Towards Improving Machine Learning Models for Predicting Human Estimates of Image Similarity

Florica Margaritescu

MInf Project (Part 2) Report
Master of Informatics
School of Informatics
University of Edinburgh

2024

Abstract

While advancements in machine learning have resulted in significant performance improvements for traditional vision tasks such as object detection and image classification, as well as numerous extensive studies exploring further improvement methods and emerging correlations between effectiveness and facets of the task, the task of emulating human judgements in the context of image similarity remains an open problem. This project aims to fill this gap and contribute a comprehensive guide on improvement techniques and their effectiveness on the task of predicting human estimates of image similarity. These techniques and their effectiveness are discussed in great detail in light of current recent work in the field of deep learning, as well as of the particularities of each dataset that they are applied to. The objective of the project is to shed light on the most effective techniques, whilst constantly providing indications on which conditions these techniques are best applied to.

Research Ethics Approval

This project was planned in accordance with the Informatics Research Ethics policy. It did not involve any aspects that required approval from the Informatics Research Ethics committee.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Florica Margaritescu*)

Acknowledgements

I would like to express my gratitude to my supervisor, Oisin Mac Aodha, who provided invaluable suggestions and insightful ideas throughout the entire process. Thank you very much for all of the feedback on my work and for the genuine interest you showed in the project.

Table of Contents

Chapter 1

Introduction

This chapter aims to give an high-level view of the task undertaken by this project, the motivation behind it, as well as contributions made as part of this project. It is to be noted that this project represents the second part of the MInf degree qualification and is a continuation of the project undertaken in the previous year of study [67]. Thus, last year's work will be summarised and the manner in which this project extends the past work will be explained.

1.1 Machine Learning Task Description

We refer to a "machine learning task" as a concept, representing a problem or objective, that we would like a complex mathematical model to address or emulate. In our case, we would like a mathematical model to emulate properties of human perception and vision by learning how to solve the problem of correctly choosing the most similar image, out of two image choices, to some given reference image. By a "correct choice", we mean a choice identical to that made by a human. We say that the mathematical model is given the task of **predicting human estimates of image similarity**. The discussion to follow concerns the collection and structure the datapoint set for the desired concept to be emulated, and how the model makes use of this set to assess whether it correctly aligns with human judgements.

1.1.1 Human Visual Similarity Judgements

The image datapoints that we will apply the mathematical model to are of triplet structure: we see one main image and two secondary images - the puzzle we need to solve is choosing which of the two secondary images are more similar to the main image. By human estimates of image similarity above, we mean the judgement from a human with regards to which secondary image is most similar to the main image. These judgements are collected by conducting a survey [85, 5] involving multiple human participants who are given a single main image and a number of secondary images and asking them to rank the most N, in our case $N = 2$, similar secondary images to the main image. Such as survey is depicted in Figure 1.1. Such a survey will result

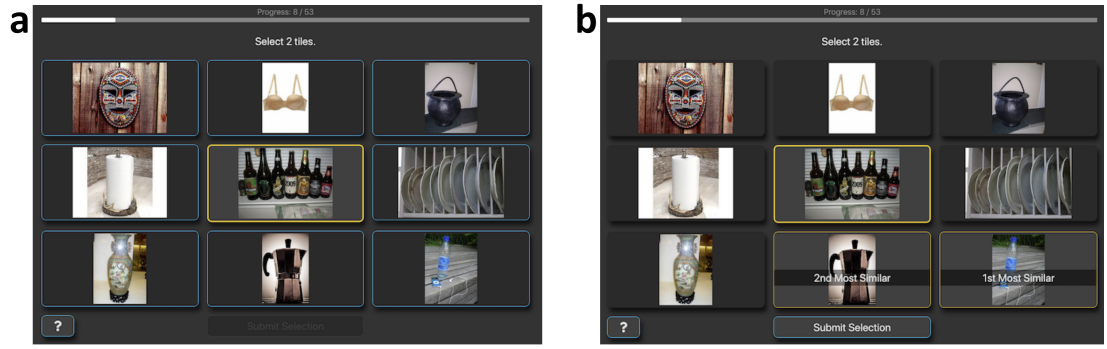

Figure 1.1: Example of the interface used in a human similarity judgement collection survey. (a) The participant is given 9 images, the highlighted middle image represents the main image, the rest are the image set. (b) The participant chooses (in order of their ranking) the most and second most similar image to the main image. Image and study credit: [85], also used in [67] - Chapter 1.

in a datapoint collection where each datapoint is a triplet of the following images: a main image - called the **query**, the first most similar secondary image to the query - **reference 1**, and the second most similar image to the query - **reference 2**.

1.1.2 Image Triplet Datasets of Human Judgements

Following the collection process described above, we end up with a number of triplet sets. The triplet sets used in this project have been collected as part of three separate studies [85, 109, 5]. The three sources of triplets to be used in this project are: a general Image-Net [20] based source spanning multiple categories [85], a birds species centric source presenting a manageable number of categories [5], and lastly, a vastly different dataset from the previous two which presents image distortion-based triplets [109]. A triplet example from each source is depicted in Figure 1.2. Chapter 4 details these datasets and each of their purposes. Each triplet contains a query image and two reference images.

1.1.3 Machine Learning Framework

We can denote our machine learning model, momentarily simplified to a complex mathematical function, as f, which takes as input variable x and produces output y: $f(x) = y$. In our case, x is an image in a triplet datapoint, and y is a vector representation of this image which is somehow generated by our function f. The way in which the function is applied to the triplet set is: given triplet datapoint $(x1, x2, x3)$, where $x1$ is the query and the other variables are the references, we apply the function f to each image in the triplet and obtain three output vectors: (y_1, y_2, y_3). We then calculate the distance between the vector output of the query image and of each reference image, the reference image whose vector is closest to the vector of the query image is considered the answer that our model gives to the question: "Which of the two references images is closest to the query image?". Figure 1.3 illustrates this framework.

We both evaluate and train the framework exemplified above. By evaluating, we mean

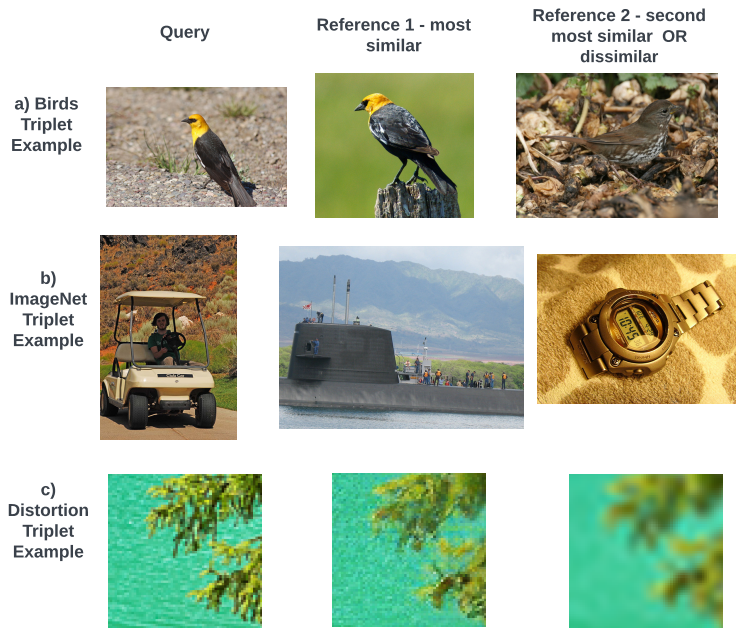

Figure 1.2: (a) An image triplet example from the bird species dataset [5]; (b) An image triplet example from the ImageNet-based dataset [85]; (c) An image triplet example from the distortion-based dataset [109].

that we check its predictions - the answers it gives to the aforementioned question- against the actual human judgements for that triplet. By training, we refer to the process of teaching the model to recognise the patterns and intricacies necessary to adjust its predictions to better adhere to the expected human judgements. The model adjusts its predictions by modifying its parameters.

1.2 Previous Work

As previously noted, this project is the second part of the MInf degree qualification and builds directly on top of the findings from the first part of the project [67]. The findings of the first part are detailed in Chapter 2. The first part of the project focused on drawing parallels between human perception and model predictions by evaluating model predictions on the three aforementioned datasets [5, 85, 109]. Human vision and perception properties and biases [84, 57, 5, 99, 28, 105, 65] were compared against experiments results at every step of evaluating model predictions. The purpose of drawing these parallels was to find the model architectures and frameworks which were best-suited for emulating human vision and perception and which model properties were correlated to better triplet similarity prediction (to exemplify, such a property was the number of parameters in a model).

The second part of the project focuses less on drawing parallels to human perception, and instead, more on the effectiveness of different **improvement techniques** on the triplet prediction task. The way in which the second part builds on top of the first part is by selecting the top performing architectures and fine-tuning (training on the specific triplet sets) paradigms found in the first part of the project, and take steps towards

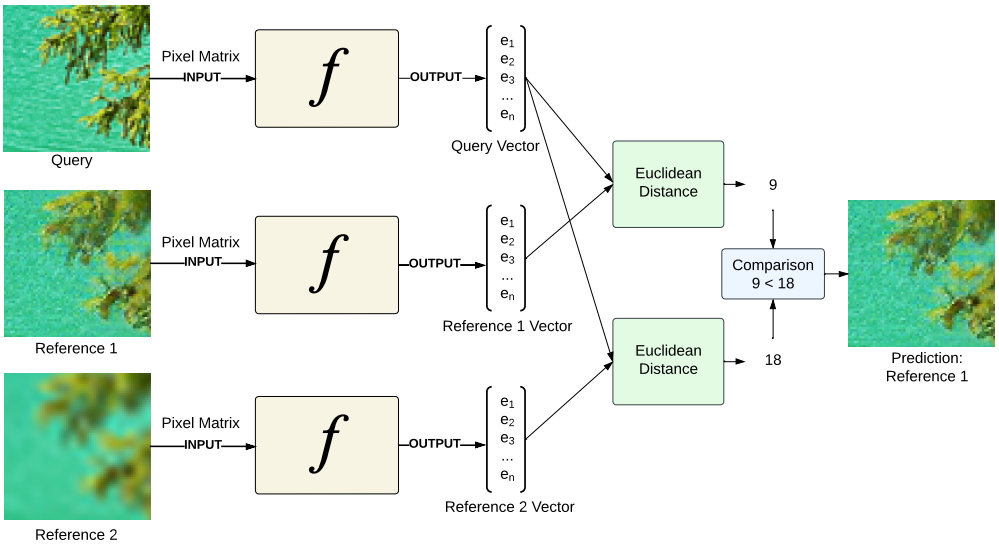

Figure 1.3: Depiction of the framework used to apply models of interest to a single triplet datapoint and obtain a prediction. In this instance, the prediction is correct as the model predicts the first reference which is also that chosen as most similar to the query image by human judgement. Image is inspired from: [67] - Chapter 1, Figure 1.2.

improving their performance using a variety of techniques proven to be effective by past studies.

1.3 Project Scope

The scope of the project is to explore various improvement techniques to be applied to the models undertaking the triplet similarity prediction task. The goal of these methods is to improve the accuracy of the models on the aforementioned task - by accuracy, we mean the percentage of correct similarity predictions. To do so, the intention of these techniques is to "encourage" the models to output vectors for images in such a way that we maximise the similarity between vectors of the query and the first reference, and minimise the similarity of the vectors of the query and the second reference. These techniques can be applied to various aspects of the framework shown in Figure 1.3: to the model itself [11, 12, 25, 102, 62, 104, 46] - namely f in our previous simplified discussion; to the similarity metric used to compare image output vectors [109, 25]; as well as to the way in which we train [53, 45, 2] our model framework on the triplet sets.

1.4 Motivation

The scope of the project - namely, to increase the accuracy of our model predictions on the task of predicting human estimates of image similarity - is motivated by the possibility to consequently emulate desirable human perception and vision properties [34, 85, 98, 96, 28, 5]. Not only, emulating human vision properties - whether desirable

or factual, such as the neutral fact that human vision is biased towards shape [84, 57] - has been shown to translate to increased accuracy on other image-based (vision) machine learning tasks [98] as well. To enumerate desirable properties of human vision:

- Ability to generalise better to new tasks and datasets [28] - respond more appropriately to new stimuli. Contextual understanding in humans [85] plays a great role in achieving this advantage - namely, visually dissimilar images may be connected due to the connection in meaning or scope of the objects that they represent;
- Proven robustness of human vision when faced with adversarial perturbations of the data [24]: for example, imperceptible perturbations to human vision can, oppositely, greatly change model predictions [34];
- Human vision allows to learn from very few examples - few-shot learning. This property is highly desirable for models as it would greatly reduce the amount of data required for training models, and not only, it would enhance the use of machine learning in fields where adapting to new stimuli is crucial - such as autonomous systems and rare event detection [82].

Not only, approaching model representations to human perception, and consequently, aligning model predictions to human values, is crucial in order to avoid unintended consequences and allow for ethical decision making [60] - especially applicable in nuanced domains such as criminal justice, for example [49].

1.5 Contributions

We see existing past work in the field of machine learning in vision tasks focusing on approaching and comparing neural network representations to human perception and vision [98, 27, 5, 29, 109, 85, 58, 87, 67], as well as on generally improving the performance of neural networks on a variety of vision tasks such as classification and object-detection [64, 81, 104, 77, 61, 70, 104, 46], however, current work in the field is yet to provide a comprehensive evaluation of improvement techniques applied to the task of triplet similarity prediction on a variety of triplet sources - this project is meant to fill in this gap. The main contributions of this project thus are:

- A comprehensive guide on improving triplet similarity task prediction on three human judgement datasets [85, 109, 5], 7 fine-tuning paradigms [7, 36, 16, 31, 73, 15, 39], 3 similarity metrics, and 8 base models (Table 3.1);
- An evaluation of meaningful patterns and correlations between improvement technique effectiveness and facets of the datasets - difficulty-related and structural factors;
- An evaluation of meaningful patterns and correlations between improvement technique effectiveness and facets of the backbone model - training-related and structural factors.

Chapter 2

Background

This chapter aims to give a high-level overview of the essential knowledge required for comprehending the project scope and findings. To understand the purpose of the project, we shall discuss: the concepts our machine learning models intend to emulate - the task they are evaluated on - namely, human vision and perception, and how the concepts differ; details on the models of choice - neural networks; as well as, related work and state-of-the-art in the field of aligning model predictions to human perception, and of improving machine learning predictions in general.

2.1 Human Vision and Perception

As discussed in the first part of the project [67], human vision and human perception, distinctive from each other [54, 95, 26], are both utilised in assessing image similarity. Whilst vision is concerned with processing the raw visual information (colours, shapes, textures, and so on), perception is concerned with obtaining a subjective image interpretation using complex cognitive processes (memory, contextual understanding, and so on) and the visual information. Estimating image similarity involves both concepts. We highlight this distinction to explain how, due to the involvement of complex cognitive processes in perception, machine learning models emulating the human visual system would not necessarily obtain near-human image similarity predictions.

Thus, this still remains an open problem due to the fact that human image similarity judgements are not only based on visual properties of the images [37, 109] such as Gestalt shape ([95] - Chapter 5, [84, 57]), for example, but also on our own individual experiences and subjectivity. To exemplify: although specific machine learning architectures have been heavily based on human vision [63] - namely Convolutional Neural Networks [75] - past work has highlighted different architectures - Transformers [100] - as better emulators [30, 98] of human vision and perception. Not only, the first part of the project [67] observed consistently better results when applying Transformer-based architectures to the given triplet prediction task detailed in Chapter 1.

2.1.1 Similarity Perception

Having discussed human vision and perception in general, we shall now discuss similarity perception for the purpose of better understanding the cognitive processes involved in the participants' similar image selection in the collection surveys introduced in Chapter 1 and not only, to better understand the underlying complexity of similarity judgements. Similarity perception concerns the cognitive process responsible for recognizing similarities between different concepts or objects.

Similarity, in the field of machine learning, may be considered from the simplistic categorical point of view [23] - in other words, we use as criterion for deeming two images as similar, whether the images belong to the same category (from a pre-defined set of categories). We may thus think about similarity perception from the perspective of category learning [4, 9]: naturally, all beings assign concepts to separate categories in order to respond to them differently. Human categorical learning studies [4, 9] in the field of cognitive psychology present various ways of explaining human category learning. The most relevant concept to our project being prototype distortion [41]. This theory suggests that categorical decisions in humans involve perceptual learning and similarity judgement processes, and categories are based on distorted versions of a prototype. Namely, to assign a concept - an image in our case - to a category, we compare this concept to our subjective version of the average member of the category. However, in the context of vision, our mental representation of this typical image in a category is in fact imperfect and different from the true central sample of that category - it is *distorted*. This distortion is a result of personal, social and cultural experiences, cognitive biases and so on [90]. An example of this is the way that individuals from different regions perceive geometric illusions, as explained in Figure 2.1.

However, it is to be noted that similarity is much more than image categorisation [101]. The concepts are correlated [93] - visual similarity and picture categorisation - however, not equivalent. The reason why we bring about the discussion of simplifying similarity judgements to categorical judgements, is to highlight that even the simplified version of the problem still involves great complexity due to the involvement of human perception and thus individuality and subjectivity. The correlation between categorisation and similarity will also be of use in latter section when discussing improvements to be

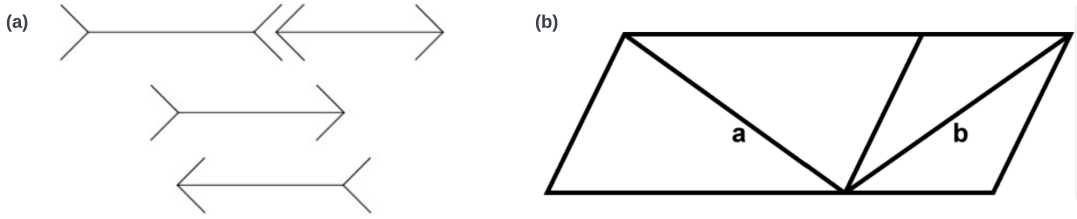

Figure 2.1: (a) The Müller-Lyer illusion [19] and the (b) Sander parallelogram illusion [89] were used in a study [90] where people from various countries were sampled. The study shows that culture does have an impact on illusion susceptibility: European and American participants being much more susceptible to the illusion compared to their non-western peers. Image credit: [90].

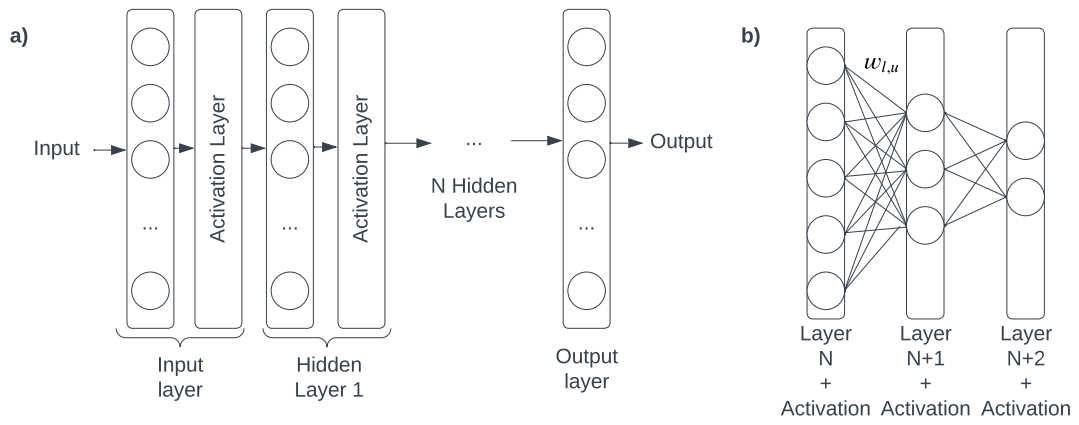

Figure 2.2: (a) A representation of a simple neural network architecture with activation layers (note: we often see activation layers tied to previous whose outputs they apply activations to). (b) A representation of a number of fully-connected layers - where each unit in a layer is connected to each unit in the following layer and each connection presents weight $w_{layer,unit}$.

applied to our current models: namely, applying experiments in which we improve categorisation predictions in order to positively influence similarity predictions.

2.2 Introduction to Machine Learning Models

This section provides an overview of the machine learning models of interest, as well as how they are applied in the context of our task. The information encompassed in this section is high-level and serves the purpose of offering enough information for understanding the discussion of past work from later sections in this chapter.

The specific machine learning models which are of interest to us are neural networks [33] due to the fact that they have been proven to be the state-of-the-art in various vision tasks [91, 21, 1]. Neural networks' effectiveness is explained by their ability to learn complex and high-dimensional data representations. They present the ability to learn intricate patterns from raw data ([33] - Introduction), due to their layered design which incrementaly capture more complex raw data representations. These machine learning models and the specific architectures used in the project are detailed in Chapter 3.

We can imagine these models as mathematical functions structured in a manner that is roughly based on the human neural system [55] - thus the name. They are made up of individual units, also called nodes, grouped into layers - Figure 2.2 (a). In the simplest case of a neural network, namely that of a feed-forward neural network ([33] - Chapter 6) - an *FNN*, the input is passed from the first layer (input layer) to the second layer and so on, until it reaches the final layer (output layer), which generates the output. Middle layers are called hidden layers. Each node from Layer N is connected to the nodes in Layer N+1, and each connection has a weight - we may think of this weight - Figure 2.2 (b) - as a coefficient which indicates the importance of the previous node to the following node. We also see activation layers - which are non-linear functions applied to Layer N outputs (multiplied by weight) to generate Layer N+1 inputs. These allow

the network to approximate any non-linear function [18].

The two neural network architectures used in this project and further detailed in Chapter 3 are: **Convolutional Neural Networks (CNN)**, designed for grid-like data such as images and based on the human vision system [44, 63]; and **Transformers** [100], initially designed for sequential data and later adapted to images [22] - these networks deploy attention mechanisms to weigh the importance of different input components. Despite being based on a non-vision intuition from CNNs, past work [98, 30] has shown that internal representations of Transformers may be better aligned to human image perception.

The project will use the terms ”features” and ”embeddings” interchangeably. These are matrix-like data structures obtained by sampling the output of hidden layers in our network - they are representations of the input data that make sense to the network, in the sense that they convey meaningful information, such as the presence of edges, for example. As the network progresses through its hidden layers, it learns increasingly more intricate representations of the input. The output from the final hidden layer serves as the feature representation for the input image. The way in which these models are deployed in the context of our similarity prediction task is shown at a high-level in Figure 1.3, where the image output vectors are in fact model embeddings, and detailed in Chapter 3.

2.2.1 Training Neural Networks

As part of our experiments aimed at improving model predictions, we will train some of the networks on image sets obtained from the three sources described in Section 1.1.2. Training a neural network means teaching it to perform a specific task, such as, for example, teaching it to get embeddings of similar images closer and those of dissimilar images farther apart, by adjusting the parameters of the neural network in an iterative manner, after inputting a number of triplets and assessing the performance of the model. This process is described in Chapter 3.

We may either train the models on the task that they will be evaluated on - namely, triplet similarity prediction - or, on a different vision task. Models can be trained on a different task to learn general image features - they thus gain some general knowledge which can be applied to our own task. This represents the concept of transfer learning [10] - the knowledge of the model gained from a task (such as classification training on a large dataset such as ImageNet [20]) can be transferred to a different, related, task (such as our triplet similarity prediction task).

2.3 Related Work

Following the sections which established the necessary foundational knowledge to understand past work, this section now aims to give an overview of how this project aligns with current and previous research on the topic of approaching machine learning models to human perception, and on the topic of exploring improvement methods for neural network predictions. Predicting human estimates of image similarity remains

an open problem [58, 85] due to the subjective and individualistic [90] nature of the expected outcome. This expected outcome cannot be reached by following logical rules (rule-based categorical learning [4, 9]) and may not be agreed upon by all (a "universal truth").

The past decades have seen significant progress in the domain of improving neural network predictions at various vision tasks [64, 81, 104, 77, 61, 70], of better aligning neural network predictions to human judgements [98, 27, 5, 29, 109, 85, 58], as well as in the domain of available human judgement datasets [85, 5, 109, 87]. The subsections to follow shall discuss past work undertaken in the field, of particular relevance to our experiments, in more detail.

2.3.1 MInf: Part I

Chapter 1 discusses the contributions of the first part of the project [67] at a very high-level, this section aims to provide more detail, as well as present the concrete results obtained in the first part [67]. To briefly summarise the contributions of the previous project: it provided an evaluation of a number of neural network architectures and fine-tuning frameworks on the three similarity judgement datasets [5, 85, 109]; and it highlighted correlations between certain model properties (such as accuracy on a different vision task, or the number of its hyper-parameters)/dataset properties (such as general similarity amongst images) and the triplet prediction accuracy achieved.
The first part of the project [67], Chapter 6, also predicted different paths to continue the work such as branching out and exploring the addition of language and mutimodality in the models we use to obtain the model features, as well as, varying the training loss or similarity metric (applied as indicated in Figure 1.3). Both of these improvement methods appear in the current work undertaken and will be detailed in the sections to follow.

2.3.2 Approaching Model Representations to Human Perception

Research work on aligning model properties to those of human perception [67] highlight a number of emerging vision properties: human visual perception presents a **shape bias** [84, 57] - we assign higher importance to shapes rather than other image properties (colours, textures and so on) - CNN models tend to be biased towards texture instead [27]; whilst human similarity perception is **asymmetric** [99, 5] (pair-wise similarity of images A and B may be different to that of image pair B and A). One study [27] has shown that by reducing the texture bias of CNN and allowing the model to re-focus on shape, image classification results on ImageNet [88] are improved. Similarly, it has been shown that by allowing models to make asymmetric similarity judgements, accuracy can improve [5].

These represent main differences between human and machine vision. The previous part of this project ([67] - Chapter 2) enumerates many more studies which delve into approaching model internal representations to human perception.

2.3.3 Large Vision Models

Due to the increased availability of training data sets - such as ImageNet-21k [83] in our case - and computing resources - especially GPU capabilities [42] - we see the popularisation of **large** machine learning models [35]. By large model, we may refer to a machine learning model with a large number of parameters, such as the GPT-3 model [11], or a model trained on a very large dataset, such as CLIP [79] and Distillation with No Labels - DINO [12]. The latter two are of interest to us.

The reason why we are interested in large vision models is due to the fact that, by being trained on very extensive datasets, we expect them to may be able to better capture intricacies of their training data and thus present superior performance on vision tasks when compared to regular neural network models [102]. Not only, we also expect them to generalise well [62] - in the sense, a large model trained on a different dataset from ours is expected to still perform well on our own dataset. CLIP and DINO are the two large vision models used in this project. Both represent training paradigms in which models of regular size are trained on very large datasets. To picture the extent of the training data: a typical PyTorch pre-trained CNN model is typically trained on 1.2 million datapoints (from ImageNet-1k ILSVRC 2012 [88] - a subset of the full ImageNet dataset [20] [1]), oppositely, a CLIP CNN is trained on 400 million datapoints [79] and a DINO CNN on 14 million datapoints (on the whole dataset of ImageNet [20]).

Another reason why CLIP is of great interest to us is due to its multimodality: whilst DINO is trained on images only, CLIP is trained simultaneously on both images and text. Namely, the task that CLIP is trained on is matching images to their corresponding text description. CLIP's multimodal approach has proven to improve predictions on vision and language tasks alike [13, 59, 8] due to its enhanced contextual understanding stemming from the model analysing two different perspectives (visual and textual) and also, from CLIP's ability to learn more fine-grained visual concepts due to the use of language in its training. Other studies [104, 46] also highlight the effectiveness of using language-based models for extracting image features in vision tasks.

2.3.4 Varying Similarity Metrics

The effectiveness of different similarity metrics has also been highlighted in previous work [109]. We use similarity metrics to compare the resulting image features from our models - namely, to calculate how similar vectors are to one another - the manner in which we use similarity metrics is described in Chapter 1. The previous part of the project [67] made use of cosine similarity to compare feature vectors. **Cosine similarity** is used by a number of other studies to obtain similarity between image feature vectors [106, 72, 109, 85] in a variety of tasks. The measure only takes into account the angle formed by two vectors. Euclidean distance, also called **L2 distance**, is also used in vision tasks [109, 66] and is a much more intuitive similarity metric between feature vectors as it measures the actual straight-line distance. It is also the

[1] ImageNet-1k contains only on the 1k high-level categories, rather the approximately 22k original categories of ImageNet. Usually studies are conducted and models pre-trained using ImageNet-1k.

similarity measure of choice for Learned Perceptual Patch Similarity - LPIPS [109]. LPIPS is a popular image similarity metric which measures the similarity between two images by first obtaining their feature vectors from some pre-trained neural network and then comparing these features using the L2 distance - this framework is in fact what inspired the framework we shall be using and which is depicted in Chapter 1. Lastly, we see a number of studies[107, 66] making use of Manhattan distance, or **L1 distance**, to obtain image embedding similarity measures. Alike L2 distance, L1 distance is also more intuitive than cosine similarity, however, L1 distance may provide additional robustness to outliers compared to the L2 distance [48].

Using a similar architecture to LPIPS, DreamSim [25] claims to align much better to human judgements than LPIPS by using a concatenation of image features obtained from pre-trained large vision models such as CLIP and DINO. DreamSim thus inspired our exploration of a LPIPS-adjacent architectures which utilises features from large vision models in our experiments.

2.3.5 Varying Fine-tuning

Fine-tuning refers to training an already pre-trained neural network on our own data - namely, on images from the sources discussed in Section 1.1.2. The manner in which we fine-tune a model can vary greatly, as well as the model's final performance, by varying the fine-tuning parameters: such as the loss - a measure of how well the actual outcome meets the expected outcome, or vision task that we use for training (classification, object detection, triplet similarity prediction, and so on). A number of studies [53, 45, 2] explore the effectiveness of different training variations on vision tasks and inspire our choice of training variations to explore in later chapters.

Not only, other studies [53, 52] have also compared a number of losses used to train models for classification tasks on the ImageNet [20] dataset: the study not only compares the prediction accuracy obtained after training models with different losses, however it also explores how different losses lead to different transferability of features. More specifically, the study deduced that losses which yield good accuracy on the task they were trained for, namely image classification on ImageNet, have a decreased performance on different vision tasks - less transfer learning [10] power. To put this discussion in the light of our triplet similarity prediction task: a model which is pre-trained on ImageNet and yields great classification accuracy on it, may not also yield great accuracy on our own triplet prediction task. This finding has also been shown as part of the first part of the project [67] - Chapter 5, Section 5.1.2: there is no evident correlation between the accuracy a pre-trained model achieved on ImageNet classification and that model's resulting accuracy on our task of triplet similarity prediction.

Chapter 3

Methodology

The purpose of this chapter is to offer a more detailed overview of the methods involved in the project's experiments. We shall first discuss the machine learning models of choice, namely neural networks, the specific architectures to be used in our projects, as well as concrete training paradigms. It is to be noted that this information is also available in the first part of this project [67], as well as in a number of official published resources [33], and it will thus not be covered to the same extent as in last year's methodology chapter [67] in order to instead prioritise offering more valuable information on experiment frameworks. Following the discussion on neural networks, the chapter then delves into the specific variations to be used in experiments of the general model architectures discussed previously, as well as the structure of the frameworks in which these models are used.

3.1 Neural Networks

Whilst neural networks, and more specifically feed-forward neural networks (FNN), have been described at a very high-level in Chapter 1, this section aims to offer more detail into their inner workings in order to understand the way in which the improvements, the scope of the project, will be applied. We shall first elaborate on manner in which an FNN works, to provide the necessary foundational knowledge for discussing the more intricate architectures to be used in the project.

In an FNN, each node in a layer is connected to all nodes in adjacent layers as depicted in Figure 2.2 (b) in Chapter 1, and each node-to-node connection is weighted by $w_{l,u}$ where l is the layer and u is the unit number of the start end of the connection. A non-linear activation function is found between layers of fully-connected nodes as depicted in Figure 2.2 (a). Given an input x to a node, its output to the next node is the input, multiplied by the weight, plus a bias $b_{l,u}$: $x \times w_{l,u} + b_{l,u}$ - to which we then apply the activation function. The weights and biases - the model parameters - are gradually adjusted during training to better match expected outputs - they are "learnt". To represent this calculation on a per-layer basis, for a layer N we have parameter matrices: $\mathbf{W}_N$ of size $I \times O$ where I is the number of nodes of layer $N-1$ and O of $N+1$ respectively; layer N bias vector $\mathbf{b}_N$ of size $O \times 1$ where O is the size of layer

$N+1$. The element-wise activation function is denoted by ϕ_N. The output of layer N, denoted $\mathbf{z}_N$, when given input vector $\mathbf{x}$ can be seen in Equation 3.1.

$$\mathbf{z}_N = \phi_N(\mathbf{W}_N\mathbf{x} + \mathbf{b}_N) \tag{3.1}$$

3.1.1 Convolutional Neural Networks

To re-iterate the introductory information presented in Chapter 1, Convolutional Neural Networks (CNN) [44, 63] have been designed with grid-data in mind and have become the de facto standard for a number of computer vision tasks with the emergence of CNN architectures such as AlexNet [56]. CNNs leverage hierarchical features and spatial relations within images. This concept is inspired by the hierarchical processing observed in the visual cortex of animals [44]. Earlier layers extract low-level features from raw pixel data - simple patterns such as edges and shapes, while deeper layers produce higher-level features of a more abstract and non-interpretable nature.

CNN's architecture is more intricate than that of an FNN as it contains convolutional layers, pooling layers, fully-connected layers, as well as activation functions which can be inserted after any layer. A convolutional layer applies filters or kernels - matrices designed to capture patterns, also representing the learnable weights of a convolutional layer - to the input data and generates a set of feature maps, representing extracted input features - as output. Each kernel, typically smaller in dimension than the input, is slid over the input and the dot product between the input patch where the filter is currently applied is calculated and inserted into the output feature map. This matrix-based convolution is usually implemented as cross-correlation - $\bigotimes$. Given a layer N and an input (image or feature map) with a single channel $\mathbf{x}$ of size $i \times j$, a single layer kernel $\mathbf{w}$ of size $m \times n$, a bias term $\mathbf{b}$ and an activation function ϕ_N, the convolution operation formula can be seen in Equation 3.2. Generally, the input has multiple channels - an RGB image has three colour channels for example - and a layer has multiple kernels. This dimensionality makes convolutions expensive and pooling layers, which downsample spatial dimensions by aggregating - via a maximum or average function - nearby values in a feature map, are introduced.

$$\mathbf{z}_N(i,j) = \phi_N((\mathbf{w}\bigotimes\mathbf{x})(i,j) + \mathbf{b}) = \phi_N(\Sigma_{k=0}^{m-1}\Sigma_{I=0}^{n-1}x_{i+k,j+I}w_{k,I} + \mathbf{b}) \tag{3.2}$$

A CNN generally employs a sequence of convolutional layers to extract features of an image, followed by a series of fully-connected layers which use these features to fulfil a task, such as predicting whether or not the image contains a certain object. The first part of the project [67] convolutions and downsampling in detail in Chapter 4.

3.1.2 Transformers

Also introduced in Chapter 1, Transformers [100] have been designed with language sequences in mind and became the de facto standard for language tasks, however, a recent contribution called Vision Transformer (ViT) [22] adapted the network to be used in vision tasks, where it has sometimes surpassed the performance of CNNs. The image is processed as a sequence by being divided in equal patches and fed to the ViT network as a sequence of patches.

ViT presents components - each made up of several layers - such as: patch embedding, feature extraction and a classification head. The patch embedding component takes an input image $\mathbf{x}$ of dimension $H \times W \times C$, breaks it down into M equal patches of dimensions $p \times p \times c$, and outputs the vector representation of each image patch. Each vector is of size $1 \times (p^2 * c)$. This vector sequence is then multiplied by a learnable vector $\mathbf{E}$ of shape $(p^2 * c) \times d$ - where we choose d - to output M embedded patches of size $1 \times d$. A vector $\mathbf{x}_{class}$ of dimension $1 \times d$ which represents an aggregate of all patch vectors is then added at the beginning of the sequence. A positional learnable matrix $\mathbf{E}_{pos}$ of size $(M+1) \times d$ is added to the concatenated M patch vectors to output the final concatenated patch embedding component z_0. This process is explained by Equation 3.3.

$$\mathbf{z}_0 = \left[\mathbf{x}_{class}; \mathbf{x}_p^1\mathbf{E}; \mathbf{x}_p^2\mathbf{E}; ...; \mathbf{x}_p^M\mathbf{E}\right] + \mathbf{E}_{pos},\ \mathbf{E} \in \mathbb{R}^{(p^2 \cdot C) \times d}, \mathbf{E}_{pos} \in \mathbb{R}^{(M+1) \times d} \tag{3.3}$$

The feature extraction component takes embeddings $\mathbf{z}_0$ and extracts significant features using a "stack" of encoders. These encoders contain the attention mechanism denoted as *MSA*, which aids in "focusing" on the relevant parts of an input, a 2-layer FNN and normalization layers *LN* to scale inputs to reduce training time [6]. The propagation formula for the feature extraction component can be seen in Equation 3.4. The classification component, a simple FNN, is then applied to the last resulting output of the feature extractor for the $\mathbf{x}_{class}$ token. Equation 3.5 shows this last step. Appendix **??**, Figure **??**, depicts this process.

$$\mathbf{z}_l = MLP(LN(\mathbf{z}'_l)) + \mathbf{z}'_l \text{ where } \mathbf{z}'_l = MSA(LN(\mathbf{z}_{l-1})) + \mathbf{z}_{l-1} \text{ and } l = 1...L \tag{3.4}$$

$$\mathbf{y} = LN(\mathbf{z}_L^0) \tag{3.5}$$

3.2 Training Paradigms

Following the sections explaining different neural network architectures and the way input data propagates through each architecture type, we shall now discuss how to make these models "learn" the expected mapping between an input and an output by adjusting their parameters. This process is called training a neural network and has been briefly introduced in Chapter 1.

Training is performed by inputting a number of D training datapoints $\mathbf{x}_i$ where $i \in [0...D]$, each with expected corresponding output $\hat{\mathbf{y}}_i$, into our model to obtain actual output $\mathbf{y}_i$. A **loss function** ([33] - Chapter 8), which can take on a variety of forms, measures how close the actual output is to the expected output. This measure is then used to guide the model to update its weights in order to yield better predictions. For a classification task of the form: given an image of a training set with D datapoints, assign it to one of C classes, where the predicted probability of datapoint i belonging to class c, and $y(i,c)$ being the true label - 1 if the datapoint belongs to class c, 0 if not - we would make use of **cross-entropy** Loss. The loss is calculated by summing over all classes and penalizing the difference between the predicted probabilities and the actual class labels for each datapoint and across all classes. The formula can be seen in Equation 3.6

$$L = -\frac{1}{D}\sum_{i=1}^{D}\sum_{c=1}^{C} y(i,c)\log(\hat{y}(i,c)) \tag{3.6}$$

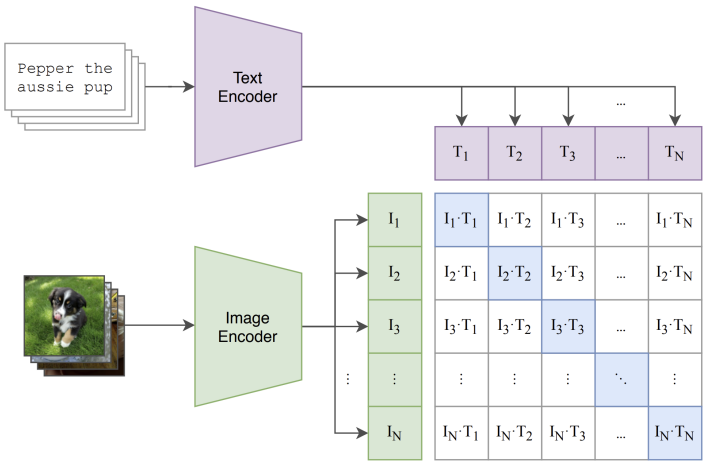

Figure 3.1: Depiction of how CLIP uses an image enconder (our chosen backbone) and a text enconder to obtain the features for a batch of datapoints and compute the pair-wise similarity between each pair of resulting feature vectors from images and text. Image credit: [79].

Backpropagation then occurs, which is a process in which the weights of the neural network are updated such that the loss function is minimised. The method involves calculating, from the output to the input layer, the gradient of the loss function with respect to each network parameter, using the chain rule. These gradients indicate the magnitude and the direction of change for the parameters to reduce the loss. Once gradients are obtained, a function called the **optimizer** updates the parameters using the gradients and a hyper-parameter named the learning rate α, which dictates the size of steps taken towards minimising the loss. This process is repeated iteratively, an epoch representing a full pass over the training dataset, until loss is minimised to a desired level.

Our optimizers of choice are Stochastic Gradient Descent [50, 33] (SGD) and Adaptive Moment Estimation [51] (Adam). There is no "best" optimizer for a given task or model, finding an optimal optimizer is an experimental process. SGD updates the model parameters by moving in the opposite direction of the gradient of the loss function and using a subset of the training data for each step. Adam presents a more intricate method that does not only consider the average of the past gradients from a subset alike SGD, but also adapts the learning rates for each parameter individually, using estimates of the first two averages of gradients. Adam is typically used for complex datasets and tasks as it may be prone to overfitting on smaller datasets ([33] - Chapter 5). A much more in-depth explanation of optimizers can be found in the first part of the project [67], Chapter 4. The information was not repeated due to its availability in other resources, as well as the fact that our experiments do not involve experimenting with optimizer choices.

3.2.1 Supervised Training

In supervised training ([33] - Chapter 5.7), we are aware of the correct input-output mapping. For example, in the case of our triplet sets depicted in Chapter 1, Figure 1.2, we know the correct answer to the question which represents the machine learning

model task: "Which of the two reference images are more similar to the query image?". We are aware that the first reference is more similar to the query. We will discuss the supervised losses used in model pre-training (models we choose come already trained on some task and present some foundational "knowledge"), and model fine-tuning.

Model Pre-training Loss: Cross-Entropy Loss is used in model pre-training for all non-large vision models we shall use in our experiments. These models are pre-trained on ImageNet [20], on a classification (matching the class to the image) task, using Cross-Entropy Loss. On the other hand, CLIP's supervised training paradigm is much more intricate and involves the use of text labels as mentioned in Chapter 1. CLIP trains its models, regular CNN or ViT architectures, on a vast dataset of 400 million image-text pairs found across the Internet, from various sources rather than ImageNet alone. The machine learning task given is to match the image with the correct text description. During training, a feature vector is generated for the image and one for the text, CLIP then uses a variant of **contrastive** loss (explained later in this section) such that the model trains to minimise the distance between feature vectors of correct image-text pairs and maximise it between mismatched pairs. CLIP uses the Adam optimizer, which is a typical choice for large datasets. To explain CLIP's training process for CLIP model using a specific CNN backbone: CLIP uses two models, one to get the feature vector of the image (the CNN backbone) and one to get the feature vector of the text label (some set transformer-based model). CLIP processes batches of image-text pairs and it computes the feature vectors of the image and text in each pair per batch. It then calculates the similarity between every image and text in the batch - this process is depicted in Figure 3.1. Contrastive loss is then applied, with the intent to increase the similarity measure between correct pairs and decrease the distance between mismatches. Both the image and the text enconder are trained simultaneously.

Model Fine-tuning Loss: In our experiments, we use three different supervised losses: **triplet** loss [7], **contrastive** loss [36], and **quadruplet** loss [16].

Triplet loss is the most intuitive choice due to the structure of the datapoints in our task: image triplets. After we obtain feature vectors for each image in our triplet, the query and two references, we input them into the triplet loss function as: $\mathbf{a}$ - the query feature vector, $\mathbf{p}$ - reference 1 vector which is most similar to the query (positive example), $\mathbf{n}$ - reference 2 vector which is less similar/dissimilar to the query (negative example). The function aims to minimise the distance between the vectors $\mathbf{a}$ and $\mathbf{p}$ of similar images, and maximise that between vectors $\mathbf{a}$ and $\mathbf{n}$ - of dissimilar images. We make use of simplistic Euclidean distance in the formula below: $|\mathbf{a}-\mathbf{p}|^2$ being the Euclidean distance between $\mathbf{a}$ and $\mathbf{p}$. An arbitrary margin hyperparameter is used to indicate the minimum required separation between positives and negatives. The triplet loss function is shown in Equation 3.7.

$$\text{Triplet Loss}(\mathbf{a},\mathbf{p},\mathbf{n}) = \max(|\mathbf{a}-\mathbf{p}|^2 - |\mathbf{a}-\mathbf{n}|^2 + \text{margin}, 0) \tag{3.7}$$

Contrastive loss instead takes pairs of images, rather than triplets, as its input. Its goal is similarly to minimise the distance between the query $\mathbf{a}$ and a positive example $\mathbf{p}$ feature vectors, and maximise that between the query $\mathbf{a}$ and a negative example $\mathbf{n}$ feature vector. Given an image triplet with resulting feature vectors $(\mathbf{a},\mathbf{p},\mathbf{n})$, we make use of contrastive loss in the following way: we first give the loss function pair $(\mathbf{a},\mathbf{p})$

for which it adapts the weights to minimise the distance using the Adam optimiser; we then give it pair $(\mathbf{a},\mathbf{n})$ to perform the opposite action.The function can be seen in Equation 3.8. $T = 0$ if we look at similar pair (A,P) and 1 otherwise.

$$\text{Contrastive Loss}(\mathbf{a},\mathbf{p},\mathbf{n}) = (1-T)\frac{1}{2}(|\mathbf{a}-\mathbf{p}|^2)^2 + T\frac{1}{2}\max(\text{margin} - |\mathbf{a}-\mathbf{n}|^2, 0)^2 \tag{3.8}$$

Quadruplet loss can be considered an extension of triplet loss where we consider instead quadruplet $(\mathbf{a},\mathbf{p},\mathbf{n1},\mathbf{n2})$ as the query, positive and respectively two negative image feature vectors. We thus modify our training datasets for each source so that their datapoints are quadruplets where the first reference is similar to the query, and the last two references are dissimilar/less similar - this process is detailed in Chapter 4. Equation 3.9 shows the process. The formula uses two margins, margin_1 indicates minimum separation between positive pairs margin_2 between negative pairs.

$$\begin{aligned}\text{Quadruplet Loss}(\mathbf{a},\mathbf{p},\mathbf{n1},\mathbf{n2}) = \max(|\mathbf{a}-\mathbf{p}|^2 - |\mathbf{a}-\mathbf{n1}|^2 + \text{margin}_1, 0) + \\ \max(|\mathbf{a}-\mathbf{p}|^2 - |\mathbf{p}-\mathbf{n2}|^2 + \text{margin}_2, 0)\end{aligned} \tag{3.9}$$

3.2.2 Unsupervised Training

Opposite to supervised learning, in unsupervised learning ([33] - Chapter 5.8), we do not know the correct input-output mapping. Self-supervised learning is used both to pre-train models, as well as fine-tune them. Self-supervision is a sub-type of unsupervision where the model creates its own supervisory signal from the available data - namely, it creates its own input-output mappings that it considers to be correct. The network essentially creates a dummy task using the available input data: this is called a pre-text task. A very simple example may be: given an input datapoint, we apply two transformations to it $T1$ and $T2$, to obtain two datapoints. We give the model these two datapoints as input, one by one, and give the model the task of asigning each input to a class of transformations, either $T1$ or $T2$.

Model Pre-training Loss: The only pre-trained models using a self-supervised paradigm are the backbones of DINO. Despite the lack of supervision, DINO matches the performance of a number of supervised state-of-the-art networks [12]. As mentioned in Chapter 1, DINO is a large vision model and trained on the 14 million unlabelled images from the complete ImageNet dataset [20]. Each datapoint is a single image in DINO's training. The self-supervision method used is novel and involves training two separate networks called a student and a teacher. Each image datapoint is transformed to obtain two images which are different augmented views of the original image. The two distinct augmented view images are given to the two networks and the feature vectors are obtained. The task of the student network is then to match its output feature vectors as closely as possible to that of the teacher network. DINO uses, unusually, cross-entropy loss to measure the similarity between the student and the teacher predictions. We discussed cross-entropy loss in the context of classification tasks, however, DINO makes use of it by treating the teacher's feature vectors as "pseudo-labels" that must be matched by the feature vectors originating from the student model. The student parameters are updated through backpropagation, whilst the teacher weights are updated as an exponential moving average of the student model's parameters in order to allow

for a slower and smoother evolution. When using a DINO model, it is the student network's features that we actually make use of.

Model Fine-tuning Loss: To fine-tune pre-trained models on the set of unique images that make up the triplet sets for one of the three sources mention in Chapter 1 - the process of obtaining these training sets is detailed in Chapter 4 - we use the pre-text tasks: **rotation prediction** [31], **Jigsaw puzzle** [73], **SimCLR** [15], **MoCo** [39]. We shall explain how each of these will be applied to our training datasets detailed in Chapter 4 - keeping in mind that each datapoint is now a single image.
Rotation prediction works by taking the aforementioned input image and applying a random rotation to it by some degree in a list of fixed degrees: for example, we can apply either rotation of [0, 90, 180, 270] - the model randomly picks to apply a 270 degree rotation. The rotation options are essentially classes and the task that we give the model is to predict the class of the rotation using the input image - the expected output is 270. Cross-entropy is used as a loss function as this is essentially a 4-way classification task. The reason for choosing rotation prediction is due to its intuitive nature and simplicity, yet effectiveness as it forces the model to understand the orientation of object and consequently enhance spatial understanding.
Jigsaw puzzle is less simplistic and found to outperform a number [109] of self-supervised learning paradigms used in a similar triplet prediction context to ours when analysed as part of the BAPPS dataset study [109]. The input image is first divided into a number of titles or patches (in grid-like 9 x 9 division, for example, alike a puzzle) and the order of the titles is shuffled - using a random permutation option from a fixed list of N possible permutations (this fixed list of possible permutations will become our classes). The shuffled titles are then given as input to the network. This will be treated as a classification problem where the model, given as input an image of permutated equal patches, choose the correct permutation from a fixed list of permutations. The supervision method makes use of cross-entropy loss for a N-way classification problem.
SimCLR is closer in intuition to the supervised tasks applied to the triplet datasets as it equally involves bringing image embeddings of similar images closer and those of disimilar images farther apart. We thus deploy the method in the interest of observing whether a logical link between the pre-text task and the actual task the trained model will be applied to increases performance. SimCLR was deemed state-of-the-art by its creators [15] at the time of its creation, surpassing other self-supervised training paradigms tested on ImageNet and using ResNet [38] backbones. The way in which it works is by taking a single image and creating two different augmentations for it. The two augmentations, different images in themselves, are then passed to the network to obtain the feature vectors of each. The task given to the model is to bring augmentations of the same image closer in distance, whilst distancing augmentations of different images. Training is achieved using contrastive loss.
MoCo (its most recent version) surpasses the performance of SimCLR on ImageNet classification tasks using a ResNet [38] backbone. Identically to SimCLR, given an image, we perform two augmentations of it and feed it into the network. The task is again similar to SimCLR, involving distinguishing between pairs of augmentations originating from the same image or different images, using contrastive loss. The difference, however, is that MoCo has a queueing mechanism which keeps track of the negative image samples (dissimilar) it has seen for a batch. This allows to have a variety

of negative examples on hand for comparison: for an iteration, we bring augmentations of the same image closer, and we distance the augmentations of the current image and the images in the queue.

3.3 Similarity Metrics

Lastly, before delving into the concrete frameworks used in the project experiments, we detail the last remaining component: the similarity metric. As depicted in Chapter 1, Figure 1.3, a similarity metric takes the feature vectors and determines the similarity of the image pairs. Three different similarity metrics have been chosen according to their proven effectiveness in past work in indicating image embedding similarity: cosine similarity [106, 72, 109, 85], Euclidean distance [109, 66] - L2, as well as Manhattan distance - L1 [107, 66].

Cosine similarity is invariant to vector scale, only taking into the account the angle that they form, and it is thus of interest when the magnitude of the vectors is not of interest - as it is in our case as all feature vectors to be compared will have the same number of components. Given two feature vectors $\mathbf{a}$ and $\mathbf{b}$, which are outputs of some pre-trained feature extractor applied to two arbitrary images, with n as the dimension of the feature vectors, and $\mathbf{a}_i$ and $\mathbf{b}_i$ as the i-th components of the feature vectors, the formula for calculating the cosine similarity between the two vectors is detailed in Equation 3.10. It is to be noted that the greater the cosine similarity, the more similar we deem the features to be.

$$\text{Cosine Similarity}(\mathbf{a},\mathbf{b}) = \frac{\mathbf{a}\cdot\mathbf{b}}{||\mathbf{a}||_2 \cdot ||\mathbf{b}||_2} = \frac{\sum_{i=1}^{d} \mathbf{a}_i \cdot \mathbf{b}_i}{\sqrt{\sum_{i=1}^{d} \mathbf{a}_i^2} \cdot \sqrt{\sum_{i=1}^{d} \mathbf{b}_i^2}} \tag{3.10}$$

On the other hand, L2 distance is a much more intuitive similarity metric between feature vectors as it measures the actual straight-line distance. This is also the metric of choice for LPIPS models [109]. It is widely used in classification tasks - we may thus imagine that image features from a model which was pre-trained in a supervised way using classification may benefit more from the use of the L2 distance - detailed in Equation 3.11. Oppositely to cosine similarity, the greater the distance, the more dissimilar we deem the features to be.

$$\text{L2 Distance}(\mathbf{a},\mathbf{b}) = \sqrt{\sum_{i=1}^{d} (\mathbf{a}_i - \mathbf{b}_i)^2} \tag{3.11}$$

Alike L2 distance, L1 distance is also more intuitive than cosine similarity. However, L1 distance may provide additional robustness to outliers compared to the L2 distance [48] due to summing the absolute difference in each dimension. The formula can be found in Equation 3.12. Similarly to the L2 distance, greater L1 distance signifies greater dissimilarity.

$$\text{L1 Distance}(\mathbf{a},\mathbf{b}) = \sum_{i=1}^{d} |\mathbf{a}_i - \mathbf{b}_i| \tag{3.12}$$

3.4 Model Frameworks

Following our discussion of the separate components making up the frameworks to be used in project experiments, we shall now discuss the concrete structure of the frameworks and how they are applied in the context of evaluation and fine-tuning.

Firstly, we shall define the separation between a model's feature extractor [78] and classification block: a neural network is typically split into a feature extractor - which obtains the feature vectors of the input data - and a classification component which is typically a linear block (a sequence of fully-connected layers). The classification component takes the features and uses them to make predictions for a given task: be it assigning these features to categories or embedding the features in a smaller dimsenional space. The feature extractor is of main interest to us as it is the component which captures the relevant patterns, defining of a model's inner representations, of the input data. Typically, the feature extractor is made up of convolutional, downsampling and residual connection [38] (allows us to skip passing information over layers to mitigate the vanishing gradient problem [40]) layers for CNN architectures; and of the self-attention mechanism described in Section 3.1.2 for ViT architectures. Figure 3.2 depicts this separation using a simplistic CNN-like architecture.

Following the above foundational knowledge of the feature extractor/classification component split, we can now discuss the concrete variations of CNN and ViT architectures to be used in this project. All of the architectures have been pre-trained extensively on image datasets on an array of tasks. We only make use of the feature extraction component of each pre-trained model: we evaluate the feature extractor on the triplet sets exemplified in Chapter 1, as well as fine-tune these feature extractors using either triplet or single image sets - explained later in this chapter.

3.4.1 Pre-trained Models

This section aims to discuss the concrete CNN and ViT variations to be used in experiments. These models have been discussed extensively as part of the first part of project [67] - Chapter 4, Section 4.2.2, however, will be briefly summarised in this text. Firstly, we choose the Resnet [38] architecture, a CNN, which pioneered the use of residual connections to mitigate the vanishing gradient problem, due to its proven top perfor-

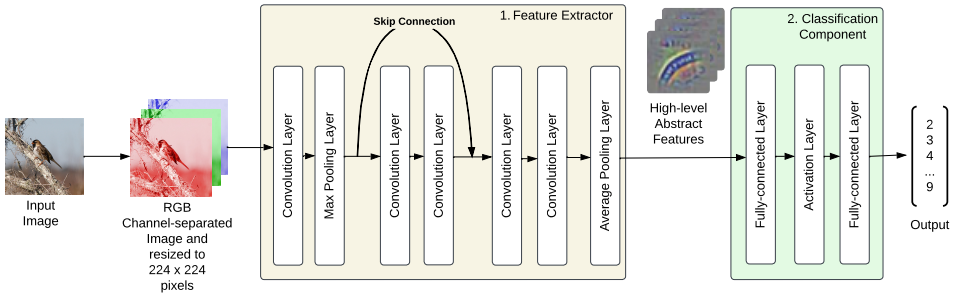

Figure 3.2: Illustration of the separation into a feature extractor and classification component of a CNN network. It is to be noted that images are processed as tensors by the network: that is, three - per colour channel in RGB - matrices of values. Image inspiration credit: [67].

Model ID	Pre-training Supervision	Pre-training Task	Pre-training Loss	Pre-training Dataset
ResNet-50_BASE	Supervised	Classification	Cross-Entropy	ImageNet-1k [88]
EfficientNetV2-M_BASE	Supervised	Classification	Cross-Entropy	ImageNet-1k [88]
ViT-B-16_BASE	Supervised	Classification	Cross-Entropy	ImageNet-1k [88]
AlexNet_BASE	Supervised	Classification	Cross-Entropy	ImageNet-1k [88]
CLIP_ViT-B-16_BASE	Supervised	Image-Text Map	Contrastive	Custom [79]
CLIP_ResNet-50_BASE	Supervised	Image-Text Map	Contrastive	Custom [79]
DINO_ViT-B-16_BASE	Unsupervised	Student-Teacher	Contrastive	ImageNet [20]
DINO_ResNet-50_BASE	Unsupervised	Student-Teacher	Contrastive	ImageNet [20]
CLIP_ResNet-50x4	Supervised	Image-Text Map	Contrastive	Custom [79]
CLIP_ResNet-50x16	Supervised	Image-Text Map	Contrastive	Custom [79]
CLIP_ResNet-50x64	Supervised	Image-Text Map	Contrastive	Custom [79]

Table 3.1: Pre-trained feature extractors to be used in experiments. Only the feature extractors with "_BASE" in their identification will be fine-tuned.

mance on the ImageNet-HSJ [85] dataset on image similarity estimation specifically. Another CNN of choice is EfficientNet [97] due to its distinct scaling procedures that allows efficient computation despite the depth of the network. The last CNN of choice is AlexNet [56], which, although the oldest and simplest - presenting only 8 layers - of the two CNN-architectures, it still manages to almost match the performance of many deeper CNN architectures. Lastly, we make use of the transformer architecture ViT [22] due to its proven efficiency on mirroring human perception [98, 29, 27] and on vision tasks in general, matching that of state-of-the-art CNN architectures.

The specific model variations used in this project have been chosen based on the top-performing model feature extractors from the first part of the project [67] - Chapter 5. These chosen top-performers are: **ResNet-50**, **ViT-B-16**, **AlexNet** and **EfficientNet-V2-M**. It is to be noted that although ViT-B-32 offered slightly better performance than ViT-B-16, we selected ViT-B-16 due to its more extensive use as a backbone for large vision models, CLIP and DINO - which allows for more meaningful comparisons. Moreover, we selected EfficientNet-V2-M instead of EfficientNet-B7 due to the new version's significantly faster training time and its comparable performance to EfficientNet-B7.

Table 3.1 presents all pre-trained feature extractors to be used in our experiments. It is to be noted that only those identified by "_BASE" will be fine-tuned as well. It is to be noted that the top-performing feature extractors from the first part of the project [67] are exclusively PyTorch pre-trained models: the scope of this part extends to include CLIP and DINO pre-trained models using the top-performing architectures from the previous project as backbones (if available - only ViT-B-16 and ResNet-50 are available as CLIP/DINO backbones). It is also to be noted that ResNet-50-based CLIP models with different increases in the number of filters (namely: x4, x16 and x64) are **not** base models - namely, we will not fine-tune them.

3.4.2 Evaluation Framework

The evaluation framework builds on top of the simplified version shown in Chapter 1, Figure 1.3 and is used to evaluate the quality of the features predicted by the models

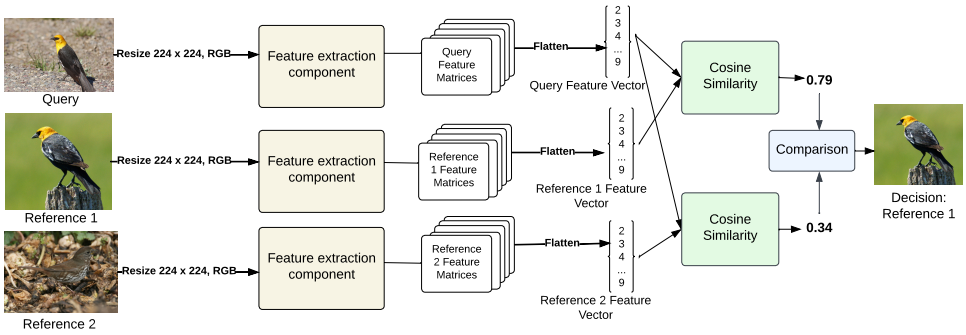

Figure 3.3: Illustration of evaluation framework used to assess whether a feature extractor correctly predicts the most similar reference to the query image. Image credit: [67].

enumerated in Table 3.1 (as well as the quality of the features of the fine-tuned extractors of these models). The evaluation framework always takes as input a triplet from one of the triplet sets described in Chapter 4. It then obtains the feature vectors for each image in the triplet, and compares, using one of the similarity metrics described in Section 3.3. Figure 3.3 illustrates the evaluation framework.

3.4.3 Fine-tuning Framework

The fine-tuning framework is more complex due to the fact that experiments involve 3 supervised fine-tuning paradigms (using triplet, constrastive and quadruplet loss), and 4 self-supervised fine-tuning paradigms (rotation prediction, Jigsaw puzzle, SimCLR and MoCo). Thus, in reality, there are 6 distinct fine-tuning frameworks - as SimCLR and MoCo share the same framework - MoCo simply keeps track of past negative examples during training. All frameworks, however, have one component in common: a linear block to either reduce feature dimensionality or aid with classification problems. This linear block - simply a sequence of fully-connected layers - is added at the end of the feature extractor and the entire model (feature extractor and linear block) is fine-tuned on the relevant data. The linear block is identical for all supervised tasks (as well as SimCLR and MoCo) - where it is used with dimensionality reduction in mind - and distinct for the class-based self-supervised tasks: for rotation prediction for example, the block will output a vector of size four (rotation classes); for Jigsaw puzzle a vectors of size 36 (possible permutation). Figure 3.4 (a) depicts the fine-tuning framework used for fine-tuning models using supervised triplet loss, whilst Figure 3.4 (b) depicts the fine-tuning framework used for fine-tuning models using self-supervised SimCLR. The frameworks for the rest of the supervised and self-supervised paradigms can easily be extrapolated from the illustrations in Figure 3.4.

It is also to be noted that whilst supervised fine-tuning makes use of triplet datasets, self-supervised fine-tuning instead uses single image datasets where the images are unique and form the triplets belonging to a triplet set for a specific source. The creation of the fine-tuning datasets used in the supervised and self-supervised versions is detailed in Chapter 4. It is also to be noted that whilst the first part [67] of the project experimented with fine-tuning using a frozen feature extractor - in the sense that weights of the feature extractor are not updated during fine-tuning, only those of the classification block - this

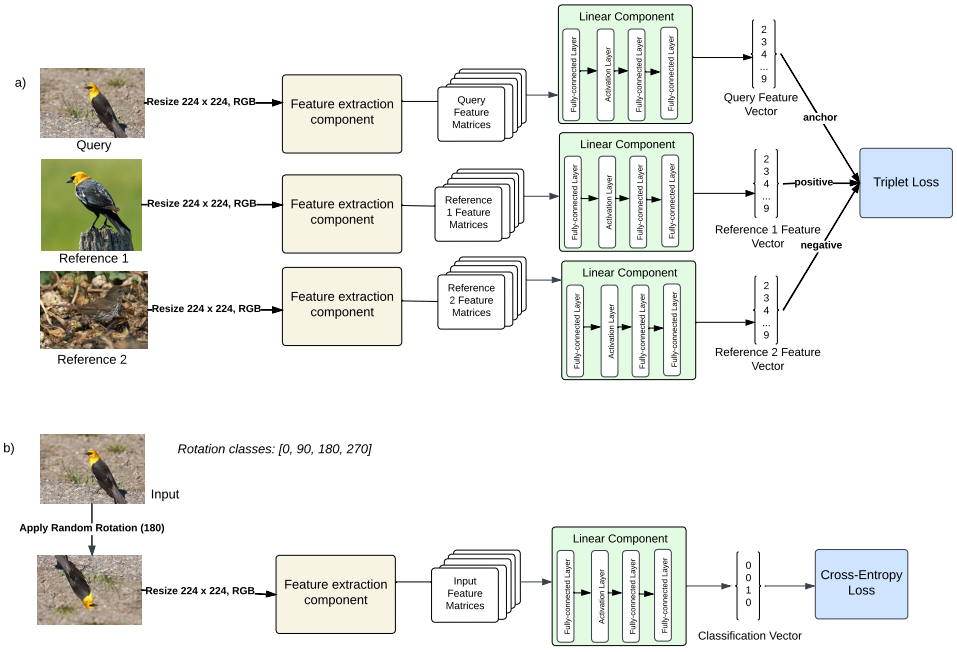

Figure 3.4: (a) Illustration of the fine-tuning framework used for supervised fine-tuning with triplet loss. Image credit: [67]. (b) Illustration of the fine-tuning framework used for self-supervised fine-tuning pretext task of rotation prediction.

project is only making use of unfrozen fine-tuning due to better proven results.

Chapter 4

Datasets

This chapter shall define all data sets used to evaluate and fine-tune models in the project, as well as how these triplet datasets were obtained and pre-processed from three distinct sources [5, 85, 109]. This section presents two main dataset types: datasets used in the evaluation process of models, and datasets used in the fine-tuning process of models. Information is presented briefly as it is also present in detail in the first part of project [67] - Chapter 3.

4.1 Data Sources

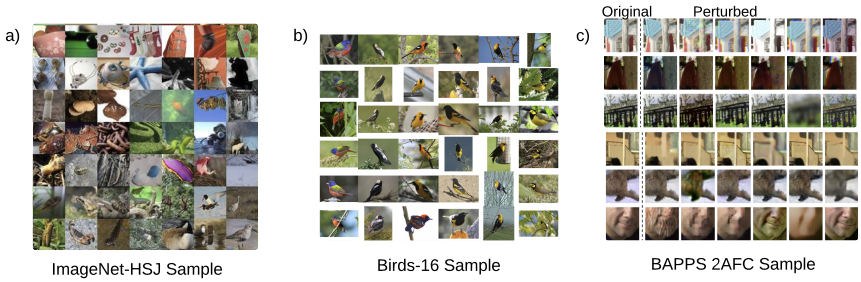

Figure 4.1: Sample images from the three image sources used in the project: (a) [85], (b) [5], (c) [109].

We make use of three triplet data sources [85, 5, 109] which present human judgements of image similarity collected via collection surveys involving human participants - Figure 1.1. Figure 1.1 - 1 query image and **8** reference images and the task of ranking the **2** most similar to the query image - shows an **8 rank 2** trial. The human judgements for the datasets have also been collected using **2 rank 1** trials: where human participants are given three images (query and **2** references) and are asked to deem **1 single** reference which is more similar to the query. The datasets have been chosen as they are vastly different from each other and may each bring to light different valuable observations with regards to model performance on triplet similarity prediction. As part of pre-processing, each image - dictated by common machine learning resizing patterns - is resized to 224 by 224 pixels image before being given as input to the model.

ImageNet-HSJ: ImageNet Human Similarity Judgements (ImageNet-HSJ) [85] triplet data was collected using images from the ILSVRC (Large Scale Visual Recognition

Challenge (ILSVRC) [88]) validation set - containing 50,000 images and 1000 categories - and conducting collection surveys of the type **8 rank 2** trials. The 9-image pairs to include in each individual step of the trials were not randomly chosen from the 50,000-image ILSVRC dataset, but rather were carefully selected using an active learning paradigm ([85], Section 4). Engagement for each trial was also measured by introducing catch trials - where the answer was highly obvious. We only make use of trials where all catch trials were correctly answered. Figure 4.1 (a) shows a sample of the ImageNet-HSJ trial images. The motivation behind choosing the ImageNet-HSJ dataset its generality and high object category number, as well as the quality of the data collection.

Birds-16: The Birds-16 [5] triplet dataset was collected using trails of type **8 rank 2** and **2 rank 1**, and images from a bird species dataset - 208 unique images spanning 4 bird families [86] - Figure 4.1 (b). The triplet set source was chosen due to the fact that images concerning bird species are more similar to each other compared to ImageNet-HSJ where each image may depict completely different objects, and may thus pose a more difficult challenge from a perceptual point of view. The existence of both **8 rank 2** and **2 rank 1** trials may highlight interesting performance variations in our models of choice.

BAPPS 2AFC: The Berkeley-Adobe Perceptual Patch Similarity (BAPPS) dataset 2AFC [109] contains image patches, not full images. The collection survey involved **2 rank 1** trials - each trial involved an image - the query - and two alternative distortions of that same image - the references. Table 2 of the original study [109] enumerates all distortion operations used. To summarise the manner in which we refer to all available distortions: colour, deblur, interpolation and super resolution-based distortions as real algorithm distortions; the others as CNN-based and traditional distortions. The survey involved catch trials alike the ImageNet-HSJ. Figure 4.1 (c) depicts images occurring in the BAPPS 2AFC triplet sets. The triplet dataset was chosen due to its distinct content: images in a triplet do not differ by content, but rather, they differ by the way in which they visualise the same query image. Contextual knowledge of individuals cannot thus play a part in similarity judgements.

4.2 Evaluation Triplet Sets

The triplet sets described in this section are those resulting from the trials conducted by each of the three source dataset studies [109, 5, 85]. We apply our own pre-processing to these triplets to obtain 11 triplet sets in total. These triplet sets will be used to evaluate the performance of feature extractors as depicted in Chapter 3, Figure 3.3. Each triplet set contains 1000 triplets obtained from the aforementioned sources by selecting 1000 trials at random (with correct catch trials). Each triplet is of the form: query image, reference 1 (most similar image to the query image), reference 2 (either second most similar **or** a random dissimilar image from the query image). We define three triplet types based on the trial they originate from, as well as the second reference image we choose to include in the final triplet: **8rank2 similar** triplet - comes from an 8 rank 2 trial and its reference 2 image is the image deemed the second most similar to the query; **8rank2 dissimilar** triplet comes from an 8 rank 2 trial and its reference 2 image is an **indirectly chosen** dissimilar image to the query image, randomly chosen from

the images which have not been chosen as the most similar or second most similar images to the query image; **2rank1 dissimilar** triplet comes from a 2 rank 1 trial and its reference 2 image is a **directly chosen** dissimilar image to the query - as the user did not choose this image as the most similar to the query. Figure 4.2 depicts all 11 triplet sets: 6 BAPPS 2AFC triplet sets correspnding to the number of distinct distortions, 2 ImageNet-HSJ triplet sets and 3 Birds-16 triplet sets (as it presents both **8 rank 2** and **2 rank 1** trials).

4.3 Fine-tuning Data Sets

Different datasets need to be used to fine-tune the base models presented in Table 3.1 using the frameworks depicted in Figure 3.4, both in the interest of keeping the number of experiments manageable, as well as to create datasets that adhere to the pre-text tasks used in the self-supervised fine-tuning methods. If we would fine-tune the models on each of the 11 triplet sets, whilst also varying the supervision/self-supervision methods enumerated in Chapter 3 - this would result in: 11 (triplet sets to train on) x 8 (base models) x 7 (supervised and self-supervised paradigms) = **616** framework versions to train. As a solution, we instead create a single dataset per supervision method and triplet source pairing to be used in fine-tuning - of **1000** datapoints each. A table summarising all fine-tuning datasets can be found in Appendix A - Table A.1.

Supervised Fine-tuning Data Sets: We make use of triplet, contrastive and quadruplet loss. Both triplet and contrastive loss-based supervised paradigms will make use of the same triplet sets, whilst quadruplet loss will make use of quadruplet sets. There will be **3** triplet datasets and **3** quadruplet datasets, for each of the three distinct data sources. The way in which we create the **3 triplet datasets**, one per triplet source (ImageNet-HSJ, Birds-16, BAPPS 2AFC): for all triplets originating from the same source, we sample 1000 random triplets, ensuring a balanced selection from each triplet set in a source. To exemplify: for all 6 triplet datasets originating from the BAPPS 2AFC data source, we sample 1000 random triplets, ensuring that each of the 6 triplet datasets are equally represented (approximately 1000/6 images in the fine-tuning BAPPS 2AFC dataset to represent each BAPPS 2AFC triplet dataset based on a different distortion as shown in Figure 4.2). The way in which we create the **3 quadruplet datasets**: we once again sample 1000 random triplets (ensuring balance) from all triplet sets for a data source, however, we then add an extra image to each triplet form a quadruplet. This image is dissimilar to the query and is obtained in **8 rank 2** triplet sets by selecting a random (dissimilar) image from the same trial that the triplet came from. In **2 rank 1** triplet sets, it is obtained by selecting a random image from a random, different, 2 rank 1 trial triplet.

Self-supervised Fine-tuning Single Image Sets: All self-supervised fine-tuning paradigms use single-image data sets, one for each of the three triplet sources. We create the **3 single-image datasets** by: creating a set of all unique images used in the triplet datasets for a specific data source (e.g.: all unique images making up the 6 BAPPS 2AFC triplet datasets), and then we sample 1000 random images from this set.

Figure 4.2: Triplet set examples from the 11 triplet sets from each of the three data sources [5, 109, 85]. Figure best viewed on screen. Figure credit: [67].

Chapter 5

Experiments and Results

This chapter aims to explain the details of each experiment and its purpose, as well as interpret the experiments results on a hypothesis-structured basis. Discussion of experiments results shall be done in light of past academic work and findings. The experiments performed aim to vastly extend those performed in the first part of this project [67], both in scope and in breadth, whilst exploring the task of similarity prediction and human perception alignment from an improvement-driven perspective.

5.1 Experiments

All experiments concern improving the performance of our chosen neural networks on the task of image triplet similarity prediction. We divide our experiments into two high-level categories: those applied on pre-trained feature extractors and a similarity metric - the framework depicted in Chapter 3 - Figure 3.3, and those applied to the fine-tuning process of a feature extractor and a custom linear block - framework depicted in Chapter 3 - Figure 3.4.

5.1.1 Pre-trained Model Experiments

Firstly, the most simplistic improvement experiment on the pre-trained models is varrying the similarity metric which is used as depicted in Figure 3.3. Three different **similarity metrics** - detailed in Chapter 3 have been chosen according to their proven effectiveness in past work in indicating image embedding similarity: cosine similarity [106, 72, 109, 85], Euclidean distance [109, 66] - L2, as well as Manhattan distance - L1 [107, 66].

Another experiment involves utilising image features from **large vision models** CLIP and DINO - detailed in Chapter 3. CLIP and DINO models are considered large due to the extensive size of the datasets that they are trained on - which may result in better foundational understanding of the the image data and consequently, richer internal representations. The two large vision models both employ vastly different pre-training techniques which result in additional valuable experiments themselves. **Pre-training supervision type** experiments come from the fact that DINO deploys

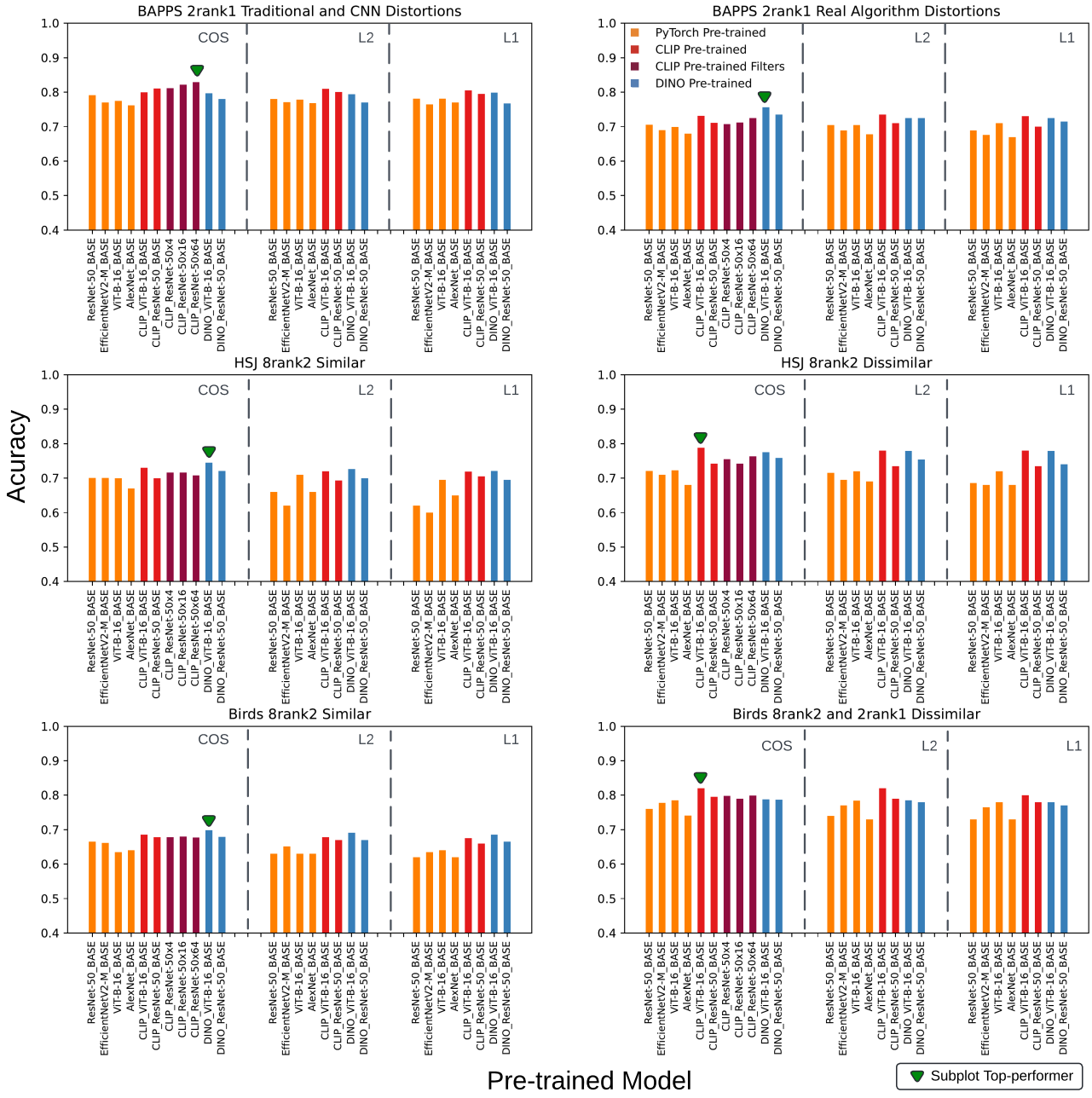

Figure 5.1: Figure is recommended to be viewed on a screen. Figure shows all pre-trained model evaluation results on all triplet sets. It is to be noted that the three different quadrants in each subplot, denoted by "COS", "L2" and "L1", represent the similarity metric used. Each subplot top-performer is indicated by a green triangle.

a self-supervised pre-training paradigm. All other pre-trained models from Table 3.1 use a supervised paradigm. The choice for self-supervision is due to the supposed increased generalisation power - ability to perform well on new, unseen data - of the features. On the other hand, **language-based pre-training experiments** originate from the fact that CLIP makes use of multimodality - image-text pairs - in its supervised training, which has been shown to improve training stability in some scenarios [43]. Applying CLIP-based models to our triplet similarity prediction task may offer more semantically rich feature representations as CLIP's output vectors are not only based on

pixel-level features, instead, they also incorporate high-level semantic understanding derived from the training text labels which contained descriptions. **Backbone scaling experiments** will also be executed using CLIP's backbone offering and namely ResNet-50x4, ResNet-50x16 and ResNet-50x64 - corresponding to the last three rows in Table 3.1. The "x4", "x16", and "x64" indicate the width - the number of channels or feature maps learnt from the input data - of the backbone architecture compared to the original ResNet-50. This may allow for capturing more detailed features from input images and, as a consequence, leading to improved performance.

5.1.2 Fine-tuned Model Experiments

The fine-tuning experiments revolve mainly around varrying the **supervised loss and self-supervised pre-text task** for all base models in Table 3.1. This experimentation aims to expose models to broader range of learning scenarios and challenges and possibly establish correlations between certain supervision paradigm's effectiveness and the datasets/models they were applied to. It is to be noted that to assess the performance of fine-tuned feature extractors on the 11 triplet sets, we remove the linear block and evaluate the feature extraction component only. This is done in order to ensure fairness accross supervision paradigm comparisons - as self-supervised tasks' linear block is non-applicable to triplet similarity prediction task. Fine-tuning implementation details for reproductibility purposes can be found in Appendix A.

5.2 Results and Discussion

Following the discussion concerning the experiments carried out as part of this project, the results shall now be discussed with respect to a number of hypotheses. Due to the extensive number of experiments - over 300 concerning both evaluation and fine-tuning - experiment results will only be displayed and discussed if they bring informational value in relation to a hypothesis. Colours chosen for the graphs have been selected with colour blindness friendliness in mind using the process described in Appendix C. It is to be noted that due to great similarity in results and patterns, we average and group together results for the following triplet types (Chapter 4, Figure 4.2): CNN and colour distortions from BAPPS 2AFC, real algorithm distortions for BAPPS 2AFC (alike the grouping in the original paper [109]), as well as, 2 rank 1 and 8 rank 2 dissimilar triplets from Birds-16. We begin by discussing evaluation-based hypotheses, followed by fine-tuning-based hypotheses.

5.2.1 Large Vision Models Pre-Trained Features

***Hypothesis**: Large vision model-generated features align better with human similarity judgements.*

We shall first discuss the reason for which the above hypothesis is of interest to us. Due to the extensive training datapoints that our large vision models, CLIP and DINO, have been pre-trained on, we infer greater potential in extracting more intricate and nuanced image features and more interesting emerging patterns and relationships between images.

We deploy these models in the hope that this added feature complexity may breach the gap between machine and human vision. Previous studies have explored the capabilities of large models as achievers of state-of-the-art performance [103, 74] on datasets such as ImageNet on classification and object-detection tasks. On the other hand, studies [94] have also assessed the "cognitive" (language, vision, problem-solving processes) abilities of these large models and have concluded that a large cognitive gap still remains between large vision models and humans.

Figure 5.1 shows the results from evaluating the pre-trained models listed in Table 3.1 on all triplet sets, using three similarity metrics. It is to be noted that large-vision model-based frameworks have been shown consistently to be top-performers. However, interesting patterns emerge from this observation: we notice that DINO ViT-based models are consistently top-performers on triplets sets where the overall accuracy is lower (*harder* sets); whilst CLIP ViT and CNN-based models are top-performers on triplet sets where overall accuracy is higher (*easier* sets). A look at Figure 4.2 suggests that easier sets involve triplets where the second reference is visually different to the query, as well as easier, more *describable*, distortions related to colour. We may thus say that CLIP models perform best on triplets where (dis)similarity can be "put into words". CLIP is pre-trained in such a way that it learns **visual** concepts from **natural** language - this allows the model to understand images in a manner closely aligned with human language descriptions - thus explaining its increased accuracy on easier triplets. Human judgement similarity predictions' link to language has been observed in previous studies [68], highlighting how models can predict human similarity judgments using word-embedding representations - this illustrates the benefit of text-image multimodality and how the link between image and text similarity aligns in some measure with human similarity conceptualisation.
On the other hand, harder triplet sets instead present references which are both similar to the query image and not only, also more difficult "to put into words" distortions. CLIP's lack of efficiency, when compared to DINO, concerning these models may be the fact that language cannot capture such granularity of image intricacies that would make similar images (in an 8 rank 2 similar triplet - the references) in fact be predicted as dissimilar. On the other hand, DINO models, via self-supervised training solely on images, build a deep understanding of visual content, and thus capture nuanced visual similarities and differences. It has been shown before [74] that certain versions of DINO outperform CLIP models, despite the lack of supervision, on image similarity prediction tasks involving specific datasets.
Not only, another property of large vision models when compared to regular PyTorch pre-trained models is their robusteness to the change in the similarity metric: whilst switching to L1 and L2 distance from cosine similarity has a negative impact of considerable magnitude on PyTorch pre-trained models, large vision pre-trained models' accuracy is impacted a lot less, their accuracy remaining somehow consistent irrelevant of the similarity metric.

We thus affirm that the experiments **prove** the hypothesis that large vision models are indeed better emulators of human perception as indicated by their superior performance, as well as robustness to similarity metric variation, on human judgement similarity prediction.

5.2.2 Pre-training Supervision Type

Hypothesis: *There is a specific pre-training supervision type which better aligns with human judgements.*

This hypothesis is of interest to us due to the rise in popularity [32] of self-supervised pre-trained backbones, which showcase certain generalisation advantages [3]. Studies [32, 3] suggest that self-supervision's advantage lies in its increased capability for generalisation and adaptability to new datasets, especially valuable in transfer learning [10], as pre-trained models will ultimately be used on a different dataset from the one they were pre-trained on. Not only, similarly to our case where we use triplets from a number of different sources, a model's ability to adapt to differently structured images is highly important. On the other hand, supervised pre-trained models may sometimes learn "too well" (overfit [33] - Chapter 5) the intricacies of their pre-training dataset - such as ImageNet-pretrained models only performing well on evaluation triplets originating from the ImageNet dataset (ImageNet-HSJ), for example.

We compare ViT-B-16 and ResNet-50 pre-trained models from PyTorch/ CLIP (supervised), and DINO (self-supervised). With the justification above in mind, we would thus expect DINO models to be the sole top-performers of triplet sets originating from datasets which differ vastly from the datasets on which our supervised backbones have been trained on, such as BAPPS 2AFC triplets. None of the pre-training datasets present distortions alike those seen in BAPPS 2AFC triplet sets, yet our previous affirmation about DINO as a sole top-performer is quickly invalidated by the top subplots in Figure 5.1. Not only do CLIP models (and on one occasion, a PyTorch-pre-trained ResNet-50) outperform DINO models for CNN and colour-based distortions, but we also see that performance of DINO models does not greatly surpass that of CLIP and PyTorch models on the distortion-derived triplet sets. The effects of the pre-training, as proven in a previous section, may thus not be related to the supervision type, as much as they are to the number of datapoints in the pre-training dataset and the specific pre-training task. On the other hand however, as discussed in the previous section, DINO-based models (pre-trained via self-supervision) display best results on *harder* (conceptually more difficult) triplets than on *easier* triplets. A similar pattern is shown in the original study [109] for the collection of the BAPPS-2AFC dataset, where self-supervised networks seem to outperform supervised networks on more difficult (measured by lower overall model accuracy) distortion types. However, DINO's effectiveness in our case may be explained by the competition that we draw parallels to: CLIP uses language whose semantics may not capture the granularity of difficult image similarity tasks, whilst PyTorch models are inherently disadvantaged by the low number of data points they have been pre-trained on.

The results are thus **inconclusive** in light of the hypothesis that an emerging pattern proves a pre-training supervision type superior in estimating human perception.

5.2.3 Optimal Similarity Metric

Hypothesis: *There is an optimal similarity metric which benefits all architectures and supervision types.*

Figure 5.1 applies three similarity metrics to our pre-trained models, as explained in Section 5.1.1. The overall observation is that cosine similarity outperforms both other metrics for all models, and on all triplet sets. However, despite the straightforward result of this hypothesis, more meaningful observations can be drawn. Firstly, surprisingly, although the switch to L1/L2 distance negatively (sometimes drastically) impacts most architectures, simpler architectures such as AlexNet seem to be the least affected, presenting consistent performance accross similarity metrics. AlexNet is an older model and presents 8 layers only: its features may thus not be as abstract and as nuance-enconding as the features resulting from deeper or more complex models, but rather simpler and easier to measure by cosine similarity and L1/L2 distance alike. The combination between AlexNet and L2 distance has been shown to be optimal before by the original Learned Perceptual Image Patch Similarity (LPIPS) metric paper [109].

Going back to the results in Figure 5.1: we observe that more complex PyTorch CNN pre-trained models are the most negatively affected by the switch from cosine similarity to L1/L2 distance. On the other hand, large vision model predictions are not nearly as negatively affected - despite being on the opposite end of the complexity spectrum to AlexNet. We may thus say that features of large vision models are complex enough to be meaningful and informational in themselves, without heavy reliance on optimal similarity metrics. ViT-based models, no matter the origin, seem to be robust in the face of similarity metric variation - this can be explained by the surprising property of ViT-based features compared to CNN-based features shown in Figure 5.2 and discussed in Section 5.2.6: ViT-based features seem to offer better spatial separation of images into categories and this may be why ViT-based models' performance is not as affected by traditional distance measuring metrics such as L1/L2.

Between L1 and L2 distance however, L1 seems to generate the worst performance. Not only, if we detail our observation, L1's sub-optimal performance is not as sereve on triplet sets of image patches (originating from BAPPS 2AFC), as it is on full-size image triplets (ImageNet-HSJ, Birds-16). This may be due to the fact that L1 is more sensitive to exact pixel differences (essentially how our distortions are structured) and thus unsuitable for capturing broader structural (dis)similarities in images (essential for full-image triplet sets). On the other hand, L2 distance is capable to some extent to also capture overall differences in images, making it more effective that L1 distance on full-image tasks.

The results thus **prove** the hypothesis which affirms the existence of an optimal similarity metrics, namely cosine similarity, irrespective of supervision type.

5.2.4 Language in Pre-training

Hypothesis: *The addition of language in pre-training vision models may provide semantic understanding of images and minimise the gap with human judgements.*

The interest in the hypothesis stems from curiosity involving the link between images and text in the context of both human perception and tasks: studies have previously proposed (and demonstrated advantages) utilising language-based models [108] and image-text multimodality [79, 43] for obtaining image features for pure vision tasks.

As discussed in Section 5.2.1 and shown in Figure 5.1, CLIP models (trained on image-text pairs) provide especially strong performance on triplet sets where (dis)similarity is defined in a context that aligns well with human textual descriptions. However, CLIP models are outperformed by DINO models for a number of tasks, despite DINO's lack of supervision. CLIP's efficiency due to its use of language in pre-training is thus fully dependent on the nature of the dataset it is evaluated on - the inclusion of language in pre-training is thus not a universal solution to improving accuracy.

The results thus **disprove** the hypothesis as, despite language in pre-training providing a level of semantic understanding that can be applied if structural differences of triplets can be "put into words", the use of language is not a universal solution for minimising the gap between machine and human vision.

5.2.5 Increased Number of CNN Filters

Hypothesis: *Increasing the number of filters in a CNN architecture will lead to greater complexity of patterns retained by features and thus minimise the gap with human judgements.*

Following the popularisation of CNN architectures by AlexNet [56], the de-facto standard for improving model performance on vision tasks was to increase the number of parameters [92] up until the apparition of EfficientNet [97]. The hypothesis thus considers this old-fashioned approach and looks into increasing the efficiency of CNN architectures on the human similarity judgement task by increasing the number of filters - detailed in Chapter 3. This experiment is possible due to the fact that CLIP offers 4 ResNet-50 backbones: regular, x4, x16, x64 - Chapter 3. Increasing the number of CNN filters may result in more detailed and nuanced feature representations, potentially improving model performance.

The results shown in Figure 5.1 aid us conclude that whilst increasing the number of filters in CLIP CNN-architectures improves performance in patch-based triplet sets (BAPPS 2AFC) - even yielding a top-performer for CNN and colour distortions - it makes negligible difference in full-image-based triplet sets (ImageNet-HSJ, Birds-16). This improvement in patches only may be due to the fact that patches focus on smaller regions of interest in an image and thus, increasing the number of filters allows the model to capture more intricate details about these smaller information areas. On the other hand, full-images provide broad context that can be captured with lower number of filters as well. CLIP models with an increased number of filters are not amongst top-performer models for most triplet sets in Figure 5.1.

The results thus **disprove** the hypothesis that increasing the number of CNN filters yields better results on estimating human judgdments of image similarity.

5.2.6 Pre-trained Transformers Versus CNN Architectures

Hypothesis: *Transformer architectures present emerging properties which explain their proven efficiency, compared to CNN architectures, at estimating human judgements.*

The hypothesis is of interest to us as multiple studies [17, 103, 5, 98] showcase ViT

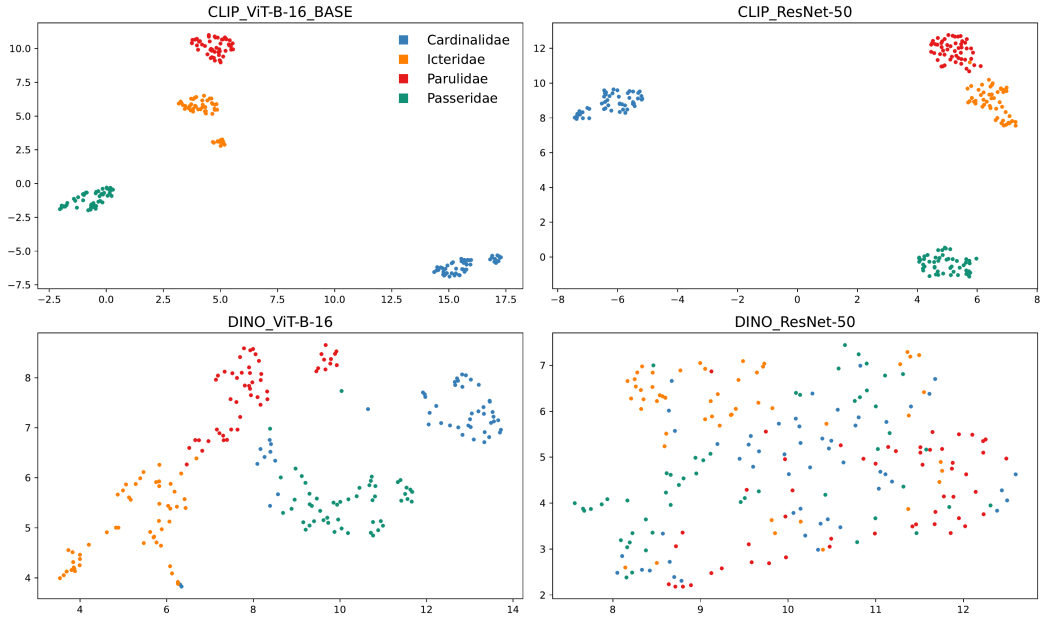

Figure 5.2: Features of ViT and CNN-based large vision models plotted using non-linear dimensionality reduction tool UMAP [69]. Features are generated for each unique image in the Birds-16 data set.

models as better estimations of human vision and not only, as being able to achieve this without the supervision or pre-training that a CNN would make use of [17]. The first part of this project [67] also observed the increased performance of ViT over CNN-based architectures. The reason for this is the vastly different manner in which ViT and CNN networks are built [80] - attention mechanism versus filters, as detailed in Chapter 3.

Figure 5.2 depicts the dimensionality reduced features of the 208 unique bird images making up the Birds-16 image triplets, of ViT-B-16 and ResNet-50 models from both CLIP and DINO. The reason why CLIP models provide vastly superior separation into categories is due to the fact that they are trained in a supervised manner, whilst DINO is trained in an unsupervised manner. However, another observation comes to light: ViT-based models are much better at separating images into categories, irrespective of the pre-training paradigm or the use/lack of language. This may be one facet of why ViT networks are the top-performers in all subplots (so when applied to all triplet sets) with the exception of the CNN and colour-based distortion triplets subplot. This may be explained due to how ViT's process images: the self-attention mechanism applied to patches of the image may allow the network to capture both local and global relationships [22, 17] and this global perspective may be the factor that allows this nuanced differentiation into bird categories. In the case of CNN's, due to their hierarchical application of filters, they have the capacity to capture local and afterwards increasingly more abstract embeddings, however lacking the global perspective [17].

The results illustrated in Figures 5.2 and 5.1 thus **prove** the hypothesis that transformer-based architectures present emerging properties which partly explain the quality of their alignment with human perception.

5.2.7 Link Between Fine-tuning And Pre-training Supervision

Hypothesis: *The effectiveness of the improvement applied by fine-tuning a pre-trained feature extractor depends on the possible link between the supervision type of the fine-tuning and that of the pre-training.*

The purpose of this hypothesis is to affirm or deny that there may be a link between the optimal-fine-tuning supervision type and the pre-training supervision. Past work [14] suggests the existence of such a relation, although nuanced and complex, between pre-training and fine-tuning supervision types influencing optimal fine-tuning approaches. In essence, the pre-training of the model sets a foundational "knowledge" that can determine the most effective fine-tuning strategy.

Figure 5.3 shows the accuracies obtained form fine-tuning the base models - with the exception of AlexNet and EfficientNet architectures, which brought little informational value due to their similarity to PyTorch's ResNet-50 and consistently lower-end results. The results in Figure 3.4 show a pattern in the efficiency of self-supervised training depending on the supervision of the pre-trained backbone. Firstly, it is to be noted that supervised fine-tuning remains the most effective fine-tuning paradigm - generating the top-performers for each plot - all with the use of quadruplet loss. Secondly, we see that self-supervised fine-tuning generates a smaller increase on supervised pre-trained backbones, and a much more significant increase on self-supervised pre-trained backbones. This may be due to the fact that self-supervised backbones still benefit from gaining a deeper understanding of images, whilst supervised backbones - due to the specificity of their training - already possess "knowledge" of image basics. To further support this claim, we also notice that self-supervised fine-tuning has even less of an effect on large vision backbone models trained in a supervised way, compared to PyTorch supervised model backbones. Another hypothesis [110, 76] may be that self-supervised backbone models simply benefit from fine-tuning approaches that continue the theme of self-supervision as these models, not having been trained on labelled data, gain more from being allowed to explore the input space instead during fine-tuning.

The results, despite the interesting emerging patterns on self-supervision, **disprove** the hypothesis as ultimately, supervised fine-tuning still obtained superior results compared to self-supervised fine-tuning in all use cases.

5.2.8 Fine-tuning Loss or Pre-text Task

Hypothesis: *There is a universally better supervised fine-tuning loss or self-supervised pre-text task which approaches models to human perception.*

This is a hypothesis of interest because it is not immediately obvious which loss is optimal: although triplet loss is highly aligned to the logic of the triplet similarity prediction task, quadruplet loss is more complex and provides a link to an additional dissimilar reference image which may uncover new relations in the dataset and also aid in adding robustness to the feature space. On the other hand, the choice of self-supervised loss is similar: both SimCLR and MoCo are the closest in logic to our similarity prediction task, yet Jigsaw and Rotation tasks may again reveal previously uncovered nuanced structure in our data. The fine-tuning paradigm variation results

presented in Figure 5.3 are straight-forward in showcasing optimal supervision losses. On the other hand, we see less obvious patterns in self-supervised pre-text tasks. Firstly, we see that in all cases, quadruplet loss is the best performing training paradigm in general, followed by triplet and lastly, contrastive - this is a sustained pattern in the vast majority of experiments. From a supervised point of view, we may thus say that the hypothesis is **proven**.

Self-supervised fine-tuning pre-text tasks present less direct patterns: generally, we see that pre-text tasks SimCLR and MoCo are better performing than Jigsaw and Rotation. This excludes BAPPS 2AFC triplet sets where SimCLR and MoCo offer especially poor performance which may be explained due to the fact that BAPPS 2AFC images are patches and both SimCLR and MoCo rely on augmentation. Augmenting an image patch - a patch itself already focuses on a part of a whole image and thus has lower informational value - may result is an overly cropped portion of an image that holds little informational value with the exception of pixel-level differences. However, the other full-image-based triplet sets suggests that self-supervised fine-tuning pre-text tasks which approach the logic of the evaluation task yield better improvements. From a self-supervised point of view, we may thus also say that the hypothesis is **proven**.

5.2.9 Fine-tuning Results' Linked to Triplet Type

Hypothesis: *There is a link between the optimal fine-tuned supervision type and the perceptual difficulty of the triplet set that it is evaluated on.*

As mentioned in Chapter 4, triplets may be of two types: 8 rank 2 or 2 rank 1 similar - where the second reference image is also similar to the query, however we expect the model to deem it dissimilar - a *harder* triplet set; and 8 rank 2 dissimilar, an *easier* triplet set. We may thus assume that the perceptual difficulty of the task has an impact on the optimal fine-tuning loss or pre-text task - a harder dataset may require the use of quadruplet loss (which is more strict), for example, whilst an easier dataset may require the use of a less strict loss such as triplet loss.

The results displayed in Figure 5.3 indicate an overall loss choice, despite of the triplet type it is applied to - namely, quadruplet loss is a top performer. All triplet types, no matter their perceptual difficulty, seem to benefit the most from the use of supervised fine-tuning, as opposed to self-supervised. We notice the same difference in effectiveness between supervised and self-supervised paradigms for all triplet sets and not only, the same amount of general improvement. The only interesting observation is the lack of effectiveness of augmentation-based methods on patch triplet sets as described in Section 5.2.8. Another interesting pattern, applicable to non-patch-based triplet sets only, is the fact that SimCLR and MoCo pre-text tasks in self-supervised learning appear to uncover performance benefits on more difficult tasks - namely, similar triplets. When applied to perceptually difficult tasks, SimCLR and MoCo almost perform to the level of supervised contrastive loss. For easier tasks however, the accuracy gap between supervised and self-supervised methods grows. It may be that SimCLR and MoCo uncover less obvious patterns and nuanced differences that make similar images in fact dissimilar: as the model needs to predict the second reference as being dissimilar, despite having been deemed similar.

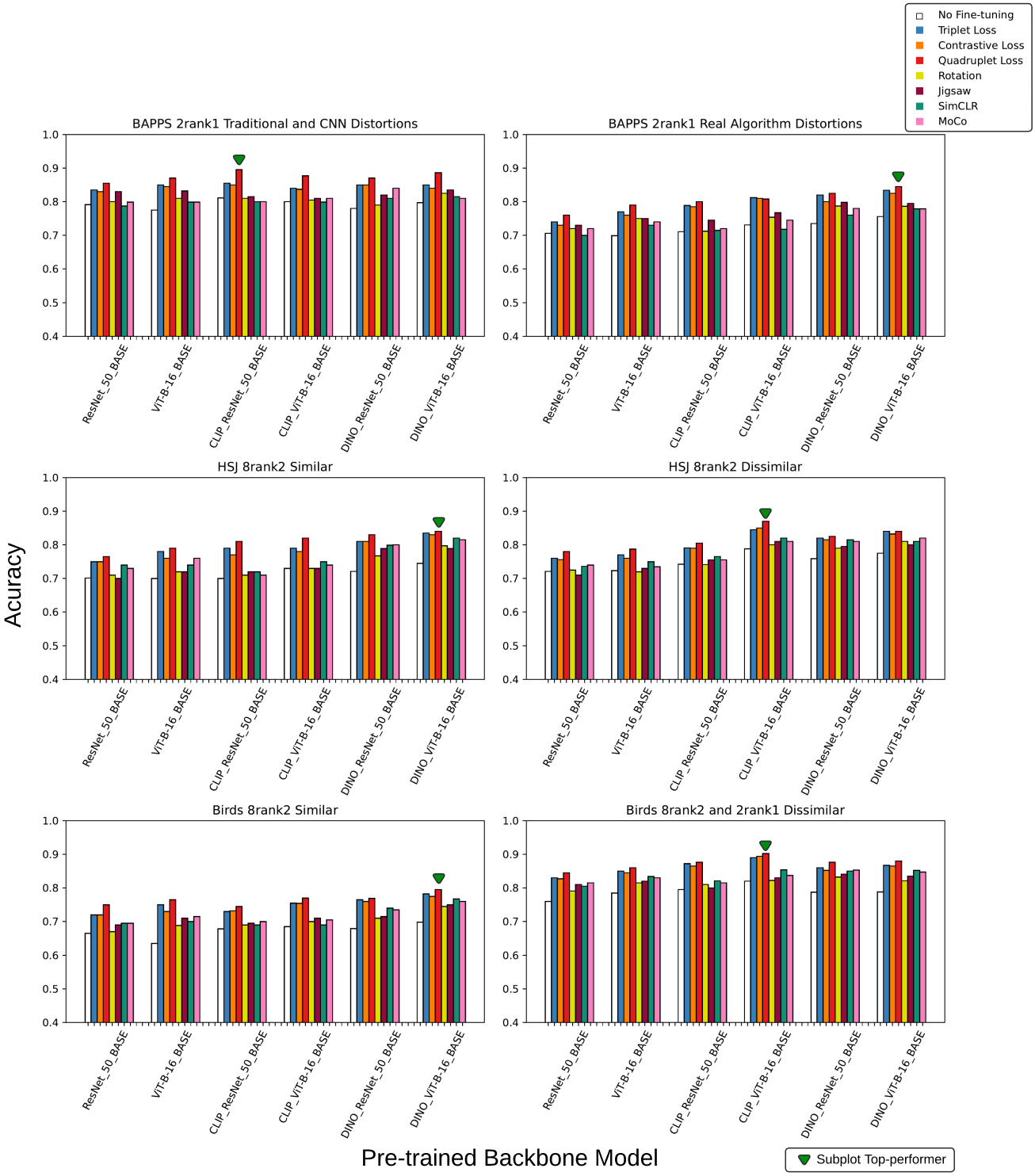

Figure 5.3: Figure is recommended to be viewed on a screen. Figure shows all fine-tuned base models (with the exception of AlexNet and EfficientNet - of particularly low informational value). Each cluster in each subplot represents a base model and the 7 different fine-tuning paradigm results applied.

Thus, whilst there is a link between the perceptual difficulty of the triplet set and the results of fine-tuning, the optimal method remains the same (quadruplet loss) which **disproves** the hypothesis.

Chapter 6

Conclusion

Optimal improvements techniques on the task of predicting human estimates of image similarity have been highlighted in Chapter 5, as well as meaningful patterns and correlations between improvement methods and various task facets. Firstly, we generally observed a superiority in the quality of predictions of both pre-trained and fine-tuned large vision models (top-performers), as well as of ViT architectures. A clear correlation between perceptual difficulty of a triplet set and optimal model choices and fine-tuning paradigms has also been shown. We also observe links between the structure of the image in a triplet, patch or full-image, and the effectiveness (or lack thereof) of certain fine-tuning paradigms and similarity metrics. Lastly, we also demonstrate the clear superiority of supervised fine-tuning, however observe an unexplained potential of self-supervised fine-tuning on perceptually difficult triplet sets. To conclude, this project fulfils its scope of acting as a guide for providing a number of methods, and their effectiveness given task particularities, aimed at closing the gap between machine and human vision and perception via the triplet similarity prediction task.

6.1 Future Work and Limitations

Future work may concern the introduction of new fine-tuning losses. We have proven the effectiveness of fine-tuning supervised losses, with quadruplet loss as a top-performer. Alike models being improved by increasing their parameters post-CNN emergence [92], we may follow the same strategy and attempt to improve predictions by considering more and more positive/negative image samples (N-tuplet loss). Additionally, N may change each epoch based on the quality of the predictions from the previous training epoch. Secondly, another valuable future direction, may be the collection of an extensive dataset of human similarity judgements of both images and text cues/hints.
Limitations in approaching machine and human perception stem from the complex nature of human perception [68, 71], and may lie in the intersection of understanding human perception inner workings, and machine learning models. Human perception is too nuanced and subjective for one to provide a computational model of it. Alternatively, despite recent advances, models present an inherent complexity and non-linearity [47] which still obscures part of their decision making process.

Bibliography

[1] Mahbubul Alam, Manar D Samad, Lasitha Vidyaratne, Alexander Glandon, and Khan M Iftekharuddin. Survey on deep neural networks in speech and vision systems. *Neurocomputing*, 417:302–321, 2020.

[2] Saleh Albelwi. Survey on self-supervised learning: auxiliary pretext tasks and contrastive learning methods in imaging. *Entropy*, 24(4):551, 2022.

[3] Jonah Anton, Liam Castelli, Mun Fai Chan, Mathilde Outters, Wan Hee Tang, Venus Cheung, Pancham Shukla, Rahee Walambe, and Ketan Kotecha. How well do self-supervised models transfer to medical imaging? *Journal of Imaging*, 8(12):320, 2022.

[4] F Gregory Ashby and W Todd Maddox. Human category learning. *Annu. Rev. Psychol.*, 56:149–178, 2005.

[5] Maria Attarian, Brett D Roads, and Michael C Mozer. Transforming neural network visual representations to predict human judgments of similarity. *arXiv preprint arXiv:2010.06512*, 2020.

[6] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[7] Vassileios Balntas, Edgar Riba, Daniel Ponsa, and Krystian Mikolajczyk. Learning local feature descriptors with triplets and shallow convolutional neural networks. In *Bmvc*, volume 1, page 3, 2016.

[8] Manuele Barraco, Marcella Cornia, Silvia Cascianelli, Lorenzo Baraldi, and Rita Cucchiara. The unreasonable effectiveness of clip features for image captioning: an experimental analysis. In *proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4662–4670, 2022.

[9] Ivana Bianchi and Roberto Burro. The perception of similarity, difference and opposition. *Journal of Intelligence*, 11(9):172, 2023.

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

[11] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

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

[13] Chang Che, Qunwei Lin, Xinyu Zhao, Jiaxin Huang, and Liqiang Yu. Enhancing multimodal understanding with clip-based image-to-text transformation. In *Proceedings of the 2023 6th International Conference on Big Data Technologies*, pages 414–418, 2023.

[14] Tianlong Chen, Sijia Liu, Shiyu Chang, Yu Cheng, Lisa Amini, and Zhangyang Wang. Adversarial robustness: From self-supervised pre-training to fine-tuning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 699–708, 2020.

[15] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[16] Weihua Chen, Xiaotang Chen, Jianguo Zhang, and Kaiqi Huang. Beyond triplet loss: a deep quadruplet network for person re-identification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 403–412, 2017.

[17] Xiangning Chen, Cho-Jui Hsieh, and Boqing Gong. When vision transformers outperform resnets without pre-training or strong data augmentations. *arXiv preprint arXiv:2106.01548*, 2021.

[18] George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

[19] Ross H Day and Hannelore Knuth. The contributions of fc müller-lyer. *Perception*, 10(2):126–146, 1981.

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

[21] Samuel Dodge and Lina Karam. Understanding how image quality affects deep neural networks. In *2016 eighth international conference on quality of multimedia experience (QoMEX)*, pages 1–6. IEEE, 2016.

[22] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[23] Matthijs Douze, Giorgos Tolias, Ed Pizzi, Zoë Papakipos, Lowik Chanussot, Filip Radenovic, Tomas Jenicek, Maxim Maximov, Laura Leal-Taixé, Ismail Elezi, et al. The 2021 image similarity dataset and challenge. *arXiv preprint arXiv:2106.09672*, 2021.

[24] Nathan Drenkow, Numair Sani, Ilya Shpitser, and Mathias Unberath. A systematic review of robustness in deep learning for computer vision: Mind the gap? *arXiv preprint arXiv:2112.00639*, 2021.

[25] Stephanie Fu, Netanel Tamir, Shobhita Sundaram, Lucy Chai, Richard Zhang, Tali Dekel, and Phillip Isola. Dreamsim: Learning new dimensions of human visual similarity using synthetic data. *arXiv preprint arXiv:2306.09344*, 2023.

[26] Nicole M. Gage and Bernard J. Baars. Chapter 4 - the art of seeing. In Nicole M. Gage and Bernard J. Baars, editors, *Fundamentals of Cognitive Neuroscience (Second Edition)*, pages 99–141. Academic Press, San Diego, second edition edition, 2018. ISBN 978-0-12-803813-0. doi: https://doi.org/10.1016/B978-0-12-803813-0.00004-0.

[27] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. *arXiv preprint arXiv:1811.12231*, 2018.

[28] Robert Geirhos, Carlos RM Temme, Jonas Rauber, Heiko H Schütt, Matthias Bethge, and Felix A Wichmann. Generalisation in humans and deep neural networks. *Advances in neural information processing systems*, 31, 2018.

[29] Robert Geirhos, Kristof Meding, and Felix A Wichmann. Beyond accuracy: quantifying trial-by-trial behaviour of cnns and humans by measuring error consistency. *Advances in Neural Information Processing Systems*, 33:13890–13902, 2020.

[30] Robert Geirhos, Kantharaju Narayanappa, Benjamin Mitzkus, Tizian Thieringer, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Partial success in closing the gap between human and machine vision. *Advances in Neural Information Processing Systems*, 34:23885–23899, 2021.

[31] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*, 2018.

[32] Micah Goldblum, Hossein Souri, Renkun Ni, Manli Shu, Viraj Prabhu, Gowthami Somepalli, Prithvijit Chattopadhyay, Mark Ibrahim, Adrien Bardes, Judy Hoffman, et al. Battle of the backbones: A large-scale comparison of pretrained models across computer vision tasks. *Advances in Neural Information Processing Systems*, 36, 2024.

[33] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[34] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[35] Ziyan Guo and Jun Liu. Trustworthy large models in vision: A survey. *arXiv preprint arXiv:2311.09680*, 2023.

[36] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE computer society conference on computer vision and pattern recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE, 2006.

[37] James A Hampton. Similarity-based categorization and fuzziness of natural categories. *Cognition*, 65(2-3):137–165, 1998.

[38] 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*, pages 770–778, 2016.

[39] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

[40] Sepp Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02):107–116, 1998.

[41] Mingjia Hu and Robert M Nosofsky. Exemplar-model account of categorization and recognition when training instances never repeat. *Journal of experimental psychology: Learning, memory, and cognition*, 48(12):1947, 2022.

[42] Yunxiang Hu, Yuhao Liu, and Zhuovuan Liu. A survey on convolutional neural network accelerators: Gpu, fpga and asic. In *2022 14th International Conference on Computer Research and Development (ICCRD)*, pages 100–107. IEEE, 2022.

[43] Yu Huang, Chenzhuang Du, Zihui Xue, Xuanyao Chen, Hang Zhao, and Longbo Huang. What makes multi-modal learning better than single (probably). *Advances in Neural Information Processing Systems*, 34:10944–10956, 2021.

[44] David H Hubel and Torsten N Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, 160 (1):106, 1962.

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

[46] Qirui Jiao, Daoyuan Chen, Yilun Huang, Yaliang Li, and Ying Shen. Enhancing multimodal large language models with vision detection models: An empirical study. *arXiv preprint arXiv:2401.17981*, 2024.

[47] Md Rezaul Karim, Md Shajalal, Alexander Graß, Till Döhmen, Sisay Adugna Chala, Alexander Boden, Christian Beecks, and Stefan Decker. Interpreting black-box machine learning models for high dimensional datasets. In *2023 IEEE 10th International Conference on Data Science and Advanced Analytics (DSAA)*, pages 1–10. IEEE, 2023.

[48] Qifa Ke and Takeo Kanade. Robust subspace computation using l1 norm. 2003.

[49] Zubair Ahmed Khan and Asma Rizvi. Ai based facial recognition technology and criminal justice: Issues and challenges. *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, 12(14):3384–3392, 2021.

[50] Jack Kiefer and Jacob Wolfowitz. Stochastic estimation of the maximum of a regression function. *The Annals of Mathematical Statistics*, pages 462–466, 1952.

[51] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

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

[53] Simon Kornblith, Ting Chen, Honglak Lee, and Mohammad Norouzi. Why do better loss functions lead to less transferable features? *Advances in Neural Information Processing Systems*, 34:28648–28662, 2021.

[54] LW Koster. Three little words–vision, perception, seeing. *Journal of Biological Photography*, 66(2):41–44, 1998.

[55] Nikolaus Kriegeskorte. Deep neural networks: a new framework for modeling biological vision and brain information processing. *Annual review of vision science*, 1:417–446, 2015.

[56] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6): 84–90, 2017.

[57] Sarah C Kucker, Larissa K Samuelson, Lynn K Perry, Hanako Yoshida, Eliana Colunga, Megan G Lorenz, and Linda B Smith. Reproducibility and a unifying explanation: Lessons from the shape bias. *Infant Behavior and Development*, 54: 156–165, 2019.

[58] Manoj Kumar, Neil Houlsby, Nal Kalchbrenner, and Ekin Dogus Cubuk. Do better imagenet classifiers assess perceptual similarity better? *Transactions of Machine Learning Research*, 2022.

[59] Naresh Kumar Lahajal et al. Enhancing image retrieval: A comprehensive study on photo search using the clip mode. *arXiv preprint arXiv:2401.13613*, 2024.

[60] Jiyoung Lee, Seungho Kim, Seunghyun Won, Joonseok Lee, Marzyeh Ghassemi, James Thorne, Jaeseok Choi, O-Kil Kwon, and Edward Choi. Visalign: Dataset for measuring the degree of alignment between ai and humans in visual perception. *arXiv preprint arXiv:2308.01525*, 2023.

[61] Youngseok Lee and Jongweon Kim. Robustness of deep learning models for vision tasks. *Applied Sciences*, 13(7):4422, 2023.

[62] Feng Lin, Wenze Hu, Yaowei Wang, Yonghong Tian, Guangming Lu, Fanglin Chen, Yong Xu, and Xiaoyu Wang. Universal object detection with large vision model. *International Journal of Computer Vision*, pages 1–19, 2023.

[63] Grace W Lindsay. Convolutional neural networks as a model of the visual system: Past, present, and future. *Journal of cognitive neuroscience*, 33(10):2017–2031, 2021.

[64] Dengsheng Lu and Qihao Weng. A survey of image classification methods and techniques for improving classification performance. *International journal of Remote sensing*, 28(5):823–870, 2007.

[65] Maya Malaviya, Ilia Sucholutsky, Kerem Oktar, and Thomas L Griffiths. Can humans do less-than-one-shot learning? *arXiv preprint arXiv:2202.04670*, 2022.

[66] MD Malkauthekar. Analysis of euclidean distance and manhattan distance measure in face recognition. In *Third International Conference on Computational Intelligence and Information Technology (CIIT 2013)*, pages 503–507. IET, 2013.

[67] Florica Margaritescu. Predicting human estimates of image similarity. In *2023*, pages 1–40. UoE, 2023.

[68] Raja Marjieh, Ilia Sucholutsky, Theodore R Sumers, Nori Jacoby, and Thomas L Griffiths. Predicting human similarity judgments using large language models. *arXiv preprint arXiv:2202.04728*, 2022.

[69] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[70] Gaurav Menghani. Efficient deep learning: A survey on making deep learning models smaller, faster, and better. *ACM Computing Surveys*, 55(12):1–37, 2023.

[71] Lukas Muttenthaler, Lorenz Linhardt, Jonas Dippel, Robert A Vandermeulen, Katherine Hermann, Andrew Lampinen, and Simon Kornblith. Improving neural network representations using human similarity judgments. *Advances in Neural Information Processing Systems*, 36, 2024.

[72] Hieu V Nguyen and Li Bai. Cosine similarity metric learning for face verification. In *Asian conference on computer vision*, pages 709–720. Springer, 2010.

[73] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VI*, pages 69–84. Springer, 2016.

[74] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

[75] Keiron O'shea and Ryan Nash. An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*, 2015.

[76] Haolin Pan, Yong Guo, Qinyi Deng, Haomin Yang, Jian Chen, and Yiqun Chen. Improving fine-tuning of self-supervised models with contrastive initialization. *Neural Networks*, 159:198–207, 2023.

[77] Sungwon Park, Sungwon Han, Sundong Kim, Danu Kim, Sungkyu Park, Seunghoon Hong, and Meeyoung Cha. Improving unsupervised image clustering with robust learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12278–12287, 2021.

[78] Dong Ping Tian et al. A review on image feature extraction and representation techniques. *International Journal of Multimedia and Ubiquitous Engineering*, 8 (4):385–396, 2013.

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

[80] Maithra Raghu, Thomas Unterthiner, Simon Kornblith, Chiyuan Zhang, and Alexey Dosovitskiy. Do vision transformers see like convolutional neural networks? *Advances in neural information processing systems*, 34:12116–12128, 2021.

[81] Yazhou Ren, Jingyu Pu, Zhimeng Yang, Jie Xu, Guofeng Li, Xiaorong Pu, Philip S Yu, and Lifang He. Deep clustering: A comprehensive survey. *arXiv preprint arXiv:2210.04142*, 2022.

[82] Mahdi Rezaei and Mahsa Shahidi. Zero-shot learning and its applications from autonomous vehicles to covid-19 diagnosis: A review. *Intelligence-based medicine*, 3:100005, 2020.

[83] Tal Ridnik, Emanuel Ben-Baruch, Asaf Noy, and Lihi Zelnik-Manor. Imagenet-21k pretraining for the masses. *arXiv preprint arXiv:2104.10972*, 2021.

[84] Samuel Ritter, David GT Barrett, Adam Santoro, and Matt M Botvinick. Cognitive psychology for deep neural networks: A shape bias case study. In *International conference on machine learning*, pages 2940–2949. PMLR, 2017.

[85] Brett D Roads and Bradley C Love. Enriching imagenet with human similarity judgments and psychological embeddings. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, pages 3547–3557, 2021.

[86] Brett D Roads and Michael C Mozer. Predicting the ease of human category learning using radial basis function networks. *Neural Computation*, 33(2):376–397, 2021.

[87] Amir Rosenfeld, Markus D Solbach, and John K Tsotsos. Totally looks like-how humans compare, compared to machines. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1961–1964, 2018.

[88] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252, 2015.

[89] Friedrich Sander and Felix Krueger. *Gestaltpsychologie und Kunsttheorie: ein Beitrag zur Psychologie architektonischer Gestalten*. Beck, 1932.

[90] Marshall H Segall, Donald Thomas Campbell, and Melville Jean Herskovits. *The influence of culture on visual perception*, volume 310. Bobbs-Merrill Indianapolis, 1966.

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

[92] Baifeng Shi, Ziyang Wu, Maolin Mao, Xin Wang, and Trevor Darrell. When do we not need larger vision models? *arXiv preprint arXiv:2403.13043*, 2024.

[93] Joan G Snodgrass and Brian McCullough. The role of visual similarity in picture categorization. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 12(1):147, 1986.

[94] Xiujie Song, Mengyue Wu, Kenny Q Zhu, Chunhao Zhang, and Yanyi Chen. A cognitive evaluation benchmark of image reasoning and description for large vision language models. *arXiv preprint arXiv:2402.18409*, 2024.

[95] Rose M Spielman, William Jenkins, Kathryn Dumper, Marilyn Lovett, and Marion Perlmutter. Psychology (openstax), 2019.

[96] Ilia Sucholutsky and Tom Griffiths. Alignment with human representations supports robust few-shot learning. *Advances in Neural Information Processing Systems*, 36, 2024.

[97] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.

[98] Shikhar Tuli, Ishita Dasgupta, Erin Grant, and Thomas L Griffiths. Are convolutional neural networks or transformers more like human vision? *arXiv preprint arXiv:2105.07197*, 2021.

[99] Amos Tversky. Features of similarity. *Psychological review*, 84(4):327, 1977.

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

[101] Sagar Vaze, Nicolas Carion, and Ishan Misra. Genecis: A benchmark for general conditional image similarity. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6862–6872, 2023.

[102] Jiaqi Wang, Zhengliang Liu, Lin Zhao, Zihao Wu, Chong Ma, Sigang Yu, Haixing Dai, Qiushi Yang, Yiheng Liu, Songyao Zhang, et al. Review of large vision models and visual prompt engineering. *Meta-Radiology*, page 100047, 2023.

[103] Wenhai Wang, Jifeng Dai, Zhe Chen, Zhenhang Huang, Zhiqi Li, Xizhou Zhu, Xiaowei Hu, Tong Lu, Lewei Lu, Hongsheng Li, et al. Internimage: Exploring large-scale vision foundation models with deformable convolutions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14408–14419, 2023.

[104] Wenhai Wang, Zhe Chen, Xiaokang Chen, Jiannan Wu, Xizhou Zhu, Gang Zeng, Ping Luo, Tong Lu, Jie Zhou, Yu Qiao, et al. Visionllm: Large language model is also an open-ended decoder for vision-centric tasks. *Advances in Neural Information Processing Systems*, 36, 2024.

[105] Janelle Weaver. How one-shot learning unfolds in the brain. *PLoS biology*, 13 (4):e1002138, 2015.

[106] Peipei Xia, Li Zhang, and Fanzhang Li. Learning similarity with cosine similarity ensemble. *Information sciences*, 307:39–52, 2015.

[107] Dengsheng Zhang and Guojun Lu. Evaluation of similarity measurement for image retrieval. In *International conference on neural networks and signal processing, 2003. proceedings of the 2003*, volume 2, pages 928–931. IEEE, 2003.

[108] Jingyi Zhang, Jiaxing Huang, Sheng Jin, and Shijian Lu. Vision-language models for vision tasks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[109] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

[110] Yifan Zhang, Bryan Hooi, Dapeng Hu, Jian Liang, and Jiashi Feng. Unleashing the power of contrastive self-supervised visual models via contrast-regularized fine-tuning. *Advances in Neural Information Processing Systems*, 34:29848–29860, 2021.

Appendix A

Dataset Pre-processing

A.1 Fine-tuning Datasets for Supervised and Self-Supervised Paradigms

The below table concretely summarises the datasets used in fine-tuning our models both in a supervised and self-supervised manner.

Dataset ID	**Source**	**Data Point Extraction Method**	**Supervision Method**
Triplet_ImageNet_HSJ	ImageNet-HSJ	Random selection of 1000 triplets out of 2 triplet sets.	Supervised Triplet/Contrastive
Triplet_Birds-16	Birds-16	Random selection of 1000 triplets out of 3 triplet sets.	Supervised Triplet/Contrastive
Triplet_BAPPS_2AFC	BAPPS 2AFC	Random selection of 1000 triplets out of 6 triplet sets.	Supervised Triplet/Contrastive
Quadruplet_ImageNet_HSJ	ImageNet-HSJ	Random selection of 1000 triplets out of 2 triplet sets. An addition of a random dissimilar image from the same 8rank2 trial.	Supervised Quadruplet
Quadruplet_Birds-16	Birds-16	Random selection of 1000 triplets out of 3 triplet sets. An addition of a random dissimilar image from the same 8rank2 trial or just random image from a different 2rank1 trial.	Supervised Quadruplet
Quadruplet_BAPPS_2AFC	BAPPS 2AFC	Random selection of 1000 triplets out of 6 triplet sets. An addition of a random image from different 2rank1 trial.	Supervised Quadruplet
Singleton_ImageNet_HSJ	ImageNet-HSJ	Random selection of 1000 images used in the 2 triplet sets.	Self-supervised All
Singleton_Birds-16	Birds-16	Random selection of 1000 images used in the 3 triplet sets.	Self-supervised All
Singleton_BAPPS_2AFC	BAPPS 2AFC	Random selection of 1000 images used in the 6 triplet sets.	Self-supervised All

Table A.1: Fine-tuning datasets from the source triplet sets. It is to be noted that datasets whose identification contains ”Triplet” are triplet datasets alike the original 11 triplet sets, whilst those with ”Singleton” in their identification are datasets where each data point is a single image. The distinct ”Quadruplet” datasets are used for quadruplet loss in supervised training and contain a triplet and an additional dissimilar image.

Appendix B

Fine-tuning Implementation Details

For reproductibility of results, we include the concrete fine-tuning parameter choices for our models. Optimal fine-tuning parameters were obtained in an experimental manner. Experiments observed that higher learning rates and thus more significant updates seem to work better on self-supervised paradigm. This information is contained in Table B.1.

Base ID	**Supervision of Fine-tuning**	**Loss/Pretext Task**	**Learning Rate**	**Epochs**	**Batch Size**
ResNet-50_BASE	Supervised	ALL	1e-4	20	64
EfficientNetV2-M_BASE	Supervised	ALL	1e-4	20	64
ViT-B-16_BASE	Supervised	ALL	1e-6	30	32
AlexNet_BASE	Supervised	ALL	1e-4	20	64
CLIP_ViT-B-16_BASE	Supervised	ALL	1e-5	25	32
CLIP_ResNet-50_BASE	Supervised	ALL	1e-4	20	64
DINO_ViT-B-16_BASE	Supervised	ALL	1e-6	25	32
DINO_ResNet-50_BASE	Supervised	ALL	1e-4	20	64
ResNet-50_BASE	Self-supervised	SimCLR	1e-4	25	64
ResNet-50_BASE	Self-supervised	MoCo	1e-4	25	32
ResNet-50_BASE	Self-supervised	Rotation	1e-4	25	32
ResNet-50_BASE	Self-supervised	Jigsaw	1e-4	25	32
EfficientNetV2-M_BASE	Self-supervised	SimCLR	1e-4	25	64
EfficientNetV2-M_BASE	Self-supervised	MoCo	1e-4	25	32
EfficientNetV2-M_BASE	Self-supervised	Rotation	1e-3	25	16
EfficientNetV2-M_BASE	Self-supervised	Jigsaw	1e-4	25	16
ViT-B-16_BASE	Self-supervised	SimCLR	1e-5	30	64
ViT-B-16_BASE	Self-supervised	MoCo	1e-5	30	16
ViT-B-16_BASE	Self-supervised	Rotation	1e-5	30	32
ViT-B-16_BASE	Self-supervised	Jigsaw	1e-5	20	32
AlexNet_BASE	Self-supervised	SimCLR	1e-3	30	32
AlexNet_BASE	Self-supervised	MoCo	1e-3	30	64
AlexNet_BASE	Self-supervised	Rotation, Jigsaw	1e-3	30	32
CLIP_ViT-B-16_BASE	Self-supervised	SimCLR	1e-5	30	64
CLIP_ViT-B-16_BASE	Self-supervised	MoCo	1e-5	30	32
CLIP_ViT-B-16_BASE	Self-supervised	Rotation, Jigsaw	1e-5	30	32
CLIP_ResNet-50_BASE	Self-supervised	SimCLR	1e-4	25	64
CLIP_ResNet-50_BASE	Self-supervised	MoCo	1e-4	25	32
CLIP_ResNet-50_BASE	Self-supervised	Rotation, Jigsaw	1e-4	25	32
DINO_ViT-B-16_BASE	Self-supervised	SimCLR	1e-5	30	64
DINO_ViT-B-16_BASE	Self-supervised	MoCo	1e-5	30	32
DINO_ViT-B-16_BASE	Self-supervised	Rotation, Jigsaw	1e-5	30	32
DINO_ResNet-50_BASE	Self-supervised	SimCLR	1e-4	25	64
DINO_ResNet-50_BASE	Self-supervised	MoCo	1e-4	25	32
DINO_ResNet-50_BASE	Self-supervised	Rotation, Jigsaw	1e-4	25	32

Table B.1: Concrete choices for fine-tuning our models. It is to be noted that each of the rows apply for all training dataset sources - 3 per row.

Appendix C

Graph Information

Graph colours have been chosen by testing out a number of helper tools - eventually the following was used: `https://davidmathlogic.com/colorblind`. The palette selected for the graphs can be seen in Figure C.1 - it is to be noted that not all colours are used in a single graph - green/orange or blue/purple, for example, are too visually similar to be used in a single graph.

Figure C.1: Colour palette chosen for the results graphs: the first column shows the unaltered colour and the following rows show the colour seen by individuals affected by certain conditions.