Aspect-based Few-shot Learning

Trinh, M.P. (Phuong)

Award date:
2022

EINDHOVEN UNIVERSITY OF TECHNOLOGY

Department of Mathematics and Computer Science
Data Mining Research Group

Aspect-based Few-shot Learning

Master Thesis

M.P. Trinh

Supervisors:

dr. V. Menkovski (TU/e)

Assessment Committee:

dr. R. Medeiros de Carvalho (TU/e)
L. Yin, MSc. (TU/e)

Eindhoven, October 31, 2022

Abstract

The information encapsulated in image data is inherently rich and complex in semantics. Consequently, image recognition, which commonly boils down to the association of images to a label, is not one-dimensional but is a task that requires taking different aspects into account. Few-shot learning (FSL) is an important machine learning technique in the domain of image recognition to deal with rich semantics and very few examples per concept, which is often times represented in the form of a class label. FSL consists of a query image and a support set of images, where the support set only contains a small number of classes with a few examples each, and the task is to map the query image to one of the support set images. Albeit the remarkable success in various areas in the past decade, which is attributed to its ability to learn from very few examples, FSL does not consider different aspects present in the data. Aspects are often defined by a combination of attributes. For instance, given an image of a lion, one can reason about its species, natural habitats, method of movement and many other characteristics. The use of such attributes significantly facilitates the learning of new concepts with a few visual examples. Attributed to the nature of rich semantics, the support set can describe various aspects, which allows for the association of the query to the support set to be based on multiple aspects. When the support set changes, so do the aspects, and the way entities are associated. However, many machine learning techniques, including FSL, assume that the set of attributes is known and fixed, and typically reduced to a finite set of labels, which fail to fully describe the complicated relationships and semantics present in the data. Without considering the aspects, image recognition with machine learning paradigms like FSL cannot be done properly. There is a lack of appropriate solutions for this problem, perhaps due to the incompatibility of the traditional label-based supervised learning approaches to deal with the variable aspects of high-dimensional data. Moreover, existing methods fail to recognize and incorporate important aspects as they do not consider the support set as a whole. While humans are well-capable in recognizing and understanding the various aspects conveyed in images, it remains a challenge for machine learning paradigms to do so as well.

In this work, we extend the capabilities of FSL models to learn the aspects that are defined by the support set independently of the class labels. To this end, we define a notion of commonalities and aspects. Commonalities refer to all common features, constant factors, and values shared by the support set objects, whereas aspects are the varying factors among these objects. Subsequently, we define a novel problem setting, referred to as aspect-based FSL. In this setting, the goal is to identify the correct assignment to a query by the aspects determined by the support set when there is no exact class label matching between the query and support objects. To address this problem, we propose a novel solution method capable of learning the relevant aspects for each few-shot task. The proposed model comprises a base representation network and the Category Traversal Module (CTM). The CTM allows the model to view the support set as a whole and learn the aspect relevant to each task. Based on the aspect, the model learns to derive the aspect-based representations of the input data that contain the most discriminating and salient features that are relevant to the aspect. Furthermore, we adopt a psychometric testing procedure to effectively capture the relationships between entities and how association differs when the aspects change. We evaluate our method on a synthetically generated geometric shape dataset and empirically demonstrate the feasibility of the approach by assessing its performance against a baseline.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

The process of searching for and listing attributes that can be used to distinguish exemplars from non-exemplars of various categories and the ability to recognize new instances of those categories is a part of learning. Concepts are the mental categories that help us categorize objects, entities, or events, and each of them has a set of common relevant features. Hence, learning the concepts is a strategy that requires a learner to compare and contrast groups or categories that contain concept-relevant features with groups or categories that do not include concept-relevant features [5]. Concept learning also refers to a learning task in which a human or machine learner is trained to classify objects by being shown a set of example objects and their class labels. The learner simplifies what has been observed in a given an example by deconstructing it into simpler bits of information. This simplified version of what has been learned will then be applied to future examples. This task is known as learning from examples. Take, for instance, as illustrated in Figure 1.1, when asked to classify a handwritten character (left) into a given set of different handwritten characters from different alphabets (right) that a human learner has hardly seen before, which requires the mental act of associating, comparing and contrasting, it is utterly trivial for them to give a correct answer quickly.

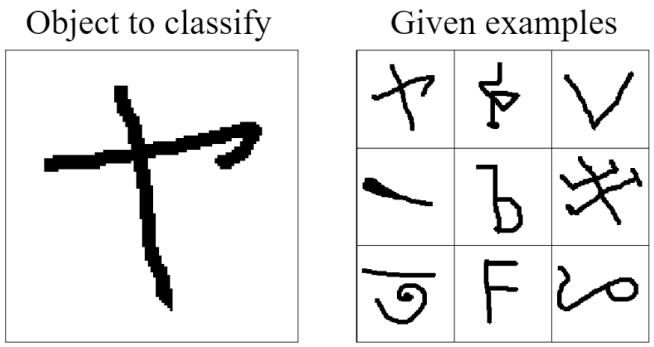


Figure 1.1: Demonstration of a task in which a learner is asked to classify a handwritten character (left) into a given set of different handwritten characters from different alphabets (right) that they have hardly seen before, which requires the mental act of associating, comparing and contrasting.

In a nutshell, learning is about grasping the relationships between different entities. There are many different ways to describe an entity, and the information that can be induced from each entity is oftentimes rich, complex, and varying. This leads to the challenge of how to effectively extract

human perception and knowledge of different entities to construct a good machine learning model. In supervised learning, such information is oftentimes characterized in the form of labels, and the goal is to learn a function that maps an entity to a label based on example entity-label pairs. The past few decades have witnessed remarkable progress and exceptional success in machine learning in general and supervised learning in particular. A substantial part of this progress can be attributed to recent advances in deep learning, a class of machine learning algorithms characterized by neural network-style models that progressively use multiple layers to extract higher-level features from the raw input data. These models have shown outstanding performance in many data-intensive applications with large amounts of labelled data, with a significant remark in image classification. However, these achievements hinge upon the fact that the optimization of these models requires many iterative updates across many labelled samples. Albeit its success, the performance of this type of optimization is oftentimes diminished in the small data regime where there are only very few labelled examples available [32]. As such, many machine learning solutions that require a trove of data oftentimes suffer from data scarcity. In many situations, obtaining or accruing more data is hardly feasible due to inherent scarcity, privacy and confidentiality concerns, time constraints, as well as the high cost of data preparation and annotation [28]. Under these circumstances, standard training approaches lead to significant overfitting, thus prompting the emergence of a setting where every category in a task has only a few examples available. Furthermore, typical supervised learning techniques rely on a predefined and finite set of labels. That is, the models learn to detect the underlying patterns and relationships between the input data and the fixed output labels, which only yield accurate results when presented with never-before-seen data if that data truly belongs to the fixed, predefined set of labels. Such methods face many limitations when the unseen data comes from novel classes that have not been introduced to the models before. In this context, where new classes are constantly introduced to the classification task, traditional supervised learning methods are rather limited. While this is considered a challenge for many machine learning approaches, humans, on the contrary, are highly capable of dealing with such tasks. This is attributed to the humans' ability to grasp the essential connection between new concepts and their previously learned knowledge and make an association, which essentially accentuates the gap in the ability to learn from very few given examples between humans and machine learning models. These constitute the main factors that drive the increase in exploration and exploitation of few-shot learning, as few-shot learning emerges as a solution to alleviate the burden of data collection drastically and computational costs while still attaining high performance.

1.1.1 Few-shot learning

Few-shot learning (FSL) refers to the practice of learning the underlying patterns in the data just from a very few examples [38]. When there is only one example to learn from, FSL is referred to as one-shot learning. The basic principle behind FSL is to learn a function or a model to recognize unseen classes during training with a very limited number of labelled examples per class. This is done by means of learning through a set of training few-shot tasks and evaluating using a set of test few-shot tasks. Here, each few-shot task consists of a small number of classes with a few examples each. Additionally, there are further examples of the same classes, known as a query set. In each task, the objective is to correctly assign a class label to each query instance, where the classes are specified by a support set which consists of only a few examples. The few-shot tasks can be completely non-overlapping, that is, the classes from one task may not be seen in any of the others. The idea is that the machine learning model repeatedly sees different tasks during training that match the structure of the final few-shot task but contain different classes. As such, the model is enforced to learn to classify image classes in general rather than to recognize any particular set.

Given the ability to attain good learning performance while requiring only limited supervised information, FSL has received much attention in various areas as it not only addresses the many challenges that traditional supervised learning faces but also meets the industrial demands for cost-

effective learning. Many practitioners face the problem of leveraging the power of machine learning to accelerate product development but not having sufficient data. Building FSL frameworks hence leads to less mundane tasks, faster development, and accelerated product delivery.

1.1.2 Challenges of few-shot learning

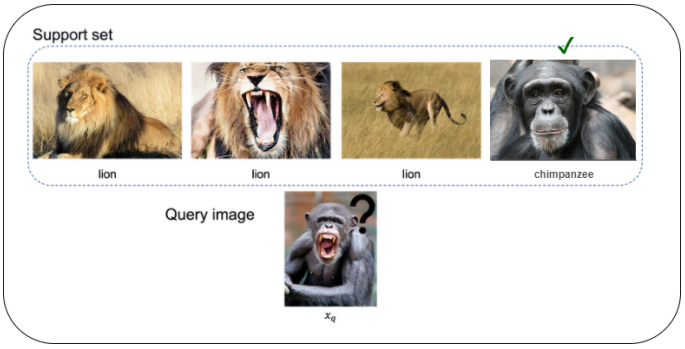
FSL has received a surge of attention since its conception and has achieved state-of-the-art performance in various fields, such as computer vision, natural language processing, audio processing, and robotics. As the goal of FSL is to learn from just a few examples, the main challenge is to make the best use of the limited data available in the support set to find the correct generalizations as suggested by the task [22]. In this work, we focus on how this challenge can be alleviated in the domain of image recognition. The information conveyed in the support set images are typically rich and complex. Hence, associating one entity with another can be multidimensional and conditioned on different factors and aspects. In the context of the support set, those factors may oftentimes be uncovered. Consider the example of different one-shot-four-way classification tasks presented in Figure 1.2. From a human perspective, in the first example (top), the association between the query image and the image from the support set that matches the most is based on the aspect of animal species, as they both represent a chimpanzee, whereas the rest are lions. In the second example with the same query image (middle), all the images in the support set are lions, with each image representing the lion in a different physical movement, whereas the query instance is an image of a roaring chimpanzee. In this case, the association is based on the aspect of physical movement, and classification based on this aspect would match the query image to the second image in the support set as they both exhibit the movement of roaring. In the third example (bottom), the support set is identical to that in the first example. However, from a different perspective, the differences between these images of lions can also be perceived as whether the lion is a real animal or a toy animal instead of physical movement. If judging by this aspect, the query image would match with the fourth image in the support set as they are both toy animals. This is to explicate that the aspect based upon which an association is made can vary; and that when the aspect changes, the relationship between different images changes. What the examples have also shown is the aspects are defined by all images in the support set and when the support set changes, so does the way associations and mappings are done.

While these details are easily recognized and well comprehended by humans, it is often overlooked by machine learning models, as such rich and complex information is typically reduced to a predefined set of labels. Labels, in their nature, create an absolute and direct mapping between entities and do not fully describe those entities. Existing few-shot learning approaches heavily rely on the assumption that all data points have a unique mapping to a class label in a given set of labels; and that the class label of the query must precisely match one of the support set class labels. When there is an exact class matching, as shown in Figure 1.2a, a few-shot learning model would be able to correctly assign a label to the query image. However, due to such dependencies on the class labels, few-shot learning models that are capable of dealing with classification based on the predefined classes fail to identify the right class assignment when there is no exact class matching, as shown in Figure 1.2b and 1.2c.

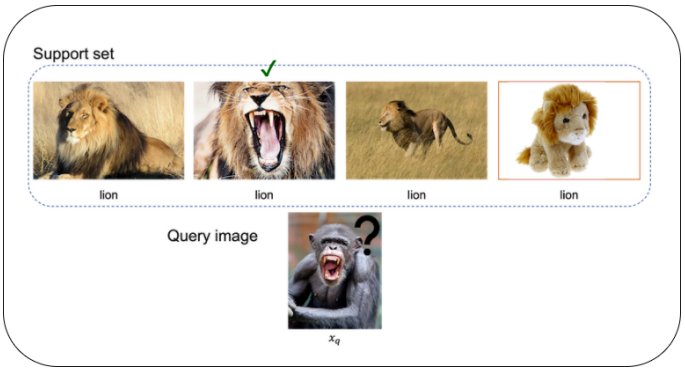
This is an intricate problem. In real life, the associations between entities are not absolute; they are conceptual. As such, image recognition, which in hindsight boils down to the association of images to a label is not one-dimensional, is a task that requires taking different aspects into account. Humans are very capable of inferring these aspects, whereas conventional few-shot learning approaches are not specifically designed for this under label-based supervision and single discriminating class descriptions. Existing methods treat each class equally without considering their intra-conceptual relationships. Without taking into account the aspects, image recognition with machine learning paradigms like FSL cannot be done properly. In the exigency of exploiting useful and valuable information in present-day society, it is of paramount importance to build aspect-based approaches that enable machine learning models to effectively conduct the task of few-shot learning by alleviating the challenges that existing approaches are facing. A possible reason for the

lack of a solution for this problem formulation lies in the incompatibility of the traditional label-based supervised learning approaches for dealing with the variable aspects of high-dimensional data. It is thus imperative to investigate and extend the capabilities of conventional few-shot learning models to aspect-based few-shot learning, i.e., to learn the aspects given the support set objects, agnostic of the predefined labels.

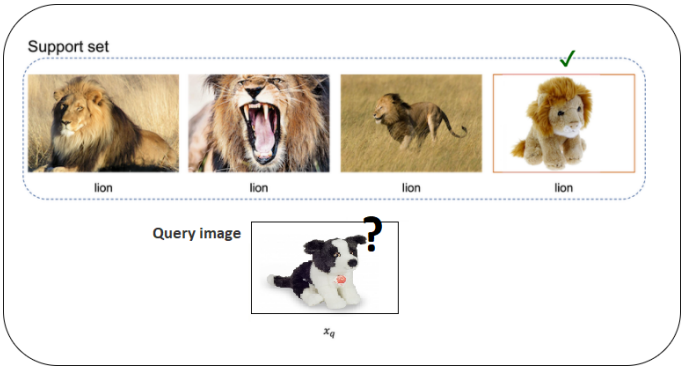
To this end, in this work, we define among the objects in the support set the notion of commonalities and *aspects*. Commonalities refer to all common features, constant factors, and values shared by the support set objects, whereas aspects are the varying factors among these objects. In other words, in a given support set, different aspects vary. With a clear notion of aspect, we formulate the problem setting of aspect-based FSL. Subsequently, we propose a framework that is capable of learning the aspect described by the support set independently of the given labels. The goal behind the developed framework is to be able to identify the correct assignment to query by the learned aspect when there is no exact class matching between query and support objects in few-shot tasks.



(a) A four-way-one-shot classification task. Classification based on the predefined label will assign the query image the class label "chimpanzee", which is correct. This is the same with classification based on the aspect of animal species.



(b) A four-way-one-shot classification task. All the images in the support set are of the class "lion", with each image representing the lion in a different physical movement, whereas the query instance is an image of a roaring chimpanzee. Classification based on the predefined label "lion" will assign the chimpanzee the class label "lion", which is not sensible. Classification based on the aspect of physical movement would match the query image to the second image in the support set as they both exhibit the movement of roaring.



(c) A four-way-one-shot classification task. All the images in the support set are of the class "lion", with each image representing the lion in a different physical movement, whereas the query instance is an image of a toy dog. Classification based on the predefined label "lion" will assign the dog the class label "lion", which is not sensible. Classification based on the aspect of whether that is a real animal or a toy animal toy would match the query image to the fourth image to the left in the support set, as they are both toy animals.

Figure 1.2: Examples of different few-shot tasks.

1.2 Research objective

Many approaches in FSL are about the correct association of a query object to one of the support set objects. The problem becomes aspect-based when the association is done conceptually, i.e., based on a specific aspect described by the support set instead of a fixed, predefined set of labels. We refer to the problem of learning from a limited number of available examples based on the aspects determined by the support set, agnostic of the predefined set of labels as aspect-based few-shot learning. The main objective of this work is to devise a solution approach that solves the problem of aspect-based few-shot learning. This is encapsulated in the main research question:

Given varying aspects in the support set, how can a method effectively learn the relevant aspect for accurately querying objects for each few-shot task in few-shot learning?

In order to address the main research question, a number of sub-research questions and high-level tasks have been defined.

(i) *How to implement an efficient procedure such that the humans' perception and knowledge of objects can be elicited without mapping to a predefined set of labels?*

(ii) *How can the relevant class-independent aspects be identified and learned from the support set?*

In order to incorporate the aspects defined by each support set, we develop a framework in which a module is bundled into the feature embeddings of the support and query set, making learning in the subsequent feature space more effective. This method is evaluated by assessing the performance against a standard few-shot learning baseline in the setting of aspect-based few-shot learning.

1.3 Contributions

This work contributes to the literature by providing insights into the following aspects:

- We define the novel problem of aspect-based few-shot learning. This problem setting aims to identify the correct assignment to query object by the aspects determined by the support set when there is no class matching between query and support objects.
- We analyze the limitations of label-based supervision under the aspect-based few-shot learning setting and propose an extension of the psychometric testing-based approach to overcome these limitations.
- We propose a solution method and evaluate our method by comparing the performance with a standard baseline in the setting of aspect-based few-shot learning.

1.4 Research outline

This section provides a general overview of what each of the following chapters entails. Chapter 2 introduces the components that are fundamental to the construction of our solution approach together with the necessary notations and definitions. Chapter 3 provides an overview of the relevant literature in the current field of few-shot learning and where our work is positioned. In Chapter 4, we discuss the role of aspects in how relative distances are perceived and introduce the notion of an aspect-based embedding space, which leads to the formal formulation of the established problem. The proposed framework, which entails data generation, representation learning, psychometric testing and the construction of the aspect-based representation network is detailed in Chapter 5. Chapter 6 presents the experiment setup and the performance metrics. The experimental results are presented in Chapter 7, which is followed by a discussion on the model performance. Chapter 8 concludes this work with a summary of our contributions and a discussion

on the limitations that this work bears, which prompts the recommendations for possible future work based on this work.

Chapter 2

Research background

In this chapter, we introduce the relevant background information together with the necessary mathematical notations and definitions. We first introduce the basis of convolutional networks. Secondly, we formally introduce few-shot learning. We conclude this chapter by presenting the fundamentals of psychometric testing.

2.1 Convolutional neural networks

Convolutional Neural Networks (CNN) are a specialized class of neural networks designed to process data in the form of multiple arrays. They have been a prominent method in computer vision tasks [21]. Examples of such data are images and signals, which belong to the class of spatially distributed data. Data is considered spatially distributed when useful correlations of the features based on their spatial position relative to other features can be exploited. In the case of images, correlations are localized in small regions, this means that pixels that are positioned close to each other are highly correlated. When it comes to dealing with such data, CNN has been a dominant method as it is designed to use local receptive fields to capture the spatial dependencies existing in the images, from low- to high-level patterns and allows for the possibility of sub-sampling [21]. Furthermore, the parameters are re-used at each location which significantly reduces the number of parameters. Since image data is inherently high-dimensional, the neural network is required to have many parameters to train which is susceptible to overfitting if not enough data is made available for training. Overfitting is a modelling error that occurs when the model corresponds too closely to the training data and may therefore fail to fit unseen data or predict future observations in a reliable manner. Using fewer parameters thus helps reduce the risk of overfitting. In addition, the memory requirement to store all the parameters can lead to limitations of computational power. CNN, therefore, has a great impact as it allows for significantly more efficient training and inference processes. CNN is a mathematical construct that is typically composed of three types of layers (or building blocks): convolution, pooling, and fully connected layers [21]. Convolution and pooling layers perform feature extraction, whereas a fully connected layer maps the extracted features to the final output. A typical architecture consists of repetitions of a stack of several convolution layers and a pooling layer, followed by one or more fully connected layers. Figure 2.1 provides an overview of a CNN. In the following sections, we present the key elements of each layer type.

2.1.1 Convolutional layers

In a CNN, the input is a tensor of the shape: (number of inputs) $\times$ (input height) $\times$ (input width) $\times$ (input channels). After passing through a convolutional layer, the image is reduced to a feature

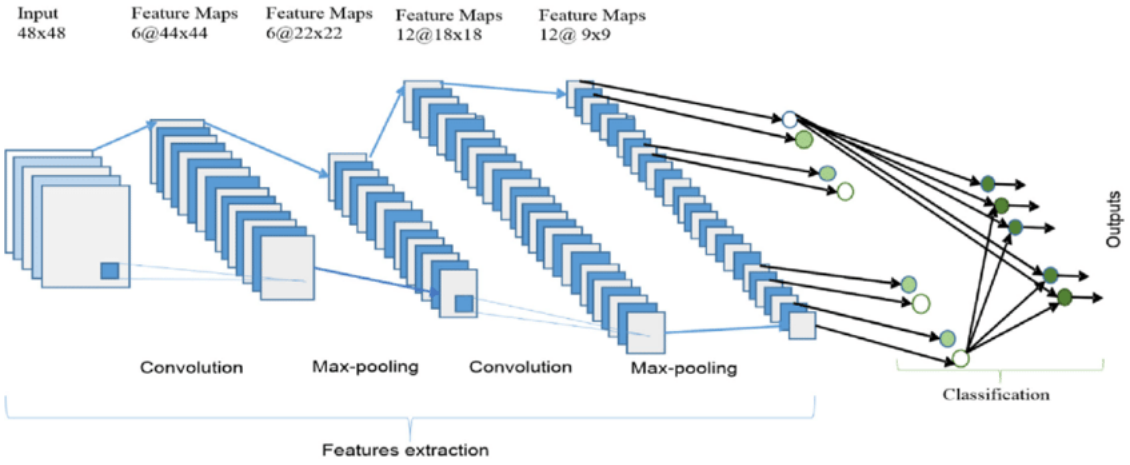


Figure 2.1: An overview of a convolutional neural network (CNN) architecture [1].

map, also called an activation map, with shape: (number of inputs) × (feature map height) × (feature map width) × (feature map channels).

Convolutional layers convolve the input and pass its result to the next layer. The neurons in a convolutional layer are organized in feature maps and each feature map receives all the feature maps from the previous layer. Each unit processes a local patch from every previous feature map using a group of weights called a kernel which is shared among all the units in the same feature map.

The key aspect of a convolution operation is weight sharing where kernels are shared across all the image positions. Weight sharing determines the following characteristics of convolution operations: First, it lets the local feature patterns extracted by kernels be translation invariant as kernels travel across all the image positions and detect local patterns. Secondly, it allows for learning the spatial hierarchies of feature patterns by downsampling in conjunction with a pooling operation, resulting in capturing an increasingly larger field of view. Lastly, it reduces tremendously the number of training parameters and thus accelerates model efficiency.

Convolution
An element-wise product between each element of the kernel and the input tensor is calculated at each location of the tensor and summed to obtain the output value in the corresponding position of the output tensor, called a feature map, see Figure 2.2. The kernel traverses the input image from the top left to the bottom right and typically is a block consisting of 3×3 pixels.

Stride
The distance between two successive kernel positions, i.e., how many pixels the kernel needs to move to perform the next filter operation is called a stride. The size of the convolutional layer output depends on the stride size where larger strides will produce smaller outputs.

Padding
The act of convolution results in a lower dimensional image. A technique to avoid this is to populate the borders of the image with pixels of value 0 in a way that the output dimension remains the same.

2.1.2 Pooling layers

Whereas the convolutional layer is to detect local patterns from the previous layer, the pooling layer merges similar features semantically. The feature maps obtained after a convolutional layer capture the exact position of the local patterns they detected and, in consequence, small changes

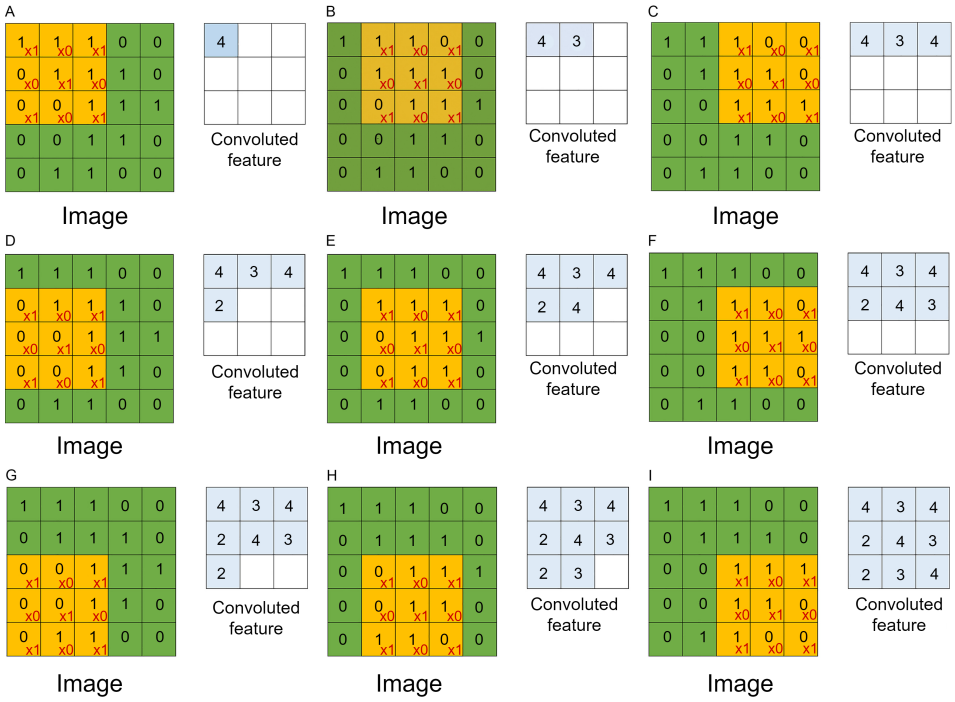


Figure 2.2: An example of convolution operation adopted from [31] with a kernel size of 3×3, no padding, and a stride of 1. A kernel is applied across the input tensor, and an element-wise product between each element of the kernel and the input tensor is calculated at each location and summed to obtain the output value in the corresponding position of the output tensor called a feature map.

in the position of the motif in the input images can result in a different feature map. This problem is alleviated by adding a local or global pooling layer after the convolutions. A pooling layer provides a typical downsampling operation by "pooling" the values of a local region into a single value, which in turn reduces the in-plane dimensionality of the feature maps in order to introduce translation invariance to small shifts. Each pooling unit aggregates the information contained in a local patch of units in a feature map and the aggregation can be done using the maximum (Maximum Pooling) or the average value (Average Pooling). Neighbour pooling units take input from patches that are shifted by more than one unit in the feature map, hence reducing dimensionality and the position of each motif is coarse-grained so the feature maps still contain important structural elements without fine details [21].

2.1.3 Nonlinear activation function

The outputs of a linear operation such as convolution are passed through a nonlinear activation function. The most common nonlinear activation function is the rectified linear unit (ReLU), which computes the function:

$$f(x) = \max(0, x).$$

Other activation functions that are frequently used are the sigmoid and hyperbolic tangent (tanh) functions (Figure 2.3) [24, 30].

2.1.4 Fully connected layer

Typically, the output feature maps of the final convolutional or pooling layer is flattened into a one-dimensional vector and connected to one or more fully connected layers. These are also known as dense layers where every input is connected to every output by a learnable weight. Once the

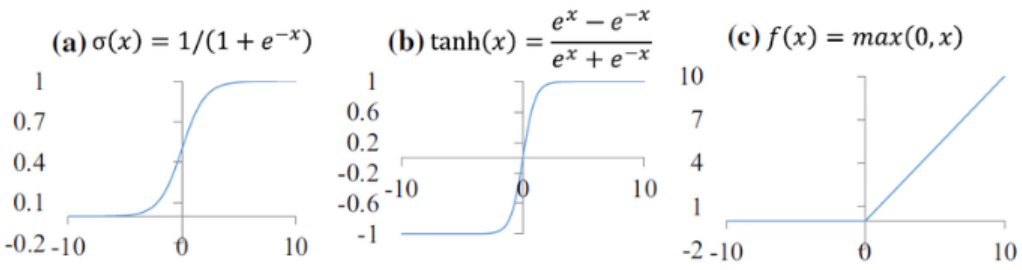


Figure 2.3: Different nonlinear activation functions [42].

features are extracted by the convolution layers and down-sampled by the pooling layers, they are mapped by a subset of fully connected layers to the final outputs of the network [46].

2.2 Few-shot learning

Inspired by how humans learn new information from previously learned information across tasks, meta-learning, also known as *learning to learn*, aims to learn new concepts or adapt to new environments efficiently from previous experiences that can lead to efficient downstream learning of new tasks. More specifically, meta-learning refers to a set of machine learning techniques that observe how other machine learning algorithms perform on a wide collection of learning tasks and then based on this prior experience (i.e., *meta-data*) learn how to learn new tasks in a faster fashion with fewer given examples in changing environment [18]. A well-known instantiation of meta-learning is few-shot learning (FSL). Few-shot learning refers to the problem of learning from very few examples. When there is only one example to learn from, FSL is referred to as one-shot learning. Contemporary approaches to FSL often decompose the training process into an auxiliary meta-learning phase where transferable knowledge is learned in the form of good initial conditions [10], embeddings [38] or optimization strategies [32]. More specifically, the transferable knowledge is learned from a "base" dataset where ample labelled examples are available to generalize to a "novel" dataset which also has very few labelled examples in an episodic training procedure. Formally, the few-shot classification problem consists of a task $\mathcal{T}$ with a dataset $\mathcal{D} = \{(x_k, y_k)\}_{k=1}^{n}$ with n data samples where x is the input and y is the output label. The dataset $\mathcal{D}$ is typically split into a training dataset $\mathcal{D}^{\text{train}} = \{(x_k, y_k)\}_{k=1}^{t}$ and a test dataset $\mathcal{D}^{\text{test}} = \{(x_k, y_k)\}_{k=t+1}^{n}$, where t denotes the number of training samples. We optimize parameters θ on the training set $\mathcal{D}^{\text{train}}$ and evaluate its generalization performance on the test set $\mathcal{D}^{\text{test}}$. Thus, the learning problem here is to approximate the function f with parameters θ as:

$$y \approx f(x;\theta) \text{ where } (x,y) \in \mathcal{D}^{\text{test}} \tag{2.1}$$

and

$$\theta = \arg\min_{\theta} \sum_{(x,y)\in\mathcal{D}^{\text{train}}} \mathcal{L}(f(x;\theta), y), \tag{2.2}$$

where $\mathcal{L}$ is a given loss function that measures the error between the prediction $f(x;\theta)$ and the true class label y.

The few-shot classification task is often referred to as a standard N-way-K-shot task, where N is the number of classes and K denotes the number of samples per class present in $\mathcal{D}^{\text{train}}$. In this setting, K is typically small (e.g., $K \leq 10$) and $t = |\mathcal{D}^{\text{train}}| = N \times K$. The performance is measured by an appropriate loss function $\mathcal{L}(\hat{y}, y)$ defined over the prediction $\hat{y} = f(x;\theta)$ and the true label y. The N-way-K-shot tasks are usually sampled from a larger dataset which contains much more than N classes. Each task of N-way-K-shot classification $\mathcal{T}_i$ comprises the following components:

- A support set $\mathcal{S}_i$ consisting of N class labels and K labelled instances for each class where K is typically small.
- A query set $\mathcal{Q}_i$ containing Q query samples.

Figure 2.4 demonstrates an example of the few-shot learning paradigm. In each task, the objective is to correctly assign a class label to each query x_q, where classes are specified by the support set, which consists of only a few examples. The target few-shot learning problem is learned by fine-tuning [10], with the learned optimisation strategy [32] or computed in a feed-forward pass [3, 38] without updating the network weights.

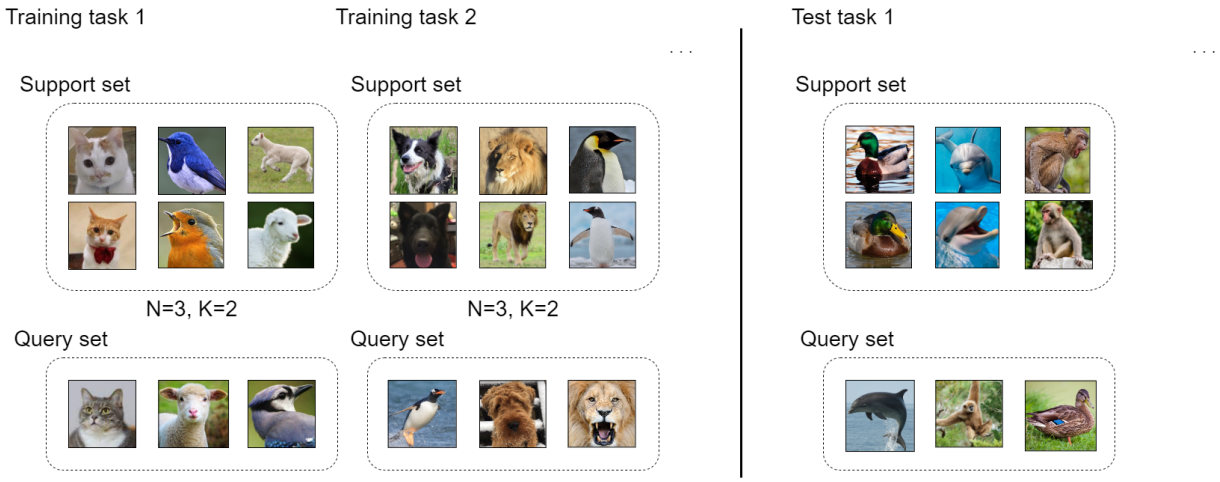


Figure 2.4: A model is trained using a series of few-shot tasks. Here, each task is a 3-way-2-shot classification problem where the support set comprises three different classes with two examples per class. During training, the cost function assesses performance on the query set for each task given the respective support set. At test time (inference), a completely different (unseen) set of tasks is used, and the performance is evaluated on the query set, given the support set. Note that there is no overlap between the classes in the training tasks and between those in the test tasks, enforcing the model to learn to classify image classes in general rather than to recognize any particular set.

Each task can be completely non-overlapping, that is, the classes from one task may not be seen in any of the others. At each step of the meta-learning scheme, the model parameters are updated based on a randomly selected training task. The loss function is determined by the classification performance on the query set of this training task. The network is presented with a different task at each time step, enforcing it to learn how to discriminate data classes in general, rather than a particular subset of classes present in one single learned task. To evaluate few-shot performance, a set of test few-shot tasks is used. Each task contains only unseen classes that were not in any of the training tasks and measure performance on the query set based on the knowledge learned from the corresponding support set.

2.3 Metric learning

A metric or distance function is a function $d(x, y)$ that defines the distance between elements of a set. If the distance is zero, both elements are said to be equivalent under that specific metric. The larger the distance, the more dissimilar the elements are. Distance functions thus provide a way to measure the similarity between two objects, e.g., vectors or matrices. The use of distances in machine learning has been present since its inception. The smallest distance between two points is known as the Euclidean Distance, which is the most prevalent distance metric in machine learning. Mathematically, this is the L_k-norm where $k = 2$. The k norm for any $k \in \mathbb{Z}$ and $x, y \in \mathbb{R}^N$ is

defined as:

$$L_k(x, y) = \sum_{i=1}^{N} (||x_i - y_i||^k)^{\frac{1}{k}},$$

where N denotes the dimensionality of the space in consideration. This is also known as the Minkowski distance.

Metric learning is the task of learning a distance function over data samples. Consider two image-label pairs (x_1, y_1) and (x_2, y_2) and a distance metric d to measure the distance between two images. To assign a label to a query image x_3, we could compute the two distances $d(x_1, x_3)$ and $d(x_2, x_3)$ and assign the label corresponding to the image with a smaller distance. However, with high-dimensional inputs such as images, we typically calculate the distances between the image embeddings instead. The embeddings are obtained using an embedding function g to transform the input to a lower dimension before computing the distances:

$$g : \mathbb{R}^n \to \mathbb{R}^m, \text{ where } n > m.$$

Therefore, the core idea behind metric-based few-shot learning is to either learn an embedding function $g(;\theta_1)$, which is parameterized by a neural network with parameters θ_1 given an appropriate distance metric d (such as the cosine or Euclidean distance), or to learn both the embedding function $g(;\theta_1)$ (with parameters θ_1) and the distance function $d(;\theta_2)$ (usually parameterized by another neural network with parameters θ_2).

Training proceeds by randomly sampling N-way-K-shot episodes from the training set $\mathcal{D}^{\text{train}}$. Each few-shot task $\mathcal{T}_i$ has a support set $\mathcal{S}_i$ and a query set $\mathcal{Q}_i$. The average error computed on query sets across multiple training few-shot tasks is used to update the parameters of the embedding function and the distance function (if any). Finally, new N-way-K-shot tasks are sampled from the testing set $\mathcal{D}^{\text{test}}$ to evaluate the performance of the neural network.

2.4 Psychometric testing

Psychometric testing is a well-studied research area that concerns the perceptual processes under measurable psychical stimuli [12]. Such stimuli may be lights with different brightness levels or tones with various intensities. In practice, psychometric testing procedures are based on either one of the two methods, namely the absolute threshold method and the discriminative-based method. The absolute threshold method is referred to as the method of limits. In this method, a stimulus is presented to a participant in descending or ascending order of intensity and the goal is to detect the lowest intensity of a stimulus that could be noticed by the participant. An example of this method would be to find the amount of force required for one to detect the feeling of a feather lightly brushing on the arm. The discriminative-based method, on the other hand, aims to find the slightest difference between two stimuli that a participant could perceive. Participants might be asked to describe the difference in direction or magnitude between these two stimuli or forced to choose between the stimuli concerning a specific parameter of interest. The latter is also referred to as a two-alternative forced choice (2AFC) method [8]. An example of such an experiment setup would be to present two audio stimuli to a participant and ask the participant to indicate which one is perceived as louder. To scale the perception of loudness, the experiment would consist of two pairs of stimuli, and the participant would need to answer which pair has a larger difference in loudness. Such experiments were also adopted for estimating the quality of complex multimedia content such as images or videos by measuring the perception of the participants [40, 9]. In this method, participants are asked to discriminate between two "intensities" (or "intervals") where no qualitative rating and scales are involved. Thus, it is far less susceptible to biases and the participants typically present lower variability, which prompts the 2AFC method to be the method of choice in many research designs [35].

Chapter 3

Literature Review

In this chapter, we first discuss current state-of-the-art deep metric learning techniques that are relevant to our research and provide an overview of the existing solution approaches to problems of a similar nature to our challenge. Subsequently, we provide an overview of psychometric testing, a line of research that is instrumental to the construction of our methodology.

3.1 Deep metric learning

Approaches to few-shot learning can be categorized in two groups: meta-learning-based FSL and non-meta-learning-based FSL. Meta-learning-based FSL methods aim to approximate the function f with parameters θ such that the performance on any few-shot task $\mathcal{T}_i$ sampled randomly from the task distribution $p(\mathcal{T})$ is optimal. Consider the N-way-K-shot classification task. During training, the meta-learner learns a prior θ over the distribution of N-way-K-shot classification tasks so that at test time we can solve a new task. The non-meta-learning-based FSL methods leverage the mechanism of transfer learning because in an FSL setting, where the data is too limited to train a neural network from scratch, transferring knowledge from another network can be a viable option. For a classification task, this knowledge transfer is obtained by pre-training a deep neural network on large amounts of training data on the base classes (seen), and then fine-tuning it on new few-shot classes (unseen). The caveat with this approach is that naive fine-tuning using just a few examples can lead to overfitting and thus result in poor generalizability on the new few-shot task. Therefore, many non-meta-learning-based FSL approaches focus on addressing this problem. In this work, our main focus belongs to the class of meta-learning-based FSL. Literature suggests that meta-learning-based FSL approaches can be categorized into three groups, namely metric-based, optimization-based and model-based meta-learning. Consider a few-shot task $\mathcal{T}$ with support set $\mathcal{S}$ and query set $\mathcal{Q}$. Let f be a few-shot classification model with parameters θ. Then, for $(x, y) \in \mathcal{Q}$, where x is the input image and y is the true class label, these groups of meta-learning approaches to FSL can be differentiated in the way the the output probability $P_\theta(y|x)$ is modeled [28], as summarized in Table 3.1. Our solution approach belongs to the metric-based method. Figure 3.1 provides an overview of the approaches to the problem of FSL and shows the position of our work in recent meta-learning-based FSL literature.

Metric-based methods aim to learn a distance function over examples by making the prediction conditioned on the distance to a few labelled instances during the training phase [6]. Models that fall into this category are designed to learn the embedding vectors extracted from the input data explicitly and use them to design proper kernels. Metric learning, in combination with deep learning, collectively form the concept of deep metric learning. The core idea is to train a CNN that embeds the extracted prominent image features that are semantically similar (i.e., belonging to the same class) onto nearby locations in the learned metric space while keeping dissimilar image

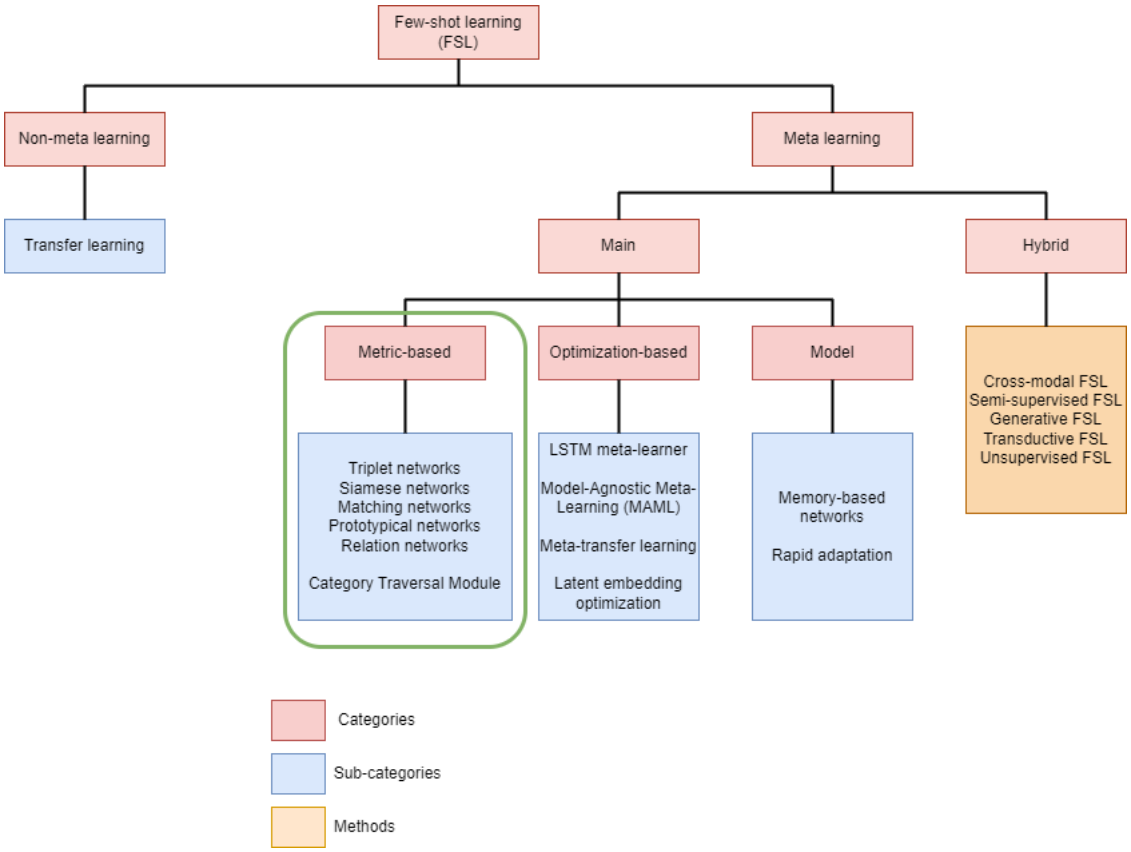


Figure 3.1: Overview of FSL approaches, which are categorized into meta-learning-based methods and non-meta-learning-based methods. Variations of the FSL problem which use meta-learning are categorized as hybrid approaches. The scope of our work is highlighted in the green box.

features (i.e., belonging to different classes) apart [14]. Examples of models and an appropriate distance metric include the Euclidean distance in convolutional Siamese neural networks [20] and in prototypical networks [38], the cosine distance in matching networks [44], or the CNN-based relation module in relation networks [41].

Convolutional Siamese neural networks for the one-shot learning problem comprise a pair of identical CNNs with shared weights [20]. These two networks are joined at their output, where the joining neurons measure the distance between the feature vector output of each network. The network was trained to distinguish whether the two input images belong to the same class and to output a probability if the two images belong to the same class as a measure of similarity. This idea was further extended to perform one-shot recognition by comparing the similarity score between a query image and support images from different classes. The learned embedding function g_θ is a CNN and the distance metric to compute the similarity between the two embeddings x_1, x_2 is the L1 distance, i.e., $|g_\theta(x_1) - g_\theta(x_2)|$.

The idea behind prototypical networks is that there exists an embedding in which points cluster around a single prototype representation for each class. Leveraging this concept, Snell et al. [38] proposed a 4-layer CNN model that learns a non-linear mapping of the input into an embedding space. In this embedding space, a prototype for each class is defined by taking an average of embedding vectors obtained from the support images belonging to that class. Few-shot classification is then performed for an embedded query point by simply finding the nearest class prototype using the Euclidean distance.

	Metric-based	**Optimization-based**	**Model-based**
Key idea	Metric learning	Gradient descent	Memory, Recurrent Neural Networks
How $P_\theta(y\|x)$ is modeled	$\sum_{(x_k,y_k)\in\mathcal{S}} k_\theta(x,x_k)y_k$ where $k_\theta(x_1,x_2)$ is the kernel function measuring similarity between two vectors x_1 and x_2	$P_{\theta'}(y\|x)$ where $\theta' = g_\phi(\theta,\mathcal{S})$	$f_\theta(x,\mathcal{S})$
Advantages	Faster inference. Easy to deploy.	Offers flexibility to optimize in dynamic environments. $\mathcal{S}$ can be discarded post-optimization.	Faster inference with memory models. Eliminates the need for defining a metric or optimizing at test time.
Disadvantages	Less adaptive to optimization in dynamic environments. Computational complexity grows linearly with $\|\mathcal{S}\|$ at test time.	Optimization at inference is undesirable for real-world deployment. Prone to overfitting.	Less efficient to store data in memory as $\mathcal{S}$ grows. Hard to design.

Table 3.1: Approaches to meta-learning-based FSL. The summary is provided by Parnami and Lee [28].

Matching networks define a probability distribution over the output labels y using an attention kernel [44]. The attention kernel computes the cosine similarity between the embeddings of support and query examples. Furthermore, the authors introduced the notion of full context embeddings, that is, to embed the elements of the set through a function that takes as input the full support set $\mathcal{S}$ in addition to the image input x.

A well-established deep metric learning model is triplet networks, which comprises three identical CNNs (with shared parameters) that are trained using triplet loss with triplets of samples: an anchor sample, a positive sample of the same class as the anchor, and a negative sample of a different class [16]. When fed with a sample triplet, the network outputs two intermediate values: the Euclidean distance between the embedded representations of the positive sample and the anchor, and between the negative sample and the anchor. The idea is that the distance from the anchor to the positive sample is minimized whereas the distance from the anchor to the negative sample is maximized. By enforcing the order of distances, triplet models embed the data in such a way that a pair of samples with identical labels are closer than those with different labels. Triplet networks are superb to Siamese networks in the sense that they can learn both distances simultaneously, and the number of combinations of training data circumvents overfitting. A great part of the research has been devoted to the exploration of more effective methods for selecting informative positives and negatives for training triplets [14, 16, 36, 45, 13, 26, 43, 25].

Recent years have also witnessed interesting works that explore the relationship between the support and query sets to enable more complex comparisons between support and query features. The relation networks evaluates the relationship of each query-support pair using a CNN, where the representations of the support and query examples, obtained from the *embedding module* are concatenated together and fed into the *relation module* to obtain a relation score between each query and a support class [41]. The relation score for a match should be 1 and 0 for a no-match. The network is then trained to minimize the mean squared error of relation scores. During test time, a relation network classifies images of new classes by computing relation scores between query images and the few examples of each new class without further updating the network. A transductive propagation network learns a graph construction module that exploits the manifold structure in the data to propagate labels from known labelled instances to unlabeled test instances [23]. The notion of the metric space was extended in [27] by making it task-dependent via conditioning the feature extractor on the specific task. Furthermore, the interaction between the conditioned feature extractor and the training procedure based on auxiliary co-training on a simpler task was exploited.

3.2 Category Traversal Module

While attempting to exploit the relationship between the support set and the query using different approaches, what all aforementioned studies share is that their approaches are based on a metric learning framework that compares the query to each support set object separately and independently. Therefore, these approaches are constrained to use a single set of features for all possible tasks during inference, which hinders the ability to distinguish the most relevant dimensions for the task at hand. Category Traversal Module (CTM) was introduced to overcome this hurdle [22].

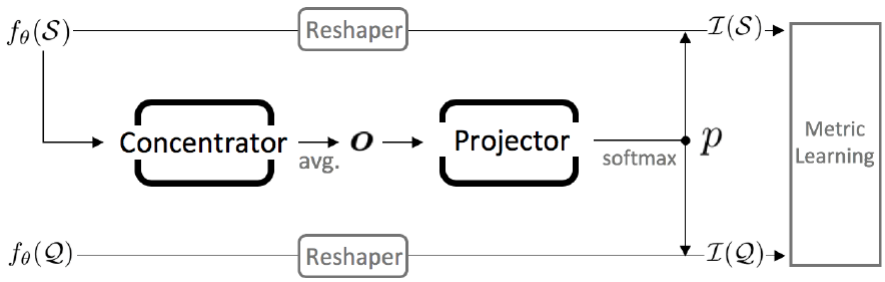


Figure 3.2: Overview of CTM [22].

A simplistic example was presented in which the instances in the support set are simple shapes that can be distinguished in two dimensions: color and shapes. Each example in the support set has a unique color (purple, blue, green, yellow and red), but shares its shape with other support examples (either circle or triangle) whereas the query at hand is a green circle. In this case, the relevant feature is color, hence, the correct label assignment to the query should be the class label "green". However, if feature similarity to the query is calculated independently for each class, it is impossible to determine which dimension (color or shape) is more relevant, hence the distance from the query to all support set examples that share either shape (circle) or color (green) is the same. Viewing the support set as a whole, CTM effectively finds the dimensions that are most relevant to each task. That is, CTM selects the most relevant feature dimensions after traversing both across and within categories to find the intra-class commonality and inter-class uniqueness in the feature space. The output of CTM is bundled into the feature embeddings of the support and query set, making metric learning in the subsequent feature space more effective. More specifically, CTM consists of a concentrator unit to extract the embeddings within a category for commonalities between the support examples, and a projector unit to consider the output of the concentrator across categories for uniqueness within the support examples 3.2. By looking at the support set as a whole, i.e, considering the global feature distribution in the entire support set, CTM allows for straightforward and easy integration into existing metric-based FSL algorithms. Prior to the introduction of CTM, metric-based few-shot learning approaches had not yet incorporated any of the available contexts by traversing through the classes in the support set for each task.

3.3 Psychometric testing

In this section, we describe how psychometric testing can be applied to alleviate the challenge of exploiting full-depth knowledge from human experts in deep metric learning.

Oftentimes, eliciting and conveying the domain knowledge and constructing a training dataset is a challenging and unscalable task, which is the reason why many of the proposed solutions are formulated as mappings from high-dimensional data points to a fixed set of labels. Consequently, this limits the models from learning the full depth of domain knowledge. This conflicts with the fact that exploiting a rich data structure adds a significantly larger understanding of the finer subtleties of the data, which stems from the relationships between the concepts and aspects in the domain [48]. This has prompted the necessity to elicit domain or human knowledge by

employing psychometric testing in machine learning and specifically in deep metric learning. Yin et al. have become the pioneer in incorporating psychometric testing as a method for efficient knowledge elicitation of data in deep metric learning [48]. The authors developed a hierarchical representation of the data that facilitates more efficient annotation. The proposed method adopts a variation of 2AFC, the three-alternative-force choice (3AFC) test, in which three samples are presented in one test to elicit the annotator's perception regarding the conception similarity. In the 3AFC test variant, the participants are presented with three objects o_1, o_2, o_3, and are forced to choose the object that is most dissimilar. The response captures the perceived relative distance of the examples to develop a distance metric using a deep metric learning method, where the distances reflect the representation of the expert. The extracted embeddings are structured in a hierarchical fashion that allows for assigning properties and annotations to a large number of samples. The results show that the proposed framework has achieved better performance when compared to the baseline, and the proposed 3AFC testing with dual metric learning can extract latent hierarchical representations of data in a scalable way.

Psychometric testing is also applied to collect high-fidelity knowledge from domain experts and map it to semantic hierarchical segmentation [47]. Yin et al. first created super-pixel patches by the simple linear iterative clustering approach, and with the 3AFC psychometric test measured the patches' perceived similarity to each other [47]. Subsequently, a deep metric learning algorithm was proposed to project the perceived relative distances to an embedded space where the patches are structured in a hierarchical fashion, thereby proceeding to image hierarchical segmentation. The proposed framework was empirically shown to effectively capture and represent the high-fidelity semantics in synthetic, aerial and histology images, highlighting the ability of this method to elicit knowledge for image segmentation in a broad range of image analysis applications.

Following the success of applying psychometric testing in deep metric learning, Yin et al. proposed a method to address the problem setting of semantic-based FSL by capturing the inner semantic relationships in the data using interactive psychometric learning [49]. Sharing an analogous motivation to our work, they addressed the problem of having a single class mapping to each image, causing the failures of machine learning models in identifying the proper class assignment when there is no exact matching between the support and query classes. It is important to note that in this study, Yin et al. made the assumption that there is a shared concept hierarchy covering both base and novel classes in FSL [49]. In their proposed framework, self-supervised learning (SSL) was applied for feature learning at the initial stage. This is then followed by the interactive 3AFC psychometric testing procedure in which the similarities of the semantic concepts from the base dataset are captured. The captured semantic similarities are then used to fine-tune the learned features from SSL and map them to a semantic embedding space where they transfer the learned hierarchical knowledge from base classes to novel classes for semantic few-shot prediction.

It is well postulated that while label-based supervision reduces the comprehensive semantic relationships in images data to predefined discrete labels, psychometric testing-based methods can elicit the relative conception of similarities and full-depth knowledge by transmitting the annotations progress to derive an appropriate loss function for a deep metric learning model. One detail that all the aforementioned research share in common is the setup of the psychometric testing procedure. The 3AFC variant is the method of choice as it allows for the construction of the dual-triplet loss function, represented by:

$$\mathcal{L} = \sum_{i=1}^{N} \left[d(x_{p1}^i, x_{p2}^i) - d(x_n^i, x_{p1}^i), +m \right]_+ + \left[d(x_{p1}^i, x_{p2}^i) - d(x_n^i, x_{p2}^i), +m \right]_+,$$

where x_n is the "negative" image indicated by the annotator and x_{p1}, x_{p2} denote the two unpicked "positive" images that have closer concept similarity, captured by the 3AFC test. The loss is constructed such that it pushes images that are perceived as similar by the annotator to be close to each other in the embedded space and enforces a distance margin $m > 0$ between positive pairs and negative pairs.

3.4 Position of this work in literature

The problem setting of aspect-based FSL that we address in this work exhibits resemblance to the semantic-based FSL setting in the work of Yin et el. [49], and that the Category Traversal Module introduced by Li et al. [22] is an effective approach to a similar problem. Our work distinguishes from the preceding works by:

1. The problem setting of semantic-based FSL assumes a shared concept hierarchy covering both base and novel classes. In our work, we postulate that such hierarchical relations may not always exist. Revisiting the examples illustrated in Chapter 1, in Figure 1.2c. a correct assignment could be made by matching the query image with the image of the toy lion, as the hierarchical semantic concept of non-living things can be inferred. This, however, is not the case in Figure 1.2b, as a physical movement - the aspect that matches the query image with the image of the roaring lion, does not have a hierarchical semantic relationship with species; and inferring the concept of non-living things does not result in a sensible matching in this case.
2. In our work, we extend the 3AFC method to a variant of the m-AFC method ($m > 3$) to derive the appropriate loss function in our deep metric learning model. This is elaborated in detail in Chapter 5.3.
3. Most aforementioned studies learn to represent the support set and query examples in one embedding space in which embeddings that are more semantically similar are closer to each other, where the similarity is represented by an appropriate distance metric. In our work, we postulate that the aspect, which is given by the support set, determines the embedding space and hence the distance between two embeddings. For example, in the *color* embedding space, the embedding of a red circle is closer to the embedding of a red square than the embedding of a blue circle, given that the aspect is *color*; whereas in the *shape* embedding space, the embedding of a red circle is closer to that of a blue circle, rather than the red square. Chapter 4 provides a more elaborated explanation of this matter.

Chapter 4

Problem Description

4.1 The role of aspects in perceived relative distances

When presented with three stimuli and tasked to discriminate between the relative difference between any two pairs formed by these three examples, humans or artificial agents oftentimes face the challenge of ambiguity and may not be able to make a congruent decision. That is, the presented images may seem to have the same level of similarity, or there may be more than one aspect based upon which one can decide whether two examples are more similar to each other than the other. To deal with triplets of images which present different aspects that are not directly comparable in a pair-wise fashion, Yin et al. adapted the psychometric testing such that when presented with three samples, the participants are forced to select the most dissimilar one among them [48]. This, however, does not eliminate the ambiguity as any image among the three can be the most dissimilar depending on the aspect in consideration. We illustrate a simplistic example in Figure 4.1, where any two images among the three can be indicated to be more similar depending on the aspect: color, shape or thickness.

If one were to consider the aspect of shape, Figure 4.1b and Figure 4.1c are perceived to be most similar, as they are both squares. However, if one were to judge based on color instead, then Figure 4.1c is "closer" to Figure 4.1a than Figure 4.1c as they both have the color blue. The perceived relative distance also changes if the aspect is thickness. In this case, the pair of Figure 4.1a and Figure 4.1b are most similar. In psychometric testing when presented with these types of questions if no aspect is specified a priori, participants may give different correct answers. It is thus instrumental to incorporate such aspects into the design of metric-based FSL frameworks. Moreover, the illustrated examples are synthetically generated simple shapes, where the aspects are easily identifiable. In real-life datasets where more complicated aspects are present, measuring the similarity between two images often requires performing complex reasoning along

(a) A thick blue circle. (b) A thick red square. (c) A thin blue square.

Figure 4.1: A triplet of three images where any two images can be indicated to be more similar depending on the aspect: color, shape or thickness.

not just different axes such as color, texture, or shape, but also the semantic relationship between the images, prompting an unambiguous, robust way of determining the aspects relevant to the few-shot tasks at hand.

4.2 Aspect-based embedding space

We revisit the notion of an embedding space and discuss the extension to an aspect-based embedding space. In deep learning, embeddings are referred to as dense numerical representations of real-world objects and relationships, represented by a vector. The embeddings represent salient and meaningful features from raw data in a lower dimension that facilitates the task of learning. Embedding extraction refers to mapping the raw, high-dimensional data into an inner product space. In this vector space, the semantic similarity between categories can be quantified. Data points that describe similar semantic concepts can be represented by similar feature vectors. Two feature vectors are said to be similar whenever they are close w.r.t. the metric induced by the inner product. Within this embedding space, similarities can be directly computed using an appropriate metric distance such as the Euclidean distance or the cosine metric. As previously discussed in Section 4.1, the perceived similarity changes depending on the aspect in consideration. Hence, one embedding space may not be able to accommodate or reflect accurately the semantic similarity between pairs of images. This prompts us to the notion of aspect-based embedding space. We postulate that once the aspect is determined, the input data can be embedded in such a way that the most relevant, meaningful features w.r.t the aspect are captured. The embedding of the same data point is thus different per aspect, and specific to the aspect at hand. We refer to the embedding space in which different data points are mapped specifically based on a particular aspect as the aspect-based embedding space. Within an aspect-based embedding space, data points that elicit similarities w.r.t the aspect can be represented by similar feature vectors where similarity can be calculated using a metric distance. For instance, in the *shape*-based embedding space, the distance between the embedding of the blue square and the embedding of a red square is smaller than that of the blue circle, whereas in the *color*-based embedding space, the distance between the embedding of the blue square would be closer to the blue circle, rather than the red square (Figure 4.2). This suggests that the design of a solution method for this problem formulation must allow for the model to learn an embedding function given the determined aspect.

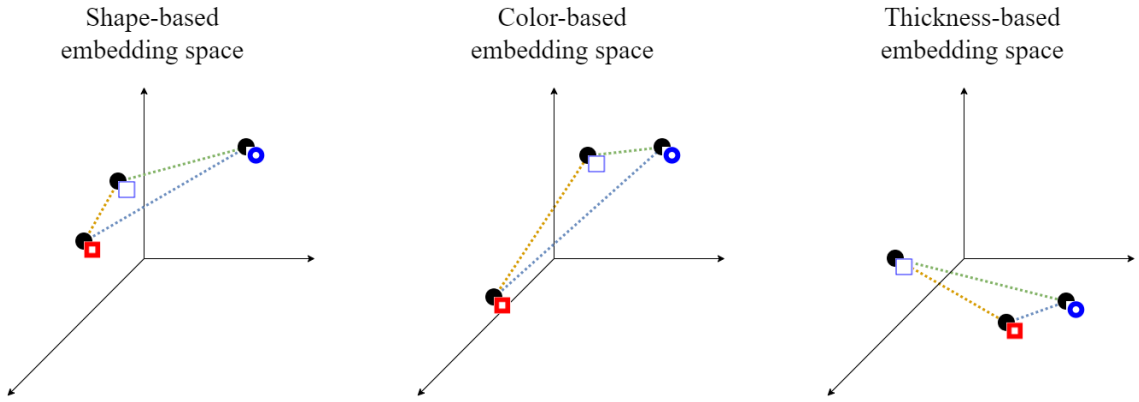


Figure 4.2: Aspect-based embedding space. The relative distances between the embeddings are different per aspect. In the *shape*-based embedding space, the distance between the embedding of the blue square and the embedding of a red square is smaller than that of the blue circle, whereas, in the *color*-based embedding space, the distance between the embedding of the blue square would be closer to the blue circle, rather than the red square. In the *thickness*-based embedding space, the embedding of the blue circle is closer to the embedding of the red square as they both have the same thickness.

4.3 Scope of the problem

This work addresses the aspect-based few-shot learning problem where the novel challenge is to determine the relevant aspects given in the support set and learn the embedding space where the features that are most salient w.r.t this aspect can be extracted. In order to solve this challenge, the following two aspects are of importance to this thesis project:

- This project solely focuses on the domain of image recognition, where there are different factors based upon which comparison for similarities can be made such as shapes, color, texture or semantic relationships.
- The aspects given in the support set are learned and incorporated in a deep metric learning model where a psychometric testing procedure is applied to design the loss function.

4.4 Problem formulation

Let us consider the situation in which given a base dataset with classes $\mathbb{C}_{\text{base}}$ and a novel dataset containing class $\mathbb{C}_{\text{novel}}$, both containing an ample amount of labelled images. The classes in the novel dataset are not made known or seen in the base dataset, i.e., $\mathbb{C}_{\text{base}} \cap \mathbb{C}_{\text{novel}} = \emptyset$. The premise of FSL is to leverage the limited data observed in $\mathbb{C}_{\text{base}}$ to learn a good classification model for the unseen classes $\mathbb{C}_{\text{novel}}$. In the specific setting of a few-shot classification task, given a support set of N distinct, previously unseen classes, with K examples each and a query, the goal is to find a correct matching for the query agnostic of fixed labels. We now introduce the notion of an aspect in few-shot learning. Given a support set $\mathcal{S} = \{(x_i, y_i)\}_i^N$, the common features are all constant factors and values that are shared by all support set images $x_0, \ldots, x_N$, whereas what varies among these images depict the aspects.

Let f_θ denote an embedding function that is used to extract the embeddings of the images $\mathbf{x}$, represented by $f_\theta(\mathbf{x})$, which typically has a CNN module as its backbone. The objective of the aspect-based few-shot classification task is to learn a model f'_ϕ parameterized by ϕ, capable of exploiting the few training examples provided in each support set $\mathcal{S}$ to determine the relevant aspect correctly. That is, given the set of embeddings $\{f_\theta(x_0), \ldots, f_\theta(x_N)\}$ for $x_0, \ldots, x_N \in \mathcal{S}$, we need to learn an embedding function that extracts the aspect determined by this set, denoted $\mathcal{I}$, and use this encoding to compute the aspect-based embeddings for the images in the query set and the support set in a metric space in which the distance can be calculated. The aspect-based embeddings, also regarded as *aspect-based* representations of the support set and the query are $\mathcal{I}(f_\theta(\mathcal{S}))$ and $\mathcal{I}(f_\theta(\mathcal{Q}))$, respectively.

As the objective is to address this problem of label-based supervision, we cannot specify a few-shot task using the label information in the typical few-shot learning setting, i.e., sampling multiple images with the same labels to create a class. For this reason, we sample N images without specifying their class labels from $\mathbb{C}_{\text{novel}}$ but instead based on pre-defined aspects to create the support set and sample one image as a query to form a N-way 1-shot aspect-based few-shot learning task.

Chapter 5

Solution Approach

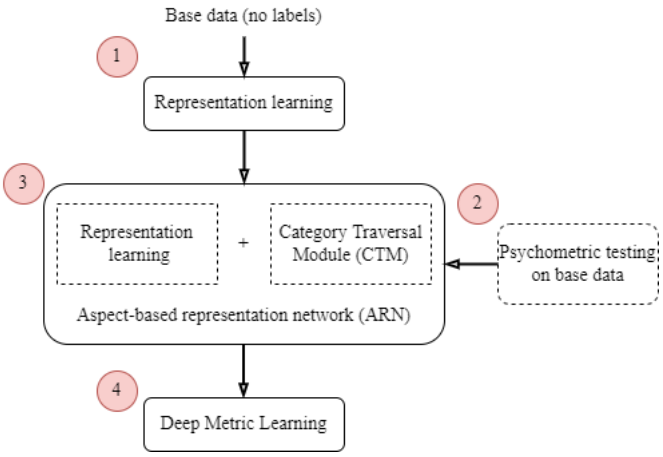


Figure 5.1: Overview of the framework of aspect-based FSL.

This chapter details the proposed framework of aspect-based FSL. This framework consists of three main components. The first component is representation learning, where the features are extracted from the raw input data by a neural network model (i.e., feature extractor). The second component is a psychometric testing procedure that relies on discriminative testing to capture the transferable aspect-based conception relationships, which allows us to tackle the limitation caused by label-based supervision. Using the captured responses, the perceived similarities are then used to fine-tune the learned features using the Category Traversal Module cf. [22] in an aspect-based representation network. The premise behind the aspect-based representation network is to learn the aspect and embed the data into the aspect-based embedding space that captures the relationship between data instances when taking such an aspect into consideration. To achieve this, a deep metric learning model is trained using a triplet loss function variant. During inference, the aspect-based representation network allows one to retrieve for each query the most semantically similar image in the support set by the Euclidean metric, even when the target image and the query image are not from the same class. The proposed framework is visualized in Figure 5.1.

5.1 Data generation

5.1.1 Data description

As a basis for the experiments conducted in Chapter 6, we synthetically generate a two-dimensional geometric shape dataset out of six simple shapes, where each shape is a 224×224 RGB image with different attributes. This specific resolution requires minimum space while the geometric shapes can still be generated with different patterns. The base dataset consists of $6^5 = 7776$ images, where each image contains one of the following single-holed polygon shapes: triangles, rectangles, pentagons, hexagons, circles or rhombus. For each image, the following attributes are also described: the filling color, the thickness, the deformation (stretch) and the patterns inside each shape. The motivation behind this choice of data is two-fold: (1) The dataset is clean, i.e., strictly consists of drawn geometrical shapes, which rules out any inconsistencies that would influence the benchmark of any machine learning model; and (2) the specified attributes allow for straightforward and unambiguous comparisons between images, this, in turn, enables us to create sets of images where the aspect can be predefined and thus allows for controlled benchmark experiments.

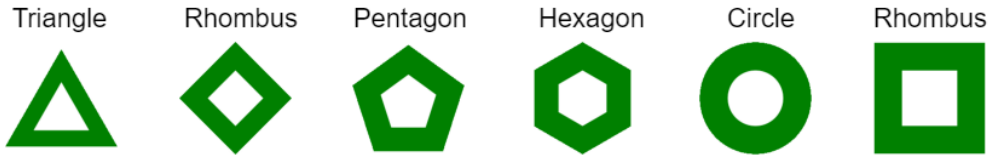
5.1.2 Design setup

Each of the aforementioned attributes is considered a parameter of the geometric shape generator. The generation program is developed using the Python programming language and the functionalities from the Matplotlib library to draw such geometric shapes [17]. The generation program receives as input the range for each parameter. Each image is generated with the following attributes:

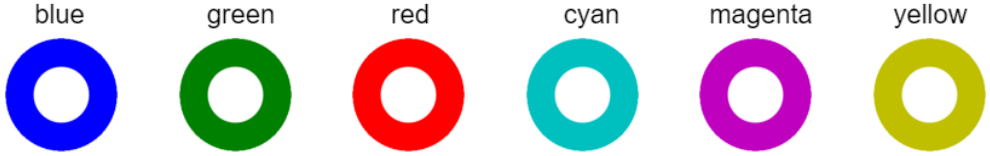
- Shape: triangles, rectangles, pentagons, hexagons, circles or rhombus (Figure 5.2a).
- Color: We select six out of the eight base colors provided by Matplotlib, namely blue, green, red, cyan, magenta, or yellow (Figure 5.2b).
- Stretch (level of deformation): indicated by a value in the range of $[0.5, 1.0, 1.5, 2.0, 2.5, 3.0]$. The value of 1 defines an undeformed shape, the higher the value the more compressed the shape is (deformed inwards) and the lower the value the more stretched the shape is (deformed outwards) (Figure 5.2c).
- Thickness: indicated by a value in the range of $[0.4, 0.5, 0.6, 0.7, 0.8, 0.9]$; the higher the value the thinner the shape is (Figure 5.2d).
- Pattern: crossed diagonal pattern, small circles, stars, no pattern, dots or horizontal stripes (Figure 5.2e).

Note that the base set contains each combination of the aforementioned attributes. One is not limited to these parameter ranges, however, we settle on these ranges as the correspondingly generated images are adequately easy for humans to distinguish (i.e., it is difficult to visually recognize the difference between a thickness of 0.4 and a thickness of 0.42) while still allowing for a sufficient number of shapes per attribute. Figure 5.2 demonstrates an example of a generated shape with each parameter.

Besides the images, the generation program also stores the attribute information in the form of a dictionary (a key-value store). This allows for the query of an image or a set of images with one or multiple attributes, without having to rely on sets of labels. For example, querying a set of images of the shape circle with the color green, stretch of 1 and no pattern results in the images illustrated in Figure 5.2d.



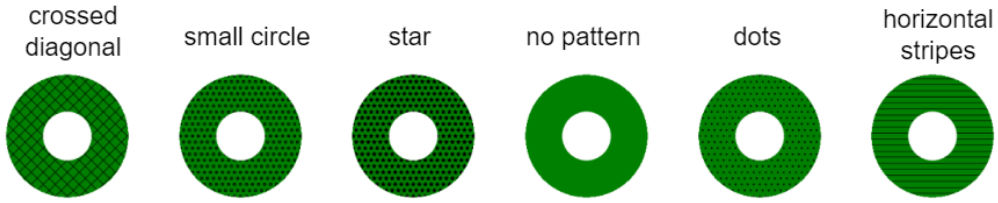
(a) All shapes generated in the geometric shape dataset with the color green, stretch of 1.0, thickness of 0.5 and no pattern.



(b) All colors generated in the geometric shape dataset with the circle shape, stretch of 1.0, thickness of 0.5 and no pattern.

(c) All stretch levels generated in the geometric shape dataset with the circle shape, color green, thickness of 0.5 and no pattern.

(d) All thickness levels generated in the geometric shape dataset with the circle shape, color green, stretch of 1.0 and no pattern.



(e) All patterns generated in the geometric shape dataset with the circle shape, color green, thickness of 0.5 and no pattern. From left to right: crossed diagonal pattern, small circles, stars, no pattern, dots and horizontal stripes.

Figure 5.2: Examples of generated geometric shapes with the pre-defined ranges of parameters

5.1.3 Generation of triplets

Having generated the geometric shapes based on different attributes along with the attribute information, we can generate meaningful image triplets that allow for comparison based on a specific aspect. A triplet is considered meaningful if there exist three aspects where per aspect there is a pair of images that is more similar. Figure 4.1 demonstrates a triplet consisting of a thick blue circle, a thick red square and a thin blue square. The three aspects that one can consider are color, shape and thickness. Note that when generating the geometric shapes, each attribute can be used as an aspect. Thus, in this dataset, the following aspects can be defined: shape, color, stretch, thickness and pattern. Such aspects are easily recognizable by humans and hence unambiguity is ensured.

Aspect \ Image	x_0	x_1	x_2
Color	Blue	Red	Blue
Shape	Circle	Square	Square
Thickness	Thick	Thick	Thin

Table 5.1: A look-up table for generating a triplet of images based on three different aspects.

The principle behind creating meaningful triplets is described in Table 5.1. First, three different aspects are selected, for example, one may select color, shape and thickness. For each image, an attribute value is assigned such that per aspect two of the three images share the same attribute value; and that no pair has the same value for more than one aspect. Each column in Table 5.1 describes the generated image in terms of these aspects. In this example, the image pair (x_0, x_1) has the same thickness, the image pair (x_1, x_2) has the same shape whereas the image pair (x_0, x_2) has the same color. The design of the look-up table allows one to extend it to be able to accommodate more aspects and more images, which in turn increases the complexity.

Having established the look-up table, the triplet generator is created as follows. First, an image x_0 is selected uniformly at random from the pool of unchosen images (without replacement). Given the three predefined aspects, an image x_1 that differs from image x_0 in two aspects while sharing all other aspects, including the remaining aspects that are not considered for the task at hand, with image x_0 is chosen. Having drawn the first two images, the image x_3 is constructed such that for the two aspects where x_0 and x_1 differ, x_2 shares one aspect with x_0 and the other aspect with x_1; and for the aspect where x_0 and x_1 are the same, x_2 is assigned a different value in the parameter range. The drawn images x_0, x_1 and x_2 form a triplet.

5.1.4 Generation of aspects

Recall the notion of an aspect of a support set, which is the varying factors among the images in the support set. In a wider scope, this notion also applies to any set of images. In a set of images, multiple aspects may exist and may be perceived differently. The geometric shape dataset consists of images in which the aspects can be easily identified as one can explicitly control what differs among the images. For example, Figure 5.3 depicts a set of four images in which color is the only factor that varies among the set. It is thus said that color is the aspect given this particular set of geometric shapes.

Figure 5.3: A set of geometric shape images in which color is the only varying factor. Color is thus deemed to be the aspect determined by this particular set.

Given a meaningful triplet cf. Section 5.1.3, when making a pairwise comparison, one could provide auxiliary instances to clarify the relevant aspect. For example, to make it explicit that the relevant aspect when comparing two images is shape, one could provide an auxiliary set of images in which shape is the only factor that varies. Hence, to provide a context for each aspect, we extend Table 5.1 to include a criterion to generate an auxiliary set of images that adequately describes the relevant aspect. This results in Table 5.2. The criterion describes the condition that all images in this auxiliary set must satisfy, i.e., all auxiliary images must have the same value for every attribute while having a unique value for the attribute which is deemed the current aspect.

The triplet with auxiliary image sets that describe the aspects according to Table 5.2 is illustrated

Aspect \ Image	x_0	x_1	x_2	Criterion for generating the auxiliary set
Color	Blue	Red	Blue	Unique color, same value for all other attributes
Shape	Circle	Square	Square	Unique shape, same value for all other attributes
Thickness	Thick	Thick	Thin	Unique thickness, same value for all other attributes

Table 5.2: The extended look-up table for generating a triplet of images based on three different aspects with a criterion for the generation of an auxiliary set of images that describes the aspect explicitly.

in Figure 5.4.

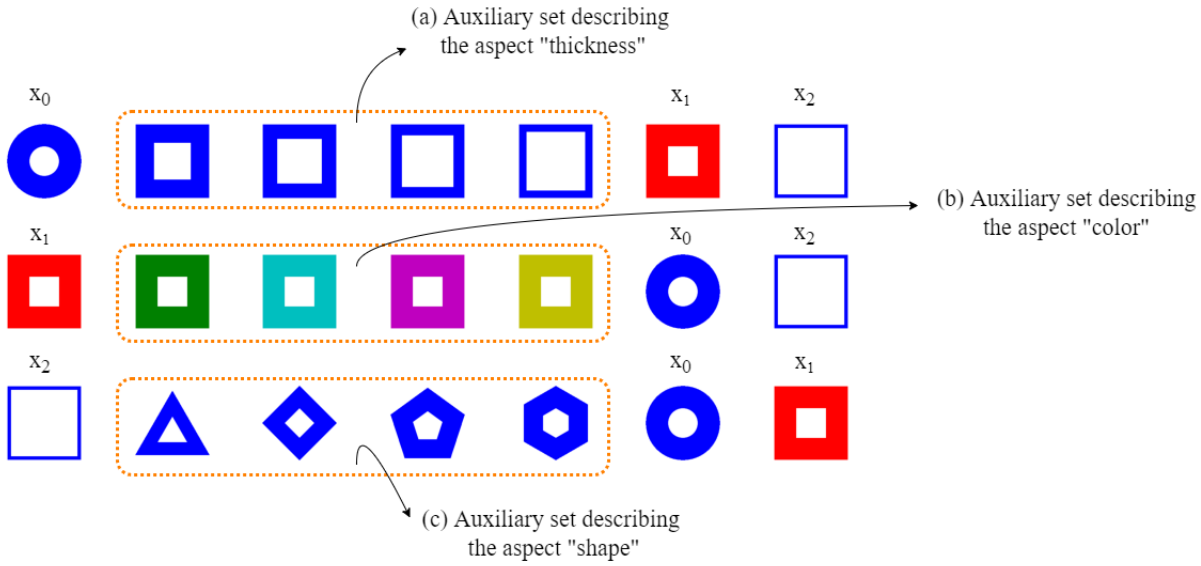


Figure 5.4: The triplet with auxiliary image sets that describe the aspects according to Table 5.2. Each row consists of the triplet (x_0, x_1, x_2) and the set of images (highlighed in orange) that describes the corresponding aspect. The images in this set are unique in thickness and identical in all other attributes. Thus, for the first row, this auxiliary set of images describes the aspect of "thickness". Similarly, the highlighted set in the second row and the third row describes the aspect of "color" and "shape", respectively.

5.2 Representation learning

Representation learning refers to the practice of training machine learning algorithms to learn useful representations, i.e., those that are interpretable, have latent features, or can be used for transfer learning. Deep neural networks are typically considered representation learning models that can encode meaningful information which is projected into a smaller subspace. Evidence strongly suggests that CNNs are one of the most effective techniques to learn visual representations [2]. Two popular CNN architectures for the task of representation learning, or feature extraction are ResNet by He et al. [15] and VGG by Simonyan and Zisserman [37].

VGG: The VGG network is constructed with increasing depths and very small convolutional filters, that is, VGG's convolutional layers leverage a minimal receptive field, i.e., 3×3, the smallest possible size that still allows for capturing the up-down and left-right patterns [37]. Moreover, there are also 1×1 convolution filters acting as a linear transformation of the input, followed by a ReLU unit. The convolution stride is fixed at 1 pixel to keep the spatial resolution preserved after convolution. VGG has three fully connected layers, where the first two have 4096 channels each, and the last one has 1000 channels.

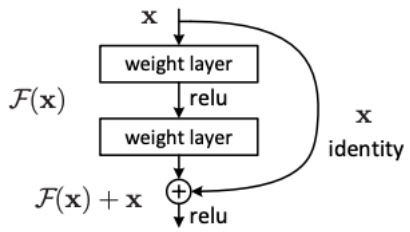


Figure 5.5: The architecture of a residual building block [15].

ResNet: The ResNet architecture is a deep residual learning model to learn the image features from the base dataset [15]. The ResNet architecture is composed of multiple residual blocks forming the input layers, and their operating principle is concerned with optimizing a residual function. More specifically, instead of hoping each few stacked layers in a deep CNN directly fit a desired underlying mapping, He et al. explicitly let these layers fit a residual mapping. Denote by $H(x)$ the underlying mapping and with $F(x) := H(x) - x$ the residual mapping. The original mapping is recast into $F(x) + x$, and can be realized by feed-forward neural networks with the so-called "shortcut connections", as shown in Figure 5.5. Shortcut connections refer to those connections that skip one or more layers. In the architecture of a ResNet model, the shortcut connections simply perform the identity mapping, and their outputs are added to the outputs of the stacked layers. It is notable that identity shortcut connections inflict neither extra parameters nor computational complexity. Different variants of a ResNet architecture exist, i.e., they share the same concept but consist of a different number of neural network layers.

These special architectures allow for performance gains from increasing layer depth. The rationale behind the choice of these network architectures for representation learning is that the representation depth is instrumental for high accuracy in classification whereas both of these architectures are extremely robust in accommodating much deeper architecture while achieving breakthrough performance. It has also been shown that these models generalise well to a wide range of tasks and datasets, which level with or even outperform more complex recognition pipelines built around shallow image representations.

The output of the representation learning module is the embeddings that represent the most meaningful features from the raw data in a lower dimension. These embeddings, however, do not accommodate the different aspects present in the data and thus are to be fine-tuned to facilitate the task of aspect-based learning. This prompts the second component in the proposed framework, which is the psychometric testing procedure.

5.3 Psychometric testing

As we are parting from label-based supervision, no label information is factored into the construction of the solution approach. We adopt a psychometric testing procedure that is based on discriminative testing. The psychometric test collects information about the human perceived relative differences between different stimuli, for example, a few support set images and a query image. During this test, the target labels are not presented to the human participants, which circumvents the need for a mapping from the presented images to a predefined set of concepts. By implementing the psychometric test, we can extract knowledge from the data including the variable aspects and the rich semantics by a simple ternary decision. This is fundamental as there is no necessity for using a fixed set of labels while the aspects and the semantic relationships built on the aspects are well-captured. The perceived differences between the objects can then be expressed as distances in a deep metric learning model.

Existing literature (as presented in Chapter 3) has shown that the common denominator between

all deep metric learning approaches that employ psychometric testing is the choice of the 3AFC variant. We have identified the perplexing situation when any choice would be considered correct as there exists an aspect that justifies the perceived similarity. Thus, in our proposed framework, we extend the 3AFC method to a variant of the $m-$AFC method. In this variant, the participants are presented with m objects that consist of:

- An anchor image x_0,
- two images x_1 and x_2, and
- k auxiliary images that are unique in only one attribute and identical in all other attributes.

Here, $m = k + 3$ where $k > 3$. The psychometric testing procedure entails a series of questions. For each question, we select an anchor image and the participants are tasked to discriminate between the distance given by the anchor and the $m - 1$ options. The distance is with respect to a particular attribute in the image such as the size of the object, the category of the objects or the value of the object in terms of shape, color, thickness, stretch or pattern. The answers are then captured by marking the image which is chosen as closer to the anchor (denoted positive) and the others as farther than from the anchor (indicated as negative). Figure 5.6 provides an example of our proposed m-AFC variant. Note that since we are more interested in the relationships among the triplet (x_0, x_1, x_2) and less on the k auxiliary images, the occurrence of the same triplet is repeated three times, but each time is accompanied by a different set of k auxiliary images. For each occurrence, the supposed answer for which anchor-option pair is the closest would be different. In Figure 5.6, in the first test, the thin blue circle is closest to the anchor (thin red square) but this is not necessarily the case in the second test when the thin blue circle becomes the anchor and the thin red square also appears as one of the options. This is not intended to introduce confusion and ambiguity, on the contrary, this is aimed to enforce the learning of the relevant aspect that dictates the differences in similarities. In addition, the setting of this psychometric test is conforming with the generation of triplets together with auxiliary sets of images that emphasize on the relevant aspects as detailed in Section 5.1.

The justification behind the design of the psychometric testing procedure in our proposed framework is that it allows the participants to view the set of options as a whole and define the relevant aspect based on which comparisons are most sensible. On that account, the psychometric testing procedure enables the elicitation of the perceived aspect that is relevant for each question and subsequently the perceptions of concept similarities are captured.

In this variant of the m-AFC test, the choice of m directly depends on the choice of k as $m = k+3$. The intended purpose of including k auxiliary images in each test is to make the aspect relevant to the task at hand clear. Thus, k must not be too small as it will not adequately present the aspect, nor too large as it will increase the complexity of the test. For this reason, we settle on the choice of $k = 4$, making this procedure a 7-AFC test.

Using the captured responses, the perceived similarities dictated by the aspects are then used to fine-tune the learned features learned using the Category Traversal Module in an aspect-based representation network, which is entailed in the following sections.

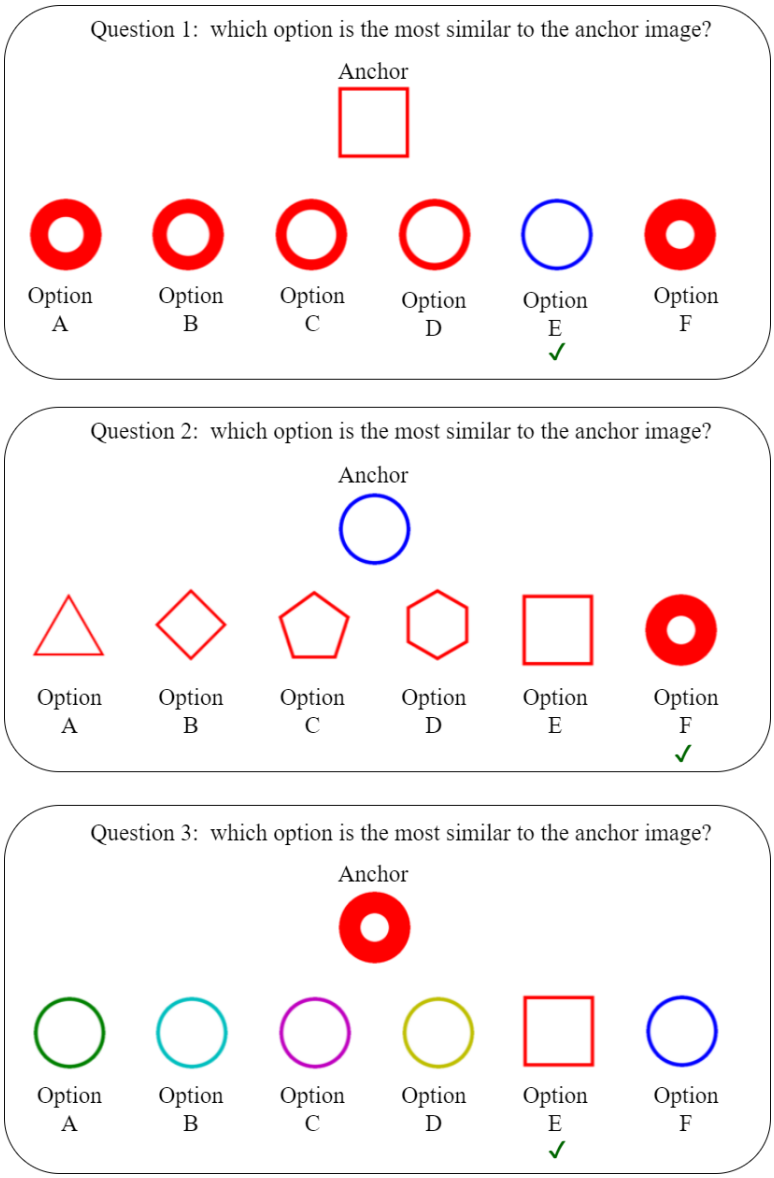


Figure 5.6: Examples of a m-AFC psychometric testing setting. In the first test, the participants are more likely to choose the thin blue circle (option E) as most similar to the anchor since they have an identical thickness. In the second test, the participants tend to choose the thick red circle (option F) as they are most similar in shape. In the third test, the thin red square is most likely to be selected as it has the same color as the anchor while the others do not. Note that the occurrence of the thin red square, the thin blue circle and the thick red circle all is repeated three times.

5.4 Aspect-based representation network

Given the ability to consider the entire support set as a whole to determine the aspect relevant to each task and its robustness, the CTM has become a prominent candidate structure for facilitating deep metric learning in high-dimensional spaces. The aspect-based representation network constitutes the CTM on top of the learned representations. The obtained representations are then fine-tuned based on the perceived similarities obtained from the psychometric tests which are mapped to embedding distances. The basic premise of the CTM component is as follows. The support set features $f_\theta(\mathcal{S})$ of shape (NK, m_1, d_1, d_1), i.e., the learned representations, obtained

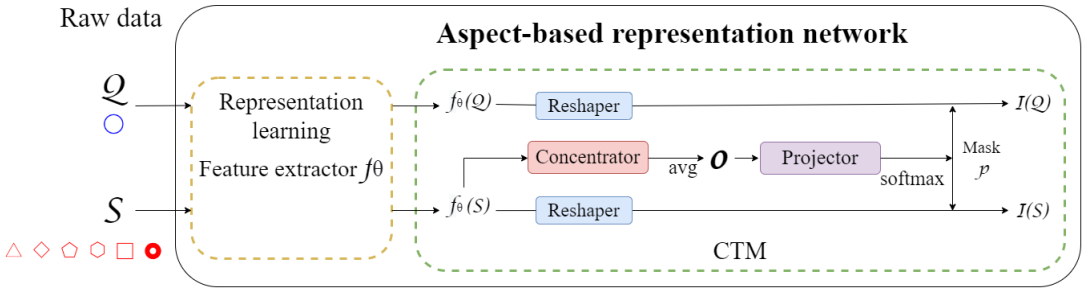


Figure 5.7: Overview of the aspect-based representation network.

from the feature extractor f_θ in the first stage serve as input to the CTM, where m_1, d_1 denote the number of channels and the dimension of the raw input, respectively. The concentrator and the projector, which learn the commonalities and uniqueness between the support classes, are used to create a mask p. This mask is then applied to both the support and query, producing improved embeddings with the aspect relevant to the current task (Figure 5.7). The essential constituents of this design are the aforementioned components: the concentrator and the projector, both implemented as convolutional layers.

5.4.1 Concentrator

The concentrator is used to find common features that are shared by all instances for one class in the support set:

$$f_\theta(\mathcal{S}) : (NK, m_1, d_1, d_1) \xrightarrow{\text{Concentrator}} \mathbf{o} : (N, m_2, d_2, d_2), \tag{5.1}$$

where m_2 and d_2 denote the number of channels of the output and the spatial size, respectively. The purpose is to remove the difference among instances and extract the commonality among instances within one category, which is achieved by appropriate downsampling from (m_1, d_1) to (m_2, d_2). Note that the input to the concentrator is first fed to a CNN module to perform dimension reduction, and then samples within each class are averaged to have the final output $\mathbf{o}$.

5.4.2 Projector

Subsequently, the projector filters irrelevant features and keeps the ones that are most discriminative for the task at hand by looking at the concentrator features from all support classes simultaneously:

$$\hat{\mathbf{o}} : (1, Nm_2, d_2, d_2) \xrightarrow{\text{Projector}} \mathbf{o} : (1, m_3, d_3, d_3), \tag{5.2}$$

where $\hat{\mathbf{o}}$ is a reshaped version of $\mathbf{o}$ and similarly, m_3 and d_3 denote the number of channels of the output of the projector and the spatial size, respectively. The objective of traversing across all support classes is achieved by concatenating the class prototypes in the first dimension (N) to the channel dimension (m_2), applying a CNN module to the concatenated features to produce a map of size $(1, m_3, d_3, d_3)$, and finally applying a softmax over the channel dimension m_3, which is applied separately for each of the $d_3 \times d_3$ spatial dimensions to produce a mask p. This is used to mask the relevant feature dimensions for the task in the query and the support set and thus the obtained embeddings only contain the most discriminative features that are specific to the task at hand.

5.4.3 Reshaper

Once the mask p is obtained from the projector, in order for p to influence the feature embeddings f_θ, the shape between these components in the network needs to be matched. This is done through

a reshaper network, which is implemented with one CNN layer. The reshaper is applied separately to each of the NK samples as follows:

$$f_\theta(\cdot) \xrightarrow{\text{Reshaper}} \mathbf{r}(\cdot)(NK, m_3, d_3, d_3). \tag{5.3}$$

There is an underlying aspect based on which the embeddings of the query and the support set are learned. With the CTM, the resulting embeddings only contain the features that are most discriminative to this aspect. Such embeddings are referred to as *aspect-based* representations, which are projected onto an aspect-based embedding space where their distances can be calculated with an appropriate distance metric.

5.5 Deep Metric Learning

5.5.1 Loss functions

The application of the 3AFC method in psychometric testing has been shown to be a robust and proper fit for the implementation of the triplet network, together with the triplet loss. The triplet network f takes an anchor x_a, a positive sample x_p (which is of the same class as the anchor) and a negative sample x_n (from a different class than the anchor) as inputs. The objective is to learn embeddings such that the anchor is enforced to be closer to the positive sample than it is to the negative sample by some margin value $m > 0$:

$$||f(x_a) - f(x_p)||_2^2 < ||f(x_a) - f(x_n)||_2^2.$$

Schroff et al. [34] defined the triplet loss as follows.

$$\mathcal{L}^i_{\text{triplet}}(x_a, x_p, x_n) = \max(0, m + ||f(x_a) - f(x_p)||_2^2 - ||f(x_a) - f(x_n)||_2^2). \tag{5.4}$$

In this work, we propose a loss function that follows similar principles as the triplet loss, but is more centric to the setting of aspect-based FSL. The rationale behind this is two-fold: (1) to align with our proposed solution method that employs the m-AFC variant of the psychometric testing procedure, which is an extension of the 3AFC variant, and (2) the objective is to learn the aspect dictated by the support set, and given this aspect, the support set and the query are properly mapped to an aspect-based embedding space, where data points that are perceived to be more similar based on the aspect are enforced to be closer.

We deviate from the usual definition of an anchor, a positive and a negative sample used in triplet networks. Here, we consider the positive sample the one that is perceived to be the closest to the anchor, which is elicited from psychometric testing and the rest to be negative samples. To avoid confusion, hereinafter, when the term "triplet" is solely mentioned, it refers to a set of anchor, positive and negative samples as conventionally used in literature. On the other hand, when we address the set of three images (x_0, x_1, x_2) that have the different conceptual similarities dictated by the aspects, we refer to it as "the triplet (x_0, x_1, x_2)". Conform Section 5.3, given a triplet (x_0, x_1, x_2), there are three tasks that involve (x_0, x_1, x_2) and a different aspect is present in each task. In these occurrences, each image x_i, $i = 1, 2, 3$ takes turn to be the anchor and the positive and negative samples are determined based on the aspect. For example, x_0 is the chosen anchor in the first psychometric test where the first aspect is present and x_1 is perceived to be the closest to x_0. Then, x_1 is considered the positive sample, whereas x_2 and the auxiliary images are the negative samples. The same principle applied to the second test and the third test, where x_1 and x_2 are the anchors, respectively. This exposition motivates the formalization of a 3-component loss, where the loss is calculated separately per aspect and the total loss is the sum of the three individual losses.

The aspect-based representations obtained from the CTM are the embeddings projected onto the aspect-based embedding space. In this aspect-based embedding space, one can either concentrate

on the relative distances between the triplet at hand, or one can take into account all the relative distances between the anchor and the auxiliary samples that form the embedding space. This allows us to formally derive two candidates for the loss function. Per aspect, the individual loss is defined as follows.

Aspect-based tuplet loss

In this formulation, the notion of a tuplet generalizes the triplet to explore multiple negative examples. The tuplet constitutes the triplet (x_0, x_1, x_2) at hand and a set of k auxiliary images. The tuplet per aspect is thus defined as:

$$t = (x_a, x_p, x_{n_1}, \ldots, x_{n_{k+1}}),$$

where k is the size of the auxiliary set of images. Note that the relationship between the tuplet and triplet can be viewed as: each tuplet $(x_a, x_p, x_{n_1}, \ldots, x_{n_{k+1}})$ contains $k+1$ triplets that share the same positive pair (x_a, x_p). That is:

$$(x_a, x_p, x_{n_j}), \forall j = 1, \ldots, k+1.$$

On that account, the tuplet constraint can be defined as follows. For all $j = 1, \ldots, k+1$,

$$||f(x_a) - f(x_p)||_2^2 < ||f(x_a) - f(x_{n_j})||_2^2.$$

Subsequently, similar to [39], the tuplet loss per aspect i can be defined as:

$$\mathcal{L}^i_{\text{tuplet}}(t) = \log\left(1 + \sum_{j=1}^{k+1} e^{d(x_a, x_p) - d(x_a, x_{n_j})}\right),$$

where $d(\cdot, \cdot)$ is the Euclidean distance, i.e., $d(x_1, x_2) = ||f(x_1) - f(x_2)||_2^2$. In this case, the tuplet loss pulls the anchor and the positive sample together while pushing $k+1$ negative samples all at once. This means that with the triplet (x_0, x_1, x_2), if x_0 becomes the anchor and x_2 is the positive sample, then the tuplet loss pushes not only x_1 but also the k auxiliary images away (Figure 5.8).

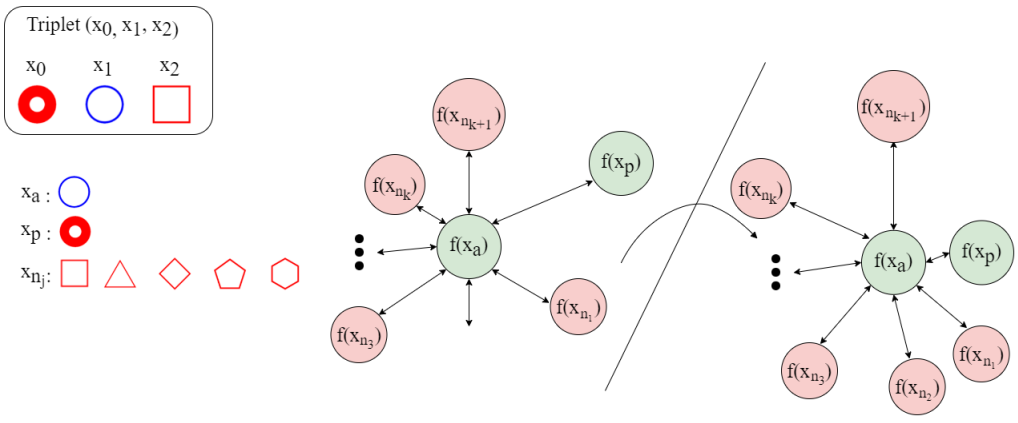


Figure 5.8: Demonstration of the tuplet $t = (x_a, x_p, x_{n_1}, \ldots, x_{n_{k+1}})$. The tuplet loss pulls the anchor and the positive sample together while pushing $k+1$ negative samples all at once.

Aspect-based triplet margin loss

An alternative choice for the loss function design is to focus solely on the relative distances between the triplet (x_0, x_1, x_2) and ignore the auxiliary set. In this situation, we revert to the simple triplet loss function. Thus, for each aspect i, the triplet loss function is as shown in Eq. (5.4).

Total aspect loss

The loss is calculated individually per aspect, as the pair (anchor, positive) is different for each aspect. Subsequently, the total aspect loss is the sum of the losses for every aspect:

$$\mathcal{L}_{\text{total}} = \mathcal{L}^{\text{aspect}_1} + \mathcal{L}^{\text{aspect}_2} + \mathcal{L}^{\text{aspect}_3},$$

where all loss components $\mathcal{L}^{\text{aspect}_i}$ for $i = 1, 2, 3$ are calculated using the same type of loss function, either $\mathcal{L}_{\text{tuplet}}$ or $\mathcal{L}_{\text{triplet}}$.

5.5.2 Deep metric network

The three-component loss emerges from the design of a triplet (x_0, x_1, x_2) where three aspects are present, which are described by three different sets of auxiliary images. As the goal is to learn different representations of the query and the support set given various aspects, the input data into the network needs to be structured in a combined manner that allows for the three-component loss to be calculated.

For training, each mini-batch contains three tasks $\mathcal{T}_i$ with a query $\mathcal{Q}_i$ and support set $\mathcal{S}_i$ for $i = 1, 2, 3$ sampled from the base dataset. In each task, an aspect $\mathcal{A}_i$ is presented and is described by the set of auxiliary images $X_i = \{x_{n_1}, \ldots, x_{n_k}\}$ of size k. The data is sampled such that in these three tasks, a triplet (x_0, x_1, x_2) occurs three times.

- Task $\mathcal{T}_1$: $x_0 \in \mathcal{Q}_1$ whereas x_1, x_2 and the auxiliary set $X_1 \in \mathcal{S}_1$.
- Task $\mathcal{T}_2$: $x_1 \in \mathcal{Q}_2$ whereas x_0, x_2 and the auxiliary set $X_2 \in \mathcal{S}_2$.
- Task $\mathcal{T}_3$: $x_2 \in \mathcal{Q}_3$ whereas x_0, x_1 and the auxiliary set $X_3 \in \mathcal{S}_3$.

For each task $\mathcal{T}_i$, the component loss $\mathcal{L}^{\text{aspect}_i}$ is calculated and subsequently the total loss. The proposed loss functions are minimized using the Adam optimizer [19], which is the superior optimizer due to its robustness and performance improvement over other optimizers [33]. A small learning rate results in significantly slow convergence, however, a large learning rate can hinder convergence and lead to the loss function fluctuating around the minimum or even diverging. Thus, choosing the learning rate is a delicate task. The strategy for selecting the learning rate is by observing the loss curve; for instance, if the loss decrease is too slow, then we may increase the learning rate; and if the loss decreases too fast or the loss fluctuates a lot or even diverges, we may decrease the learning rate.

After fine-tuning the aspect-based representation network with m-AFC tests from the base dataset, we can extract the aspect-based representation of the query and the support sets sampled from the novel dataset. As such, for a query image in a task, we compute its normalized Euclidean distance to each support image and find the closest one, which is the predicted most semantically similar image to the query when considering the aspect defined by the support set.

Chapter 6

Experiment Setup

This chapter introduces the experiment setup. In order to assess whether the proposed method can fulfil the formulated objective, detailed experiments need to be drafted. Before this can happen, suitable data needs to be in place in order to reflect the problem setting's characteristics. Lastly, this chapter provides performance metrics that are used to evaluate the proposed methods.

6.1 Dataset

6.1.1 2D synthetic geometric shape data

In Chapter 5.1, we presented a comprehensive description of the 2D synthetic geometric shape data from the design setup to the implementation of the generation program. The base dataset contains 7776 unique shape images with different attributes such as shape, color, stretch, thickness and pattern. The way the triplets and aspects are generated allows us to build the dataset with the structure required for both the psychometric testing procedure and for training the aspect-based representation network, as specified in Chapter 5. One episode contains a triplet (x_0, x_1, x_2) and three sets X_i of size $|X_i| = 4$ that describes three aspects, structured into three tasks, each with a query set $\mathcal{Q}_i$ and a support set $\mathcal{S}_i$.

We consider 5 attributes and each attribute of the generated shapes can be chosen as an aspect, thus resulting in 5 unique aspects. Each episode requires 3 aspects and hence there are $\binom{5}{3} = 10$ possible combinations of aspects based on which the tasks can be generated. Table 6.1 lists all combinations of aspects that define each training episode.

	Aspect combination
1	(color, thickness, shape)
2	(color, stretch, shape)
3	(stretch, shape, thickness
4	(color, stretch, thickness)
5	(color, pattern, shape)
6	(color, pattern, thickness)
7	(color, pattern, stretch)
8	(pattern, shape, thickness)
9	(pattern, shape, stretch)
10	(pattern, stretch, scale)

Table 6.1: List of all possible combinations of aspects.

As the same aspect can appear in multiple aspect combinations, it is important to ensure the number of episodes per aspect combination is balanced and to avoid bias towards any aspect. In addition, the episodes must be sampled such that the dataset size must be adequately large to obtain a reasonably good model. To this end, for each aspect combination, we generate an equal amount of episodes, which is set to 1000. The total number of episodes in the dataset is thus 10 $\times$ 1000 = 10000. Note that each episode is comprised of three tasks, with each task containing the triplet (x_0, x_1, x_2) and the auxiliary set of images of size 4. Hence, each episode consists of 21 images in total.

With regards to data pre-processing, the only step that is considered in practice is data normalization with mean and standard deviation. Normalizing the images means transforming the images into such values that the mean and standard deviation of the image become 0.0 and 1.0, respectively. This is done by first subtracting the first channel mean from each input channel and then dividing the result by the channel standard deviation. No data augmentation techniques are deemed necessary since the generated geometric shape is a very controlled dataset.

6.2 Model setup

In this section, we detail the setup of the aspect-based representation network.

6.2.1 Representation learning

For learning rich and meaningful representations of the input data, we rely on a deep CNN backbone. Chapter 5 motivates the choice of CNN architectures that are used for representation learning, namely VGG and ResNet. To extract the representation, we discard the classification component of the model that is specific to the original classification task and to the set of classes on which the model was trained. The base convolutional network already contains features that are generically useful. The CTM module, which will be trained from scratch, is placed on top of the pre-trained model so that the feature maps learned previously for the dataset can be repurposed.

Shallow representation network

VGG and ResNet are deep, complicated architectures. In order to get a broader overview, we also assess the model performance in the case of a simple backbone network. Specifically, a 4-layer neural network is adopted as the backbone for representation learning. Each convolutional layer has a kernel size of 3. The first two layers have a padding of 0 and each is followed by a batch normalization layer, subsequently, a ReLU activation layer after which max pooling is applied. The two subsequent layers have a padding of 1 where each layer is followed by a batch normalization layer and a ReLU activation layer.

VGG

VGG-16 is a convolutional neural network that is 16 layers deep (14 convolution layers, 2 fully connected layers) with 5 MaxPool layers and 1 softmax layer that is trained on more than a million images from the ImageNet database [7]. The network is pre-trained to classify images into 1000 different object categories. As a result, the network has learned rich feature representations for a wide range of images. The network has an image input size of 224x224. The output representation is extracted from the last layer before the classification component. The extracted feature is thus of shape (batch size, 517, 7, 7).

ResNet

We consider the ResNet-18 architecture which is also trained on the large database of ImageNet. Similarly, the network receives an input size of 224x224. The feature is extracted from the last

convolution block, which also results in a representation of shape (batch size, 517, 7, 7).

6.2.2 Category Traversal Module

The concentrator and the projector of the CTM are implemented as convolutional layers of stride 1. The architecture of these components consists of two convolutional blocks.

- The first block is comprised of bottleneck building blocks, each bottleneck block has three convolutional layers, each followed by a batch normalization layer and the last layer is the ReLU activation layer. The kernel size of each convolutional layer is 1, 3 and 1, respectively. The first bottleneck includes a downsampling element.
- The second block follows a similar structure but only consists of two bottleneck building blocks.

6.3 Experiment description

In this section, we describe the experiment setup which is used to analyze the performance of the proposed framework. This setup contains several components whereafter three experiments are drafted. This first component is the mapping of the aspect-based representation of the same triplet into a specific aspect-based embedding space, which is the basis for the comparison between the two proposed loss functions in the same settings. The second component is to study the aspects. The way that the geometric shape dataset is constructed allows us to monitor and control the generated aspects. This prompts an experiment that studies how well the network learns each aspect. The third experiment compares different base networks for representation learning.

6.4 Performance evaluation

In this section, we present different evaluation metrics so as to assess the performance and efficacy of the proposed framework.

6.4.1 Aspect-based few shot learning accuracy

We report the mean accuracy (%) of 1940 generated episodes on the test set. Every episode during inference has three few-shot tasks, each described by an aspect. For each task, we calculate the Euclidean distance between the aspect-based query embedding and the embedding of each instance of the support set. The task is successfully achieved if the support set instance whose smallest distance to the query matches a so-called "ground truth", typically in the form of labels. Since label-based supervision is a hindrance that limits the model performance in the aspect-based FSL setting, we assume no label information. Typical FSL accuracy is calculated by the number of test cases in which the predicted label of the query matches the true label. However, we are then not able to assess whether the aspect assignment to the query is correct using the defined metric. Therefore, we simulate a virtual participant in the psychometric test who always precisely responds to the tests based on a given knowledge about the aspects so that the accuracy could be measured objectively by this similarity. Furthermore, we also report the percentage of episodes where none, one, two or all three of the aspect-based tasks are done correctly.

6.4.2 Pair-wise relative distance accuracy

The model is shown to serve its object if, for a triplet (x_0, x_1, x_2), the relative pairwise distances between the learned representation are logical. As demonstrated in Chapter 4.2, in aspect-based embedding space, the embeddings of the two objects that are perceived to be more similar in regards to that aspect should be smaller than in other aspect-based embedding spaces. In this regard, we report the mean accuracy (%) on the same 1940 generated test episodes. For each task

in an episode, the pairwise Euclidean distances between the aspect-based embeddings of the triplet (x_0, x_1, x_2) are calculated, and the pair with the smallest distance matches is checked against the response from the aforementioned virtual participants. Similar to the previous performance metric, we also report the number of episodes where the relative distances are sensible in none, one, two or all three aspect-based tasks.

6.5 Environment setup

The experiments are implemented in PyTorch (version 1.12.1+cu113) [29] on Google Colab Pro+ [4]. The train-validation-test split is done in the 60%-20%-20% ratio. For training, the learning rate is initialized as 0.0001. A learning rate scheduler is adapted, which seeks to adjust the learning rate during training by reducing the learning rate according to a pre-defined schedule. Specifically, the scheduler reads a metrics quantity, and if no improvement is seen for a 'patience' number of epochs, the learning rate is reduced by a factor of $2 - 10$. The margin m for deep metric learning is set to 0.1. L2 regularization was added to the weights and was set to 0.00001. Furthermore, a regularization technique called early stopping is applied. This is done by storing the network parameters of the moment where the validation error is the lowest, thus using the network parameters which generalize the best instead of the ones that capture the training set the best. If the model does not improve, i.e., the validation error does not increase after a number of epochs, which is defined to be 7 in our case, the training process terminates. The output dimension of the network is set to 16.

Chapter 7

Results

This chapter assesses the performance of the proposed solution method in several experiments. Each experiment exposes different factors of the solution methods, which can be of help in the evaluation of the general performance. The analyses are done on a test set that comprises 1940 episodes, each containing three few-shot tasks, in total there are $3 \times 1940 = 5820$ individual few-shot tasks. The results and analyses of these experiments are structured as follows. We first present the experimental results and analysis of the performance of the aspect-based representation network for both proposed loss functions in Section 7.1. The results are provided for different base representation networks. Next, we investigate individually each aspect of how they are learned and how good the network is at doing so in Section 7.2. The experimental results are then assessed against a baseline performance. Attributed to the ability to view the support set as a whole, CTM has been shown to be able to learn to represent the input data with embeddings that are discriminative and informative to the specific relevant feature dimension (which we refer to as "aspect"). We design the baseline experiment such that the performance of the CTM without training with the proposed deep metric learning model can be evaluated in the exact same settings. We conclude with an assessment of the experimental results that constitute the proposed approach and a discussion of the insights.

7.1 Experimental results and analysis of the proposed loss functions

In the proposed framework, we present two different loss functions, namely $\mathcal{L}_{\text{tuplet}}$ and $\mathcal{L}_{\text{triplet}}$. Each function characterizes a slightly different aspect-based representation network. Table 7.1 displays the results on the aspect-based FSL accuracy for each loss function with different base representation networks, namely a simple 4-layer CNN, ResNet-18 and VGG-16 on the test set. In this table, one can find the total aspect-based FSL accuracy calculated on individual few-shot task levels, and the percentage of episodes where none, one, two or all three of the aspect-based tasks are done correctly, respectively.

The baseline in the first row evaluates a the model with the CTM component that is trained on batches of labelled data that contains N classes and K samples for each class. Rows (ii)-(vii) show a model that includes our episode structure, trained on the proposed loss functions $\mathcal{L}_{\text{tuplet}}$ and $\mathcal{L}_{\text{triplet}}$ and with different base representation network architectures. In terms of total accuracy, on average there is a 4-9% relative improvement using our proposed framework compared to the baseline. Notably, the improvements using the shallow representation network that comprises a 4-layer CNN are substantial for both proposed loss functions, whereas using deep, complex neural network architectures only marginally improves over the original baseline. In addition, the percentage of episodes where all aspect-based few-shot tasks fail decreases significantly. On a more

		Total accuracy (%)	No aspect correct (%)	One aspect correct (%)	Two aspects correct (%)	All aspects correct (%)
(i)	Baseline	13.81	62.68	33.4	3.71	0.21
(ii)	Tuplet loss + ResNet	17.34	54.85	38.61	6.24	0.31
(iii)	Triplet loss + ResNet	20.33	47.99	43.45	8.14	0.41
(iv)	Tuplet loss + VGG	19.73	46.75	47.37	5.82	0.05
(v)	Triplet loss + VGG	18.33	52.84	39.48	7.53	0.15
(vi)	Tuplet loss + 4-layer CNN	22.29	42.89	47.68	9.12	0.31
(i)	Triplet loss + 4-layer CNN	21.12	44.43	47.94	7.47	0.46

Table 7.1: Total aspect-based FSL accuracy calculated on individual few-shot task level, and the percentage of episodes where none, one, two or all three of the aspect-based tasks are done correctly, respectively for each loss function and base representation network architecture.

granular level of episodes, the percentages of episodes where one, two and all of the aspect-based few-shot tasks succeed all increase compared to the baseline. This shows that the performance increase obtained by using the proposed aspect-based representation network is indeed due to its ability to learn the relevant aspect for each task.

One caveat with looking at the aspect-based FSL accuracy as a performance metric is that it does not reflect the relative pairwise distance comparison. Consider a triplet (x_0, x_1, x_2) and its three corresponding few-shot tasks where each x_i, $i = 1, 2, 3$ takes turns to be the query. In the first task, x_0 is the query whereas the rest belongs in the support set. While aspect-based FSL accuracy can indicate if (x_0, x_2) is indeed the correct matching pair, which means that $d(x_0, x_2) < d(x_0, x_1)$, the measure fails to include the relative distances with $d(x_1, x_2)$. On this account, as elaborated in Chapter 6.4, we assess the pair-wise relative distance accuracy. In a similar manner to Table 7.1, we provide in Table 7.2 the total pairwise relative distance accuracy, and the percentage of episodes where the relative distance in none, one, two or all three of the aspect-based tasks are reflected accurately.

		Total accuracy (%)	No aspect correct (%)	One aspect correct (%)	Two aspects correct (%)	All aspects correct (%)
(i)	Baseline	33.14	28.25	47.16	21.49	3.09
(ii)	Tuplet loss + ResNet	33.99	21.60	56.86	19.54	2.01
(iii)	Triplet loss + ResNet	34.19	13.56	71.80	13.14	1.49
(iv)	Tuplet loss + VGG	33.75	10.46	78.30	10.77	0.46
(v)	Triplet loss + VGG	34.45	19.54	59.18	19.69	1.60
(vi)	Tuplet loss + 4-layer CNN	33.92	12.63	73.92	12.53	0.93
(i)	Triplet loss + 4-layer CNN	33.81	13.35	72.89	12.73	1.03

Table 7.2: Total pairwise relative distance accuracy calculated on individual few-shot task level, and the percentage of episodes where none, one, two or all three of the aspect-based tasks are done correctly, respectively for each loss function and base representation network architecture.

Here, in terms of total accuracy, the improvements are only marginal compared to the baseline performance. The percentage of episodes where the relative distances are reflected correctly in none of the aspects decreases significantly. While the percentage of episodes where the relative distances are reflected correctly in one or two of the aspects increases, the framework fails to be able to do so for all aspects of each episode.

We visualize a case where the pairwise relative distances are accurately reflected in an episode in Figure 7.1. There are three few-shot tasks in this episode, each based on a different aspect: pattern, color and shape as displayed on the left. For each aspect-based few-shot task, the aspect-based embeddings are obtained for x_0, x_1, x_2 and the pairwise Euclidean distance is calculated. It is apparent in the first task, where the aspect is pattern, x_0 and x_2 are closest as they share the same pattern which also matches the response from the psychometric test. In the second task, the relevant aspect is color which changes the pairwise relative distance as, given the aspect

color, x_0 and x_1 become the most matching pair. In the third task, when the aspect changes to shape, the pairwise relative distance also changes. This aligns with our proposition that given different aspects, different aspect-based embedding spaces exist and the relative distances between embeddings that are projected onto such spaces also change. This further indicates that the proposed aspect-based representation network is not only able to learn the task-relevant aspects but is also able to learn the corresponding aspect-based embedding space.

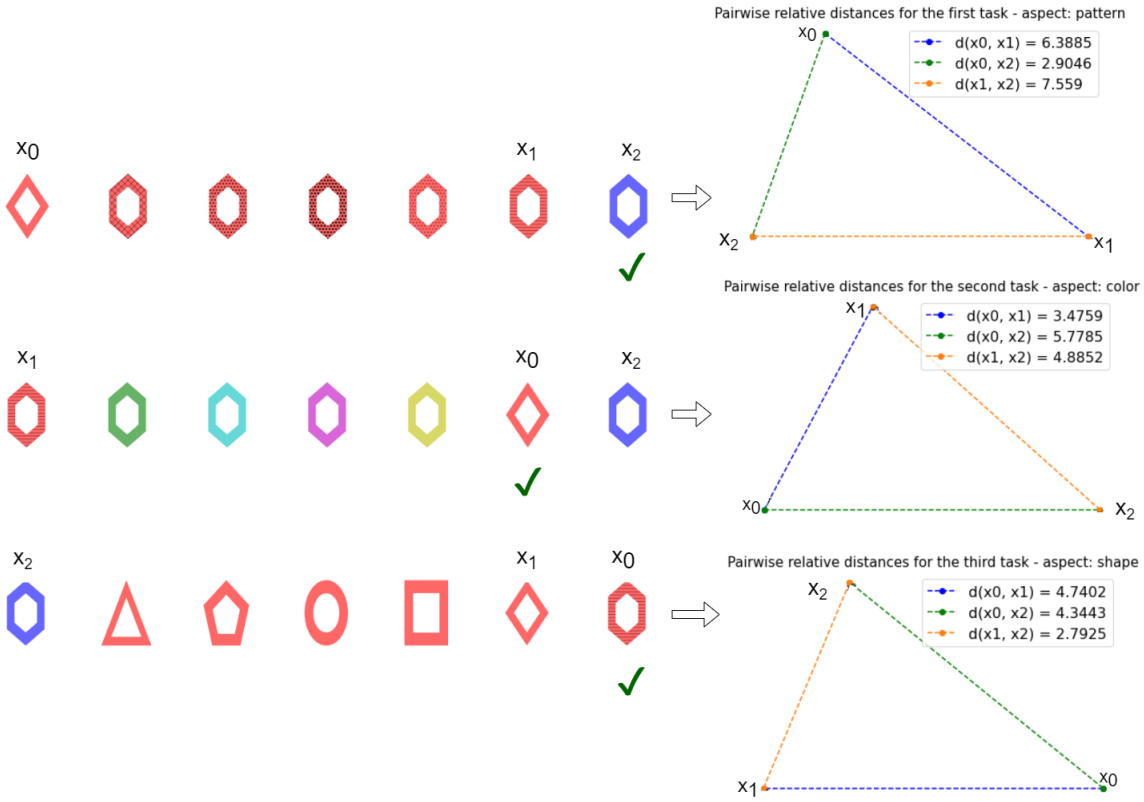
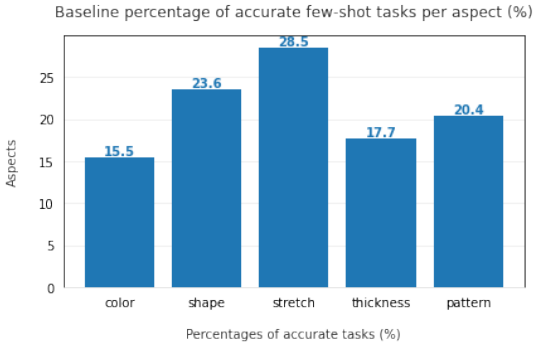


Figure 7.1: An episode containing three few-shot tasks, each based on a different aspect: pattern, color and shape as displayed on the left. For each aspect-based few-shot task, the aspect-based embeddings are obtained for x_0, x_1, x_2 and the pairwise Euclidean distance is calculated, as shown on the right.

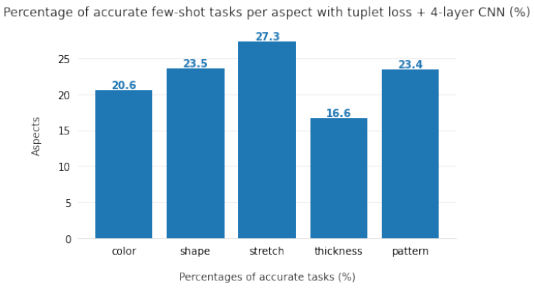
7.2 Experimental results and analysis of the learned aspects

There are five attributes that constitute a geometric shape, namely color, shape, stretch, thickness and pattern, which can correspond to five different aspects. We investigate individually each aspect on how they are learned and how well the network performs per aspect. This is done by extracting all individual few-shot tasks that are of the same aspect from all episodes. Per aspect, we assess the aspect-based FSL accuracy and compare the performance against the baseline. Figure 7.2 presents a bar chart that illustrates the accuracy per aspect.

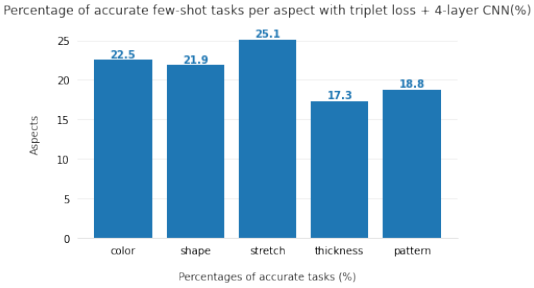
There are significant differences in the baseline percentage of accurate few-shot tasks among the aspects. While the aspect of stretch appears to have the highest baseline accuracy, the lowest is observed in color. This is to be expected as the baseline model uses a CNN backbone. At different layers, the network learns to detect features at different levels of abstraction, with low-level features being edges and color patterns, with higher-level patterns combining these into texture, and shape



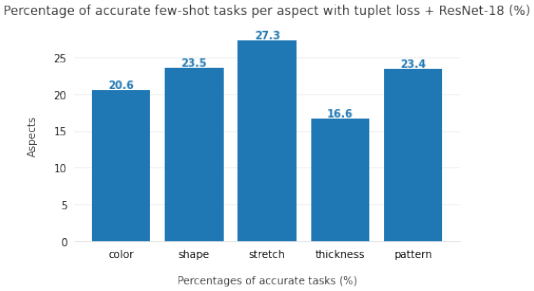
(a) Baseline.



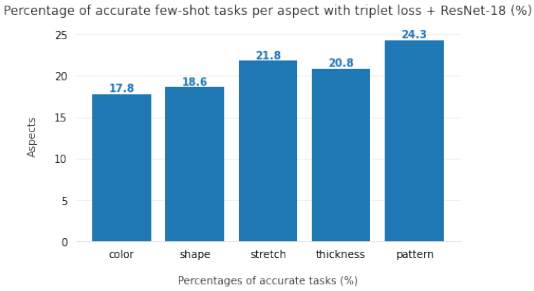
(b) Tuplet loss $\mathcal{L}_{\text{tuplet}}$ + 4-layer CNN.



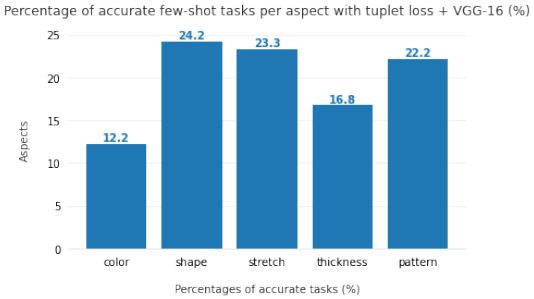
(c) Triplet loss $\mathcal{L}_{\text{triplet}}$ + 4-layer CNN.



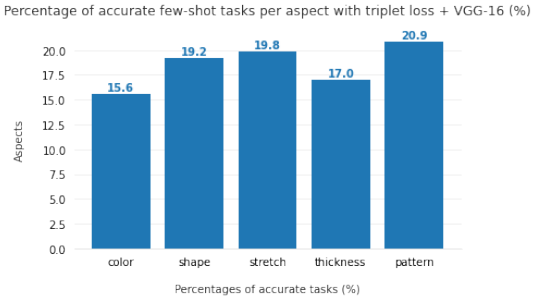
(d) Tuplet loss $\mathcal{L}_{\text{tuplet}}$ + ResNet-18.



(e) Triplet loss $\mathcal{L}_{\text{triplet}}$ + ResNet-18.



(f) Tuplet loss $\mathcal{L}_{\text{tuplet}}$ + VGG-16.



(g) Triplet loss $\mathcal{L}_{\text{triplet}}$ + VGG-16.

Figure 7.2: Model performance on each aspect for each loss function and base representation network architecture.

features and the model learns the most characteristics that help it best distinguishing different data input. This explains why shape, stretch and pattern have higher accuracy not only in the baseline but also in variations of the proposed framework. Moreover, the invariance of the CNN to an artefact is derived from the data. The CNN backbone only has the data to learn whether color is a decisive factor for recognizing an object or not, which is not the case for geometric shapes. That is, by being exposed to a large number of different geometric shapes that are colored differently, the CNN backbone will learn that color has negligible influence in recognizing a particular geometric shape (i.e., one is not able to recognize a shape solely based on its color).

With the proposed aspect-based representation network, the model is forced to learn the features that are most discriminating given an aspect. This means that when color is the aspect, it becomes the most discriminative feature to carry out a successful few-shot task. This is reflected in the improvements in accuracy for the aspect of color, as can be seen in variations of the proposed framework, with the exception in Figure 7.2f.

7.3 Discussion

In this section, we provide a brief summary of the insights that are gained from the experimental results. Given the ability to consider the support set as a whole, CTM has been shown in previous works that it can represent the input data with the most discriminative and informative features to the task-specific aspect. However, in this work, we have demonstrated through baseline performance that a base representation network with CTM on top performs relatively poorly in our setting. In an attempt to improve this, we have proposed an aspect-based representation framework that is trained on two different loss functions: the aspect-based tuplet loss and the aspect-based triplet loss. In addition, we also experiment with different base networks for representation learning, namely a shallow network that comprises a 4-layer CNN, ResNet and VGG. While there is no clear indication of which loss function or base architecture for representation learning is better, it is notable that the improvements using the proposed framework are evident when compared against the baseline. This implies that the performance increase obtained using the proposed aspect-based representation network is indeed due to its ability to learn the relevant aspect for each task. In this part, CTM plays a central role by first learning the commonalities with the concentrator and then the discriminative features that are aspect relevant with the projector. Even though not discussed by the original authors, it is notable that this concentrator-projector component in the design of CTM conforms with the structural psychological approaches to similarity in humans. Gentner and Markman argued that commonalities and differences are not psychologically independent in humans [11]. As a matter of fact, determining the differences between a pair of examples requires first finding their commonalities. As this difference requires first finding a commonality between the pair, it is referred to as an alignable difference, which has significant impact on people's judgments of similarity.This thus highlights the importance of the relationship between the commonalities of a pair and the differences in understanding people's assessments of similarity.

Furthermore, the experimental results on the pairwise relative distances for each triplet of images (x_0, x_1, x_2) confirms our proposition in Chapter 4.2 that each aspect defines an aspect-based embedding space. In such spaces, entities that are more similar based on this aspect are closer together by a proper distance metric. Even though there are some improvements when compared to the baseline with regards to the number of episodes where the relative distances are reflected correctly on one of two of the aspects. However, the accuracy of all correct aspect-based few-shot tasks is relatively low. This shows that the current proposed method does learn the aspects but only to some extent.

Chapter 8

Conclusions and Recommendations

This thesis project addresses the intricate problem of label-based supervision approaches in few-shot learning (FSL). When it comes to dealing with image data, FSL faces some limitations as it does not take into account the varying aspects in the data. Due to the semantic richness of the image data, associating one example with another can be multidimensional and conditioned on different factors and aspects. We explicate that the support set may describe and exhibit various aspects and that when the support set changes, so do the way associations and mappings are done. While humans are intrinsically fast in recognizing and understanding the various aspects conveyed in an image, it remains a challenge for machine learning models to do so. This thus motivates our work to extend the capabilities of few-shot learning models to learn the aspects between unseen data instances independently of the class labels. In this work, we propose a method that not only deviates from label-based supervision but also considers the aspects defined by the support set for each few-shot task. Toward this end, different research questions have been devised. The main research question concerns how to effectively learn the relevant aspect for correctly matching the query to a support set object for each few-shot task agnostic of fixed labels. We have developed an aspect-based representation framework that incorporates the Category Traversal Module, which learns the most discriminative and meaningful representations based on the aspect learned from the support set. While doing so, this begs the question of how to implement an effective and efficient procedure such that the humans' perception and knowledge of objects can be elicited and expressed without the mapping to a predefined set of concepts as labels. In our solution method, we employ a variant of the psychometric testing procedure, which plays a central role in fine-tuning the model and creating the ground truth for evaluating our approach. In addition, we devise the notion of aspect-based representation and aspect-based embedding space. We postulate that once the aspect is determined, the input data can be embedded to capture the most relevant, meaningful features w.r.t the aspect. The embedding of the same data point is thus different per aspect and specific to the aspect at hand. We refer to the embedding space in which various data points are explicitly mapped based on a particular aspect as the aspect-based embedding space.

The experimental results and analyses have shown that the proposed aspect-based representation network is capable of learning the relevant aspect for each task. In this part, CTM plays a central role in discovering the commonalities and the changing discriminative factors in each support set that is defined to be the aspect. Furthermore, the experimental results on the pairwise relative distances for each triplet of images confirm our proposition on the aspect-based embedding space: different aspects define different aspect-based embedding spaces. Even though there are some improvements compared to the baseline, the chances that the model is able to perform well on all

aspect-based few-shot tasks in our proposed setting are not yet significant. The reason for this is that psychometric testing in a natural setting requires one or more human participants. Albeit its efficacy, this is still an expensive procedure to implement. Within the feasibility scope of this thesis, we simulate a virtual participant that always precisely responds to the tests based on a given knowledge about the aspects present in the geometric shape dataset. This is quite simplistic as there are only five aspects that can be derived from the data compared to other datasets with more complex attributes. Thus, the improvement is not significant and salient.

The slight improvement shows that the proposed method does learn the aspects but only to some extent, which implies that there are several limitations to the proposed methods which lead to the recommendations for future research. Firstly, the proposed method is developed and evaluated on one dataset and only on the 6-way-1-shot problem setting, i.e., each support set has 6 classes and 1 instance each. Thus, it is not possible to conclude how well the model can generalize to different datasets with different characteristics and complexities. This requires a more extensive evaluation of more datasets when the number of classes and the number of instances per class change.

Secondly, the training setup, which includes a triplet of images and three different few-shot tasks describing three aspects, is rather complicated. The way that the geometric shape dataset is generated and its simplicity has enabled the data to be sampled accordingly to the setup of this training procedure. Thus, when dealing with when dealing with more complex datasets, it becomes a challenge to generate triplets and tasks that are adequately meaningful for both the psychometric testing procedure and for training the model. An approach to address this is to derive different semantic hierarchical trees such that each pair from the triplet comes from a different hierarchy. This allows for the auxiliary instances to be sampled such that an aspect can be defined, and thus the tasks can be structured as in the proposed model. The psychometric testing can then be simulated with a virtual participant that always responds according to its given latent hierarchical models. In this manner, one can objectively evaluate the precision of the model against the one given to the virtual participant.

Furthermore, since the base representation network also plays a role in learning the most salient, discriminative embeddings, a possible direction is to investigate how to improve the base architecture for more effective representation learning. A prominent approach is self-supervised learning. When human supervision is expensive to obtain, self-supervised learning can serve as a general framework that learns features without human annotations by solving pretext tasks. Employing self-supervised learning as a base representation network has shown remarkable performance in deep metric learning, as shown by Lu et al. [49].

Bibliography

[1] Md Zahangir Alom, Tarek M Taha, Chris Yakopcic, Stefan Westberg, Paheding Sidike, Mst Shamima Nasrin, Mahmudul Hasan, Brian C Van Essen, Abdul AS Awwal, and Vijayan K Asari. A state-of-the-art survey on deep learning theory and architectures. *Electronics*, 8(3):292, 2019. 9

[2] Hossein Azizpour, Ali Sharif Razavian, Josephine Sullivan, Atsuto Maki, and Stefan Carlsson. From generic to specific deep representations for visual recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 36–45, 2015. 27

[3] Luca Bertinetto, João F. Henriques, Jack Valmadre, Philip Torr, and Andrea Vedaldi. Learning feed-forward one-shot learners. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. 12

[4] Ekaba Bisong. *Google Colaboratory*, pages 59–64. 09 2019. 38

[5] Jerome S Bruner, JJ Goodnow, and George A Austin. A study of thinking. new york: Science editions. *Tinc., l962*, 1967. 1

[6] Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Frank Wang, and Jia-Bin Huang. A closer look at few-shot classification. *arXiv preprint arXiv:1904.04232*, 2019. 14

[7] 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. 36

[8] Gustav Theodor Fechner. *Elemente der psychophysik*, volume 2. Breitkopf u. Härtel, 1860. 13

[9] Hsin-Chang Feng, Michael W Marcellin, and Ali Bilgin. A methodology for visually lossless jpeg2000 compression of monochrome stereo images. *IEEE Transactions on Image Processing*, 24(2):560–572, 2014. 13

[10] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1126–1135. PMLR, 06–11 Aug 2017. 11, 12

[11] Dedre Gentner and Arthur B Markman. Structure mapping in analogy and similarity. *American psychologist*, 52(1):45, 1997. 43

[12] George A Gescheider. *Psychophysics: the fundamentals*. Psychology Press, 2013. 13

[13] Xufeng Han, Thomas Leung, Yangqing Jia, Rahul Sukthankar, and Alexander C Berg. Matchnet: Unifying feature and metric learning for patch-based matching. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3279–3286, 2015. 16

[14] Ben Harwood, Vijay Kumar BG, Gustavo Carneiro, Ian Reid, and Tom Drummond. Smart mining for deep metric learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2821–2829, 2017. 15, 16

[15] 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. 27, 28

[16] Elad Hoffer and Nir Ailon. Deep metric learning using triplet network. In *International workshop on similarity-based pattern recognition*, pages 84–92. Springer, 2015. 16

[17] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007. 24

[18] Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren. *Automated machine learning: methods, systems, challenges*. Springer Nature, 2019. 11

[19] Diederik P Kingma. &ba j.(2014). adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2015. 34

[20] Gregory Koch, Richard Zemel, Ruslan Salakhutdinov, et al. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, volume 2. Lille, 2015. 15

[21] Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995. 8, 10

[22] Hongyang Li, David Eigen, Samuel Dodge, Matthew Zeiler, and Xiaogang Wang. Finding task-relevant features for few-shot learning by category traversal. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1–10, 2019. 3, 17, 19, 23

[23] Yanbin Liu, Juho Lee, Minseop Park, Saehoon Kim, Eunho Yang, Sung Ju Hwang, and Yi Yang. Learning to propagate labels: Transductive propagation network for few-shot learning. *arXiv preprint arXiv:1805.10002*, 2018. 16

[24] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Icml*, 2010. 10

[25] Hyun Oh Song, Stefanie Jegelka, Vivek Rathod, and Kevin Murphy. Deep metric learning via facility location. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5382–5390, 2017. 16

[26] Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4004–4012, 2016. 16

[27] Boris Oreshkin, Pau Rodríguez López, and Alexandre Lacoste. Tadam: Task dependent adaptive metric for improved few-shot learning. *Advances in neural information processing systems*, 31, 2018. 16

[28] Archit Parnami and Minwoo Lee. Learning from few examples: A summary of approaches to few-shot learning. *arXiv preprint arXiv:2203.04291*, 2022. 2, 14, 16

[29] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017. 38

[30] Prajit Ramachandran, Barret Zoph, and Quoc V Le. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, 2017. 10

[31] CR Rao and Venkat N Gudivada. *Computational analysis and understanding of natural languages: principles, methods and applications*. Elsevier, 2018. 10

[32] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *International Conference on Learning Representations*, 2017. 2, 11, 12

[33] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016. 34

[34] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015. 32

[35] Roger N Shepard. Psychological relations and psychophysical scales: On the status of "direct" psychophysical measurement. *Journal of Mathematical Psychology*, 24(1):21–57, 1981. 13

[36] Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. Training region-based object detectors with online hard example mining. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 761–769, 2016. 16

[37] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 27

[38] Jake Snell, Kevin Swersky, and Richard S Zemel. Prototypical networks for few-shot learning. *arXiv preprint arXiv:1703.05175*, 2017. 2, 11, 12, 15

[39] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. *Advances in neural information processing systems*, 29, 2016. 33

[40] IY Son, M Winslow, B Yazici, and XG Xu. X-ray imaging optimization using virtual phantoms and computerized observer modelling. *Physics in Medicine & Biology*, 51(17):4289, 2006. 13

[41] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1199–1208, 2018. 15, 16

[42] Zhiqiang Teng, Shuai Teng, Jiqiao Zhang, Gongfa Chen, and Fangsen Cui. Structural damage detection based on real-time vibration signal and convolutional neural network. *Applied Sciences*, 10(14):4720, 2020. 11

[43] Evgeniya Ustinova and Victor Lempitsky. Learning deep embeddings with histogram loss. *arXiv preprint arXiv:1611.00822*, 2016. 16

[44] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. *Advances in neural information processing systems*, 29, 2016. 15, 16

[45] Jiang Wang, Yang Song, Thomas Leung, Chuck Rosenberg, Jingbin Wang, James Philbin, Bo Chen, and Ying Wu. Learning fine-grained image similarity with deep ranking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1386–1393, 2014. 16

[46] Rikiya Yamashita, Mizuho Nishio, Richard Kinh Gian Do, and Kaori Togashi. Convolutional neural networks: an overview and application in radiology. *Insights into imaging*, 9(4):611–629, 2018. 11

[47] Lu Yin, Vlado Menkovski, Shiwei Liu, and Mykola Pechenizkiy. Hierarchical semantic segmentation using psychometric learning. *arXiv preprint arXiv:2107.03212*, 2021. 18

[48] Lu Yin, Vlado Menkovski, and Mykola Pechenizkiy. Knowledge elicitation using deep metric learning and psychometric testing. *arXiv preprint arXiv:2004.06353*, 2020. 17, 18, 20

[49] Lu Yin, Vlado Menkovski, Yulong Pei, and Mykola Pechenizkiy. Semantic-based few-shot learning by interactive psychometric testing. *arXiv preprint arXiv:2112.09201*, 2021. 18, 19, 45