE journal of solid-state circuits, 52(1):127–138, 2016.

[20] Gordon Christie, Neil Fendley, James Wilson, and Ryan Mukherjee. Functional map of the world. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 6172–6180, 2018.

[21] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 3606–3613, 2014.

[22] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2016.

[23] Andrea Cossu, Antonio Carta, Lucia Passaro, Vincenzo Lomonaco, Tinne Tuytelaars, and Davide Bacciu. Continual pre-training mitigates forgetting in language and vision. Neural Networks, page 106492, 2024.

[24] Jorge Cuadros and George Bresnick. Eyepacs: an adaptable telemedicine system for diabetic retinopathy screening. Journal of diabetes science and technology, 3(3):509–516, 2009.

[25] Yin Cui, Yang Song, Chen Sun, Andrew Howard, and Serge Belongie. Large scale fine-grained categorization and domain-specific transfer learning. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 4109–4118, 2018.

[26] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. Advances in neural information processing systems, 26, 2013.

[27] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition, pages 248–255. Ieee, 2009.

[28] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In International conference on machine learning, pages 647–655. PMLR, 2014.

[29] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In International Conference on Learning Representations, 2020.

[30] D.C. Dowson and B.V. Landau. The fréchet distance between multivariate normal distributions. Journal of multivariate analysis, 12(3):450–455, 1982.

[31] Simon S Du, Wei Hu, Sham M Kakade, Jason D Lee, and Qi Lei. Few-shot learning via learning the representation, provably. In 9th International Conference on Learning Representations, ICLR 2021, 2021.

[32] Vincent Dumoulin, Neil Houlsby, Utku Evci, Xiaohua Zhai, Ross Goroshin, Sylvain Gelly, and Hugo Larochelle. Comparing transfer and meta learning approaches on a unified few-shot classification benchmark. arXiv preprint arXiv:2104.02638, 2021.

[33] Alaaeldin El-Nouby, Gautier Izacard, Hugo Touvron, Ivan Laptev, Hervé Jegou, and Edouard Grave. Are large-scale datasets necessary for self-supervised pre-training? arXiv preprint arXiv:2112.10740, 2021.

[34] Jorge Will Cukierski Emma Dugas, Jared. Diabetic retinopathy detection, 2015.

[35] Utku Evci, Vincent Dumoulin, Hugo Larochelle, and Michael C Mozer. Head2toe: Utilizing intermediate representations for better transfer learning. In International Conference on Machine Learning, pages 6009–6033. PMLR, 2022.

[36] Simone Fabbrizzi, Symeon Papadopoulos, Eirini Ntoutsi, and Ioannis Kompatsiaris. A survey on bias in visual datasets. Computer Vision and Image Understanding, 223:103552, 2022.

[37] Chun-Mei Feng, Kai Yu, Yong Liu, Salman Khan, and Wangmeng Zuo. Diverse data augmentation with diffusions for effective test-time prompt tuning. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 2704–2714, 2023.

[38] Yutong Feng, Biao Gong, Jianwen Jiang, Yiliang Lv, Yujun Shen, Deli Zhao, and Jingren Zhou. Vim: Vision middleware for unified downstream transferring. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 11696–11707, 2023.

[39] Robert M French. Catastrophic forgetting in connectionist networks. Trends in cognitive sciences, 3(4):128–135, 1999.

[40] Samir Yitzhak Gadre, Gabriel Ilharco, Alex Fang, Jonathan Hayase, Georgios Smyrnis, Thao Nguyen, Ryan Marten, Mitchell Wortsman, Dhruba Ghosh, Jieyu Zhang, et al. Datacomp: In search of the next generation of multimodal datasets. Advances in Neural Information Processing Systems, 36, 2024.

[41] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario March, and Victor Lempitsky. Domain-adversarial training of neural networks. Journal of machine learning research, 17(59):1–35, 2016.

[42] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. International Journal of Robotics Research (IJRR), 2013.

[43] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A. Wichmann, and Wieland Brendel. Imagenet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In International Conference on Learning Representations, 2019.

[44] Mozhdeh Gheini, Xiang Ren, and Jonathan May. Cross-attention is all you need: Adapting pretrained transformers for machine translation. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 1754–1765, 2021.

[45] Golnaz Ghiasi, Barret Zoph, Ekin D Cubuk, Quoc V Le, and Tsung-Yi Lin. Multi-task self-training for learning general representations. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 8856–8865, 2021.

[46] Shiry Ginosar, Kate Rakelly, Sarah Sachs, Brian Yin, and Alexei A Efros. A century of portraits: A visual historical record of american high school yearbooks. In Proceedings of the IEEE International Conference on Computer Vision Workshops, pages 1–7, 2015.

[47] Raphael Gontijo-Lopes, Yann Dauphin, and Ekin Dogus Cubuk. No one representation to rule them all: Overlapping features of training methods. In International Conference on Learning Representations, 2021.

[48] Dipam Goswami, Yuyang Liu, Bartłomiej Twardowski, and Joost van de Weijer. Fecam: Exploiting the heterogeneity of class distributions in exemplar-free continual learning. Advances in Neural Information Processing Systems, 36, 2024.

[49] Priya Goyal, Mathilde Caron, Benjamin Lefaudeux, Min Xu, Pengchao Wang, Vivek Pai, Mannat Singh, Vitaliy Liptchinsky, Ishan Misra, Armand Joulin, et al. Self-supervised pretraining of visual features in the wild. arXiv preprint arXiv:2103.01988, 2021.

[50] Sophia Gu, Christopher Clark, and Aniruddha Kembhavi. I can't believe there's no images! learning visual tasks using only language supervision. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 2672–2683, 2023.

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

[52] Ruite Guo and Vic Patrangenaru. Testing for the equality of two distributions on high dimensional object spaces. arXiv preprint arXiv:1703.07856, 2017.

[53] Yunhui Guo, Yandong Li, Liqiang Wang, and Tajana Rosing. Adafilter: Adaptive filter fine-tuning for deep transfer learning. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 34, pages 4060–4066, 2020.

[54] Yunhui Guo, Honghui Shi, Abhishek Kumar, Kristen Grauman, Tajana Rosing, and Rogerio Feris. Spottune: transfer learning through adaptive fine-tuning. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 4805–4814, 2019.

[55] Elie G Haddad, David Rifkind, and Ms Sarah Deyong. A critical history of contemporary architecture: 1960-2010. Ashgate Publishing, Ltd., 2014.

[56] William L Hamilton, Jure Leskovec, and Dan Jurafsky. Cultural shift or linguistic drift? comparing two computational measures of semantic change. In Proceedings of the conference on empirical methods in natural language processing. Conference on empirical methods in natural language processing, volume 2016, page 2116. NIH Public Access, 2016.

[57] Zeyu Han, Chao Gao, Jinyang Liu, Sai Qian Zhang, et al. Parameter-efficient fine-tuning for large models: A comprehensive survey. arXiv preprint arXiv:2403.14608, 2024.

[58] Soufiane Hayou, Nikhil Ghosh, and Bin Yu. LoRA+: Efficient low rank adaptation of large models. In Forty-first International Conference on Machine Learning, 2024.

[59] Soufiane Hayou, Nikhil Ghosh, and Bin Yu. Lora+: Efficient low rank adaptation of large models. arXiv preprint arXiv:2402.12354, 2024.

[60] Haibo He and Edwardo A. Garcia. Learning from imbalanced data. IEEE Trans. Knowl. Data Eng., 21(9):1263–1284, 2009.

[61] Haoyu He, Jianfei Cai, Jing Zhang, Dacheng Tao, and Bohan Zhuang. Sensitivity-aware visual parameter-efficient fine-tuning. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 11825–11835, 2023.

[62] Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. Towards a unified view of parameter-efficient transfer learning. In International Conference on Learning Representations, 2022.

[63] Kaiming He, Ross Girshick, and Piotr Dollár. Rethinking imagenet pre-training. In Proceedings of the IEEE/CVF international conference on computer vision, pages 4918–4927, 2019.

[64] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In Conference on Computer Vision and Pattern Recognition, CVPR, 2016.

[65] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 8340–8349, 2021.

[66] Dan Hendrycks, Kimin Lee, and Mantas Mazeika. Using pre-training can improve model robustness and uncertainty. In International conference on machine learning, pages 2712–2721. PMLR, 2019.

[67] Harold Herzog. Forty-two thousand and one dalmatians: Fads, social contagion, and dog breed popularity. Society & animals, 14(4):383–397, 2006.

[68] Joel Hestness, Sharan Narang, Newsha Ardalani, Gregory Diamos, Heewoo Jun, Hassan Kianinejad, Md Mostofa Ali Patwary, Yang Yang, and Yanqi Zhou. Deep learning scaling is predictable, empirically. arXiv preprint arXiv:1712.00409, 2017.

[69] Daniel E Ho, Emily Black, Maneesh Agrwawala, and Fei-Fei Li. Domain shift and emerging questions in facial recognition technology. policy brief, Stanford University Human-Centered Artificial Intelligence, https://hai. stanford. edu/sites/default/files/2020-11/HAI_FRT_WhitePaper_PolicyBrief_Nov2020. pdf, 2020.

[70] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. Neural networks, 2(5):359–366, 1989.

[71] Mark Horowitz. 1.1 computing's energy problem (and what we can do about it). In 2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC), pages 10–14, 2014.

[72] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In International conference on machine learning, pages 2790–2799. PMLR, 2019.

[73] Joachim Houyon, Anthony Cioppa, Yasir Ghunaim, Motasem Alfarra, Anaïs Halin, Maxim Henry, Bernard Ghanem, and Marc Van Droogenbroeck. Online distillation with continual learning for cyclic domain shifts. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 2437–2446, 2023.

[74] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861, 2017.

[75] Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. In International Conference on Learning Representations, 2021.

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

[77] Mohammadreza Iman, Khaled Rasheed, and Hamid R Arabnia. A review of deep transfer learning and recent advancements. arXiv preprint arXiv:2201.09679, 2022.

[78] Leo F Isikdogan, Bhavin V Nayak, Chyuan-Tyng Wu, Joao Peralta Moreira, Sushma Rao, and Gilad Michael. Semifreddonets: Partially frozen neural networks for efficient computer vision systems. In Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVII 16, pages 193–208. Springer, 2020.

[79] Miro-Panades Ivan, Lorrain Vicent, Billod Lilian, Kucher Inna, Templier Vincent, Choisnet Sylvain, Ali Nermine, Rossigneux Baptiste, Bichler Olivier, and Alexandre Valentian. A 772μj/frame imagenet feature extractor accelerator on hd images at 30fps. In Proceedings of IEEE Asia Pacific Conference on Circuits and Systems, 2024.

[80] Ashish Jaiswal, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. A survey on contrastive self-supervised learning. Technologies, 9(1):2, 2020.

[81] Paul Janson, Wenxuan Zhang, Rahaf Aljundi, and Mohamed Elhoseiny. A simple baseline that questions the use of pretrained-models in continual learning. arXiv preprint arXiv:2210.04428, 2022.

[82] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In European Conference on Computer Vision, pages 709–727. Springer, 2022.

[83] Junguang Jiang, Yang Shu, Jianmin Wang, and Mingsheng Long. Transferability in deep learning: A survey. arXiv preprint arXiv:2201.05867, 2022.

[84] Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 2901–2910, 2017.

[85] Norman P Jouppi, Cliff Young, Nishant Patil, David Patterson, Gaurav Agrawal, Raminder Bajwa, Sarah Bates, Suresh Bhatia, Nan Boden, Al Borchers, et al. In-datacenter performance analysis of a tensor processing unit. In Proceedings of the 44th annual international symposium on computer architecture, pages 1–12, 2017.

[86] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. arXiv preprint arXiv:2001.08361, 2020.

[87] Kimmo Karkkainen and Jungseock Joo. Fairface: Face attribute dataset for balanced race, gender, and age for bias measurement and mitigation. In Proceedings of the IEEE/CVF winter conference on applications of computer vision, pages 1548–1558, 2021.

[88] Diederik P Kingma. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

[89] Seunghee Koh, Hyounguk Shon, Janghyeon Lee, Hyeong Gwon Hong, and Junmo Kim. Disposable transfer learning for selective source task unlearning. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 11752–11760, 2023.

[90] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. In Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16, pages 491–507. Springer, 2020.

[91] Simon Kornblith, Jonathon Shlens, and Quoc V Le. Do better imagenet models transfer better? In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 2661–2671, 2019.

[92] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In Proceedings of the IEEE international conference on computer vision workshops, pages 554–561, 2013.

[93] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[94] Anders Krogh and John Hertz. A simple weight decay can improve generalization. Advances in neural information processing systems, 4, 1991.

[95] Vivek Kulkarni, Rami Al-Rfou, Bryan Perozzi, and Steven Skiena. Statistically significant detection of linguistic change. In Proceedings of the 24th international conference on world wide web, pages 625–635, 2015.

[96] Ananya Kumar, Aditi Raghunathan, Robbie Matthew Jones, Tengyu Ma, and Percy Liang. Fine-tuning can distort pretrained features and underperform out-of-distribution. In International Conference on Learning Representations, 2022.

[97] A Kutuzov, L Øvrelid, E Velldal, and T Szymanski. Diachronic word embeddings and semantic shifts: A survey. In COLING 2018-27th International Conference on Computational Linguistics, Proceedings, pages 1384–1397, 2018.

[98] Christiaan Lamers, René Vidal, Nabil Belbachir, Niki van Stein, Thomas Bäeck, and Paris Giampouras. Clustering-based domain-incremental learning. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops, pages 3384–3392, October 2023.

[99] Neal Lawton, Anoop Kumar, Govind Thattai, Aram Galstyan, and Greg Ver Steeg. Neural architecture search for parameter-efficient fine-tuning of large pre-trained language models. arXiv preprint arXiv:2305.16597, 2023.

[100] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. Neural computation, 1(4):541–551, 1989.

[101] Yann LeCun, Fu Jie Huang, and Leon Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004., volume 2, pages II–104. IEEE, 2004.

[102] Namhoon Lee, Thalaiyasingam Ajanthan, and Philip Torr. Snip: Single-shot network pruning based on connection sensitivity. In International Conference on Learning Representations, 2018.

[103] Peihao Li, Jie Huang, Shuaishuai Zhang, Chunyang Qi, Chuang Liang, and Yang Peng. A novel backdoor attack adapted to transfer learning. In 2022 IEEE Smartworld, Ubiquitous Intelligence & Computing, Scalable Computing & Communications, Digital Twin, Privacy Computing, Metaverse, Autonomous & Trusted Vehicles (SmartWorld/UIC/ScalCom/DigitalTwin/PriComp/Meta), pages 1730–1735. IEEE, 2022.

[104] Yiming Li, Yong Jiang, Zhifeng Li, and Shu-Tao Xia. Backdoor learning: A survey. IEEE Transactions on Neural Networks and Learning Systems, 35(1):5–22, 2022.

[105] Baohao Liao, Yan Meng, and Christof Monz. Parameter-efficient fine-tuning without introducing new latency. arXiv preprint arXiv:2305.16742, 2023.

[106] Chun-Hsien Lin and Bing-Fei Wu. Mitigating domain mismatch in face recognition using style matching. Neurocomputing, 487:9–21, 2022.

[107] Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin A Raffel. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. Advances in Neural Information Processing Systems, 35:1950–1965, 2022.

[108] Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. Dora: Weight-decomposed low-rank adaptation. arXiv preprint arXiv:2402.09353, 2024.

[109] Vincenzo Lomonaco and Davide Maltoni. Core50: a new dataset and benchmark for continuous object recognition. In Conference on Robot Learning, pages 17–26. PMLR, 2017.

[110] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In International Conference on Learning Representations, 2019.

[111] Hua Ma, Huming Qiu, Yansong Gao, Zhi Zhang, Alsharif Abuadbba, Minhui Xue, Anmin Fu, Jiliang Zhang, Said F Al-Sarawi, and Derek Abbott. Quantization backdoors to deep learning commercial frameworks. IEEE Transactions on Dependable and Secure Computing, 2023.

[112] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In Proceedings of the European conference on computer vision (ECCV), pages 116–131, 2018.

[113] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens Van Der Maaten. Exploring the limits of weakly supervised pretraining. In Proceedings of the European conference on computer vision (ECCV), pages 181–196, 2018.

[114] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. arXiv preprint arXiv:1306.5151, 2013.

[115] Gary Marcus. Deep learning: A critical appraisal. arXiv preprint arXiv:1801.00631, 2018.

[116] Marc Masana, Xialei Liu, Bartłomiej Twardowski, Mikel Menta, Andrew D Bagdanov, and Joost Van De Weijer. Class-incremental learning: survey and performance evaluation on image classification. IEEE Transactions on Pattern Analysis and Machine Intelligence, 45(5):5513–5533, 2022.

[117] Loic Matthey, Irina Higgins, Demis Hassabis, and Alexander Lerchner. dsprites: Disentanglement testing sprites dataset. https://github.com/deepmind/dsprites-dataset/, 2017.

[118] Mark D McDonnell, Dong Gong, Amin Parvaneh, Ehsan Abbasnejad, and Anton van den Hengel. Ranpac: Random projections and pre-trained models for continual learning. Advances in Neural Information Processing Systems, 36, 2024.

[119] Arthur Mensch and Gabriel Peyré. Online sinkhorn: Optimal transport distances from sample streams. Advances in Neural Information Processing Systems, 33:1657–1667, 2020.

[120] Timo Milbich, Karsten Roth, Samarth Sinha, Ludwig Schmidt, Marzyeh Ghassemi, and Bjorn Ommer. Characterizing generalization under out-of-distribution shifts in deep metric learning. Advances in Neural Information Processing Systems, 34:25006–25018, 2021.

[121] George A. Miller. WordNet: a lexical database for english. 38(11):39–41.

[122] M Jehanzeb Mirza, Marc Masana, Horst Possegger, and Horst Bischof. An efficient domain-incremental learning approach to drive in all weather conditions. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 3001–3011, 2022.

[123] Pavlo Molchanov, Arun Mallya, Stephen Tyree, Iuri Frosio, and Jan Kautz. Importance estimation for neural network pruning. pages 11256–11264. IEEE Computer Society.

[124] Ingrid Moons and Patrick De Pelsmacker. Emotions as determinants of electric car usage intention. Journal of marketing management, 28(3-4):195–237, 2012.

[125] Stylianos Moschoglou, Athanasios Papaioannou, Christos Sagonas, Jiankang Deng, Irene Kotsia, and Stefanos Zafeiriou. Agedb: the first manually collected, in-the-wild age database. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshop, volume 2, page 5, 2017.

[126] Moin Nadeem, Anna Bethke, and Siva Reddy. Stereoset: Measuring stereotypical bias in pretrained language models. arXiv preprint arXiv:2004.09456, 2020.

[127] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

[128] Jiquan Ngiam, Daiyi Peng, Vijay Vasudevan, Simon Kornblith, Quoc V Le, and Ruoming Pang. Domain adaptive transfer learning with specialist models. arXiv preprint arXiv:1811.07056, 2018.

[129] Jiquan Ngiam, Daiyi Peng, Vijay Vasudevan, Simon Kornblith, Quoc V Le, and Ruoming Pang. Domain adaptive transfer learning with specialist models. arXiv preprint arXiv:1811.07056, 2018.

[130] Thao Nguyen, Gabriel Ilharco, Mitchell Wortsman, Sewoong Oh, and Ludwig Schmidt. Quality not quantity: On the interaction between dataset design and robustness of clip. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, Advances in Neural Information Processing Systems, volume 35, pages 21455–21469. Curran Associates, Inc., 2022.

[131] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In Indian Conference on Computer Vision, Graphics and Image Processing, Dec 2008.

[132] Alexandra Olteanu, Carlos Castillo, Fernando Diaz, and Emre Kıcıman. Social data: Biases, methodological pitfalls, and ethical boundaries. Frontiers in big data, 2:13, 2019.

[133] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel HAZIZA, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. Transactions on Machine Learning Research, 2023.

[134] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. IEEE Transactions on knowledge and data engineering, 22(10):1345–1359, 2009.

[135] Aristeidis Panos, Yuriko Kobe, Daniel Olmeda Reino, Rahaf Aljundi, and Richard E. Turner. First session adaptation: A strong replay-free baseline for class-incremental learning. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pages 18820–18830, October 2023.

[136] German Ignacio Parisi, Ronald Kemker, Jose L. Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. Neural Networks, 113, 2019.

[137] Chanwoo Park, Sangdoo Yun, and Sanghyuk Chun. A unified analysis of mixed sample data augmentation: A loss function perspective. Advances in Neural Information Processing Systems, 35:35504–35518, 2022.

[138] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In 2012 IEEE conference on computer vision and pattern recognition, pages 3498–3505. IEEE, 2012.

[139] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake VanderPlas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Edouard Duchesnay. Scikit-learn: Machine learning in python. CoRR, abs/1201.0490, 2012.

[140] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In Proceedings of the IEEE/CVF international conference on computer vision, pages 1406–1415, 2019.

[141] Xingchao Peng, Ben Usman, Neela Kaushik, Dequan Wang, Judy Hoffman, and Kate Saenko. Visda: A synthetic-to-real benchmark for visual domain adaptation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pages 2021–2026, 2018.

[142] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In Proceedings of the AAAI conference on artificial intelligence, volume 32, 2018.

[143] Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. Adapterfusion: Non-destructive task composition for transfer learning. In Proceedings

of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, pages 487–503, 2021.

[144] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In International conference on machine learning, pages 8748–8763. PMLR, 2021.

[145] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In International conference on machine learning, pages 8748–8763. PMLR, 2021.

[146] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In International Conference on Machine Learning, pages 8821–8831. PMLR, 2021.

[147] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. Learning multiple visual domains with residual adapters. Advances in neural information processing systems, 30, 2017.

[148] Jiawei Ren, Cunjun Yu, Xiao Ma, Haiyu Zhao, Shuai Yi, et al. Balanced meta-softmax for long-tailed visual recognition. Advances in neural information processing systems, 33:4175–4186, 2020.

[149] Simon Reynolds. Retromania: Pop culture's addiction to its own past. Macmillan, 2011.

[150] Tal Ridnik, Emanuel Ben-Baruch, Asaf Noy, and Lihi Zelnik-Manor. Imagenet-21k pretraining for the masses. In Thirty-fifth Conference on Neural Information Processing Systems, Datasets and Benchmarks Track, 2021.

[151] William A Gaviria Rojas, Sudnya Diamos, Keertan Ranjan Kini, David Kanter, Vijay Janapa Reddi, and Cody Coleman. The dollar street dataset: Images representing

the geographic and socioeconomic diversity of the world. In Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track, 2022.

[152] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 10684–10695, 2022.

[153] Eleanor Rosch. Principles of categorization. In Eleanor Rosch and B. B. Lloyd, editors, Cognition and Categorization, pages 27–48. Erlbaum, Hillsdale, NJ, 1978.

[154] Evgenia Rusak, Steffen Schneider, Peter Vincent Gehler, Oliver Bringmann, Wieland Brendel, and Matthias Bethge. Imagenet-d: A new challenging robustness dataset inspired by domain adaptation. In ICML 2022 Shift Happens Workshop, 2022.

[155] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Fei-Fei Li. Imagenet large scale visual recognition challenge. International Journal of Computer Vision, 115(3):211–252, 2015.

[156] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Fei-Fei Li. Imagenet large scale visual recognition challenge. International Journal of Computer Vision, 115(3):211–252, 2015.

[157] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 4510–4520, 2018.

[158] Mert Bülent Sarıyıldız, Karteek Alahari, Diane Larlus, and Yannis Kalantidis. Fake it till you make it: Learning transferable representations from synthetic imagenet clones. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 8011–8021, 2023.

[159] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. Advances in Neural Information Processing Systems, 35:25278–25294, 2022.

[160] Christoph Schuhmann, Richard Vencu, Romain Beaumont, Robert Kaczmarczyk, Clayton Mullis, Aarush Katta, Theo Coombes, Jenia Jitsev, and Aran Komatsuzaki. Laion-400m: Open dataset of clip-filtered 400 million image-text pairs. arXiv preprint arXiv:2111.02114, 2021.

[161] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition workshops, pages 806–813, 2014.

[162] Tianfeng Shi, Rong Huang, and Emine Sarigöllü. Consumer product use behavior throughout the product lifespan: A literature review and research agenda. Journal of environmental management, 302:114114, 2022.

[163] Peeyush Singhal, Rahee Walambe, Sheela Ramanna, and Ketan Kotecha. Domain adaptation: challenges, methods, datasets, and applications. IEEE access, 11:6973–7020, 2023.

[164] Leslie N Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates. In Artificial intelligence and machine learning for multi-domain operations applications, volume 11006, pages 369–386. SPIE, 2019.

[165] Yisheng Song, Ting Wang, Puyu Cai, Subrota K Mondal, and Jyoti Prakash Sahoo. A comprehensive survey of few-shot learning: Evolution, applications, challenges, and opportunities. ACM Computing Surveys, 55(13s):1–40, 2023.

[166] Krishna Srinivasan, Karthik Raman, Jiecao Chen, Michael Bendersky, and Marc Najork. Wit: Wikipedia-based image text dataset for multimodal multilingual machine learning. In Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval, pages 2443–2449, 2021.

[167] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. The german traffic sign recognition benchmark: a multi-class classification competition. In The 2011 international joint conference on neural networks, pages 1453–1460. IEEE, 2011.

[168] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. Neural networks, 32:323–332, 2012.

[169] Andreas Steiner, Alexander Kolesnikov, Xiaohua Zhai, Ross Wightman, Jakob Uszkoreit, and Lucas Beyer. How to train your vit? data, augmentation, and regularization in vision transformers. arXiv preprint arXiv:2106.10270, 2021.

[170] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In Proceedings of the IEEE international conference on computer vision, pages 843–852, 2017.

[171] Tao Sun, Mattia Segu, Janis Postels, Yuxuan Wang, Luc Van Gool, Bernt Schiele, Federico Tombari, and Fisher Yu. Shift: a synthetic driving dataset for continuous multi-task domain adaptation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 21371–21382, 2022.

[172] Yi-Lin Sung, Varun Nair, and Colin A Raffel. Training neural networks with fixed sparse masks. Advances in Neural Information Processing Systems, 34:24193–24205, 2021.

[173] Indranil Sur, Karan Sikka, Matthew Walmer, Kaushik Koneripalli, Anirban Roy, Xiao Lin, Ajay Divakaran, and Susmit Jha. Tijo: Trigger inversion with joint optimization for defending multimodal backdoored models. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 165–175, 2023.

[174] Harini Suresh and John Guttag. A framework for understanding sources of harm throughout the machine learning life cycle. In Proceedings of the 1st ACM Conference on Equity and Access in Algorithms, Mechanisms, and Optimization, pages 1–9, 2021.

[175] Vivienne Sze, Yu-Hsin Chen, Tien-Ju Yang, and Joel S Emer. Efficient processing of deep neural networks: A tutorial and survey. Proceedings of the IEEE, 105(12):2295–2329, 2017.

[176] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 2818–2826, 2016.

[177] Gábor J Székely, Maria L Rizzo, et al. Testing for equal distributions in high dimension. InterStat, 5(16.10):1249–1272, 2004.

[178] Youssef Tamaazousti, Hervé Le Borgne, and Céline Hudelot. Mucale-net: Multi categorical-level networks to generate more discriminating features. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 6711–6720, 2017.

[179] Chuanqi Tan, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu. A survey on deep transfer learning. In Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part III 27, pages 270–279. Springer, 2018.

[180] Jin Tan, Taiping Zhang, Linchang Zhao, Xiaoliu Luo, and Yuan Yan Tang. A robust image representation method against illumination and occlusion variations. Image and Vision Computing, 112:104212, 2021.

[181] Ningyuan Tang, Minghao Fu, Ke Zhu, and Jianxin Wu. Low-rank attention side-tuning for parameter-efficient fine-tuning. arXiv preprint arXiv:2402.04009, 2024.

[182] Rohan Taori, Achal Dave, Vaishaal Shankar, Nicholas Carlini, Benjamin Recht, and Ludwig Schmidt. When robustness doesn't promote robustness: Synthetic vs. natural distribution shifts on imagenet, 2020.

[183] David Thiel. Identifying and eliminating csam in generative ml training data and models. Technical report, Technical Report. Stanford University, Palo Alto, CA. https://purl.stanford . . . , 2023.

[184] Bart Thomee, David A Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. Yfcc100m: The new data in multimedia research. Communications of the ACM, 59(2):64–73, 2016.

[185] Neil C Thompson, Kristjan Greenewald, Keeheon Lee, and Gabriel F Manso. The computational limits of deep learning. arXiv preprint arXiv:2007.05558, 10, 2020.

[186] Mamatha Thota, Dewei Yi, and Georgios Leontidis. Lleda—lifelong self-supervised domain adaptation. Knowledge-Based Systems, 279:110959, 2023.

[187] Sebastian Thrun. A lifelong learning perspective for mobile robot control. In Intelligent robots and systems, pages 201–214. Elsevier, 1995.

[188] Songsong Tian, Weijun Li, Xin Ning, Hang Ran, Hong Qin, and Prayag Tiwari. Continuous transfer of neural network representational similarity for incremental learning. Neurocomputing, 545:126300, 2023.

[189] Adrián Tormos, Dario Garcia-Gasulla, Victor Gimenez-Abalos, and Sergio Alvarez-Napagao. When & how to transfer with transfer learning. In Has it Trained Yet? NeurIPS 2022 Workshop, 2022.

[190] Antonio Torralba and Alexei A. Efros. Unbiased look at dataset bias. In CVPR 2011, pages 1521–1528, 2011.

[191] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In International conference on machine learning, pages 10347–10357. PMLR, 2021.

[192] Eleni Triantafillou, Tyler Zhu, Vincent Dumoulin, Pascal Lamblin, Utku Evci, Kelvin Xu, Ross Goroshin, Carles Gelada, Kevin Swersky, Pierre-Antoine Manzagol, and Hugo Larochelle. Meta-dataset: A dataset of datasets for learning to learn from few examples. In International Conference on Learning Representations, 2020.

[193] Cheng-Hao Tu, Zheda Mai, and Wei-Lun Chao. Visual query tuning: Towards effective usage of intermediate representations for parameter and memory efficient transfer learning.

In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 7725–7735, 2023.

[194] Gido M van de Ven, Tinne Tuytelaars, and Andreas S Tolias. Three types of incremental learning. Nature Machine Intelligence, 4(12):1185–1197, 2022.

[195] Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alex Shepard, Hartwig Adam, Pietro Perona, and Serge Belongie. The inaturalist species classification and detection dataset. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 8769–8778, 2018.

[196] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. Advances in neural information processing systems, 30, 2017.

[197] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), July 2017.

[198] Rudi Volti. Cars and culture: The life story of a technology. JHU Press, 2006.

[199] Danilo Vucetic, Mohammadreza Tayaranian, Maryam Ziaeefard, James J Clark, Brett H Meyer, and Warren J Gross. Efficient fine-tuning of bert models on the edge. In 2022 IEEE International Symposium on Circuits and Systems (ISCAS), pages 1838–1842. IEEE, 2022.

[200] Angelina Wang and Olga Russakovsky. Overwriting pretrained bias with finetuning data. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 3957–3968, 2023.

[201] Jie Wang, Rui Gao, and Yao Xie. Sinkhorn distributionally robust optimization. arXiv preprint arXiv:2109.11926, 2021.

[202] Shuo Wang, Surya Nepal, Carsten Rudolph, Marthie Grobler, Shangyu Chen, and Tianle Chen. Backdoor attacks against transfer learning with pre-trained deep learning models. IEEE Transactions on Services Computing, 15(3):1526–1539, 2020.

[203] Yeming Wen, Dustin Tran, and Jimmy Ba. Batchensemble: an alternative approach to efficient ensemble and lifelong learning. arXiv preprint arXiv:2002.06715, 2020.

[204] Paul N Whatmough, Chuteng Zhou, Patrick Hansen, Shreyas Kolala Venkataramanaiah, Jae-sun Seo, and Matthew Mattina. Fixynn: Efficient hardware for mobile computer vision via transfer learning. arXiv preprint arXiv:1902.11128, 2019.

[205] Stefano Woerner and Christian F Baumgartner. Navigating data scarcity using foundation models: A benchmark of few-shot and zero-shot learning approaches in medical imaging. In International Workshop on Foundation Models for General Medical AI, pages 30–39. Springer, 2024.

[206] Baoyuan Wu, Hongrui Chen, Mingda Zhang, Zihao Zhu, Shaokui Wei, Danni Yuan, and Chao Shen. Backdoorbench: A comprehensive benchmark of backdoor learning. Advances in Neural Information Processing Systems, 35:10546–10559, 2022.

[207] Junrui Xiao, Zhikai Li, Lianwei Yang, and Qingyi Gu. Patch-wise mixed-precision quantization of vision transformer. In 2023 International Joint Conference on Neural Networks (IJCNN), pages 1–7. IEEE, 2023.

[208] Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang. Learning from massive noisy labeled data for image classification. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 2691–2699, 2015.

[209] Zheng Xie, Zhiquan Wen, Jing Liu, Zhiqiang Liu, Xixian Wu, and Mingkui Tan. Deep transferring quantization. In European Conference on Computer Vision, pages 625–642. Springer, 2020.

[210] Yi Xin, Siqi Luo, Haodi Zhou, Junlong Du, Xiaohong Liu, Yue Fan, Qing Li, and Yuntao Du. Parameter-efficient fine-tuning for pre-trained vision models: A survey. arXiv preprint arXiv:2402.02242, 2024.

[211] Xiaowei Xu, Yukun Ding, Sharon Xiaobo Hu, Michael Niemier, Jason Cong, Yu Hu, and Yiyu Shi. Scaling for edge inference of deep neural networks. Nature Electronics, 1(4):216–222, 2018.

[212] Adam X. Yang, Maxime Robeyns, Xi Wang, and Laurence Aitchison. Bayesian low-rank adaptation for large language models. In The Twelfth International Conference on Learning Representations, 2024.

[213] Chenbin Yang and Huiyi Liu. Channel pruning based on convolutional neural network sensitivity. Neurocomputing, 507:97–106, 2022.

[214] Lu Yang, He Jiang, Qing Song, and Jun Guo. A survey on long-tailed visual recognition. International Journal of Computer Vision, 130(7):1837–1872, 2022.

[215] Shuo Yang, Lu Liu, and Min Xu. Free lunch for few-shot learning: Distribution calibration. In International Conference on Learning Representations (ICLR), 2021.

[216] Huaxiu Yao, Caroline Choi, Bochuan Cao, Yoonho Lee, Pang Wei W Koh, and Chelsea Finn. Wild-time: A benchmark of in-the-wild distribution shift over time. Advances in Neural Information Processing Systems, 35:10309–10324, 2022.

[217] Yuanshun Yao, Huiying Li, Haitao Zheng, and Ben Y Zhao. Latent backdoor attacks on deep neural networks. In Proceedings of the 2019 ACM SIGSAC conference on computer and communications security, pages 2041–2055, 2019.

[218] Burak Yildiz, Seyran Khademi, Ronald Maria Siebes, and Jan Van Gemert. Amstertime: A visual place recognition benchmark dataset for severe domain shift. In 2022 26th International Conference on Pattern Recognition (ICPR), pages 2749–2755. IEEE, 2022.

[219] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? Advances in neural information processing systems, 27, 2014.

[220] Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. Journal of the Royal Statistical Society Series B: Statistical Methodology, 68(1):49–67, 2006.

[221] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. arXiv preprint arXiv:1605.07146, 2016.

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

[223] Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 12104–12113, 2022.

[224] Xiaohua Zhai, Joan Puigcerver, Alexander Kolesnikov, Pierre Ruyssen, Carlos Riquelme, Mario Lucic, Josip Djolonga, Andre Susano Pinto, Maxim Neumann, Alexey Dosovitskiy, et al. A large-scale study of representation learning with the visual task adaptation benchmark. arXiv preprint arXiv:1910.04867, 2019.

[225] Jeffrey O Zhang, Alexander Sax, Amir Zamir, Leonidas Guibas, and Jitendra Malik. Side-tuning: a baseline for network adaptation via additive side networks. In Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16, pages 698–714. Springer, 2020.

[226] Longteng Zhang, Lin Zhang, Shaohuai Shi, Xiaowen Chu, and Bo Li. Lora-fa: Memory-efficient low-rank adaptation for large language models fine-tuning. arXiv preprint arXiv:2308.03303, 2023.

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

[228] Qingru Zhang, Simiao Zuo, Chen Liang, Alexander Bukharin, Pengcheng He, Weizhu Chen, and Tuo Zhao. Platon: Pruning large transformer models with upper confidence bound of

weight importance. In International conference on machine learning, pages 26809–26823. PMLR, 2022.

[229] Yuanhan Zhang, Kaiyang Zhou, and Ziwei Liu. Neural prompt search. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2024.

[230] Zhiyuan Zhao, Qingjie Liu, and Yunhong Wang. Exploring effective knowledge transfer for few-shot object detection. In Proceedings of the 30th ACM International Conference on Multimedia, pages 6831–6839, 2022.

[231] Zihan Zhong, Zhiqiang Tang, Tong He, Haoyang Fang, and Chun Yuan. Convolution meets lora: Parameter efficient finetuning for segment anything model. arXiv preprint arXiv:2401.17868, 2024.

[232] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. Proceedings of the IEEE, 109(1):43–76, 2020.

Appendix A

Résumé en français

Cette thèse explore des techniques d'apprentissage par transfert efficientes pour des environnements contraints, où la réduction du nombre de paramètres ajustables devient centrale.

Dans un premier temps, nous examinons l'impact de la quantité de données de pré-entraînement ainsi que du nombre de classes associées. Lors du transfert, nous étudions également la méthode de transfert employée ainsi que le nombre d'exemples par classe dans la tâche cible. Nos résultats montrent que, durant le pré-entraînement, les performances saturent après une certaine quantité de données, et qu'une fois ce seuil atteint, le nombre de classes a peu d'influence, même s'il est élevé et que la précision du modèle sur la tâche de pré-entraînement diminue. Nous observons également que l'ajout de nouvelles classes, bien qu'il augmente la diversité des données, n'améliore pas significativement les performances en aval après saturation. Nous avons également observé que, bien que l'apprentissage d'une couche de classification linéaire soit généralement moins performant qu'un fine-tuning complet, il peut être plus efficace lorsque la quantité de données est faible et que la tâche cible est similaire à la tâche source.

Nous étudions ensuite l'impact du temps sur les extracteurs de caractéristiques basés sur des modèles neuronaux profonds. Pour cela, nous introduisons un nouveau jeu de données qui met en évidence le manque de robustesse des modèles préentraînés face aux décalages temporels. Ce dernier rassemble des images tirées de Flickr représentant 107 classes et réparties en 5 périodes, basées sur leurs dates de mise en ligne. Nous évaluons la capacité de plusieurs stratégies de

préentraînement et méthodes d'adaptation à mitiger ce problème. Cela inclut des préentraînements multimodaux, supervisés ou non, ainsi que des méthodes d'apprentissage incrémental. Nos résultats soulignent l'importance de mettre régulièrement à jour les modèles pour s'adapter aux changements dans le temps des distributions des classes visuelles, même lorsque le modèle est fortement préentraîné. De plus, la vitesse de ce changement varie selon les classes, et nous testons les capacités de plusieurs métriques de distances entre distributions à prédire les classes subissant une perte de performance.

Enfin, nous proposons une nouvelle méthode d'apprentissage par transfert optimisée en termes de paramètres ajustables. Cette méthode se base sur notre découverte d'un biais dans l'approximation de la métrique de sensibilité, permettant de déterminer les paramètres importants à ajuster. Ce biais, favorisant les couches ayant une grande variance dans les valeurs des poids, a été éliminé grâce à une nouvelle approximation de la sensibilité. Nous réalisons une attaque adverse pour mettre en évidence ce biais et tester la robustesse de l'ancienne et de la nouvelle formulation de la sensibilité face à des variations de l'écart-type entre les différentes couches. Les analyses effectuées montrent également que les approximations faites pour déterminer les poids à fine-tuner à partir de la sensibilité initiale sont fortement bruitées. Cela remet en cause l'utilisation de cette métrique pour guider un fine-tuning non structuré. La méthode fournit donc également une alternative au fine-tuning non structuré grâce à des adaptations dont le rang est déterminé avant le transfert, en fonction de la sensibilité des paramètres du réseau. Les gains apportés par cette méthode ont été mis en évidence sur un ensemble de 19 datasets.

Ainsi, nos travaux contribuent à l'amélioration de l'apprentissage par transfert dans des environnements contraints en optimisant l'utilisation des données de pré-entraînement, en renforçant la robustesse des modèles face aux décalages temporels, et en proposant une nouvelle méthode de fine-tuning efficiente en termes de paramètres. Ces contributions ouvrent de nouvelles perspectives pour répondre aux défis d'efficacité et de généralisation dans des contextes variés où les paramètres des modèles peuvent être figés.