POLITECNICO DI TORINO

Master's Degree in Data Science and Engineering

Politecnico
di Torino

Master's Degree Thesis

Data Selection via Semantic Similarity for Resource-Efficient Transfer Learning of Image Classifiers

Supervisors

Prof. Andrea CALIMERA

Ph.D Valentino PELUSO

Candidate

Pedro Antonio HERNÁNDEZ ZAMARRÓN

July 2022

Summary

Over the past 20 years, data has increased on a large scale [1]. This data takes many forms, such as text, audio, video and images.
Companies and governments are interested in extracting the value present in data, and they have announced their intentions to accelerate research and applications in this field [2].
However, said data is often unstructured and does not convey meaningful information. Hence, many data analysis methods have arisen. For instance, methods like deep learning can glean meaningful information from images by labelling them. This information can help companies and governments make educated decisions by providing helpful insights.
In particular, the process of managing image data and labelling it is called image classification, and in some cases, it can surpass human-level accuracy [3]. The computing system used to classify images is called a neural network, and the process through which the neural network learns to label the pictures is called *training*.
Training a neural network is a process that generally requires a copious quantity of data to perform satisfactorily. Unfortunately, this requirement leads to training times for neural networks that are pretty long, and the availability of such an amount of data is often an issue.
Transfer learning addresses some of these obstacles. Transfer learning is a technique within the machine learning field whose objective is improving the accuracy of a given task via pretraining while reducing the amount of data needed. This pretraining step precedes the actual task and entails training a neural network on a similar assignment on an unrelated domain. For example, in the context of image classification, the pretraining step could consist of training a model on a generic image dataset such as ImageNet, while the training proper is a task where the network must recognise several different races of cats and dogs. The neural network can perform better on the pets dataset because of the visual features learnt in the previous phase. However, the pretraining step still requires a dataset of a considerable dimension; therefore, the computational cost of creating a neural network remains a problem.
This thesis introduces a novel way of optimising transfer learning, particularly

image classification tasks training time. The proposed approach selects a subset of images from the source dataset via a semantic approach, thus reducing the extent of data needed.
Specifically, a lexical database chooses the most relevant source dataset categories by determining the semantic distance between classes and choosing the target categories closest to the source classes.

Acknowledgements

I want to extend my gratitude to Prof. Andrea Calimera and ph.D Valentino Peluso, whose feedback was crucial to developing this thesis.
Moreover, this would not have been possible without the computational resources lent to me by the Polytechnic of Turin.
I'm also thankful to my colleagues and friends. Their help and suggestions were invaluable during the entirety of my academic career.
Lastly, I want to mention my family. Their support gave me the push and encouragement necessary to reach this goal.

"If each of my words were a drop of water, you would see through them and glimpse what I feel: gratitude, acknowledgement."
Octavio Paz,

Table of Contents

List of Tables

List of Figures

Acronyms

NN
Neural Network

CNN
Convolutional Neural Network

ML
Machine Learning

TL
Transfer Learning

FC
Fully Connected

Chapter 1

Introduction

The generation of data such as text, audio and video has rapidly increased over the past 20 years [1]. This data contains information that can help make educated decisions by providing helpful insights. Hence, there is much interest in extracting the value present in data. For example, companies and governments have announced their intentions to accelerate research and applications in this field [2].
However, said data is often unstructured and does not convey meaningful information in its raw form. Therefore, data analysis methods are employed to extract meaningful information from datasets. This thesis deals with deep learning methods that look at images and assigns a label to them. This process of managing image data and labelling it is called *image classification*, and in some cases, it can surpass human-level accuracy [3].
The computing system used to classify images is called a *neural network*, and the process through which the neural network learns to label the pictures is called *training*. *Image classification* is a machine learning task that relies on convolutional neural networks (CNN). Unlike the manual selection of image features, CNNs provide an automated and efficient way of identifying image parts most suited for visual categorisation. This is indeed the main property of convolutional neural networks. Throughout their training, these models learn not only which images belong to which class but also they grasp which parts of the image are most predictive for categorisation purposes. These image parts used for categorisation are called features. The initial convolutional layers produce lower-level features and tend to identify more straightforward visual concepts such as edges. Higher-level features are an output of later convolutional layers, and the information they convey is much more complex, such as patterns or objects.
One of the main issues with CNNs as of the current state of the art is that they need copious amounts of data to achieve satisfactory accuracy. This leads to training times that take up ample resources such as time, energy and data.
A commonly utilised technique in data science is transfer learning, and we use

this to solve the problems mentioned earlier. *Transfer learning* is a method which uses a neural network that has been previously trained on a large dataset (source dataset) and uses this past knowledge to aid in a new classification endeavour with new data(target dataset). In this way, the amount of data needed for a specific classification task is reduced, and one only needs to find a suitable dataset for pretraining.
ImageNet has been one of the most frequently utilised source datasets, and this thesis explores why this is so. Answering some of the assumptions made about it gives insight into how one might address creating suitable datasets for source training. [4] provides interesting results; in one of the experiments, they find out that Imagenet has performance issues with specific target categories and not with others.
Optimising a neural network can be done at different phases of creating a model but mainly during inference and training. Indeed some techniques have been developed to reduce resource consumption during inference. Inference refers to the stage where an ML model makes predictions; for instance, inference means taking an image and predicting its correct class in the context of image recognition. The low power image recognition challenge [5] is a testament to all the work being carried out in this field. Moreover, advances in recent years have primarily focused on this; as a result, there is less literature dealing with the optimisation of visual recognition during training.
Nevertheless, some techniques have been developed over the years to reduce the volume of images needed to train a source dataset. Some of these methods propose a selective approach when choosing the entries for the pretraining phase. In particular, some rely on automatically learning the importance of classes via optimisation problems like [6], where a weight vector is learned, and each vector entry assigns importance to certain parts of the source dataset. In other works, like [7], features from each target image are extracted via a neural network, and via these features, the authors seek to find its nearest neighbours in the source dataset.
Similarly, our efforts are focused on selecting a subset of the source dataset. However, the selection is made via a semantic analysis of the categories. In addition to our methods, we also explore how other works in literature have dealt with selective training of the source dataset.
Our approach might seem more superficial than the techniques mentioned above used by other related works. Moreover, while automating the data selection process has its merits, it is also computationally expensive. Additionally, one needs to already possess the data for these other techniques, while this is not needed in our paper. And having a semantic approach results in having a list of classes which are most pertinent to the target task; this allows any data-gathering efforts to be more focused on the most related and impactful categories.
On the other hand, the target dataset will be dealing with a particular domain.

This type of visual classification is called FGVC (fine-grained visual categorisation), a specialised computer vision domain. FGVC has its own set of issues given the specificity of the categories (e.g. classifying different bird species). We will expound on how the issues of FGVC have been addressed so far by the state-of-the-art, and we will introduce the novel methodology mentioned above that seeks to mitigate the obstacles in fine-grained computer vision.
This methodology exploits two main insights proven by past research. The first is that transfer learning dramatically benefits from having a source dataset that is visually similar to the target [8]. The second comes from paper [9]; the researchers in this latter paper concluded that there exists a correlation between visual and semantic similarity.
With the insights provided by these two papers, we hypothesise that performing a semantic selection via a lexical database on the categories of the source dataset can improve the performance of the target classification task. More specifically, the lexical database provides a measure of semantic distance between two classes. The main idea is to define a measure of distance between a source category and the target domain via the distance function available in the semantic database. After this, the source classes most similar to the target domain are selected for pretraining.
We will find the most semantically pertinent classes in the source dataset, train the source dataset with only those and afterwards transfer the learned weights into the new task. Finally, some further experiments are performed in order to ascertain the validity of our findings.

Chapter 2

Background

The generation of data such as text, audio and video has rapidly increased over the past 20 years [1] through technological advancements such as IoT (internet of things) and social networks. This data contains information that can help make educated decisions by providing helpful insights. For example, social media data can provide ads tailored for users, leading to more effective advertising. On the other hand, data generated by sensors in IoT devices can tell manufacturers how to enhance the performance of their products.
Hence, there is much interest in extracting the value present in data. For example, companies and governments have announced their intentions to accelerate research and applications in this field [2].
However, data is unstructured and does not convey meaningful information without prior examination. Therefore, data analysis methods are employed to extract meaningful information from datasets. The most modern techniques rely on neural networks and machine learning. *Neural networks* are the computing system used to examine and learn from data. Analysis and data can have different scopes. This thesis deals with deep learning methods that look at images and assigns a label to them. This process of managing image data and labelling it is called *image classification*, and in some cases, it can surpass human-level accuracy [3].
Image classification forms part of a greater field of machine learning called Computer vision. Computer vision is a specific branch of Machine Learning that handles images and extracts meaningful information from them. In particular, *image classification* tasks a neural network with recognising the subject of an image and providing a class prediction for it.
Amongst the image classification practices, there is an approach called supervised learning. Here the neural network uses training data consisting of an image and a label. With the help of labels, the ML model can develop an inferred function [10] to predict output values for unseen images.
Indeed, this latter approach and the existence of high-quality labelled datasets

have been one of the most prominent factors in increasing the performance and accuracy of deep learning models [11]. In this context, a model that has seen significant large-scale adoption is the convolutional neural network or *CNN* for short. A CNN is composed of three main types of layers: (i) the convolutional layers, (ii) the pooling layers, (iii) and the fully connected layers. The convolutional layers convolve the image with several different kernels to generate these feature maps. Such feature maps encode essential information that the fully connected layer will later use to classify the image. The earlier convolutional layers produce feature maps that encode low-level features such as edges, whereas the later layers encode high-level information. This is illustrated in the figure 2.1. The pooling layer reduces the width and height dimension on the feature maps produced by the convolution layers (leaving the depth untouched). This provides the benefit of reducing the computational overhead for the later layers and avoiding overfitting. Lastly comes the fully connected layer that converts the 2d feature maps into a 1d feature vector that can be used for classification.
Machine learning is a costly endeavour and expensive in terms of the energy it consumes and the data it needs. Much effort has been put into making neural networks more efficient, and optimisation is done at mainly two stages of development. Optimisation at an inference level and during the training phase.
Inference optimisation makes networks less energetically expensive while categorising images (or whatever the task may be). The low power image recognition challenge [5] has resulted in many techniques dealing with power usage. These mainly consist of specialised NN architectures such as MobileNet and quantisation, which approximates neural networks using floating-point numbers into lower bit numbers.
On the other hand, this thesis focuses on making the training process more efficient, specifically by dealing with the data needed to train the model. Often the amount of data needed for training a CNN is unfeasible to obtain for a certain specific task. Gathering data, especially for supervised learning, is often monetarily expensive and time-consuming because data must be labelled, requiring people to define the class for each image manually. **Transfer learning** is an ML technique that addresses these problems by reducing the need to collect data.

2.1 Transfer Learning

According to [13] "Transfer learning aims to improve the performance of target learning on target domains by transferring the knowledge contained in different but related source domains"
In other words, transfer learning trains a learner on two different tasks. The training for the first task provides valuable information for the second training.

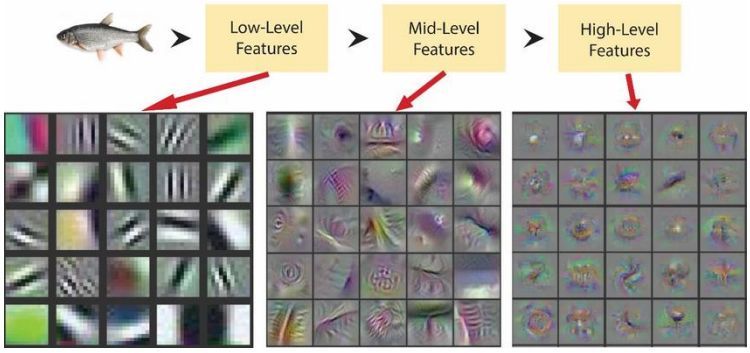

Figure 2.1: Taken from [12]. This figure illustrates how convolutional neural networks extract information from an image. The complexity of the features increases as we analyse features from later and later convolutional layers. The low-level features usually represent basic concepts such as lines, corners and edges. Mid-level features are more elaborate and frequently capture information such as object parts. And finally, high-level features tend to capture the whole object

For example, the learner might be trained at first with a generic dataset that contains multiple classes of fauna and then subsequently is trained in multiple visual categories of dog races with a more domain-specific dataset.
In this example, one would call the first and second datasets the source and target datasets, respectively.
Usually, there is much more data available for the source dataset, and the source and target classes often share low-level representations. As explained earlier, a CNN learns features through its convolutional layers. By pretraining the model on a dataset, the convolutional layers learn filters which are used as the starting point for the training on the second dataset. Starting training in such a way allows the model to converge much faster and, as a result, have much better accuracy.
In such a way, transfer learning has proven to be very useful in mitigating the need for large quantities of data. It is common practice to use a learner that has been previously trained on a large source dataset to perform visual classification on a much smaller set of images. A common choice for a source is **ImageNet**. Therefore, when presented with a certain classification problem with a scarcity of data, one can often overcome the problem presented by the limited supply of images via pretraining.
The ImageNet [14] as mentioned earlier, is a dataset that holds great importance in the realm of transfer learning and machine learning in general. It was created in 2009, and it contains more than 14 million images and 21.841 classes. ImageNet's classes are organised in a way that follows the semantic hierarchy of a lexical database for the English language called **Wordnet** [15]. Wordnet is a lexical

database where verbs, nouns and adjectives are grouped into synsets, and in such arrangement, a single entry in the database is called a synset. For instance, the words "car" and "automobile" belong to the same synset. Synsets relate to each other via a semantic hierarchy which puts the most generic words such as "vertebrate" on top and more specific words like "Burmese cat" closer to the bottom like shown in the figure 2.2.
Furthermore, semantically similar words are close to each other within this hierarchy. Now that the structure of ImagenNet is clear, we proceed to describe how the construction of this dataset came about. There were two main steps:
Firstly, candidate images were collected by querying search engines. In order to further populate the dataset with more images, parent synsets were appended to the original queries. For instance "whippet" $\rightarrow$ "whippet dog". More results were obtained by translating the query into different languages.
Lastly, the candidate images obtained during the last step were manually selected via Amazon mechanical Turks. This is a service in which anyone can pay to outsource several types of tasks to humans, such as individually labelling images. In this case, several Turks were employed to label a specific image, and an image was considered correctly labelled when the majority of them agreed on the class. For more generic classes such as cat, fewer people were asked to classify the images, whereas for more difficult classes such as "Burmese cat", more users were hired.
This process elucidates how costly creating a large-scale dataset is, and from 2009 to today, the Imagenet dataset has grown and evolved, becoming ever more extensive and complex. As of the day of writing this thesis, there are two main versions of the ImageNet dataset: ImageNet 21k consisting of more than 14 million images and ImageNet 1k, which is much easier to manage with a little bit over 1.2 million images. This latter version is the most commonly used since training the 21k version is impractical for most people. An annual competition called Imagenet Large Scale Visual Recognition Challenge uses ImageNet1k (ILSVRC) [16]. This challenge contains three possible different tasks: Image Classification task, Single Object Classification task, and Object Detection Task.

Table 2.1: [16] shows the different versions of Imagenet ILSVRC over the years.

Year	Train images (per class)	Val images (per class)	Test images
ILSVRC2010	1,261,406 (668–3047)	50,000 (50)	150,000 (150)
ILSVRC2011	1,229,413 (384–1300)	50,000 (50)	150,000 (150)
ILSVRC2012-2014	1,281,167 (732–1300)	50,000 (50)	150,000 (150)

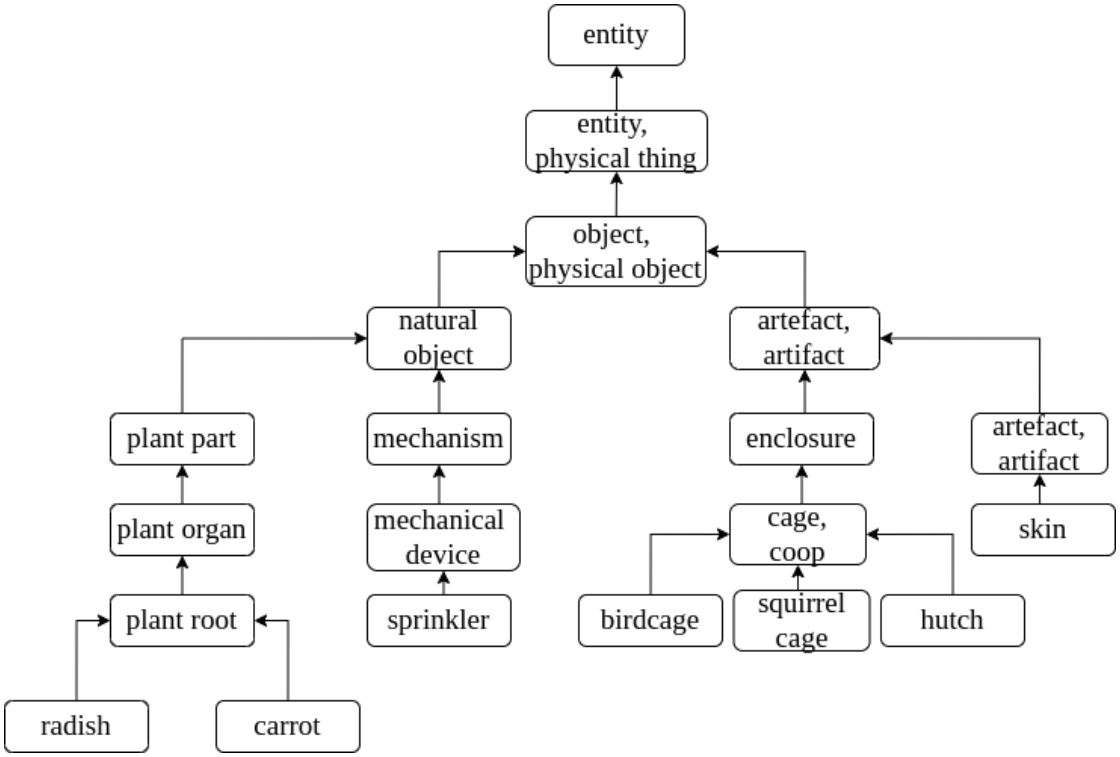

Figure 2.2: Example of Imagenet's semantic hierarchy. Furthermore this illustrates the hypernym/hyponym taxonomy. Simply put a hypernym is a more generic word than another given word. In this figure, "plant root" is a hypernym to "radish" and "carrot". Conversely a hyponym is more specific. "Radish" and "carrot" are hyponyms to "plant root".

2.2 Is Imagenet a Good Choice for a Source Dataset?

Given the noteworthiness of ImageNet, we will explore why it has been so commonly used over the years. Imagenet is broadly adopted as a source dataset for transfer learning, and it is widely believed that models that obtain high accuracy on ImageNet will be able to perform well on new domains. Is this hypothesis true? What features are learnt by pretraining with it, and are they particularly beneficial? The first research article [4] deals with the former question. And this paper studies the correlation between obtaining a high accuracy on the ImageNet dataset and other tasks.
The experiments are three: one is taking the features from a model pre-trained on ImageNet and using them on a new task; the objective is to confirm whether the features learnt through pretraining are decisive for other domains. Number

two finetunes an ImageNet pre-trained network. Lastly, number three trains a model from scratch on the target dataset. A model trained from scratch means that there is no prior training. The purpose of these last two experiments is to prove the hypothesis that pre-trained models perform better by comparing them to models trained from scratch. The third experiment, in particular, demonstrates that machine learning architectures that perform well on the source dataset tend to perform well on the target, too.
The datasets used vary in domain and scope, there are more generic datasets like: CIFAR-10[17], CIFAR-101[17], PASCAL VOC 2007[18], Caltech-101[19]. Datasets more focused on fine grained recognition were also used such as: Food-101[20],Birdsnap [21],Stanford Cars [22], FGVCA Aircraft [23] , Oxford-IIIT Pets [24]. And other texture classification datasets.
For the first experiment *fixed features* from pretraining are used in a new task; a logistic regressor then takes these features as input. The results show a strong correlation between how the model performs while training ImageNet and the accuracy of the logistic regression classification. Therefore, features learnt with ImageNet are generally transferable. However, there is a caveat when finetuning instead of using logistic regression. Finetuning allows the model to learn features from the source domain, yet such features are not fixed; they can be changed while training with the target data.
The second and third experiments, in fact, reveal that while transfer accuracy is correlated to the ImageNet top-1 accuracy, improvement was surprisingly marginal in the datasets Stanford Cars and FGVC AIR in comparison to training from scratch.

Table 2.2: Improvement in performance for FGVC datasets while using Inception v4 in the paper [4].

Dataset	Pre-training	From Scratch
FGVC Aircraft	93.7%	92.7%
Stanford Cars	89.0	88.8%

These two datasets share several things in common: they're both dealing with fine grained classes and their labels are not well represented in ImageNet. Imagenet has 12,455 images for cars; however, it only has ten high-level car categories, such as "sports car". On the other hand, Stanford cars has 8,144 images with 196 classes categorised by make, model, and year. Furthermore, both of them have less than 10,000 images, a number that is much smaller than most datasets that are used to

train CNNs.
Another interesting result is that other FGVC datasets, such as OxfordFlowers102, OxfordPets, Birdsnap, and Food-101, do not have the same performance issues. ImageNet has most of the pet classes present in OxfordPets; it also has 59 bird classes and 45 classes that pertain to the Food-101 images.
The authors conclude that there is the benefits of pretraining for fine-grained tasks dwindle as the divergence between source and target labels increases.
Lastly, the third experiment, where datasets are trained from scratch, provides further insights. The first one is that even if the target datasets are trained on a randomly initialised model, there is a correlation with the accuracy of a model pre-trained on ImageNet. Additionally, as was shown previously, some tasks barely benefit in terms of accuracy from pretraining. Nevertheless, there is a considerable benefit in the amount of time needed to reach the same comparable accuracy.
To the speed of each technique, the authors calculated the number of steps and epochs required to reach the 90% of maximum odds of correct classification (defined in 2.1) achieved at any number of steps. Subsequently, the geometric mean across datasets was calculated.

$$\log \frac{n_{correct}}{n_{incorrect}} \exp(\Delta) \tag{2.1}$$

Finetunting reached the 90% threshold in 24 epochs/1151 steps, while training from scratch reached the same goal in 44 epochs/19531 steps. Finetuning speeds up the training process by 17 times.
Therefore the results of this paper are significant to our research, mainly because they prove that similarity between the source and target labels helps improve performance. Another useful finding is that even if we are dealing with a dataset that does not benefit much in terms of accuracy, there is still an improvement in terms of time to convergence.

The paper [25] delves into several similar questions by using Imagenet as a source for transfer learning.
One of these questions asks whether or not the amount of pretraining data affects performance. In our methodology, we will be attempting to reduce the amount of data during the source training. Therefore the answer to this query is interesting.
In order to carry out this experiment, the authors use AlexNet as a model. They also train said model with 50, 125, 250, 500 and 1000 images for each of the 1000 categories in ImageNet. These pre-trained networks are then finetuned with the datasets PASCAL-DET, PASCAL-ACT-CLS and SUN-CLS.
Their results suggest that diminishing the amount of data during pretraining does not significantly affect the target task's accuracy.
We think this result is compelling since our optimisation strategy is to select and

reduce data without negatively impacting the performance.
Another important outcome of this paper is analysing how Imagenet relates to fine-grained recognition tasks. Imagenet has very specific classes, such as different dog breeds, within its 1000 classes.
More specific Imagenet classes were clustered into their hypernyms until the total number of remaining classes lowered from 1000 to a lesser number of classes. A term's *hypernym* is a word that is more generic and broader than the term itself. E.g. "primate" is a hypernym to "chimpanzee" and "human". For this analysis, specific concepts like "poodle" were incorporated into more generic classes like "dog", as shown in the figure 2.4. The hierarchy followed was the hypernym/hyponym taxonomy from wordnet.
The image 2.3 shows the relationship between the number of classes and the accuracy during classification in the target task. 486 training classes leave the target performance unaltered, whereas there is a sharp drop in accuracy when going under 79 classes. The overall conclusion for this experiment is that, while there is some benefit in pretraining with more fine classes, as long as one does not go under a critical amount of classes, there are diminishing returns in increasing the total categories. In particular, going beyond 127 classes for these experiments seemed to yield only a negligible increase in accuracy.
Subsequently, there are experiments dealing with how the number of classes affects the learning of features.
The first of these analyses answers whether learning from general classes helps grasp fine-grained features. Like in the previous experiment, this was done by pretraining on the same data but with more generalised classes and then proceeding to categorise the target domain. The k-NN (k nearest neighbours) algorithm categorised target domain images based on the features from the FC7 layer (AlexNet). The k nearest neighbours algorithm is a method of categorising data that analyses the nearest k-neighbours in space for a specific data point. The most prevalent class is the class that corresponds to the predicted label 2.5. At any rate, the outcome of this is that by training with only 127 classes, the performance of the k-NN algorithm lowers only by 15% in comparison to the baseline (features obtained by pretraining on the original classes).
After these conclusions, the opposite query arises: is training with fine-grained classes beneficial for coarse recognition? The experiment shows a mixed outcome. Some classes, such as mammal, fruit or bird, have very similar sub-classes and therefore identifying the hypernym class is done with excellent accuracy. On the other hand, classes like tool, fabric and fungus do not have this beneficial response due to the sub-classes being so disparate.
Lastly, the authors observe that the number of images and the reduction in the number of classes have a negligible effect. They speculate that these might be because the target tasks, PASCAL and SUN, might be too similar to the categories

in ImageNet.

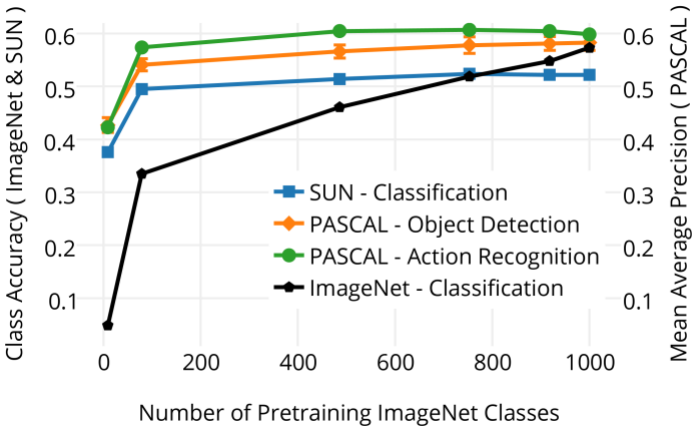

Figure 2.3: How accuracy in classification is influenced by the number of classes in the source dataset in [25]. The imagenet performance is measured by returning the model's fc layer to 1000 classes and finetuning accordingly.

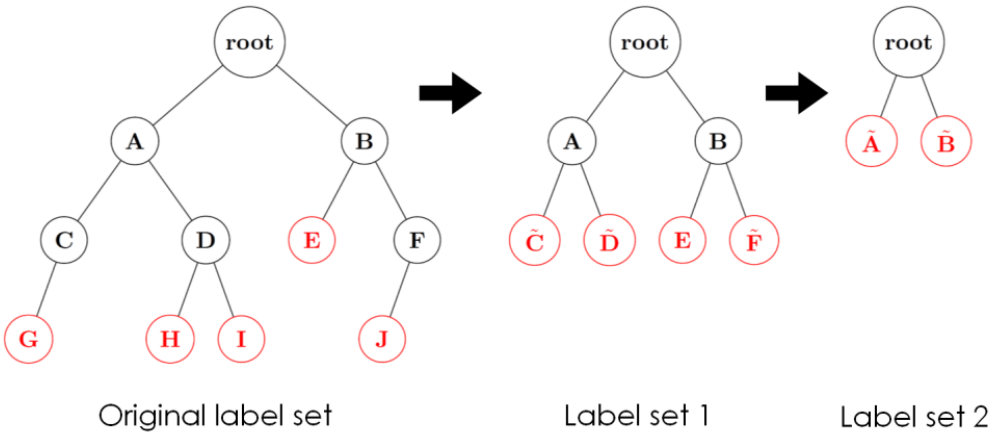

Figure 2.4: Process via which imagenet labels were transformed from specific categories into coarser more generic concepts in [25]

.

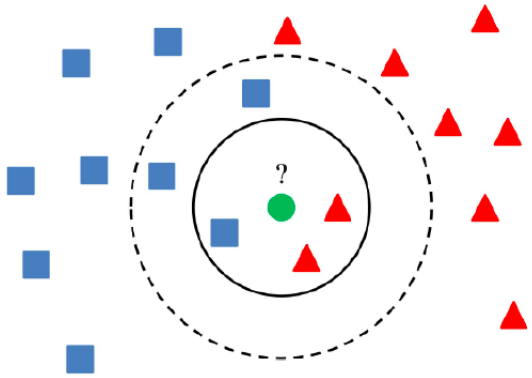

Figure 2.5: The green dot is the data-point that needs to be classified in this KNN example. If $k = 5$, then three of its neighbours are blue, and only two are red. By majority vote, the green dot is classified as blue in this particular instance. This method is used in [25] to analyse what features are learnt by pretraining on classes that are less fine-grained and more coarse. Image taken from [26].

2.3 FGVC

FGVC stands for fine-grained visual categorisation, which is a subfield of image classification where one must distinguish between visual categories that are more domain-specific (e.g. classifying different bird species).
Several problems make this task harder than the average image classification task: First, since these sub-categories are very specific, datasets tend to be fewer and smaller. Creating a dataset of this type often requires expert-level domain knowledge; therefore, it is more challenging to collect data for fine-grained datasets than it is to collect data from more general datasets like ImageNet.
Second, there is the sizeable intra-class variation: As shown in the figure 2.6, images belonging to the same class might appear very visually different.
Thirdly there is the problem of slight inter-class variation: Most of the time, there are few visual differences between classes. Backgrounds, for instance, tend to aggravate this problem since they are often redundant. If we are dealing with an FGVC problem that tries to classify birds.

We will now go into some of the standard practices and methodologies used by the state-of-the-art FGVC algorithms. According to a 2020 survey on FGVC [27] the most recurrent datasets in this field are: CUB200-2011. [28]: 11,788 images and 200 classes of birds. It is one of the most commonly used datasets in FGVC.
FGVC Aircraft [23]:10,000 images and 100 different aircraft types.
Stanford Dog [29]: 20,580 images and 120 dog species.

Figure 2.6: Intra and inter class variation, taken from [27]

Stanford Cars [22] 16,185 images and 196 brands of cars.
Oxford Flowers [30] 8,182 images and 102 flower types.
Furthermore, there are two main methods for classifying images in these datasets: the strongly supervised and the weakly supervised categorisation methods.

In **strongly supervised categorisation methods** and fine-grained tasks, detecting objects and analysing local information is of crucial importance because differences between images appear in small visual subtleties .
It is for this reason that most FGVC algorithms function in the following way: The first step is locating the essential parts of the object. Secondly, one proceeds to align the positions of the parts identified in the previous step. Finally, it is possible to extract the features for classification. Part detection and object detection often relies on the RCNN algorithm. The RCNN algorithm is similar to the CNN algorithm, only this time, the process consists of two stages. The first stage identifies portions in an image that might contain an object. In the second stage, the image portions are fed into a CNN architecture to learn significant features automatically. The problem with this type of approach is that it requires image data with bounding boxes. Unfortunately, this information is seldom available in practical applications.

Weakly supervised algorithms have been developed as an alternative when fully annotated datasets are unavailable. These algorithms only use category labels or text descriptions.
The main hurdle is detecting the crucial parts of an image, and the classification

algorithms can be divided into four categories:
(i) **Part based approaches**: two main techniques are used in this case.
Attention bases mechanisms where the main idea is to give more weight to more critical parts of the images to focus on more discriminative regions automatically.
Clustering: segmentation of an image via grouping pixels that share common characteristics such as colour, intensity and texture. The figure 2.7 is an example of using such a technique.
(ii) **End to end visual coding approaches**: this approach aims to enhance the visual information contained within the images. It does this by using high-order mixing of CNN features. In other words, instead of using the traditional global average pooling, one can use the bi-linear pooling method. This has proven to enrich the discriminative power of the feature maps.
(iii) **Approaches with external information**: The idea for this method is to use additional information such as web images, knowledge graphs or text.
Large datasets are not always available for FGVC tasks; therefore, using web images with readily available labels has been proposed as a solution. The biggest issue with this is the fact that labels are often noisy. Experimental results are trying to minimise the harmful effect of this data.
Another solution is utilising knowledge graphs. These knowledge graphs are data structures that illustrate the semantic relationships between categories.

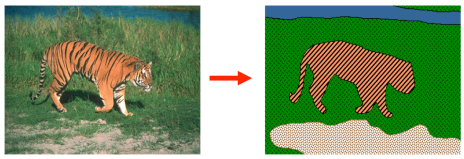

Figure 2.7: Example of segmentation taken from [31].Segmentation clusters pixels that share common characteristics to separate objects within an image. The algorithm identified four main segments in this image: water, grass, tiger and dirt.

Chapter 3

Related Works

3.1 Semantic and visual similarity

The current thesis originates from the conclusions in [9] which primarily studies the relationship between visual and semantic similarity. Is our premise of using semantic similarity as a substitute for visual similarity valid? We think that [9] answers this question thoroughly.
This paper deals with four main issues:
The first is: Within a taxonomic hierarchy, like the one provided by ImageNet, what is the relationship between the size of the semantic domain and the nature of visual variability? In other words, would a more generic class, such as animal, have more visual variability than a more specific class like "dog"?
Secondly, the authors try to provide a visual prototype for each category. This analysis considers that within cognitive science, most humans agree on a single prototype for some categories like "bird". For example, most people would agree that a bird can fly and is feathered. However, this agreement on prototypes does not hold for more generic categories. Thirdly, the relationship between semantic and visual similarity is studied. This examination relies on the cognitive psychology terms: category and super-category.
A category is a grouping of similar objects, and a super-category is a grouping of similar categories.
The fourth question is whether or not these categories belonging to the same super-category share visual similarities. Are the images of cows more visually similar to dogs (with which they share the mammal super-category), or is it equally similar to pictures of cars?
Lastly, there is proving that visual similarity inversely correlates to the broadness of a semantic category. In other words, can computer algorithms successfully classify across semantically different domains? Or is the visual similarity too scarce in

order for this to be possible across semantic boundaries?
In order to study how visually different images within a particular category are and answer the first question, a GIST representation is used. The GIST representation is a descriptor for an image, which provides a visual summary of the picture in question. It essentially uses a series of Gabor filters (a type of linear filter) and combines them into a histogram. The way in which these histograms are generated is as follows: First, images are segmented into k blocks, and each pixel in this segment goes through the Gabor filter. Next, each pixel generates a vector of values a_i as a response. Finally, vectors in the same blocks are summed as shown in figure 3.1.

$$v = \sum_{i \in B_j} a_i$$

where B_j is a particular block. Lastly, the k descriptors of all blocks are concatenated to give the GIST descriptor.

$$\text{descriptor} = [v_1, ..., v_k]$$

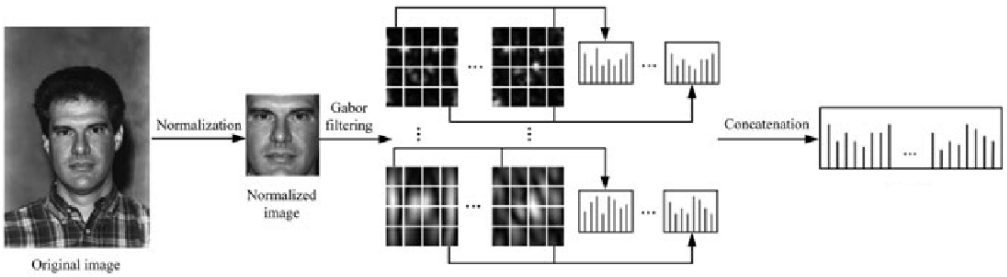

Figure 3.1: As seen in this here, the first step after normalisation is to divide the image into blocks. Denoted by the image being split into several different squares in the figure. Each pixel generates an array of values. Each square generates an array by summing the individual pixel arrays, and these arrays of the squares are concatenated at the end to form the GIST descriptor.

This GIST descriptor helps define another metric: the r_s visual scale of a category S. The visual scale gives us an idea of how visually different images in a class are; the bigger the number, the more different and disparate the images are within a given category. The visual scale is the average distance between the mean GIST descriptor of a category μ_S and the GIST descriptor of all images in S.

$$r_S = \frac{1}{|S|} \sum_{I \in S} D(I, \mu_s)$$

Where $D(I, \mu_s)$ is the squared Euclidean distance between a given image I and μ_s. This experiment concludes that as the depth increases and as categories become more and more specific, the visual variability decreases.

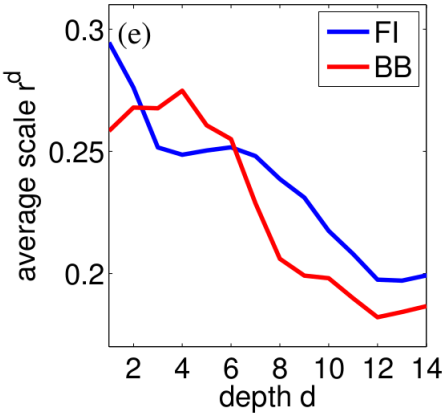

Figure 3.2: This image shows the relationship between the semantic depth of images and their average distance. This means that images in finer and more specific categories like *poodle* are more visually consistent and similar to images in broader categories like *mammal*.

Following this is the study of the relationship between semantic and visual similarity. What sort of visual relationship do semantically similar classes have to each other? Or vice-versa.
The Jiang and Conrath semantic similarity measure [32] is used for this next experiment. The definition for such a metric between two categories, S and T, is the following:

$$D^{JC}(S,T) = 2log(p(lso(S,T)) - (log(p(S)) + log(p(T)))$$

where $p(S)$ is the percentage of all images in S, $lso(S,T)$ is the lowest specific common ancestor between S and T.For instance in 2.2 the lso for "surface" and "cage" is "artifact.artefact". Subsequently, the visual distance between two categories is defined as follows:

$$D^{V}(S,T) = \frac{1}{|T|} \sum_{I \in T} D(\mu_S, I)$$

μ_s is the mean descriptor of class S. The formula above computes the average distance between μ_s and all of the images in the T category. The authors then go on to perform an analysis of the relationship between D^{JC} and D^{V}. Such analysis splits into two parts. The first part deals with full images (FI), which are images

that include the background, and bounding box images (BB), which only focus on the object of the category, excluding all other elements such as the background.
(i) At **low semantic distances**: FI images tend to have pretty similar backgrounds (for instance, cows and goats). Therefore this explains the lower D^V compared to its bounding box counterpart.
(ii) At **medium semantic distances**: Full images start to have a more diverse background, and the slope increases.
(iii) At **high semantic distances**: categories appear in extremely different environments, therefore the high slope for FF. On the other hand, there is a convergence for BB pictures, indicative of the fact that at this point, all images are equally dissimilar.

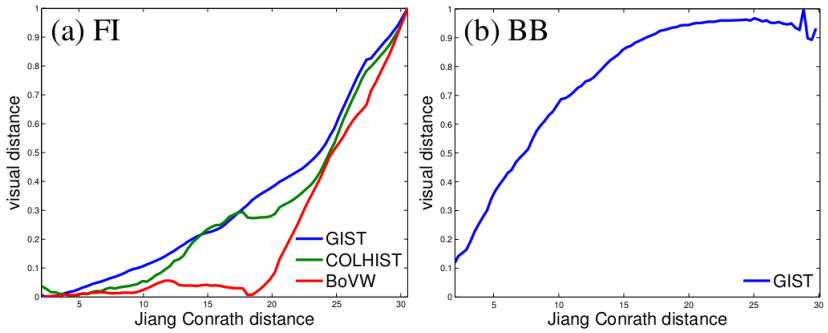

Figure 3.3: Plot of how visual distances change in relation to semantic distance. It varies depending on the type of descriptor used. However, the general trend is the same for each descriptor: as semantic distance increases, so does the distance at a visual level. Within the bounded box task (b), the visual distance increases much more rapidly.

And finally, the last question relates to the visual broadness in categories.
In order to study this concept, the authors introduce a new definition of "class". All pairs of categories (S, T) with a semantic distance smaller than a certain quantity X, $D^{JC}(S,T) \leq x$ are considered as belonging to the same class. Conversely (S', T') pairs such that $D^{JC}(S', T') > x$ belong to different classes. x is called the *semantic span*. Given these definitions, the average within-class visual distance is smaller than the average between-class visual distance, as shown in the figure 3.4.

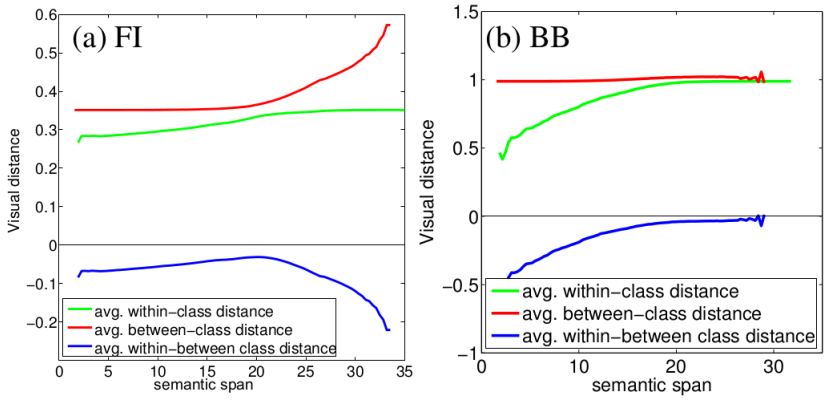

Figure 3.4: Red and green lines represent the within-class and between-class visual distance respectively. The blue line represents the difference between the two. Within class distance consistently rises as the semantic span increases.

3.2 Training Source Datasets on more Domain Specific Classes

Since finding large-scale FGVC datasets is often unfeasible, it is common practice to pre-train neural networks on more general data, such as ImageNet.
The paper [8] tackles the question: how can we design models for FGVC that perform well with generic pretraining given that we know our target task?
The technique proposed is to select a subset of the source dataset that is visually similar to the target task.
Prior to this paper, other work had explored the connection between transfer learning and domain similarity.
[33] demonstrates that random splits are better than man-made object splits when dealing with transfer learning in ImageNet.
[34] adds 512 relevant categories to ImageNet, and it improves the performance of the dataset PASCAL VOC.
[35] uses a combined dataset of ImageNet and places as a source and obtains a better accuracy on some visual recognition tasks.
The novelty of [8] is that while other studies only finetune the final layer, this finetunes every single layer. As explained before, CNNs consist, amongst other things, of fully connected layers and convolutional layers. Finetuning on the final layer entails freezing the convolutional layers. Therefore, features do not change during the training of the target task. This latter experiment changes the FC (fully connected) layer and the prior convolutional layers.
Secondly, [8] provides a way to quantify the visual similarity between two domains.

Given a source domain S and a target domain T, the distance between two images $s \in S$ and $t \in T$ is:

$$d(s,t) = ||g(s) - g(t)||$$

This formula is nothing other than the euclidean distance between their feature representations. These feature representations are obtained through $g(\cdot)$ which is the feature extractor for the image, or as we have referred to before: the convolutional layers. More specifically, the features in this paper are the output from the penultimate layer of a ResNet-101.
Furthermore, the authors ignore the effects of domain scale (the number of images). As a matter of fact, in general, more images increase the performance of transfer learning. However, the positive effects of domain scale are only logarithmic [36]. Therefore the number of images is not such an unreasonable assumption.
Having defined the distances between images, it is now possible to define the distance between domains. The distance between the source domain S and the target domain T is the least amount of work needed to move the images from S to T. Such a definition of domain similarity can be calculated via the Earth Mover's Distance (EMD) [37] [38].
Further simplifying the image features is done to make the computations more feasible. All image features within one category become, via simplification, the mean of their features. Using these definitions and simplifications, the Earth Mover's Distance between S and T is:

$$d(S,T) = EMD(S,T) = \frac{\sum_{i=1,j=1}^{m,n} f_{i,j} d_{i,j}}{\sum_{i=1,j=1}^{m,n} f_{i,j}}$$

where $d_{i,j} = ||g(s_i) - g(t_j)||$ and the optimal flow $f_{i,j}$ is the least amount of total work by solving the EMD optimisation problem. Given all of this, the domain similarity is defined as follows:

$$sim(S,T) = e^{-\gamma d(S,T)} \tag{3.1}$$

where γ is a parameter set to 0.01.

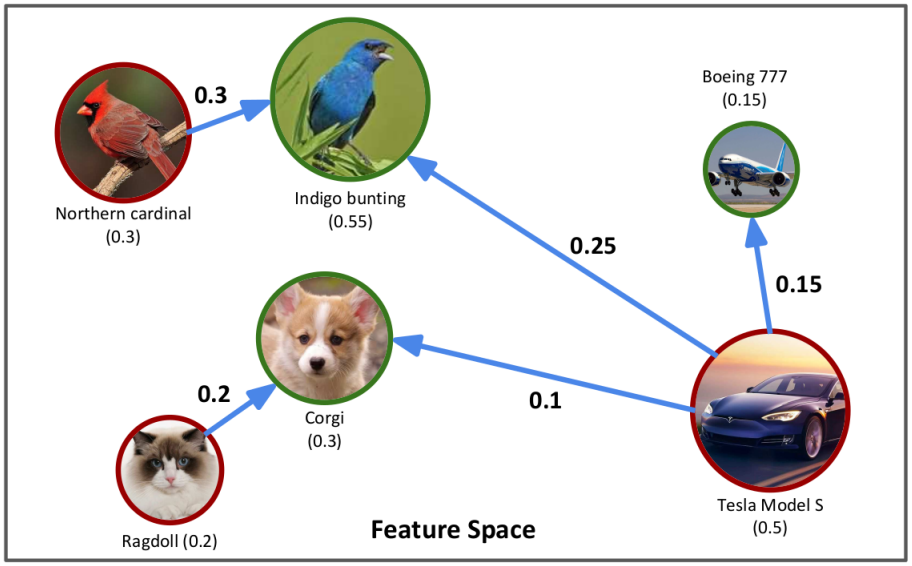

Figure 3.5: Representation of domain similarity via EMD. Red and Green mean source and target domain, respectively. The size of the circle is the normalised number of images.

The authors then use the definition of 3.1 to select a subset from the source domain that best transfers into the target domain. The strategy used is a greedy strategy where each category s_i in the source has its similarity measured to the target domain $sim(\{(s_i,1)\},T)$ 3.1. Within the realm of computer science, a greedy strategy refers to an algorithm that does not guarantee whether our solution will be optimal. The optimal solution, in this case, could be found by trying every possible subset as input for pretraining and then measuring the accuracy of the target task. This is unfeasible and computationally costly.
Hence, the greedy strategy employed here was to select the top k categories with the most similarity and use them for pretraining. To this end, Imagenet and the iNaturalist [39] datasets are used. Additionally, there is another pretraining that makes use of both datasets combined. Two different ways of selecting a subset are essential for this latter method. **Subset A** was created by including the top 200 ImageNet + iNaturalist categories for each of the 7 target FGVC datasets. **Subset B** on the other hand, used the 400 most similar categories for CUB200, NABirds, 100 categories for Stanford and 50 categories for Stanford cards and Aircraft.
Except for Food-101 there is always better transfer learning performance when pretraining from a similar source.

Figure 3.6: The number show the top-1 accuracy. In green and red are the particularly good and bad performances respectively. Overall the datasets selected based on domain similarity result in a good performance for every target task.

	CUB200	Stanford Dogs	Flowers-102	Stanford Cars	Aircraft	Food101	NABirds
ImageNet	82.84	84.19	96.26	91.31	85.49	88.65	82.01
iNat	89.26	78.46	97.64	88.31	82.61	88.80	87.91
ImageNet + iNat	85.84	82.36	97.07	91.38	85.21	88.45	83.98
Subset A (832-class)	86.37	84.69	97.65	91.42	86.28	88.78	84.79
Subset B (585-class)	88.76	85.23	97.37	90.58	86.13	88.37	87.89

3.3 Other Methods of Source Data Selection that are not Semantic

Another example of being selective with pretraining data comes from the natural language processing (NLP) side of machine learning. Natural language processing concerns itself with analysing language through neural networks. Notably, [6] deals mainly with the issues faced by heterogeneous transfer learning, but some insights can prove pertinent to this thesis.
Heterogeneous transfer learning is characterised by having different feature spaces for source and target dataset [40]. An example of different feature spaces in NLP is having two texts written in different languages.
The authors of this paper propose, amongst other things, the selection of a subset of the source dataset to improve accuracy in the target task.
The proposed technique is to have a weight vector $\{\alpha_k\}_k = 1...K$ over a data set. This weight vector represents the importance of that particular entry. In order to make optimisation more feasible, the dataset is organised into k clusters $S_1, ..., S_k$ via the k-means algorithm. The optimisation of the problem is as follows:

$$\min_{a,W_f,W_g,W_h} \mu \sum_{k=1}^{K} \frac{\alpha_k}{|L_{S_k}|} \cdot L_{HR:K}(S_k) + L_{HR}(T) + R(W) \tag{3.2}$$

where

$$\alpha_k = \frac{\exp(a_k)}{\sum_{K=1}^{K} \exp(a_k)}, 0 < \alpha_k < 1$$

and

$$L_{HR:k}(S_k) = \sum_{i \in L_{S_k}} \sum_{\tilde{y} \neq y_s^{(i)}} max[0, \epsilon - f(g(x_S^i)) \cdot (y_s^{(i)} - \tilde{y})^T]$$

f and g are transforms that encode the different target and source features into a common latent space. Latent spaces mitigate the problem of having different feature spaces. E.g. when dealing with two languages, finding a latent space means

that a neural network views the words "cane" (in Italian) and "dog" with similar features.
a is a learnable parameter that defines the importance of each class cluster, $L_{HR:K}(S_k)$ is a hinge loss for the cluster in the source task, L_{S_k} is a set of source indices for a cluster. Finally, μ is a hyper-parameter that only penalises optimising for the source task.
The authors applied this technique and others in the context of a hetero-lingual classification task. The table in figure 3.7 shows that the proposed mechanism (CTHL:2fc+ATT+AE) outperforms other baselines. The attention (AE) mechanism does improve the performance. AE refers to an auto-encoder technique, and 2fc indicates that the architecture has two fully connected layers at the end. The

Figure 3.7: Results for the hetero-lingual classification task. The proposed attention mechanism provides the best results. This suggests that not all source data is equally important for the finetuning process.

Datasets		Target Task Accuracy (%)							
S	T	MLP	ZSL	(:AE)	CHTL	(:ATT	+AE)	(:2fc	**+ATT+AE)**
RCV1	FR	39.4	55.7	56.5	57.5	58.9	58.9	58.7	**59.0**
	SP	43.8	46.6	50.7	52.3	53.4	53.5	52.8	**54.2**
	GR	37.7	51.1	52.0	56.4	57.3	58.0	57.3	**58.4**
	IT	31.8	46.2	46.9	49.1	50.6	**51.2**	49.5	51.0
20 NEWS	FR	39.4	55.7	56.5	57.7	58.2	58.4	57.0	**58.6**
	SP	43.8	46.6	50.7	52.1	52.8	52.3	52.3	**53.1**
	GR	37.7	51.1	52.0	56.2	56.9	**57.5**	55.9	57.0
	IT	31.8	46.2	46.9	47.3	48.0	**48.1**	47.3	47.7
R8	FR	39.4	55.7	56.5	56.5	56.4	57.2	55.9	**57.7**
	SP	43.8	46.6	50.7	50.6	51.3	**51.8**	50.8	51.2
	GR	37.7	51.1	52.0	57.8	56.5	56.4	57.0	**58.0**
	IT	31.8	46.2	46.9	49.7	50.4	**50.5**	49.4	**50.5**
FR	R8	48.1	62.8	**63.5**	61.8	62.6	62.8	61.5	62.3
SP					67.3	66.7	67.1	67.4	**67.7**
GR					64.1	65.1	**65.5**	64.4	65.3
IT					62.0	63.4	**64.1**	61.6	63.0

datasets used were RCV-1 [41], 20 News Groups [42] and Reuters Multilingual [43].

Another paper that handles a similar issue is [7] deals with creating a subset of the source dataset to improve the target task's performance.
The technique proposed here is called **selective joint finetuning**.
This technique measures similarity between the source and target dataset images through low-level features. The authors state that this might be better than using high-level semantic information because the specificity of the target task might lie precisely on the low-level features (such as the fur on pets for the Stanford Dogs dataset).

There are two ways of acquiring these low-level features: The first is through the response of a Gabor filter, which is particularly good at recognising textures. And the second is through using the output from the kernels of different convolutional layers of an AlexNet network trained on ImageNet.
Each bin contains roughly the same number of pixels in these feature histograms. The histograms of all filter responses are then made into a feature vector for a particular image x_i. Such a process is entirely analogous to how the GIST features were constructed, as mentioned previously.
These descriptors are also calculated for the target domain. For each training image in the target domain x_i^t, the nearest neighbours are sought in the source domain. The distance between two images x_i^t (target) and x_j^s (source) is the following:

$$H(x_i^t, x_j^s) = \sum_{h=1}^{D} w_h[\kappa(\phi_h^{(i,t)}, \phi_h^{(j,s)}) + \kappa(\phi_h^{(j,s)}, \phi_h^{i,t})]$$

Where $w_h = 1/N_h$ and N_h is the number of kernels in a particular layer. This is done to normalise the result among different convolutional layers, which might have a different number of kernels. $\phi_h^{i,t}$ and $\phi_h^{(j,s)}$ are the kernel histograms for images x_i^t and x_j^s.
$\kappa(\cdot, \cdot)$ is the KL-divergence, a function that measures how different two probability distributions are.
Furthermore, particular special attention is given to instances that are hard to classify. The uncertainty of classification is given by the metric:

$$H_i^m = -\sum_{c=1}^{C} p_{i,c}^m \log(p_{i,c}^m)$$

C is the number of classes and p_{i,c^m} is the probability that the i-th training example belongs to the class c after m iterations.
Training samples with high uncertainty have a higher number of nearest neighbours in order to help in their classification.
For the experiments, the authors used a Resnet-152 architecture with the pretraining of [44]. Source and training data are initially mixed into mini-batches. After an average pooling layer, the source and target are split, and each of them is sent to a separate soft-max classifier.
The data used is the ImageNet ILSVRC 2012 training set as source domain for Stanford Dogs [29] , Oxford Flowers [30] and Caltech 256 [45]. A combination of ImageNet and Places [46] was used as a source for preraining MIT Indoor 67[47].
We are partiularly interested in the results pertaining to Stanford Dogs and OxfordFlowers-102 since they deal with a fine grained visual recognition task. From the results of table 3.2 it is clear that the selective usage of a source dataset is beneficial. However the authors themselves state that the selection of a source

Table 3.1: Classification Results for paper [7]

Dataset	Method	Mean Acccuracy
StanfordDogs	Join fine-tuning with all source samples	85.6%
StanfordDogs	Selective joint fine-tuning	90.2%
OxfordFlowers	Join fine-tuning with all source samples	93.4%
OxfordFlowers	Selective joint fine-tuning	94.7%

domain (and its subset) for a specific target task is still an open problem for future investigation.

Next, we peruse the paper [48], the authors have three main points they wish to elucidate.
The first one is that **more pretraining data does not necessarily help**. Secondly, **matching source and target dataset distribution** is helpful. This is done by weighting all data in the source to give more importance to relevant images. Finally, **fine-grained tasks require fine-grained pretraining**. The pretraining step needs to capture the correct discriminative features to classify during the target dataset.
In order to compute the importance of each label, the authors train the model M. The first step is to pre-train such a model with the source dataset. Subsequently, the target domain images are fed into this model. What is particular about this method is that the model does not change at this phase; its layers, even the fully connected, remain frozen. The model is receiving target images at input while still predicting source labels. This model M is the first piece of the puzzle to calculate labels' importance.
The next step is to come up with a way of estimating the importance of each label in the source dataset. In order to do this, the paper starts analysing the **expected value of the loss function**. This latter concept is also known as **risk** in machine learning, and it gives us an idea of the common mistakes a model makes; by minimising the risk, one can improve the neural network.
The objective, therefore, is to define a risk function for the target problem that is a function of the source dataset. However, this is troublesome because of the differences in how data is distributed in the source and target domain. During pretraining the neural network is trying to optimise parameters θ in order to minimise the expected value of the loss function $\mathbb{E}_{x,y\sim D_s}[L(f_\theta(x), y)]$. The problem with this is that the distribution at pretraining time D_s is often different than the one for the target dataset D_t.

Since the objective is to optimise, taking into consideration the distribution D_t while finetuning, we consider the loss function:

$$\mathbb{E}_{x,y\sim D_t}[L(f_\theta(x), y)] = \sum_{x,y} P_t(x, y)L(f_\theta(x), y)$$

P_s and P_t are the distributions of the source and target datasets, respectively. Subsequently, the loss is rewritten in order to include the source D_s:

$$= \sum_{x,y} P_s(x, y)\frac{P_t(x, y)}{P_s(x, y)}L(f_\theta(x), y)$$

$$= \sum_{x,y} P_s(x, y)\frac{P_t(y)P_t(x|y)}{P_s(y)P_s(x|y)}L(f_\theta(x), y)$$

Furthermore the assumption $P_s(x|y) \approx P_t(x|y)$. This is a reasonable assumption since it says that the distribution in the target and source dataset is similar given a certain label. The "dog" class is probably similar regardless of the dataset in which it is.

$$\approx \sum_{x,y} P_s(x, y)\frac{P_t(y)}{P_s(y)}L(f_\theta(x), y)$$

$$= \mathbb{E}_{x,y\sim D_s}[\frac{P_t(y)}{P_s(y)}L(f_\theta(x), y)]$$

So far, the assumption is that target and source share the same labels. The authors solve this problem by estimating both $P_t(y)$ and $P_s(y)$ for the labels in the source domain. These values represent distributions over the target and source dataset, respectively.
$P_s(y)$ was estimated by the times a label appears by the total number of data entries in the source dataset. On the other hand, $P_t(y)$ is obtained by using the predictions of model M and averaging them. This last step computes the probabilities of source labels on target domain examples.
This methodology was tested using the Inception v3[49] and the AmoebaNet-B models[50].

3.4 Oxford Pets State of the Art

We also rely on the results and techniques shown in [51]. This paper deals with a classification task with adversarial examples that have been injected with noise. However, the main point of interest here is the authors' results and techniques when dealing with the regular Oxford pets data- without any noise.

Table 3.2: Classification Results for paper [7] with Inception v3

Pre-training	Birdsnap	Oxford Pets	Stanford Cars	FGVC Aircraft	Food-101	CIFAR-10
Imagenet-Entire Dataset	77.2	93.2	91.5	88.8	88.7	97.4
Imagenet-Adaptive Transfer	76.6	94.1	92.1	87.8	88.9	97.7
Random Initialization	75.2	80.8	92.1	88.3	86.4	95.7

Table 3.3: Classification Results for paper [7] with Amoeba. The best published results refers to results from other papers

Pre-training	Birdsnap	Oxford Pets	Stanford Cars	FGVC Aircraft	Food-101	CIFAR-10
Imagenet-Entire Dataset	80.0	94.5	94.2	90.7	91.7	98.0
Imagenet-Adaptive Transfer	80.7	95.1	93.5	89.2	91.5	98.0
Best Published Results	82.9	95.9	94.6	94.5	93.0	99.0

Oxford pets is a dataset introduced by the paper [24]. This collection of images contains 7349 pictures of 37 breeds of cats and dogs. This dataset was initially introduced to research the problem of fine-grained image categorisation.
In the paper [51], the source dataset is ImageNet, and the target dataset is Oxford-Pets which is trained with an adamW optimiser with a oneCycle scheduler while freezing the weight layers of the NN for the first 4 epochs and unfreezing them for the remaining 4. This setup reaches a **93.10** accuracy.

Chapter 4

Methodology

We explore methods of reducing the amount of data needed for the source dataset while lessening the negative impact on the performance of the target task.
In order to achieve this, we select source dataset instances that are particularly relevant to the target task. As noted in the related works section, there are different ways of selecting categories during the pretraining phase. We opt for a method that analyses semantic relationships between source and target classes. Papers like [7] have noted the shortcomings of a semantic-based selection: this method might ignore important underlying low-level features spread across semantically unrelated categories. There is, however, an advantage in using linguistic relations. Knowing which categories to select for the source domain training before the pretraining dataset is possible.
Our method makes it so that if the target task is known, the creation of the source dataset can be focused only on the classes identified by the semantic selection process.
Such an objective is achievable via a lexical database for English (or whatever language the task requires). The database, as mentioned earlier, must possess links and semantic relationships between words organised via a tree graph with a hierarchical structure.
Semantic relationships allow the usage of several metrics for quantifying the similarity between words; There exist several useful metrics: **path** and **wup** similarity are two possibilities. These are based on the structure of a taxonomic hierarchy. This hierarchy views morphemes as nodes in a tree graph, as in in 2.2. The path similarity formula is the following:

$$\text{similarity}_{path} = \frac{1}{distance(s_1, s_2) + 1}$$

The *distance* word refers to the shortest path length between the two words whose similarity is being measured s_1 and s_2. The similarity scores range from 1 to 0,

where 1 represents identity. The **wup** similarity formula, on the other hand, is the following:

$$\text{similarity}_{wup} = 2 \cdot \frac{\text{depth}(lcs(s_1, s_2))}{\text{depth}(s_1)\text{depth}(s_2)}$$

In this case, lcs is the least common subsumer. In other words, their most specific common ancestor.
We use these metrics to analyse the source dataset classes or even the morphemes in the lexical database if the source dataset is unavailable. The first step is to delineate into a list of words l_t, which is the domain of the target dataset. For example, if the target domain is animal-related like Oxford-Pets, the chosen words could be nouns related to pets like "dog, cat".Another case is "flower" for Oxford-Flowers, or "food, fruits, vegetables" for Food-101.
This array of terms can consist of any arbitrary number n of words. The list of words can even be all of the categories for the target dataset.
The next step is to calculate each source morpheme's m_i similarity to each word w_j in l_t, and this, in turn, produces a list of similarity values for each m_i in the source:

$$l_{(s,i)} = \{\text{similarity}(m_i, w_1), ..., \text{similarity}(m_i, w_n)\}$$

We define the similarity of each source morpheme m_i to the target domain as:

$$s_i = \max(l_{(s_i)})$$

The experimenter chooses the similarity that each pretraining class must have to be selected via a threshold t. Having a higher threshold implies selecting source categories with a stronger semantic relationship to the list of words l_t. Any class whose similarity to the source domain is lower than t is excluded from the pretraining phase.
For instance, we have $t = 0.15$ and Oxford-Pets as our target. The morpheme that we are trying to analyse is "speedboat".

$$\text{similarity}_{path}(\text{speedboat,dog}) = 0.067$$

$$\text{similarity}_{path}(\text{speedboat,cat}) = 0.05$$

$l_{(s,i)} = 0.067$. And therefore given t the class "speedboat" is not considered relevant enough to include in the pretraining.
On the other hand, the class "collie" is included in the dataset because it has a similarity of 0.25 with "dog" and 0.125 with "cat".
After selecting the appropriate classes, the first category uses all images from such categories as input.
This data is separated into a training and a validation dataset. Furthermore, transformations modify the data before the neural network begins its training

process. The transformations used for the training dataset are similar to the default data augmentation done for the FastAI framework: RandomResizedCrop zooms the image, resizes it and then it crops it. Resize resizes the image into a given dimension. RandomHorizontalFlip flips the image along its horizontal axis with a given probability specified by the user. Next, RandomRotation accepts two numbers representing an image's minimum and maximum rotation. Then, ColorJitter randomly affects the brightness and saturation of an image. With a given probability, RandomPerspective changes an image's perspective. Lastly, the transforms change the image into a tensor (ToTensor) format, and the tensor is normalised (Normalisation). The validation transforms are fewer. These are Resize, ToTensor and Normalization.
With this data, we are now ready to input tensor images into the model, which in this case was the ResNet-34. The model undergoes training for a certain amount of epochs, and no layer is frozen at this stage. While the network is being trained, the model uses the validation dataset after each epoch to measure its accuracy. The model's output is a tensor whose size equals the number of classes selected. It is a one-hot encoded tensor of zeroes and ones. This training step results in a model whose weights are saved.
Subsequently, we take this same model with the weights learnt from the previous step and replace the final fully-connected layer with another fully connected layer whose length equals the number of classes in the target dataset.
Then this new model is trained with the same methodology with which the authors of [51] trained their ResNet. This methodology takes the model and only trains the fully connected layers for the first half of the training epochs. Subsequently, both convolutional and fully connected layers are trained.
As aforementioned, the first experiment selects all images of the relevant classes. However, additional experimentations explore the effect of the number of images chosen and the classes chosen.
The **second** experiment measures the impact of class selection on the overall performance by using balanced classes. We took some of the results from the previous experiments, maintained the number of photos they originally used and then modified the classes employed to 1.000. We compared the most promising outcomes of the previous experiment with a pretraining done on the same number of images but with all classes from the source dataset.
Each class i was as balanced as possible. However, this was not always possible since the number of images n chosen was not divisible by the number of classes m. Therefore the number of images x_i was determined in the following way:

$$x_i = \begin{cases} \lfloor m/n \rfloor, & \text{for } i > m \mod n \\ \lfloor m/n \rfloor + 1, & \text{for } i \leq m \mod n \end{cases} \tag{4.1}$$

Essentially what this means is that we perform an integer division of the classes by the images. This results in a remainder $r = m \mod n$ most of the time. This integer division has a remainder which is distributed by giving the first r classes one extra image.
A following **third** analysis involves selecting a certain amount of classes through a given threshold. This selection process of categories is entirely identical to the one outlined in the first experiment, except now the number of selected images is arbitrary. The entries chosen per category is as balanced as possible sometimes. However, this is not achievable since certain types of images are sometimes underrepresented.
The algorithm selects images in the following way. There is a number n of images we want to select and m number of categories. These classes have already been selected through the process described above. So the first step is to compute:

$$x = \lfloor m/n \rfloor$$

This cannot be the number of images per category because $x \cdot m$ does not necessarily equal n, and not every category has x images; some have less. The methodology, therefore, is as follows. Initially, each class i is assigned y_i images:

$$y_i = \min(x, z_i)$$

where z_i is the total number of images for class i, then we define r, which is the difference between the number of images we desire and the ones we would have if each category were assigned y_i entries.

$$r = \sum_{i=1}^{n} (x - y_i)$$

The images are now distinguished into two sets. The first set consists of underrepresented classes belonging to set A with several images less than x. The second is overrepresented categories (set B) with more images than x. The number of images for A remains y_i, for B however, we must perform further computations. The number of images for the categories in set B needs to b increased in order to get to a total number of images that is equal to n, such a thing is done in the following way.

$$x'_{i \in B} = \begin{cases} \lfloor z/|B| \rfloor, & \text{for } i > |\text{B}| \mod \text{z} \\ \lfloor z/|B| \rfloor + 1, & \text{for } i \leq |\text{B}| \mod \text{z} \end{cases} \tag{4.2}$$

where z:

$$z = r + \sum_{i \in B} y_i$$

and $|B|$ is the number of categories that belong to the set B.
The number of images that could not be taken from the A set is taken from each class of B so that it is as balanced as possible. This might not always be possible since $\lfloor r/|B| \rfloor$ is not necessarily an integer. The remaining $k = |B| \mod z$ images are redistributed equally amongst the first k classes.

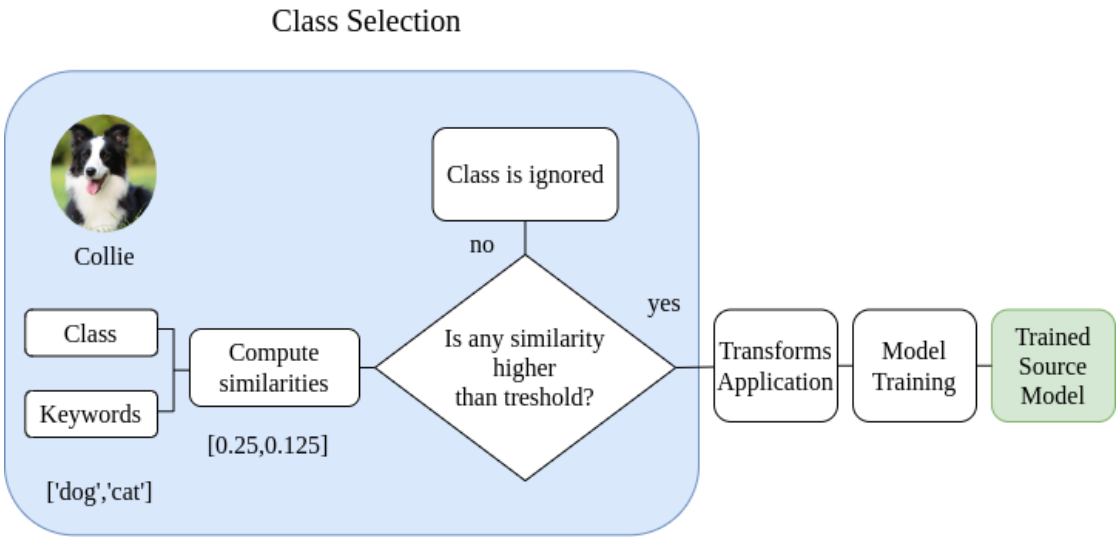

Figure 4.1: Pipeline for training the source dataset. The class selection phase is the novel part introduced by our method. In this example Collie has a similarity of 0.25 to the target domain, so if the threshold is equal to or lower than 0.25 collie is utilised in training.

Figure 4.2: Pipeline for the target domain training.

Chapter 5

Experimental Setup

The hardware utilised is an NVIDIA Corporation GP102 [TITAN Xp] on an Ubuntu "18.04.6 LTS (Bionic Beaver)" machine.
The software employed was Pytorch version 1.10.1+cu102 and Pytorch Lightning version 1.5.3. The source dataset is the ILSVRC ImageNet, and there are two target datasets: the first one is the Oxford-Pets [24] and the second one is Oxford-Flowers102 [30].
Oxford-Pets has a total of 37 classes; 25 dog classes and 12 cat classes.
Oxford-Flowers102 has a total of 102 flower classes.
The training of the ILSVRC ImageNet dataset uses the following hyperparameters: batch size =65,optimiser = SGD , scheduler = OneCycle with maximum learning rate = 0.05, training epochs = 50, learning rate = 0.05, image resolution = 224. The source training utilises the dataset's standard training and validation split. Moreover, not every image was used, and some, if not most, were filtered by the semantic filtering algorithm proposed in the previous chapter. The amount of images filtered depends on the "threshold" hyperparameter.
The threshold hyperparameter determines the number of classes, except for the balanced source training experiments, 6.5 and 6.5. These two experiments utilise all of the 1000 ImageNet classes. Conversely, the experiment 6.3 varies its number of classes while preserving the number of images at 210.273. On the other hand, target training uses another set of hyperparameters: batch size = 64, optimiser = AdamW, scheduler = OneCycle with maximum learning rate = 0.001, training epochs = 8, learning rate = 0.001, image resolution = 224. These hyperparameters are mostly similar to the ones used in [51].
Furthermore, like in [51], we split the Oxford-Pets dataset into 5912 training images and 1478 test images. For the Oxford-Flower102 dataset, we utilised the split already provided: 1020 training examples and 6149 test images. Lastly, there are the hyperparameters utilised for the transforms: RandomRezisedCrop utilises size=224 and scale=(0.91,1.0). The scale parameter is responsible for

zooming the image and size rescales it to the desired size. Resize has a size = (224,224). RandomHorizontalFlip uses p=0.5, which represents the probability of horizontally flipping an image.RandomRotation(degrees=10) randomly selects a number between [-10,10] which represents the degree rotation for the image. ColorJitter utilises (brightness=0.2,contrast=0.2,p=0.75).RandomPerspective has(distortion_scale=0.2, p=0.75), where distortion_scale regulates the degree of distortion. Normalize uses (mean=[0.485,0.456,0.406], std=[0.22,0.224,0.225]).

Chapter 6

Results

The tables 6.1 and 6.4 show the results for the the method proposed in this thesis. Alongside our proposed techniques, there are two other experiments in these tables. No TL shows the outcomes of not using any transfer learning, while the TL standard, as the name suggests, employs the traditional techniques for transfer learning.
6.1 displays information about Oxford-Pets's target domain. Unsurprisingly, No TL has the worst accuracy at 62.65% while TL standard holds the best at 93.17%. On the other hand, all TL Proposed rows display lower accuracies than TL standard but get reasonably close to the performance achieved by standard transfer learning. Moreover, every experiment surpasses No TL's accuracy.
However, the benefit of semantic selection and our contribution is most apparent in the *savings* column. This column quantifies the diminishment of iterations needed for the pretraining phase. "Savings" has the following definition:

$$1 - \frac{\text{source iterations}}{28{,}826{,}257}$$

28,826,257 is the number of iterations needed to pre-train the Pytorch ResNet-34 model. This number came about in the following manner: 1,281,167 is the number of images in the ImageNet training dataset. 8 is the number of cores used to train the ResNet34 model. 32 is the batch size and 90 is the number of training epochs. Therefore the final calculation is the following:

$$\text{iterations} = \frac{images}{batch\_size} * cores * epochs$$

A higher threshold yields higher savings since fewer images and classes are selectable as source training input. Moreover, lower thresholds produce a higher accuracy, but there are diminishing returns. The difference in accuracy between (path=0.25) and (path=0.10) is a mere 0.08%, yet the disparity amongst savings is much higher at 14.34%. Beyond a certain threshold, there is not much benefit to increasing the

number of source images.
The similarity metric is also noteworthy. The *wup* experimentations use threshold numbers that allow them to get as close to the source image values for the path experiments. However, the algorithm selects classes, and it is not practicable to directly select the number of images desired. Therefore the direct comparison between *wup* and *path* is not possible. However, we can still derive specific considerations. Wup is less predictable as a metric.$Path$ follows the trend of increasing accuracy as the threshold decreases. Contrarily *wup* does not reliably follow such a trend. In the table 6.1 (wup=0.67) has a higher accuracy than (wup=0.66), on the other hand in 6.4 lower *wup* values yield higher accuracies.
In 6.1 ,(Path=0,125) has 342,442 images, (Wup=0.67) has 314.001 and (Wup=0.66) has 400,892. Path's accuracy surpasses the two *wup* thresholds.Consequently, "path" is the metric that we focus on in the later experiments.
Table 6.4 shows the next set of outcomes. It performs the same TL experiments as before but uses the Oxford-Flowers102 as the target dataset. TL Proposed(path=0.15) and TL Proposed(wup=0.74) have lower accuracy than the No TL row. This result suggests that 7800 images are too few, leading to a drop in performance. The rest of the experiments surpass the no-transfer learning baseline. The table shows other path thresholds not present in the Oxford-Pets results (path=0.085, 0.083 and 0.075) to analyse what happens with a comparable number of images. The delta accuracies are still lower than that of 6.1.
The outcomes of the following experiment examine the impact of class selection on accuracy. We took results with the hyperparameters path=[0.15,0.125,0.10] from 6.1 and 6.4. These results maintained the number of photos originally used; however, the number of classes employed changed to 1,000. The original number of images was distributed among the 1.000 classes in a balanced way. This analysis is reported in 6.2 and 6.5, they refer to the Oxford-Pets and Oxford-Flowers102 datasets respectively.
For the Oxford-Pets dataset, the consequences of category selection are straightforward. First, semantic filtering of categories is beneficial; every entry's delta accuracy is positive. Suppose one is working with a certain amount of images. In that case, it is more advantageous to have the source images be visually similar to the target domain rather than having more categories.
On the other hand, the balanced classes experiment shows another situation entirely when dealing with the Oxford-Flowers102 dataset. In this case, the Proposed TL is damaging the categorisation accuracy. Transfer learning performs better when using 1,000 ImageNet classes instead of appointing them.
Several other studies mentioned in the previous sections provide an answer. In particular, [4] mentions that the improvement in accuracy by pretraining from scratch was particularly poor for classes that were not well represented in ImageNet. Unfortunately, despite our process of selecting the source dataset, we could not

mitigate this issue. Indeed, ImageNet has only two classes that are hyponyms of the word flower: *yellow lady's slipper* and *daisy*. On the other hand, the number of dog classes in ImageNet is 108, and the number of canines is 120. At the same time, the number of cats and felines is 6 and 10, respectively.
Lastly, there is the third experiment. The table 6.3 shows its results. It confirms that performance increases when the source dataset has sufficient images pertinent to the target. Conversely, adding more classes while maintaining the same number of images is harmful. The only exception is path=0.125. The difference in classes between path=0.125 and path=0.15 is 102. These 102 classes contain categories such as "dalmatian", "boxer", and "Madagascar cat", all extremely relevant to the target task.
The performance benefits as long as the classes added continue being relevant to the final training phase. Thresholds below 0.125 add images and categories that increasingly become more semantically distant from the visual domain of "dogs" and "cats". Accordingly, performance decreases.

Table 6.1: Comparison of the accuracy between the standard transfer learning technique and our proposed techniques.
The similarity metric and the similarity threshold are respectively written in between parenthesis in TL Proposed().
The definition of *savings* is $1 - \frac{\text{source iterations}}{28{,}826{,}257}$, where the denominator represents the number of iterations in the standard pretraining for ImageNet.
Furthermore, Delta Accuracy is the difference between each row's accuracy and the accuracy of the No TL experiment.

Source Dataset=Imagenet, Target Dataset=Oxford-Pets					
-	Source Classes	Source Images	Savings	Accuracy	Delta Accuracy
No TL	0	0	-	62.65%	0%
TL Standard	1.000	1.281.167	0%	93.17%	+30.52%
TL Proposed(path=0.15)	166	210273	99.43%	88.70%	+26.05%
TL Proposed(path=0.125)	268	342442	99.07%	90.46%	+27.81%
TL Proposed(path=0.10)	411	526226	98.57%	90.53%	+27.88%
TL Proposed(wup=0.67)	246	314001	99.15%	90.19%	+27.54%
TL Proposed(wup=0.66)	313	400892	98.91%	89.65%	+27.00%

Table 6.2: Every experiment in this table utilised all of the 1000 classes in ImageNet. The Delta accuracy refers to the difference in accuracy between the TL Proposed and the TL Balanced experiment that has the same number of source images.

Balanced Classes				
Source Dataset=Imagenet, Target Dataset=Oxford-Pets				
Source Images	TL Proposed	TL Proposed Accuracy	TL Balanced Accuracy	Delta Accuracy
210273	path=0.15	88.70%	81.66%	+6.44%
342442	path=0.125	90.46%	85.52%	+4.94%
526226	path=0.10	90.53%	86.33%	+4.20%

Table 6.3: This table pertains to an experiment where the source images remain fixed at 210.273, the number of images in TL Proposed(path=0.15).
The Delta Accuracy refers to the difference in Accuracy between TL Proposed(path=0,15) and all of the other runs.

210.273 Source Images			
Source Dataset=Imagenet, Target Dataset=Oxford-Pets			
	Source Classes	Accuracy	Delta Accuracy
TL Proposed(path=0,15)	166	88.70%	-
TL Fixed(path=0,125)	268	89.38%	+0.68%
TL Fixed(path=0,01)	411	86.40%	-2.3%
TL Fixed(path=0,075)	830	83.56%	-5.14%
TL Fixed(path=0,05)	997	81.94%	-6.76%

Table 6.4: The definition of *savings* is $1 - \frac{\text{source iterations}}{28{,}826{,}257}$, where the denominator represents the number of iterations in the standard pretraining for ImageNet. Furthermore, Delta Accuracy is the difference between the accuracy and the accuracy of the No TL row.

Source Dataset=Imagenet, Target Dataset=Oxford-Flowers102					
	Source Classes	Source Images	Savings	Accuracy	Delta Accuracy
No TL	0	0	-	47.52%	0%
TL Standard	1.000	1,281,167	0%	84.05%	+33.53%
TL Proposed(path=0,15)	6	7,800	99.98%	22.74%	-24-78%
TL Proposed(path=0,125)	31	39,881	99.89%	49.02%	+1.5%
TL Proposed(path=0,10)	77	98,827	99.73%	56.59%	+9.07%
TL Proposed(path=0.085)	123	156,720	99.58%	61.00%	+13.48%
TL Proposed(path=0.083)	247	316,133	99.14%	60.77%	+13.25%
TL Proposed(path=0.075)	423	540,393	98.54%	62.16%	+14.64%
TL Proposed(wup=0,74)	6	7,800	99.98%	26.26%	-21.26%
TL Proposed(wup=0,61)	36	46,381	99.87%	51.81%	+4.29%
TL Proposed(wup=0,55)	76	97,574	99.74%	55.93%	+8.41%

Table 6.5: Every experiment in this table utilised all of the 1000 classes in ImageNet. The Delta accuracy refers to the difference in accuracy between the TL Proposed and the TL Balanced experiment that has the same number of source images.

Balanced Classes				
Source Dataset=Imagenet, Target Dataset=Oxford-Flowers102				
Source Images	TL Proposed	TL Proposed Accuracy	TL Balanced Accuracy	Delta Accuracy
7.800	path=0.15	22.74%	34.92%	-12.18%
39.881	path=0.125	49.02%	56.22%	-7.18%
98.827	path=0.10	56.59%	63.32%	-6.73%
56.720	path=0.085	61.00%	65.18%	-4.18%
16.133	path=0.083	60.77%	67.18%	-6.41%
40.393	path=0.075	62.16%	63.44%	-1.28%

Chapter 7

Conclusions

This thesis analysed ways of optimising transfer learning problems in machine learning. Specifically, we proposed a method for reducing the number of images needed for the pretraining phase. The proposed approach is not a silver bullet; however, it provides flexibility and options for experimenters who need to diminish the amount of source training data.
A diminished pretraining time is particularly beneficial for researchers developing or experimenting on different ML architectures. The most lengthy part of training a model is the source pretraining phase. This method can expedite research significantly by reducing the time needed for the pretraining step.
The threshold hyperparameter controls the minimum semantic distance for a class to be accepted into the pretraining phase. It can vary according to the specific needs of the research. For example, the threshold can be higher or lower depending on whether researchers need resource efficiency or higher performance. These decisions are compromises since, as seen in the previous chapters, resource efficiency comes at the cost of accuracy and vice-versa. With the aid of our technique, such compromises are less harmful than the alternative of reducing the number of images used at the source.
Furthermore, the third experiment showed that increasing the number of classes can sometimes be beneficial even if the number of images remains the same.
Nevertheless, there is a caveat that concerns the source dataset's nature. The experiments show that the number of source classes pertinent to the target task must be sufficient. Unfortunately, ImageNet did not have enough "flower" classes, and therefore this method could not be applied with success. Consequently, we conclude that our technique works best when there are enough categories in the source dataset relevant to the target domain. When trying this approach on a dataset like Oxford-Pets, it is possible to use less data while sacrificing minimal accuracy.
As previously mentioned, this approach is also helpful in creating source datasets

from scratch. Using the target categories as input for the semantic selection algorithm, one can generate a list of pertinent classes that are helpful in the source dataset.
Moreover, this thesis concentrated on filtering categories from a single source. However, an open question is whether this technique can work if applied to several source datasets at the same time? [8] is a paper that incorporates several distinct source datasets to create only one. The paper authors reach their best performances by utilising a source dataset merging relevant classes from ImageNet and iNaturalist. Further research could try to semantically filter classes from different datasets and combine them into one.

Bibliography

[1] Gaudenz Boesch. *A Complete Guide to Image Classification in 2022*. [Online; accessed 22-June-2022]. 2022. URL: `https://viso.ai/computer-vision/image-classification/` (cit. on pp. ii, 1, 4).

[2] Min Chen, Shiwen Mao, and Yunhao Liu. «Big data: A survey». In: *Mobile networks and applications* 19.2 (2014), pp. 171–209 (cit. on pp. ii, 1, 4).

[3] Data Flair Team. *Active Contours – A Method for Image Segmentation in Computer Vision*. [Online; accessed 22-June-2022]. 2019. URL: `https://data-flair.training/blogs/purpose-of-data-science/` (cit. on pp. ii, 1, 4).

[4] 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 (CVPR)*. June 2019 (cit. on pp. 2, 8, 9, 37).

[5] Sergei Alyamkin et al. «Low-power computer vision: Status, challenges, and opportunities». In: *IEEE Journal on Emerging and Selected Topics in Circuits and Systems* 9.2 (2019), pp. 411–421 (cit. on pp. 2, 5).

[6] Seungwhan Moon and Jaime G Carbonell. «Completely Heterogeneous Transfer Learning with Attention-What And What Not To Transfer.» In: *IJCAI*. Vol. 1. 1. 2017, pp. 1–2 (cit. on pp. 2, 23).

[7] Weifeng Ge and Yizhou Yu. «Borrowing Treasures From the Wealthy: Deep Transfer Learning Through Selective Joint Fine-Tuning». In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. July 2017 (cit. on pp. 2, 24, 26, 28, 29).

[8] 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*. 2018, pp. 4109–4118 (cit. on pp. 3, 20, 43).

[9] Thomas Deselaers and Vittorio Ferrari. «Visual and semantic similarity in imagenet». In: *CVPR 2011*. IEEE. 2011, pp. 1777–1784 (cit. on pp. 3, 16).

[10] R. Saravanan and Pothula Sujatha. «A State of Art Techniques on Machine Learning Algorithms: A Perspective of Supervised Learning Approaches in Data Classification». In: *2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS)*. 2018, pp. 945–949. DOI: 10.1109/ICCONS.2018.8663155 (cit. on p. 4).

[11] Athanasios Voulodimos, Nikolaos Doulamis, Anastasios Doulamis, and Eftychios Protopapadakis. «Deep learning for computer vision: A brief review». In: *Computational intelligence and neuroscience* 2018 (2018) (cit. on p. 5).

[12] Shoaib Siddiqui, Ahmad Salman, Imran Malik, Faisal Shafait, Ajmal Mian, Mark Shortis, and Euan Harvey. «Automatic fish species classification in underwater videos: Exploiting pretrained deep neural network models to compensate for limited labelled data». In: *ICES Journal of Marine Science* 75 (May 2017). DOI: 10.1093/icesjms/fsx109 (cit. on p. 6).

[13] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. «A comprehensive survey on transfer learning». In: *Proceedings of the IEEE* 109.1 (2020), pp. 43–76 (cit. on p. 5).

[14] 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*. Ieee. 2009, pp. 248–255 (cit. on p. 6).

[15] George A Miller. «WordNet: a lexical database for English». In: *Communications of the ACM* 38.11 (1995), pp. 39–41 (cit. on p. 6).

[16] Olga Russakovsky et al. «Imagenet large scale visual recognition challenge». In: *International journal of computer vision* 115.3 (2015), pp. 211–252 (cit. on p. 7).

[17] Alex Krizhevsky, Geoffrey Hinton, et al. «Learning multiple layers of features from tiny images». In: (2009) (cit. on p. 9).

[18] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. «The Pascal Visual Object Classes (VOC) Challenge». In: *International Journal of Computer Vision* 88.2 (June 2010), pp. 303–338 (cit. on p. 9).

[19] Li Fei-Fei, Rob Fergus, and Pietro Perona. «Learning Generative Visual Models from Few Training Examples: An Incremental Bayesian Approach Tested on 101 Object Categories». In: *Computer Vision and Pattern Recognition Workshop* (2004) (cit. on p. 9).

[20] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. «Food-101 – Mining Discriminative Components with Random Forests». In: *European Conference on Computer Vision*. 2014 (cit. on p. 9).

[21] Thomas Berg, Jiongxin Liu, Seung Woo Lee, Michelle L Alexander, David W Jacobs, and Peter N Belhumeur. «Birdsnap: Large-scale fine-grained visual categorization of birds». In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2014, pp. 2011–2018 (cit. on p. 9).

[22] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. «3D Object Representations for Fine-Grained Categorization». In: *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*. Sydney, Australia, 2013 (cit. on pp. 9, 14).

[23] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. «Fine-grained visual classification of aircraft». In: *arXiv preprint arXiv:1306.5151* (2013) (cit. on pp. 9, 13).

[24] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. «Cats and dogs». In: *2012 IEEE conference on computer vision and pattern recognition*. IEEE. 2012, pp. 3498–3505 (cit. on pp. 9, 28, 34).

[25] Minyoung Huh, Pulkit Agrawal, and Alexei A. Efros. *What makes ImageNet good for transfer learning?* 2016. DOI: 10.48550/ARXIV.1608.08614. URL: https://arxiv.org/abs/1608.08614 (cit. on pp. 10, 12, 13).

[26] Saleh Alaliyat. «Video -based Fall Detection in Elderly's Houses». In: (June 2022) (cit. on p. 13).

[27] Chenyang Qiu and Wei Zhou. «A survey of recent advances in CNN-based fine-grained visual categorization». In: *2020 IEEE 20th International Conference on Communication Technology (ICCT)*. IEEE. 2020, pp. 1377–1384 (cit. on pp. 13, 14).

[28] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. «The caltech-ucsd birds-200-2011 dataset». In: (2011) (cit. on p. 13).

[29] Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Fei-Fei Li. «Novel dataset for fine-grained image categorization: Stanford dogs». In: *Proc. CVPR Workshop on Fine-Grained Visual Categorization (FGVC)*. Vol. 2. 1. Citeseer. 2011 (cit. on pp. 13, 25).

[30] 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 (cit. on pp. 14, 25, 34).

[31] Dulari Bhatt. *Active Contours – A Method for Image Segmentation in Computer Vision*. [Online; accessed 22-June-2022]. 2021. URL: https://www.analyticsvidhya.com/blog/2021/09/active-contours-a-method-for-image-segmentation-in-computer-vision/ (cit. on p. 15).

[32] Jay J Jiang and David W Conrath. «Semantic similarity based on corpus statistics and lexical taxonomy». In: *arXiv preprint cmp-lg/9709008* (1997) (cit. on p. 18).

[33] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. «How transferable are features in deep neural networks?» In: *Advances in neural information processing systems* 27 (2014) (cit. on p. 20).

[34] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. «The pascal visual object classes (voc) challenge». In: *International journal of computer vision* 88.2 (2010), pp. 303–338 (cit. on p. 20).

[35] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. «Places: A 10 million image database for scene recognition». In: *IEEE transactions on pattern analysis and machine intelligence* 40.6 (2017), pp. 1452–1464 (cit. on p. 20).

[36] 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*. 2017, pp. 843–852 (cit. on p. 21).

[37] Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. «The earth mover's distance as a metric for image retrieval». In: *International journal of computer vision* 40.2 (2000), pp. 99–121 (cit. on p. 21).

[38] Svetlozar T Rachev. «The Monge–Kantorovich mass transference problem and its stochastic applications». In: *Theory of Probability & Its Applications* 29.4 (1985), pp. 647–676 (cit. on p. 21).

[39] 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*. 2018, pp. 8769–8778 (cit. on p. 22).

[40] Oscar Day and Taghi M Khoshgoftaar. «A survey on heterogeneous transfer learning». In: *Journal of Big Data* 4.1 (2017), pp. 1–42 (cit. on p. 23).

[41] David D Lewis, Yiming Yang, Tony Russell-Rose, and Fan Li. «Rcv1: A new benchmark collection for text categorization research». In: *Journal of machine learning research* 5.Apr (2004), pp. 361–397 (cit. on p. 24).

[42] Ken Lang. «Newsweeder: Learning to filter netnews». In: *Proceedings of the Twelfth International Conference on Machine Learning*. 1995, pp. 331–339 (cit. on p. 24).

[43] Massih R Amini, Nicolas Usunier, and Cyril Goutte. «Learning from multiple partially observed views-an application to multilingual text categorization». In: *Advances in neural information processing systems* 22 (2009) (cit. on p. 24).

[44] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. «Deep residual learning for image recognition». In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778 (cit. on p. 25).

[45] Gregory Griffin, Alex Holub, and Pietro Perona. «Caltech-256 object category dataset». In: (2007) (cit. on p. 25).

[46] Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. «Supplementary Materials Learning Deep Features for Scene Recognition using Places Database». In: () (cit. on p. 25).

[47] Ariadna Quattoni and Antonio Torralba. «Recognizing indoor scenes». In: *2009 IEEE conference on computer vision and pattern recognition*. IEEE. 2009, pp. 413–420 (cit. on p. 25).

[48] Jiquan Ngiam, Daiyi Peng, Vijay Vasudevan, Simon Kornblith, Quoc V Le, and Ruoming Pang. «Domain adaptive transfer learning with specialist models». In: *arXiv preprint arXiv:1811.07056* (2018) (cit. on p. 26).

[49] 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*. 2016, pp. 2818–2826 (cit. on p. 27).

[50] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V Le. «Regularized evolution for image classifier architecture search». In: *Proceedings of the aaai conference on artificial intelligence*. Vol. 33. 01. 2019, pp. 4780–4789 (cit. on p. 27).

[51] Maungmaung Aprilpyone, Yuma Kinoshita, and Hitoshi Kiya. «Adversarial robustness by one bit double quantization for visual classification». In: *IEEE Access* 7 (2019), pp. 177932–177943 (cit. on pp. 27, 28, 31, 34).