Enhancing Deep Transfer Learning for Image Classification

Tasfia Shermin

A dissertation submitted in fulfilment of the requirements for the degree of
Doctor of Philosophy

School of Engineering, Information Technology and Physical Sciences
Federation University Australia

13 August 2021

I certify that except where due acknowledgement has been made, that this dissertation is of the author alone; includes nothing, which is the outcome of work done in collaboration except where specifically indicated in the text; has not been previously submitted, in part or whole to any university or institution to qualify for any degree, diploma, or other qualification; that the content of the dissertation is the result of work that has been carried out since the official commencement date of the approved research program; any editorial work, paid or unpaid, carried out by a third party is acknowledged; and that ethics procedures and guidelines have been followed.

Tasfia Shermin

13 August 2021

Guojun Lu (Principal Supervisor)

13 August 2021

Dedicated to my parents and husband for their love, encouragement and support.

Acknowledgements

Praise be to Allah, the Almighty, who has endowed me with courage, steadfastness and strength to complete this thesis successfully. I can never thank Him enough for His blessings. “Whoever puts his trust in Allah, He will suffice him.” [Qur’an 65:3]

I am eternally grateful to my supervisors, Guojun Lu, Shyh Wei Teng, Manzur Murshed and Ferdous Sohel. They inspired me to work hard and provided me with continuous feedback on my research. This journey would not have been possible without their efforts.

I especially am thankful and indebted to my husband, Tahmid Hossain, for his love, support and valuable suggestions for my research.

I want to express my sincere gratitude to my fellow postgraduate students and colleagues at Federation University Australia for their support. I am also thankful to the helping staff of the School of Engineering, Information Technology and Physical sciences, Federation University Australia, for their support. I thank Capstone Editing services for assisting me in editing this thesis.

I owe all my success to my family. I want to thank my mother, Sara Lulufar, my father, Mohammad Abu Taher, my sisters and my in-laws, who always motivated me through the ups and downs of this journey.

Tasfia Shermin was supported by an Australian Government Research Training Program (RTP) Fee-Offset Scholarship through Federation University Australia.

Abstract

Though deep learning models require a large amount of labelled training data for yielding high performance, they are applied to accomplish many computer vision tasks such as image classification. Current models also do not perform well across different domain settings such as illumination, camera angle and real-to-synthetic. Thus the models are more likely to misclassify unknown classes as known classes. These issues challenge the supervised learning paradigm of the models and encourage the study of transfer learning approaches. Transfer learning allows us to utilise the knowledge acquired from related domains to improve performance on a target domain. Existing transfer learning approaches lack proper high-level source domain feature analyses and are prone to negative transfers for not exploring proper discriminative information across domains. Current approaches also lack at discovering necessary visual-semantic linkage and has a bias towards the source domain. In this thesis, to address these issues and improve image classification performance, we make several contributions to three different deep transfer learning scenarios, i.e., the target domain has i) labelled data; ii) no labelled data; and iii) no visual data.

Firstly, for improving inductive transfer learning for the first scenario, we analyse the importance of high-level deep features and propose utilising them in sequential transfer learning approaches and investigating the suitable conditions for optimal performance. Secondly, to improve image classification across different domains in an open set setting by reducing negative transfers (second scenario), we propose two novel architectures. The first model has an adaptive weighting module based on underlying domain distinctive information, and the second model has an information-theoretic weighting module to reduce negative transfers. Thirdly, to learn visual classifiers when no visual data is available (third scenario) and reduce source domain bias, we propose two novel models. One model has a new two-step dense attention mechanism to dis-

cover semantic attribute-guided local visual features and mutual learning loss. The other model utilises bidirectional mapping and adversarial supervision to learn the joint distribution of source-target domains simultaneously. We propose a new pointwise mutual information dependant loss in the first model and a distance-based loss in the second one for handling source domain bias. We perform extensive evaluations on benchmark datasets and demonstrate the proposed models outperform contemporary works.

Contents

List of Figures

List of Tables

Acronyms

ADAM	A Method for Stochastic Optimisation
AFGN	Adversarial Feature Generation Network
AGAN	Attribute Guided Attention Network
AUC-ROC	Area Under the Receiver Operating Characteristic curve
AWA	Animals with Attributes dataset
BMCoGAN	Bidirectional Mapping Coupled Generative Adversarial Network
CAL	CalTech dataset
CMD	Central Moment Discrepancy
CNN	Convolutional Neural Network
Conv.	Convolutional
CoGAN	Coupled Generative Adversarial Network
CSN	Coarse Separation Network
CUB	CalTech UCSD Birds dataset
CVAE	Conditional Variational Auto-Encoder
DA	Domain Adaptation
DAMC	Domain Adversarial network with Multiple Classifiers
DAMI	Domain Adaptation based on Mutual Information

DAN	Domain Adversarial Network
DOG	Dogs dataset
EL	Embedding Learning
FC	Fully-Connected
FLO	Flowers dataset
FSN	Fine Separation Network
FS	Feature Synthesising
GAN	Generative Adversarial Network
GZSL	Generalised Zero-Shot Learning
KL	Kullback-Leibler
K-NN	K-Nearest Neighbour
LR	Learning Rate
MIT	Massachusetts Institute of Technology dataset
MINE	Mutual Information Neural Estimator
MLE	Maximum Likelihood Estimation
MLP	Multi-layer Perceptron
MMD	Maximum Mean Discrepancy
MION	Mutual Information Optimisation Network
NLP	Natural Language Processing
NN	Neural Network
OSDA	Open Set Domain Adaptation

OSBP	Open Set domain adaptation by Back-Propagation
PAD	Proxy $\mathcal{A}$ Distance
PET	Pets dataset
PMF	Probability Mass Function
PDF	Probability Density Function
ReLU	Rectified Linear Unit
RBF	Radial Basis Function
SGD	Stochastic Gradient Descent
SUN	Scene Understanding dataset
SVM	Support Vector Machine
TLN	Transfer Learning Network
t-SNE	t-Distributed Stochastic Neighbour Embedding
VAE	Variational Auto-Encoder
VisDA	Visual Domain Adaptation
VOC	Visual Object Classes dataset
WGAN	Wasserstein Generative Adversarial Network
ZSL	Zero-Shot learning

Nomenclature

α	Attention
χ	CNN pre-trained by using a large dataset
ϵ	Gaussian noise
γ	Mutual information bound
λ	Hyper-parameters
$\mathbb{C}$	Center of a class
$\mathbb{D}$	Domain
$\mathbb{E}$	Expectation
$\mathbb{H}$	Entropy
$\mathbb{O}$	Openness
$\mathbb{R}$	Set of real numbers
$\mathcal{C}$	Convolutional output and input channels
$\mathcal{D}_{KL}$	Kullback-Leibler divergence
$\mathcal{H}$	Divergence
$\mathcal{L}$	Loss function
$\mathcal{R}(.)$	Risk
$\mathcal{T}$	Task

$\mathcal{X}$	Input sample/feature space
$\mathcal{Y}$	Label space
$\overline{\sigma(.)}$	Leaky softmax function
$\overline{C}$	Classes private to a domain
ψ	Classification module for new classes
$\sigma(.)$	Sigmoid/Softmax function
τ	Augmented layer
θ	Network parameters
$\top$	Transpose
b	Biases
C	Classes
E	Loss function
H	Harmonic mean
I	Mutual information
$In[.]$	Indicator function
l	Conditional log likelihood
n	Number of samples
P	Probability
p	Probability
pmi	Pointwise mutual information
s	Source domain/dataset

t	Target domain/dataset
W	Network weights
w	Generated weights
X	Random variable
Y	Random variable

Chapter 1

Introduction

Understanding the contents from images is referred to as vision. It provides the most powerful perception of the world around us. Our eyes and brain collectively process a massive amount of information within a short period to learn a remarkably rich representation of the environment in a very complicated way. Thus, one of the main challenges in the pursuit of achieving artificial intelligence is developing systems that can see and understand our environment through images. This objective has encouraged research in computer vision and strives to integrate the sense of vision into machines.

Computer vision is a challenging field because it has to model the vast complexity and variation in the representation of the visual world. Statistical approaches are channelled into creating features that guide a model of what relationships and representations in the data are to be considered to perform computer vision tasks. However, hand-engineered approaches to represent or model visual world complexities lack a satisfactory level of understanding.

Deep learning approaches can design systems that understand of representations of complex visual data. In particular, convolutional neural networks (CNN) have become the ultimate choice for learning representations from visual data. CNNs [1–4] do not require manual feature engineering; they automatically learn a multi-layered hierarchy of low to high-level features. Consequently, the focus is on determining the suitable network architecture and training hyper-parameters (e.g., learning rate and number of layers).

CNNs require a huge amount of labelled training data and are trained in a supervised learning paradigm to yield optimal performance [1,4,5]. The supervised setting binds the networks to recognise only the data, domains, and tasks similar to what they have observed during training. In addition, these networks have high computational complexity when trained from very beginning. These limitations have driven deep learning research towards transfer learning.

Transfer learning provides the ability to transfer knowledge from related (source) domains to the target domain with different underlying distributions. Transfer learning approaches do not require a massive number of data and are not computationally expensive.

Transfer learning settings vary depending on the scenarios. This thesis focuses on several transfer learning scenarios related to the availability of labelled training data in the target domain and label space (classes) across domains. In particular, we work on three scenarios: i) target domain has labelled but limited data and has different source-target labels; ii) target domain is unlabelled, and source labels are a subset of target labels; and iii) target domain has no visual data but has semantic descriptions and different source-target labels. Current deep transfer learning models neglect category-specific CNN features for handling the first scenario. Existing works for the second scenario cannot correctly handle known–unknown misclassification properly for not discovering essential domain distinctive characteristics. The research on the third scenario lacks proper exploration of local and global visual–semantic relations and has source domain bias in the learned features. This thesis proposes new transfer learning models to address the issues in the above-mentioned three transfer learning scenarios. These proposed models outperform contemporary works and supervised non-transfer learning models.

1.1 Research Background and Objectives

This section briefly describes the problem settings and limitations of the existing approaches and highlights the research objectives of the thesis. The details of existing approaches are reviewed in Chapter 2.

With the aim of improving image classification performance, this thesis studies the problem of automatically learning visual feature representations using deep neural networks that are transferable across domains in inductive and transductive transfer learning settings (see Figure 1.1). The inductive setting has limited labelled training data in the target domain, while the transductive setting has no labelled training data in the target domain.

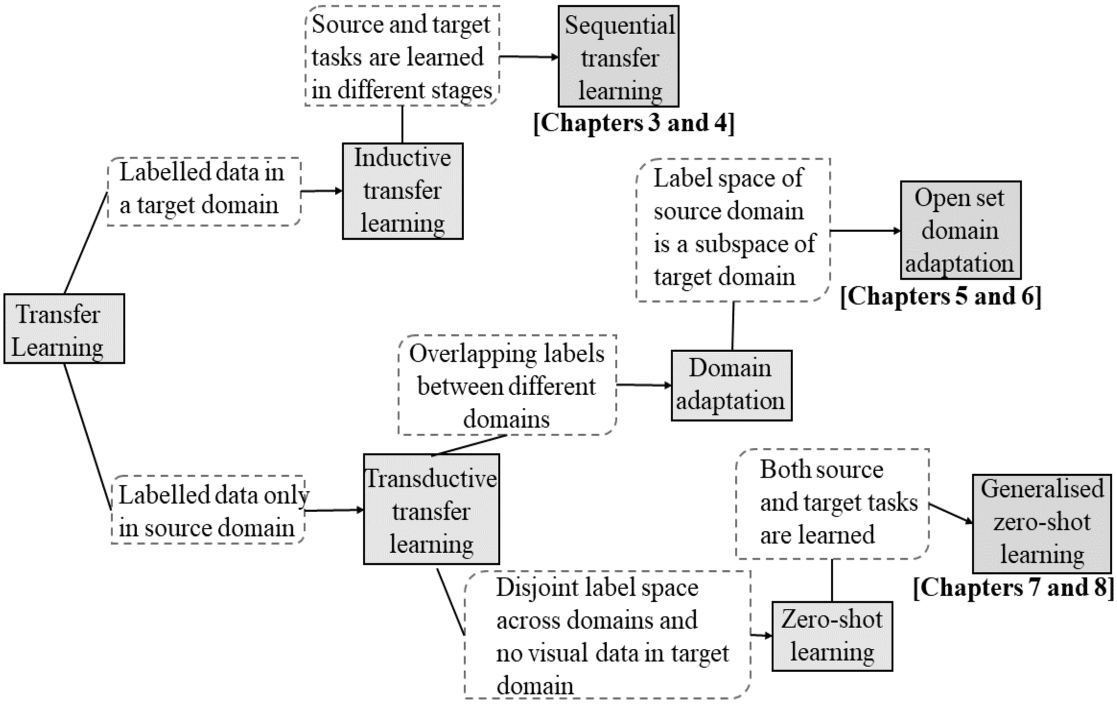

Source and target tasks are learned in different stages
Sequential transfer learning
[Chapters 3 and 4]
Labelled data in a target domain
Inductive transfer learning
Label space of source domain is a subspace of target domain
Open set domain adaptation
[Chapters 5 and 6]
Transfer Learning
Overlapping labels between different domains
Domain adaptation
Both source and target tasks are learned
Generalised zero-shot learning
[Chapters 7 and 8]
Labelled data only in source domain
Transductive transfer learning
Disjoint label space across domains and no visual data in target domain
Zero-shot learning

Figure 1.1: Contributions of this thesis.

First, this thesis aims to work on an inductive setting (i.e., *sequential transfer learning*), where the source task is learned before learning the target tasks. More specifically, a network is first trained on a dataset with a huge amount of labelled data. Then, to learn another dataset with scarce labelled samples, the knowledge is transferred by transferring the learned weights-biases or features of the network layers. Transferring learned weights and biases for tunning them on target dataset is known as *parameter fine-tuning* while transferring features is referred to as *feature representation transfer* [5,6]. In sequential transfer learning, it is expected that only generic information holding network layers are vital for transferring knowledge. It is widely believed that the final layer of CNNs hold category-specific information [5]. Therefore, the effects of this layer in sequential transfer learning have not been explored in the literature.

Second, this thesis aims to work on a transductive setting known as *open set domain adaptation* (OSDA), which originates from the *domain adaptation* (DA) setting (see Figure 1.1). The DA setting assumes a similar label set across the source and target domains. The network learns to align the target samples towards similar source classes and then learns to recognise the classes.

OSDA is a step towards a realistic and challenging setting. There are several types of OSDA settings. This thesis focuses on the type of OSDA where the source domain label set is a subset of the target domain label set. The task is to learn a classification network that aligns similar source-target samples and separate other target samples as 'unknown'. Although OSDA extends the supervised learning paradigm and allows classifying unknown target classes besides the known classes, existing works do not perform well because of known-unknown misclassification. If the DA model's performance lags behind the supervised classification model for the same task, it is referred to as negative transfer [7]. In OSDA, negative transfers occur because of faulty known and unknown target sample separation [8]. Consequently, the adaptation of known target samples to the source domain is harmed.

Finally, this thesis concentrates on *zero-shot learning* (ZSL), which has a disjoint source-target label set, and no available visual training data in the target domain. The task is to learn the target domain using visual data of the source domain and semantic descriptors of the source-target domains. *Generalised zero-shot learning* (GZSL) extends this setting where the source and target domains are simultaneously learned for classification. Existing GZSL approaches ignore essential links between visual features and semantic descriptors [9] and strong coupling between visual and semantic representations [10]. They also do not properly handle bias towards the source domain in the learned features. Consequently, they harm classification performances.

This thesis addresses the limitations and issues identified above by investigating the influence of specificity, encouraging positive transfers, exploring crucial visual-semantic relations, and reducing bias towards the source domain to improve transfer learning performances for image classification. To fulfil the ultimate objectives, we lay out the following sub-objectives to be achieved in this thesis:

1. **Improve the parameter fine-tuning approach using category-specific features of CNNs** We aim to analyse the importance of the classification layer of pre-trained CNNs (which holds category-specific features in a parameter fine-tuning transfer learning approach), propose models to utilise the layer for developmental transfer learning (another form of parameter fine-tuning) and develop optimal network architecture and training setups for proposed models.

2. **Comprehensively analyse suitable conditions for a feature representation transfer approach using the category-specific features** The goal is to find an optimal classifier and correlation between the source and the target dataset for the feature representation transfer approach using the category-specific CNN features. In particular, we aim to observe the influence of i) the similarity between source-target datasets; ii) the fine-grained target datasets; iii) the coarse target datasets; and iv) the number of target samples and classes. Moreover, we aim to employ a mutual information-based feature selection algorithm to verify the importance of category-specific features in transfer learning.

3. **Reduce negative transfer in OSDA** For improving OSDA's performance, we aim to reduce negative transfers by designing robust known-unknown separation modules, which will assist better adaptation. Our goal is to develop known-unknown target samples separation modules based on domain-distinctive characteristics, domain confidence, mutual information (I), and pointwise mutual information (pmi) to reduce negative transfers.

4. **Improve GZSL** We aim to improve GZSL performance by developing novel architectures, which will explore proper global features, local visual features related to semantic descriptors and strong bidirectional coupling between visual features and semantic descriptors.

5. **Reduce source domain bias for GZSL** For reducing bias towards the source domain in GZSL, we aim to design loss optimisations by exploring the shared information between source and target classes to transfer-learn the target classes and learning to hold domain-discriminative information.

1.2 Contributions

This thesis focuses on the setting of transfer learning that transfers across domains and requires a source and target domain in the setting. Given this basic setting, we work on three specific transfer learning settings as discussed earlier: i) sequential transfer learning; ii) open set domain adaptation; and iii) generalised zero-shot learning. Figure 1.1 demonstrated which contributions of this thesis relate to the three settings. We will discuss the settings shown in Figure 1.1 in detail in Chapter 2.

To summarise, the main contributions of this thesis are as follows:

- We propose new parameter fine-tuning approaches using the classification layer of pre-trained CNNs that hold category-specific (high-level) features. This study proves that the category-specific classification layer can transfer knowledge. We investigate the optimal architecture and hyper-parameter setting and perform an extensive evaluation to show the superior performance of the proposed transfer learning approaches on benchmark datasets (see **Chapter 3**).

- We propose to using the classification layer features of pre-trained CNNs in the feature representation transfer approach. We present comprehensive analyses to investigate the optimal network and datasets setting for this approach using the classification layer features. We perform a thorough evaluation on benchmark datasets to demonstrate the influence of classification layer features on this type of transfer learning and compare them with contemporary works (**Chapter 4**). This detailed analysis proves the positive impact of CNNs' feature specificity in the feature representation transfer approach and improves classification performance.

- We extend a threshold-dependent adversarial OSDA model to an adaptive weighting model. The proposed adaptive weighting model explores underlying domain-discriminative information to improve adversarial DA performance by reducing negative transfers. This contribution opens a new avenue to automatically learn to separate known samples from unknown samples rather than depending on a fixed threshold. We present an extensive evaluation on benchmark datasets and a detailed analysis and comparison to show the superior performance of the proposed model (see **Chapter 5**).

- We propose a novel generative adversarial model to mitigate negative transfers and improve OSDA. In a coarse-to-fine separation module, the proposed model discovers mutual dependence between domains to initiate positive transfers by optimising a new loss based on probabilistic *I*. This model reduces negative transfers by not relying only on the domain confidence of a source trained classifier. We provide a detailed analysis of evaluation on benchmark datasets and demonstrate that the proposed model outperforms all contemporary works (see **Chapter 6**).

- We propose a new integrated GZSL model with a novel two-step dense attention mechanism. The proposed model explores local information to learn essential visual-semantic relations and global information to hold a generic visual feature presentation structure. We propose novel mutual learning to benefit the integrated network. We design a new loss optimisation to reduce bias towards the source domain depending on the probabilistic *I*. Unlike existing works, this contribution explores the local and global features necessary for improving performance and avails two testing styles. We evaluate the proposed model on the benchmark datasets to demonstrate its superior performance (see **Chapter 7**).

- We propose a novel bidirectional model for GZSL. We extend a coupled generative adversarial network to a bidirectional coupled generative adversarial network to learn the joint distribution of the domains through strong visual–semantic coupling. We also design an adversarial optimisation loss to supervise the learning of features. We design a distance-based loss to hold domain discrimination in the learned features to reduce bias towards the source domain. Unlike contemporary works, the proposed model simultaneously learn the joint distribution and preserve the essential discriminative properties of the domains for improving GZSL. Our thorough evaluation shows that the proposed model outperforms other works (see **Chapter 8**).

The first and second contributions will address the first and second objectives, respectively. The third and fourth contributions will address the third objective. The fifth and sixth contributions will address the fourth and fifth objectives.

1.3 Publications

The work in this thesis relates to the following peer-reviewed articles and pre-prints:

Chapter 3 is adapted from:

- T. Shermin, S. W. Teng, M. Murshed, G. Lu, F. Sohel, and M. Paul, “Enhanced Transfer Learning with ImageNet Trained Classification Layer,” in Proceedings of the Pacific-Rim Symposium on Image and Video Technology (PSIVT), 2019. [Best Student Paper]

Chapter 4 is adapted from:

- T. Shermin, S. W. Teng, M. Murshed, G. Lu, and F. Sohel, “Suitable Conditions for Transfer Learning with the ImageNet Trained Classification Layer Features,” (to be submitted as a journal article).

Chapter 5 is adapted from:

- T. Shermin, G. Lu, S. W. Teng, M. Murshed, and F. Sohel, “Adversarial network with multiple classifiers for open set domain adaptation,” IEEE Transactions on Multimedia, 2020.

Chapter 6 is adapted from:

- T. Shermin, S. W. Teng, F. Sohel, M. Murshed, and G. Lu, "Adversarial Open Set Domain Adaptation based on Mutual Information," Revision submitted to IEEE Transactions on Image Processing, 2021. [arXiv: 2007.00384v1]

Chapter 7 is adapted from:

- T. Shermin, S. W. Teng, F. Sohel, M. Murshed, and G. Lu, "Integrated Generalised Zero-Shot Learning for Fine-Grained Classification," Pattern Recognition, 2021.

Chapter 8 is adapted from:

- T. Shermin, S. W. Teng, F. Sohel, M. Murshed, and G. Lu, "Bidirectional Mapping Coupled GAN for Generalised Zero-Shot Learning," Revision submitted to IEEE Transactions on Image Processing, 2021. [arXiv: 2012.15054]

1.4 Thesis Outline

Chapter 2 outlines the background studies relevant to this thesis's content. We briefly review the fundamentals of probability and information theory, machine learning, and deep learning for computer vision. Further, we review the transfer learning scenarios and approaches related to this thesis in detail.

Chapters 3, 4, 5, 6, 7, and 8 detail our proposed works and their performance evaluations. Chapters 3 and 4 present our works on utilising category-specific CNNs features to improve sequential transfer learning approaches and investigates the factors for optimal performance. These two chapters comprise our first two contributions (see Section 1.2).

In Chapter 5 details our third contribution. It discusses how we extend a prior threshold dependant work and propose an adaptive weighting module-based generative adversarial model for initiating positive transfers and improving OSDA.

Chapter 6 thoroughly discusses our fourth contribution. It formalises and presents our proposed adversarial DA model, which has a coarse-to-fine separation weighting module based on mutual information for mitigating negative transfers and improving OSDA.

Chapter 7 presents the proposed model for GZSL with a novel attention mechanism and mutual learning loss, which is the fifth contribution of this thesis. It also discusses the proposed loss for reducing bias towards the source domain in detail.

In Chapter 8, we present the details of our final contribution. Chapter 8 introduces our proposed bidirectional GZSL, which explores joint distribution using strong visual-semantic bidirectional coupling and optimises new distance-based loss for learning domain discrimination to handle bias.

Chapter 9 provides the conclusion summarising our contributions, findings and discussing future research directions.

Chapter 2

Literature Review

In this chapter, we provide the background information, which forms the building blocks of the approaches introduced throughout this thesis. First, we review the fundamentals of probability and information theory. Then we discuss in brief elementary machine learning approaches and subsequently, delve into deep learning models or deep neural networks for computer vision. Finally, we provide an overview of the literature of transfer learning in computer vision and for image classification in particular.

2.1 Probability and information theory

All the approaches introduced in this thesis are probabilistic in nature and can be formulated using the contents of probability theory. In the proposed approaches we aim to computationally model the likelihoods of occurrence of events. The encoded information in the events, overlap or difference among information of events can be formulated using information theory.

2.1.1 Probability theory fundamentals

Random variable A random variable is a function that outputs specific values within a range that depends on an underlying *probability distribution*. For example, a random variable X can output the values x_1 and x_2. The occurrence of x_1 and x_2 is specified by a probability distribution. A random variable can be *discrete* or *continuous*. In this thesis, we aim to design architectures that model the probability distribution of such random variables.

PMF and PDF For *discrete* random variables, the probability distribution is defined with a *probability mass function* (PMF), generally denoted as P. The probability of a random variable taking on a value is defined by its PMF. Thus, $P(X = x_1)$ indicates the probability of X taking on the value x_1. In case, we do not know to which a random variable an event belongs, we will denote as $P(x_1)$. Every event $x_i \in X$ has a probability value between 0 to 1 and all probabilities must sum to 1. For *continuous* random variable, we use *probability density function* (PDF), PDF does not denote the probability of a specific event but rather, measures the probability of a infinitesimally small region with volume δx with $P(x)\delta x$.

Joint and marginal probability distribution A probability distribution that calculates the likelihood of multiple events belonging to multiple random variables occurring at the same point in time is denoted as *joint probability distribution*. For events, $x \in X$ and $y \in Y$, $P(X = x, Y = y)$ or $P(x, y)$ denotes their joint probability.

Given a joint probability, the probability distribution over a subset of the variables is the *marginal probability distribution*. For discrete variables, the *marginal probability distribution* is:

$$P(x) = \sum_{y} P(x, y). \tag{2.1}$$

For continuous variables,

$$P(x) = \int_{y} P(x, y) dy. \tag{2.2}$$

Independence If the joint probability distribution of two random variables can be computed as a product of their individual probability distributions, the two random variables are *independent*. That is, for all $x \in X$ and $y \in Y$:

$$P(x, y) = P(x)P(y). \tag{2.3}$$

i.i.d Two random variables are *independent and identically distributed* (i.i.d) if and only if the following two conditions are satisfied: the terms of the random variables are mutually independent; they both have the same underlying probability distribution.

Expectation Machine learning concerns with functions over random variables. The value of a function $f(x)$ takes on with regard to $P(X)$ when x is drawn from P. The average value from this phenomenon is known as *expectation* or *expected value*. For discrete variables it is formulated as:

$$\mathbb{E}_{x \sim p}[f(x)] = \sum_{x} P(x) f(x). \tag{2.4}$$

For continuous variables:

$$\mathbb{E}_{x \sim p}[f(x)] = \int P(x) f(x) dx. \tag{2.5}$$

Note that, we will often write $\mathbb{E}_x[f(x)]$ when it is evident from which distribution x is drawn and $\mathbb{E}[f(x)]$ if the random variable is also clear from the context.

Variance and covariance *Variance* is measuring how much the values of a function $f(x)$ differ from the expectation as we sample or draw different values of x from P. When it comes to considering more than one random variables, we measure the *covariance* to determine the linear relation between the random variables. Two independent variables have zero covariance and two dependent variables have non-zero covariance. For instance, covariance between X and Y is:

$$Cov(X, Y) = \mathbb{E}[(X - \mathbb{E}[X])(Y - \mathbb{E}[Y])]. \tag{2.6}$$

2.1.2 Probability distributions

Machine learning often concerns with some common probability distributions. Throughout this thesis, we mainly utilise the principle of *Bernoulli distribution* for binary classifications, *multinoulli distribution* for categorical or multi-class classifications. Details on the procedures are provided in the respective sections.

2.1.3 Information theory

Information theory [11] is the core of machine learning. In this thesis, we will utilise measure from information theory to formulate the discrepancy between the encoded information of two probability distributions. This will assist us to guide the model to learn the correct probability distribution rather than the empirical distribution of the data. In this process of learning, we will also exploit shared information between two probability distributions.

Self-information and entropy The information content of an event can be measured through self-information as,

$$I(x) = -\log P(x). \tag{2.7}$$

Here, $\log$ is the natural logarithm.

Another measure of information theory is *Shannon entropy*, which determines the expected amount of information when an event x is sampled from P:

$$\mathbb{H}(x) = \mathbb{E}_{x \sim P}[I(x)] = -\mathbb{E}_{x \sim P}[\log P(x)]. \tag{2.8}$$

The amount of uncertainty contained in a probability distribution is measured by the entropy. The notion of entropy can be extended to two distributions and measure the *relative entropy* of $P(x)$ with respect to $Q(x)$. This is also known as *Kullback-Leibler divergence* (KL divergence).

$$\mathcal{D}_{KL}(P||Q) = \mathbb{E}_{x \sim P}[\log \frac{P(x)}{Q(x)}] = \mathbb{E}_{x \sim P}[\log P(x) - \log Q(X)]. \tag{2.9}$$

Kullback-Leibler divergence is asymmetric in nature, i.e., $\mathcal{D}_{KL}(P||Q) \neq \mathcal{D}_{KL}(Q||P)$. *Jensen-Shannon divergence* is a symmetric alternative, which is often used for adversarial optimisation. We will define the adversarial optimisation later in the thesis.

Another measure closely related to KL divergence, which will be used several times in this thesis is *cross-entropy*, as defined below:

$$\mathbb{H}(P, Q) = -\mathbb{E}_{x \sim P}[\log Q(X)]. \tag{2.10}$$

In this thesis, we will use the cross-entropy concept for computing multi-class classification error. For measuring binary classification error, information theory has another measure known as binary cross-entropy:

$$\mathbb{H}_P(Q) = -\mathbb{E}_{x \sim P}[\log P(X)] + \mathbb{E}_{x \sim P}[\log 1 - P(X)]. \tag{2.11}$$

Mutual information and pointwise mutual information *Mutual information* is a measure to associate the outcomes of two discrete or continuous random variables X and Y. It measures the mutual dependence between the two random variables.

For discrete variables,

$$I(X;Y) = \sum_{y \in \mathcal{Y}} \sum_{x \in \mathcal{X}} P(x,y) \log \left(\frac{P(x,y)}{P(x)P(y)} \right). \tag{2.12}$$

For continuous variables,

$$I(X;Y) = \int_{\mathcal{Y}} \int_{\mathcal{X}} P(x,y) \log \left(\frac{P(x,y)}{P(x)P(y)} \right) dxdy. \tag{2.13}$$

Pointwise mutual information (pmi) is a measure of co-occurrence between two events of two discrete random variables X and Y. The pmi is measured by computing the discrepancy between the joint probability and the product of their individual probabilities:

$$pmi(x;y) = \log \left(\frac{P(x,y)}{P(x)P(y)} \right). \tag{2.14}$$

We will use mutual information and pointwise mutual information to learn association and divergence between the samples of two domains.

2.2 Machine learning

In this section, we review the basic concepts of machine learning which will reappear throughout the thesis or form the building blocks of the advanced neural network architectures introduced in the later chapters.

In machine learning, each input sample is represented as a vector $x \in \mathbb{R}^{\mathcal{X}}$ of $\mathcal{X}$ features, where each sample is drawn from a distribution p_{data}. We focus on the *supervised* and *unsupervised* categories of machine learning in this dissertation. In supervised learning, every input x_i has a label y_i, however, unsupervised learning has no associated label with the samples. This thesis focuses on the task of *classification* in the field of machine learning. Classification refers to the task of classifying samples into predefined *classes* or *categories*. Classification can be in the form of *binary classification*, *multi-class classification*, and *multi-label classification*. Our ultimate goal is to improve multi-class classification. We often utilise binary classification to fulfill our ultimate goals throughout the thesis.

2.2.1 Maximum likelihood estimation

A *maximum likelihood estimation* (MLE) model is formulated as a function $p_{model}(x; \theta)$ which maps an input x to a probability distribution using a set of parameters θ. The true probability of the inputs is unknown so we approximate it with the empirical distribution. The objective of MLE is to decrease divergence between the probability distribution of the model parameters and the empirical distribution. In particular, MLE tries to maximise the likelihood of the samples under the model configuration as,

$$\theta_{data_{MLE}} = \underset{\theta}{\operatorname{argmax}}\, P_{model}(X; \theta) = \underset{\theta}{\operatorname{argmax}} \sum_{i=1}^{n} \log P_{model}(x_i; \theta). \tag{2.15}$$

It can be derived that this objective is similar to minimising the cross-entropy (Eq. 2.10) between the empirical and model distributions as:

$$\theta_{data_{MLE}} = \underset{\theta}{\operatorname{argmin}}\, \mathbb{H}(P_{data}, P_{model}). \tag{2.16}$$

We will frequently use this loss term in this thesis.

Conditional maximum likelihood For supervised learning, we need to extend the MLE to estimate the conditional probability $P(y|x;\theta)$ to predict the label y for x. The *conditional maximum likelihood estimator* is:

$$\theta_{data_{MLE}} == \underset{\theta}{\operatorname{argmax}} \sum_{i=1}^{n} \log P(y|x_i;\theta). \tag{2.17}$$

The conditional maximum likelihood estimator provides the single best prediction y' for the true label y.

2.2.2 Linear regression

Linear regression is a point estimator that models conditional probability distribution $p(y|x)$, i.e., it takes a vector $x \in \mathbb{R}^{\mathbb{X}}$ and predicts the value of a scalar $y \in \mathbb{R}$ using the vector $\theta \in \mathbb{R}^{\mathbb{X}}$ of *weights* and a *bias* $b \in \mathbb{R}$:

$$\hat{y}(x;\theta) = \theta^{\top} x + b. \tag{2.18}$$

Linear Regression is used for solving *regression* tasks.

2.2.3 Logistic regression

Logistic regression is another form of linear regression, which is used to solve classification tasks. For binary classification with class 0 and 1, the output of linear regression model is transformed to probability value in the interval (0, 1) using a *sigmoid* or *logistic function* σ as, $\sigma(x) = 1/1 + \exp^{-x}$. The classification probability is then computed as:

$$\hat{p}(y = 1|x;\theta) = \hat{y} = \sigma(\theta^{\top} x). \tag{2.19}$$

Since, the output random variable follows a Bernoulli distribution, computing the probability of one class is sufficient to determine the probability of other class.

In the case of multi-class classification, a separate set of weights $\theta_i \in \theta$ is learned for the label y_i of the i^{th} class. Then, *softmax* function is often used to obtain a categorical distribution from the outputs:

$$\hat{p}(y_i|x;\theta) = \frac{\exp^{\theta_i^\top x}}{\sum_{j=1}^{C} \exp^{\theta_j^\top x}}. \tag{2.20}$$

Then, cross-entropy loss between empirical conditional probability $p(y|x)$ and model proability $\hat{p}(y|x;\theta)$ is computed for each sample as:

$$\mathbb{H}(p,\hat{p};x) = -\sum_{i=1}^{C} p(y_i|x) \log \hat{p}(y|x;\theta). \tag{2.21}$$

This simplifies to binary cross-entropy as:

$$\mathbb{H}(p,\hat{p};x) = -(1-y)\log(1-\hat{y}) - y \log \hat{y}. \tag{2.22}$$

For optimising the cost function, the average cross-entropy over all samples are minimised:

$$\mathcal{L}(\theta) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{H}(p,\hat{p};x). \tag{2.23}$$

Note that we denote the cost function with E and $\mathcal{L}$ alternatively throughout this thesis.

2.2.4 Gradient descent

For machine learning models with logistic regression, we do not have any closed-form solution rather we follow an algorithm known as *gradient descent* to iteratively minimise the cost function of the model. It updates the models parameters in the opposite direction of the gradients, i.e., the vector containing the partial derivatives to reach the minima.

2.2.5 Neural networks

In recent years, neural networks have become a popular tool to solve computer vision tasks. Neural networks have the same principle as the basic machine learning models, linear and logistic regression. Logistic regression is a combination of an affine function (Eq. 2.18) and an activation function (Eq. 2.20). Neural networks are models with multiple such affine functions interleaved with non-linear activation functions (Eq. 2.19 or 2.20). The *input layer* is an affine function layer and the *output layer* is an activation function layer to obtain classification output distribution. Layers in between are known as *hidden layers*. A model with one or more hidden layers are referred to as *multi-layer perceptron* (MLP). For instance a neural network with one hidden layer can be formulated as:

$$h = \sigma_1(W_1 x + b_1), y = \text{softmax}(W_2 h + b_2). \tag{2.24}$$

Here, h is the output of the first hidden layer and σ_1 is the activation of the first hidden layer. Each layers has its own weight matrix W and bias vector b. Neurons in each layer have no connections, but every layer is connected to the next and previous layers. Computing the output by feeding an input to h and proceeding h to subsequent layers and eventually producing y is known as *forward propagation*.

Activation functions Other than sigmoid and softmax functions, *rectified linear unit* (ReLU) is a common non-linear activation function used for the hidden layers, which is defined as:

$$\sigma(x) = \max(0, x). \tag{2.25}$$

Another activation function is also used for attention models known as *hyperbolic tangent* or *tanh* function, as:

$$\sigma(x) = \frac{\exp^{x} - \exp^{-x}}{\exp^{x} + \exp^{-x}}. \tag{2.26}$$

Back-propagation The gradient descent in neural networks to minimise the cost or error is computed using an algorithm known as *back-propagation*. Back-propagation depends on the chain rule of calculus to propagate the error derivative backwards starting from the output layer. All the layer parameters are updated following the computed gradients of the layers.

2.2.6 Convolutional neural network

Convolutional neural network (CNN) is a type of neural network specially designed to handle image data efficiently. As this thesis is concerned with image classification, throughout the thesis, among different types of neural networks, we will repeatedly use CNN. CNNs are modified version of neural networks. In theory, they have similar principle as the neural networks, but they have architectural differences. We briefly discuss the basic architecture of the CNN.

For handling image data, unlike neural networks, CNNs have neurons arranged in three dimensions: *width*, *height*, and *depth*. CNN are built with four main types of layers: *convolutional layer* (conv), *pooling layer*, *fully-connected layer*, and *classification layer*.

Convolution layer A convolutional layer consists of a set of learnable filters. Every filter in a layer has same dimensions, they are spatially (width and height) small and extends to the full depth of the layer's input volume. During the forward propagation, each filter convolves across the input volume and performs convolution operation. The convolution outputs (*feature maps*) of every filters in a conv. layer are fed into an activation layer to achieve non-linearity. The activation outputs are stacked along the depth dimension and referred to as output of the layer or *activation maps*.

Pooling layer The activation maps are passed through pooling layer. The function of this layer is to reduce the spatial size of the activation maps to reduce the number of parameters and computation in the network. The pooling layer operates on every depth slice of activation map separately to reduce it spatially, using the $\max$ operation.

Fully-connected layer The fully-connected layers are similar to the regular neural network layers. A fully-connected layer has a full connectivity to all the neurons in the previous layer. Due to downsampling of the activation maps in pooling layer, such full connectivity does not overburden the architecture with massive computations.

Classification layer The classification layer is a fully-connected layer with the ability to convert the output of the network to categorical distribution similar to the neural networks. In particular, unlike other fully-connected layers, this layer has a softmax function to convert the fully-connected output of this layer to categorical distribution.

A CNN may have multiple numbers of conv. layers with or without subsequent pooling layers and multiple numbers of fully-connected layers in the architecture depending on design choices. We will discuss different architectures in the following section. The CNNs are also denoted as *deep neural networks* or *deep learning models*. We will use these term as well in this thesis to refer to CNN.

2.3 Deep learning models

In this section, we provide a brief review on several benchmark deep learning models which falls within the scope of this thesis. Deep learning models discussed in this section are designed to classify the ImageNet [12] dataset, which has 1000 categories of millions of natural object images.

2.3.1 AlexNet

The journey of CNNs in the field of computer vision was pioneered by AlexNet [1]. AlexNet has eight layers in total; the first five are conv. layers and the last three are fully-connected layers as shown in Figure 2.1. This network uses ReLU activation function for introducing non-linearity in the network. Among the five conv. layers, the first two and the final one each has a pooling layer associated with them. The final fully-connected layer is the classification layer, which has a 1000-way softmax heads to produce categorical distribution. The five conv. layers have $11 \times 11 \times 3$, $5 \times 5 \times 48$, $3 \times 3 \times 256$, $3 \times 3 \times 192$, and $3 \times 3 \times 192$ filters with 96, 256, 384, 384, and 256 kernels

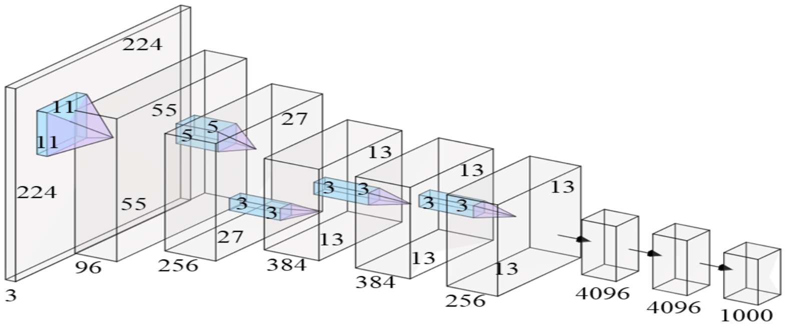


Figure 2.1: Block diagram of AlexNet.

respectively in the architecture. The first two fully-connected layers have 4096 neurons each and the final fully-connected layer has 1000 neurons.

AlexNet uses zero-mean Gaussian distribution with standard deviation 0.01 for weight initialisation and trains the network with stochastic gradient descent (SGD) optimiser with a momentum of 0.9. The network reduces error when brightness normalisation in the form of local response normalisation is applied. This network outperforms all the contemporary non-CNN image classifiers for the large-scale ImageNet dataset.

2.3.2 VGGNet

Compared to AlexNet, VGGNet [2] is an advancement in the field of CNNs. VGGNet was proposed to investigate the effect of convolutional network depth. More specifically, the impact of an architecture with very small (3×3) convolution filters on its accuracy for performing large-scale image classification. VGGNet has several networks configurations with 11, 13, 15, and 19 layers. Every configuration has three fully-connected layers and the remaining are conv. layers.

The main advancement in VGGNet compared to AlexNet is its stacked convolution layers with 3×3 filters. A stack of two 3×3 conv. layers (without spatial pooling in between) has an effective receptive field of 5×5; three such layers have an effective receptive field of 7×7.

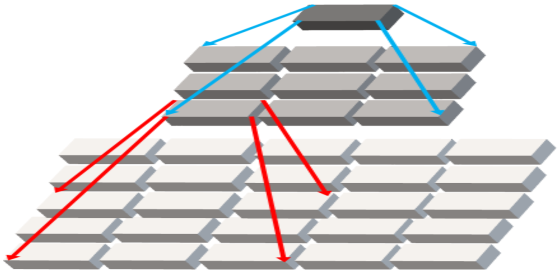


Figure 2.2: Receptive field for stack of conv. filters.

In Figure 2.2, let us assume the top layer is a block from a 3×3 filter, the second layer is another 3×3 filter. Together they have an effective receptive field of 5×5 shown in the third layer. The advantages of placing a stack of two 3×3 conv. layers instead of a single 5×5 layer are: i) instead of a single non-linear rectification layer it has two, which makes the decision function more discriminative and ii) the number of parameters are decreased: assuming that a two-layer 3×3 convolution stack has $\mathcal{C}$ input and output channels, the stack is parameterised by $2(3^2\mathcal{C}^2) = 18\mathcal{C}^2$ weights, whereas, a single 5×5 conv. layer would have $5^2\mathcal{C}^2 = 25\mathcal{C}^2$ parameters. Similarly, placing a stack of three 3×3 filters instead of a single of 7×7 filter would have $3(3^2\mathcal{C}^2) = 27\mathcal{C}^2$ weights instead of $7^2\mathcal{C}^2 = 49\mathcal{C}^2$ parameters, i.e. 81% less.

Despite the larger number of parameters and the greater depth compared to Alexnet, the VGGNets require less epochs to converge due to implicit regularisation imposed by greater depth and smaller conv. filter sizes.

2.3.3 ResNet

Deeper CNNs are exposed to an issue of degradation during their convergence. To be specific, the accuracy saturates with the increasing depth of the network and degrades rapidly at some point. This problem shows that increasing the depth of networks make them hard to optimise. Residual learning deep network (ResNet) [4] addresses this problem by having a shallow and a deeper counterpart in the architecture. The deeper part is composed by adding several shallow networks through identity mapping. This solution leads to a situation where the deeper part produces similar training error to the shallower counterpart. The residual mapping is implemented as shown in Figure 2.3. The residual/identity mapping is performed by a skip/shortcut connection, which performs parameter free identity mapping to address the degradation issue.

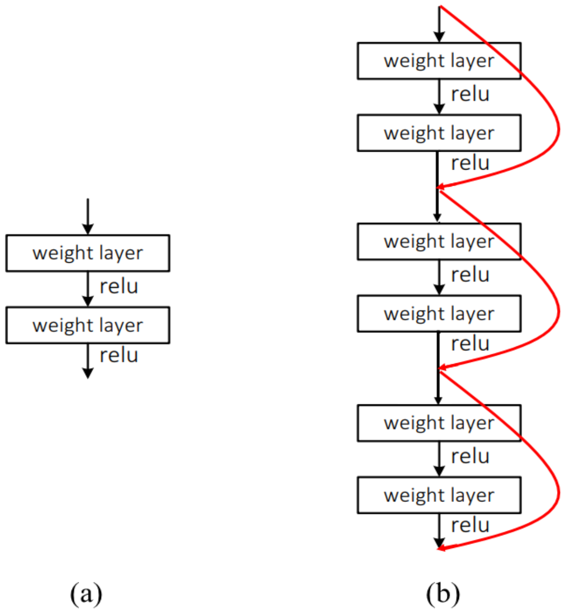


Figure 2.3: (a) Shallow network (b) Residual network [4].

ResNets have several configurations with different number of layers starting from 18 to 152. The conv. layers have 3×3 filters and follow two design rules: the layers with same output feature map size have same number of filters and the number of filters is doubled if the feature map size if halved to preserve similar time complexity per layer. ResNets are more powerful than above-mentioned CNNs (AlexNet and VGGNet) and outperforms them substantially in large-scale image classification.

2.4 Transfer Learning

In this section, we provide an overview of transfer learning in general in computer vision. The section presents a discussion on how machine learning model transfers knowledge across domains and solve tasks for data outside the training data distribution. We first define transfer learning in the context of this thesis. Based on the definition, we will present a taxonomy and finally, review three popular transfer learning settings related to this thesis.

In the supervised learning paradigm, a deep learning model requires a massive number of labelled training data of domain A to perform the task. If we intend to learn another domain B with the model, we will require huge amount of labelled data from B to train the model again. The supervised learning setting falls apart when we do not have sufficient labelled data in a domain to perform the desired task. Transfer learning allows us to use the data from related domains, i.e., *source domains*, to gather knowledge and apply it to solve a task in another domain with scarce labelled data, i.e., *target domain*.

There are different transfer learning settings comprising multiple source and target domains. This thesis focuses on the type with one source and one target domain in the setting. The transferable knowledge can be of different forms depending on the setting. This thesis concerns with the type of knowledge relating to the feature representations learned by the deep neural networks.

2.4.1 Definition

In this section, we partially follow Pan et al [7] to define transfer learning setting formally. The concept of transfer learning involves a domain and a task. A domain $\mathbb{D}$ comprises an input data space $\mathcal{X}$ with a marginal distribution $P(X)$, where $X = x_1 \dots x_n \in \mathcal{X}$. Note that $\mathcal{X}$ denotes the space of all the inputs in $\mathbb{D}$, the inputs may be images or image features depending on the task scenario and x_i represents the i^{th} instance. A task $\mathcal{T}$ has a label space $\mathcal{Y}$, a prior $P(Y)$, and a conditional distribution $P(Y|X)$. The conditional distribution is learned from the training data comprising $x_i \in \mathcal{X}$ and $y_i \in \mathcal{Y}$. $\mathcal{Y}$ is the set of all image labels and y_i is the label of i^{th} instance.

The transfer learning setting has a source domain $\mathbb{D}_s$, a source task $\mathcal{T}_s$, a target domain $\mathbb{D}_t$, and a target task $\mathcal{T}_t$ in general. The objective is to learn the target task

$P_t(Y^t|X^t)$ in $\mathbb{D}_t$ using the information learned from $\mathbb{D}_s$ and $\mathcal{T}_s$, where $\mathbb{D}_s \neq \mathbb{D}_t$ or $\mathcal{T}_s \neq \mathcal{T}_t$. In this thesis, we concentrate on the transfer learning settings with the condition $\mathbb{D}_s \neq \mathbb{D}_t$ for image classification. Under the general setting, this thesis works on three different transfer learning scenarios discussed in the next section.

2.4.2 Scenarios and Taxonomy

Given the general transfer learning setting discussed above, this thesis focuses on the scenarios below based on varying source and target conditions (Figure 2.4):

1. The target domain and task have limited labelled data. The label space, prior and the conditional distributions of the source and the target tasks may be in near proximity but are not equal. This scenario falls under the category of *inductive setting* and can be solved by *sequential transfer learning* approach.

2. The target domain and task have no labelled data. The prior, marginal, and conditional distributions for the source and the target tasks are different. The label space of the source task is a subspace of the target task $\mathcal{Y}^s \subseteq \mathcal{Y}^t$. This falls under the category of *open set domain adaptation*.

3. The target domain and task have no visual data. The prior, marginal, and conditional distributions for the source and the target tasks are different. The label space of the source and target tasks are disjoint $\mathcal{Y}^s \neq \mathcal{Y}^t$ (*zero-shot learning*) or the label space of the target task is a union of the source and target task $\mathcal{Y}^s \cup \mathcal{Y}^t$ (*generalised zero-shot learning*).

For brevity, in the rest of the thesis, we drop the concept of task $\mathcal{T}$ and follow the definition of a domain as, $\mathbb{D} = (x_i, y_i)_{i=1}^n$, where x denotes an input instance and y denotes the label.

We follow the transfer learning taxonomy of Pan et al. [13], to formulate a taxonomy for this thesis based on the discussed scenarios (Figure 2.4). This thesis contributes to improve three different settings of transfer learning that relates to the discussed

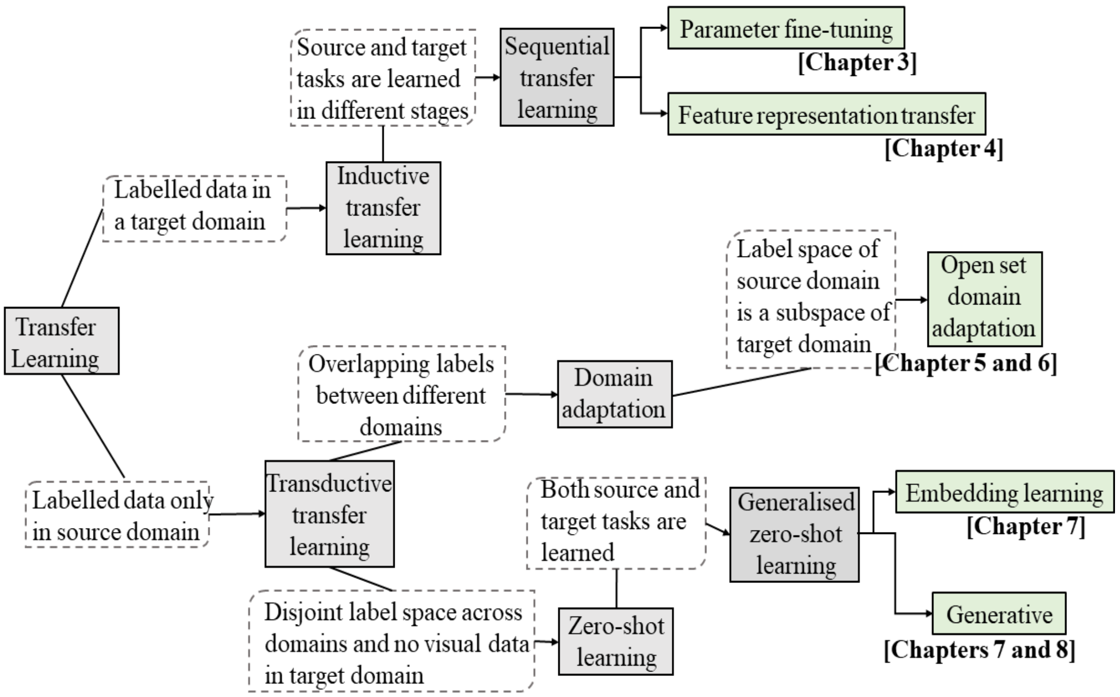

Transfer Learning
Labelled data in a target domain
Inductive transfer learning
Source and target tasks are learned in different stages
Sequential transfer learning
Parameter fine-tuning
[Chapter 3]
Feature representation transfer
[Chapter 4]
Labelled data only in source domain
Transductive transfer learning
Overlapping labels between different domains
Domain adaptation
Label space of source domain is a subspace of target domain
Open set domain adaptation
[Chapter 5 and 6]
Disjoint label space across domains and no visual data in target domain
Zero-shot learning
Both source and target tasks are learned
Generalised zero-shot learning
Embedding learning
[Chapter 7]
Generative
[Chapters 7 and 8]

Figure 2.4: Transfer learning taxonomy for this thesis. Green coloured blocks show the contribution areas of this thesis.

scenarios. The scenarios relate to *sequential transfer learning*, *open set domain adaptation*, and *zero-shot learning*, respectively. In the following sections, we will review the existing approaches for these three settings.

2.5 Sequential transfer learning

[1] In this section, we introduce sequential transfer learning and provide a brief description of the contemporary sequential approaches that relate to this thesis.

In practice, inductive sequential transfer learning for deep learning assumes a source domain with sufficient labelled data and a target domain with limited labelled data. The sequence is to train a deep learning model on source domain first and then transfer the

[1] This section is partially adapted from: **T. Shermin**, S. W. Teng, M. Murshed, G. Lu, F. Sohel, and M. Paul, "Enhanced Transfer Learning with ImageNet Trained Classification Layer," in Proceedings of the Pacific-Rim Symposium on Image and Video Technology (PSIVT), 2019.

knowledge to train on target domain. This thesis is concerned on two types of sequential transfer learning: *parameter fine-tuning* and *feature representation transfer* as shown in Figure 2.4. We will extend our discussion to developmental transfer learning as well, which is an extension of parameter fine-tuning approach.

2.5.1 Parameter fine-tuning approach

Parameter fine-tuning is defined as, first, a deep neural network is trained on a large-scale source domain, which has A categories, and then replace the classification layer of the network and train it again on the target domain to classify B categories. First, we review fine-tuning approaches for different network architectures and then, discuss works that investigate learning rates.

2.5.1.1 Network architecture

A significant number of works have studied parameter fine-tuning in CNNs, which includes various factors affecting fine-tuning, pre-training and freesing layers. Most of the works use ImageNet as the source domain as the dataset has huge amount of diverse data and classes to train the parameters in a CNN. Apparently, it has become a trend to treat CNNs [1–4, 14] trained on ImageNet as learners of features that can be reused in handling almost all categories of visualisation tasks. ImageNet pre-trained CNN features yielded impressive results in image classification [6, 15], action recognition [16], object detection [17, 18], image captioning [19, 20], human pose estimation [21], image segmentation [22], optical flow [23], and others [24].

Yosinski et al. [5] trains a network on partial ImageNet and then replaces every layer from backwards sequentially with a new classification layer. And, finally, trained on target domain to find out which layer generalises better to the target domain and performs optimally. The authors report that the networks holds general to specific features in low to high layers and consequently, the final layer has less generalisation capability. In this thesis, we argue that this claim does not hold when a network is trained on large-scale diverse dataset, i.e., full ImageNet.

Oquab et al. [25] designs an approach to reuse ImageNet trained network layers to compute mid-level features for other datasets. In particular, supporting the claim of [5], the authors sets the trend to replace the classification layer of a CNN trained on ImageNet-1000 with a new ones and fine-tunes the CNN on the target dataset. They report that the fine-tuning of the pre-trained parameters on the new datasets led to significant performance improvement. Yang et al. [26] extends previous parameter fine-tuning another step further by connecting the early layers to the classification layers. As reported, this structure assists the tuning and classification decision even better.

Besides natural object category space, parameter fine-tuning using pre-trained CNNs are applied in other domains. To extend the implementation of parameter fine-tuning in the field of medical research, Tajbakhsh et al. [27] exploit CNNs pre-trained on ImageNet to transfer knowledge to medical domain through fine-tuning. The research reports that parameter fine-tuning in a layer-wise manner outperforms training from scratch in medical image domain. Li et al. [28] improve the parameter fine-tuning approach by introducing an attention mechanism between the source and the target network. They connect the outer layers of the ImageNet trained source network with the target network through an attention block and then fine-tunes both network in a supervised manner.

Ge et al. [29] introduce a selective joint fine-tuning scheme, which identifies and uses a subset of training images from the source domain whose low-level characteristics are similar to those of the target domain, and jointly fine-tune shared convolutional layers for source and target tasks. Moreover, to integrate active learning to fine-tuning, Zhou et al. [30] introduce AIFT (active, incremental fine-tuning). The approach starts with a pre-trained CNN to find from an unlabelled pool to label and fine-tunes the CNN continuously on labelled data to improve performance. The continuous fine-tuning may inject noisy labelled data, to address that entropy and diversity based on current CNN is computed to select only a portion from the labelled data. These approaches perform well in biomedical imaging applications.

Huang et al. [31] propose a fine-tuning approach for SAR data, which has a classification and reconstruction pathway. The reconstruction pathway is first pre-trained on unlabelled SAR images and then the classification pathway is trained on limited labelled SAR images using the feedback. The network may not perform well as the reconstruction is done in an unsupervised manner. George et al. [32] exploit parameter fine-tuning for glitch classification using the gravity Spy dataset that contains hand-labelled, multi-duration spectrograms obtained from real LIGO data. Holder et al. [33] retrain a pre-trained CNN on urban-road scenes to classify off-road scenes. Transfer learning approach to fine-tune for new tasks without forgetting the old ones is proposed by [34]. To limit the need for annotated data for supervised pre-training required for transfer learning, [35] has proposed an approach of more universal representations.

All the above-mentioned works have neglected the classification layer of the pre-trained CNN while performing fine-tuning for target tasks. However, we argue that the presence of the classification layer will enhance fine-tuning when the source domain is diverse enough. Thus, in this thesis we consider publicly available ImageNet dataset as the source to prove our claim.

Developmental transfer learning Developmental transfer learning extends the traditional parameter fine-tuning by increasing the network capacity for performing the target task. Several works have exploited developmental and lifelong learning [36–38], which is in line with developmental transfer learning. A recent transfer learning work [39] increases the width and depth (append new layers) of conv. layers of a pre-trained network to investigate the optimal setup for transferability and report that increasing width of a layer performs better than increasing depth. To further investigate the developmental fine-tuning, Wang et al. [40] grow the pre-trained network beyond the pre-trained fully-connected layers and increases the width of the fully-connected layers. The research reports that increasing the model capacity allows the model to adapt better through fine-tuning and outperforms fine-tuning approaches. However, both the works [40, 41] neglect the pre-trained classification layer of the ImageNet trained source network. In contrary, we argure to grow network depth beyond the pre-trained classification layer and investigate its impact in developmental transfer learning (Chapter 3).

2.5.1.2 Learning rate

So far, we have discussed about different network architectures for fine-tuning. Now, our focus is on reviewing how existing works have investigated learning rate for the fine-tuning layers. Girshik et al. [17] report that reducing the learning rate compared to the pre-training rate during fine-tuning improves the performance. On the other hand, Nicholas et al. [42] report global learning rate for transferred layers is optimal. For growing networks, Wang et al. [40] train their newly augmented layers with ten times more learning rate than the existing layers. We aim to investigate the optimal learning rate for the proposed parameter fine-tuning approach in Chapter 3.

2.5.2 Feature representation transfer

Feature representation transfer approach refers to the approach where a CNN is trained on a domain A and the learned features from different layers are extracted and reused to learn another domain B using a different network.

Donahue et al. [43] introduce feature representation approach and presents one of the first studies on the behaviour of conv. filters. They extract features from the last conv. and the first two fully-connected layers of an ImageNet trained AlexNet [1]. They report that the first fully-connected layer features outperform other layers when trained to perform target task using a support vector machine (SVM). The contribution of Razavian et al. [6] go in a similar direction, they use the ImageNet trained OverFeat [18] to extract features from the first fully-connected layer. Then the extracted features are boosted through data augmentation and component-wise power transformation. The boosted features are fed into SVM to perform target tasks. For image classification, they evaluate other layers features and report that the first fully-connected layer outperforms others.

Another study of suitable factors for this type of transfer learning has been reported by [39, 41]. Among the parameters they considered some are related with the network architecture (network depth and width, optimisation parameters, etc), and others are related with the transfer learning process (fine-tuning, network layer to be extracted, spatial pooling, etc). The authors find that the first and second fully-connected layer features trained using SVM performs best. To exploit more advanced CNN features for transfer learning, Mahmood et al. [44] extract features from the inner layers of ImageNet trained ResNets. They report improved performance in under water coral classification. Similarly, Mormont et al. [45] have used pre-trained CNNs to detect digital pathological objects. They have explored more networks besides SVM, such as randomised tree based algorithms, multi-layer and single-layer perceptron. This works reports using inner layers for feature extractions performs optimally.

Contributions to feature representation transfer have been purely empirical so far. They were done by extracting all CNN features from a single layer and testing their performance by feeding them to a classifier. However, Dario et al. [46] statistically measure the distinctive power of every single feature within a CNN, when used for classifying every class of different datasets. This work provides new insights into the behaviour of CNN conv. features for their application to knowledge representation and reasoning. They report that low and mid-level features may behave differently to high level features, but only under certain conditions.

Gasulla et al. [32] introduce a new feature extraction called full-network embedding for transfer learning, where they extract features from different layers of a CNN to construct an embedding and then utilise that embedding along with a SVM to solve target tasks. This approach is capable of providing extra information to better characterise the data and assist transfer learning. Besides SVM and logistic regression, Bayesian least square SVM [47] and smoothed K-NN classifier [48] for transfer learning are also reported in the literature.

All the above-mentioned works, follow the footsteps of [5, 43] and neglects the classification layer features of the ImageNet trained CNNs. We aim to show the impact of classification layer features in this type of transfer learning in Chapter 4.

2.6 Domain adaptation

[2] [3] Domain adaptation (DA) is a setting of transfer learning where knowledge is transferred across domains with different distributions. Traditionally domain adaptation approaches assume to have similar class label across domains, which is also known as *closed set domain adaptation*. Open set domain adaptation is more relaxed towards realistic setting as it allows to have dissimilar labels with partial overlap across domains. In this section, first we will provide a brief review of the closed set approaches and then present detail review of recent open set domain adaptation approaches.

2.6.1 Closed set domain adaptation

Closed set DA approaches concentrate on reducing the divergence between the source and the target domains. Recent works have shown that because of its setting, closed set DA approaches can exploit domain invariant features by explicitly reducing the domain divergence upon the supervised deep neural network structures. The development of deep learning based closed set DA approaches [8,49–56] is originated from prior shallow DA approaches [57–62].

Closed set DA approaches fall into three main categories. The first category of approaches is based on static moment matching, such as Maximum Mean Discrepancy (MMD) [50, 51, 53, 63], Central Moment Discrepancy (CMD) [64], and second-order statistics matching [65]. The second category of approaches adapts the adversarial loss concept of GAN [66] and initiates the generation of images that are non-discriminative to the shared label space of source and target domain [52,56,67]. Furthermore, domain adversarial approaches align pixels and features from both domains and synthesise labelled target images for data augmentation [68–73]. Saito et al. [8] utilise the probabilistic outputs of two classifiers' to measure domain discrepancy loss and update

[2] This section is partially adapted from: **T. Shermin**, S. W. Teng, M. Murshed, G. Lu, and F. Sohel, "Adversarial network with multiple classifiers for open set domain adaptation," IEEE Transactions on Multimedia, 2020

[3] This section is partially adapted from: **T. Shermin**, G. Lu, S. W. Teng, M. Murshed, and F. Sohel, "Adversarial Open Set Domain Adaptation based on Mutual Information," Revision Submitted to IEEE Transactions on Image Processing, 2021.[preprint: arXiv: 2007.00384v1]

both the classifiers based on this loss adversarially. However, we use two different types of classifiers and a discriminator for measuring underlying domain similarity and utilise this similarity to compute final domain discrepancy loss by updating only the main classifier. Recent research has explored Cycle-Consistent GAN [74] for developing CycleGAN-based [75–77] DA approaches. The final category of approaches leverages Batch Normalisation statistics for adapting domains to a canonical cone [78,79].

2.6.2 Open Set Domain Adaptation

Open set DA models have to correctly recognise images from the shared label space and reject images from the domain-private (unknown) classes [80–82]. The main challenge in open set DA is to separate known and unknown samples. Multi-class open set SVM [83] is designed to reject images from unknown classes. In this approach, the SVMs are trained to assign probabilistic decision scores to samples and reject unknown samples by a threshold. Bendale et al. [80] integrates an OpenMax layer upon deep neural networks for exploiting them in open set recognition. The OpenMax layer assists the network in estimating the probability of an image coming from an unknown class. To generate unknown samples for open set recognition, Ge et al. [82] combines a generative model with the OpenMax layer, and to reject unknown samples during testing, this approach defines a threshold.

Assign-and-Transform-Iteratively (ATI) [81] follows the open set domain adaptation setting which has unknown classes in both the domains. ATI evaluates the distance of each target sample from the core of every source class, aligns target samples residing in near vicinity to known classes and rejects unknown target samples by aligning them towards the unknowns of the source domain. Saito et al. [84] modify the open set DA setting for addressing more practical DA cases. Their modified setting (Open set DA by back-propagation (OSBP DA)) does not require unknown classes in the source domain. The OSBP approach implements a generative adversarial domain adaptation model. Both ATI and OSBP approaches require some threshold hyper-parameters to distinguish between known and unknown classes. Since this threshold hyper-parameters are not learnable or adaptive, they encourage negative transfer by aligning known samples to the ‘unknown’ class.

Recent works follows the OSBP open set DA setting to improve open set DA such as Separate to adapt (STA) [85], Mutual to Separate (MTS) [86], and Towards Inheritable Models (TIM) [87]. STA and MTS have weighting modules, which rely on a domain confidence based similarity score to separate known-unknown target samples. STA utilises underlying domain similarity of target samples for progressive separation of known-unknown target samples before domain adaptation. MTS employs a multi-binary classifier with a domain adversarial network to separate unknown samples and simultaneously adapt shared classes across both domains. MTS proposed mutual learning between the separation network and the adaptation network based on the distance between the features of both networks. STA and MTS are exposed to negative transfers for relying only on a multi-binary classifier for known-unknown separation.

TIM trains a model jointly on the source domain and self-generated out-of-source-distribution samples. This trained source model is then used for adapting known classes and rejects unknown classes by aligning the unknown classes to the generated out-of-source-distribution samples. This approach focuses on easing the overconfidence issue [88] of the source classifier for separating known-unknown target samples. However, TIM may initiate negative transfers by confining the ‘unknown’ domain similar to [81]. We aim to partially avoid the out-of-source-distribution sample prediction issue of the separating module by employing roughly determined known-unknown target samples besides the source samples to train the separating module.

Progressive Graph Learning (PGL) [89] approach depends on a shared classifier to assign pseudo labels to the target samples for known-unknown separation. PGL is prone to negative transfer due to its dependence only on the decision of a classifier. Joint Partial Optimal Transport (JPOT) optimises a transport loss for separating known-unknown samples, which depends on a distance-based optimisation. This optimisation is not strong enough to preserve high-level semantic information [10] and may degrade performance. Factorised Representations for Open-Set Domain Adaptation (FRODA) [90] factorises low-dimensional representations of the data into shared and private parts for known-unknown separation. The low-dimensional representation may not capture high-level semantic characteristics of images and eventually lead to misclassification.

This thesis focuses on the OSBP DA setting. We aim to reduce negative transfer in the OSBP approach by integrating an adaptive weighting module instead of using a fixed threshold hyper-parameter. The weighting module will generate instance-level weights by assessing domain confidence. To further mitigate negative transfers and improve open set DA, we aim not to blindly depend on source trained classifiers and exploit both domain confidence and shared information between domains for generating instance-level weights. To explore shared information, we will utilise mutual information and pointwise mutual information score in the known-unknown separation module.

Mutual Information and Domain Adaptation Several information bottleneck principles [91–93] are used for leveraging information as a basis to learn representations using paired data in domain adaptation [94–96]. DA approaches using mutual information I can be grouped into three main categories. The first category of approaches tends to minimise I for DA tasks. Such as, for closed set DA, [95] minimises I between the source and target samples. To perform self-supervised DA, [97] minimises I between the features and domain labels to capture semantically shared information across different domains, and [98] minimises I between the representations and the domain-belonging indicators to measure the dependency. The second category of approaches maximises I for DA tasks. For example, Shi and Sha [96] maximises I between the data and the estimated label for closed set DA. For open set DA, [99] maximises I between the input feature and the two output distributions. Apart from these groups, for closed set multi-target DA, [100] maximises and minimises I between the domain labels and features. The final category of approaches use I and/or *pmi* as a heuristic to accomplish DA tasks in NLP, such as, [101] utilises I and *pmi* as a heuristic for part-of-speech tagging, [102] uses *pmi* for sentiment classification, *pmi* is also utilised by [103, 104] for pivot selection for sentiment classification and other NLP DA tasks.

Other types of open set DA settings include *partial DA* and *universal DA*.

Partial Domain Adaptation The setting of partial DA assumes a target domain that has a fewer number of classes than the source domain. Cao et al. [105] use multiple domain discriminators along with class-level and instance-level weighting mechanism to obtain class-wise adversarial distribution matching for solving partial DA. Cao et al. [106] refine Selective Adversarial Network (SAN) [105] by using only one adversarial network and integrating the class-level weight to the source classifier.

Universal Domain Adaptation Recent research introduces a universal DA setting that imposes no previous knowledge on the source, and target domain label sets [107]. This is another valuable step towards addressing a practical adaptation scenario. The authors proposed a Universal Adaptation Network (UAN), which combines domain similarity and prediction uncertainty while generating sample weights for finding shared label sets.

2.7 Zero-shot learning

[4] [5] Zero-shot learning (ZSL) setting assumes no available visual data in the target domain and the label space of the target domain is disjoint with the source domain. However, it allows to utilise available *semantic attributes* to learn visual classifier for the target domain. Generalised zero-shot learning (GZSL) extends the setting and requires to learn visual classifier for both source and target domain. ZSL approaches have two main settings: inductive (no images of target classes are used during training) [108–110] and transductive (unlabelled images of target classes are used during training) [111–113]. In this thesis, we follow the inductive GZSL setting i.e, we do not utilise images from target classes during training. Inductive zero-shot learning approaches can broadly be divided into two categories: *embedding learning* and *generative* approaches (Figure 2.4). In this section, we provide a brief overview of existing embedding learning and generative approaches.

[4] This section is partially adapted from: **T. Shermin**, S. W. Teng, F. Sohel, M. Murshed, and G. Lu, "Integrated Generalised Zero-Shot Learning for Fine-Grained Classification," Pattern Recognition, 2021.

[5] This section is partially adapted from: **T. Shermin**, S. W. Teng, F. Sohel, M. Murshed, and G. Lu, "Bidirectional Mapping Coupled GAN for Generalised Zero-Shot Learning," Revision submitted to IEEE Transactions on Image Processing, 2021. [preprint: arXiv: 2012.15054]

2.7.1 Embedding learning approaches

The embedding learning approaches learn to project either visual features to the semantic space [108, 110, 114, 115] or semantics to the visual space [116–119] based on seen/source classes for the ZSL task. A line of embedding learning approaches integrate the concept of attention to better relate semantics with visual features and perform improved classification [9, 120–126].

Visual to semantic embedding These approaches learn to transform the visual space to semantic space and perform classification in the semantic space. To transfer information from source to target domain, early approaches learn semantic attribute classifiers independently [108, 110]. To improve these works, a few approaches consider all attributes at a time and learn label embedding functions to increase the compatibilities between visual and corresponding class semantics [114, 127]. Towards learning more robust transformations, [128] introduces network that combines attribute classifier learning and semantic label embedding.

Semantic to visual embedding These approaches learn to transform the semantic space to the visual feature space and perform classification in the visual space. Consequently, they can handle hubness problem in ZSL [116, 117]. Changpinyo et al. [129] rely on the idea that the semantic representations must have information of their corresponding visual exemplars locations and convert the semantic space to visual features. To compensate for the structural difference between the semantic and visual space, Long et al. [130] propose a latent embedding space and hold the local structure simultaneously. To learn the embedding during conversion, [131] utilises knowledge graphs and [119] uses regularisers. Zhang et al. [118] introduce fusion between multiple semantic modalities to assist the conversion of visual embedding from semantic embedding.

Mapping high dimensional visual features to low dimensional semantic space may reduce the variance of features and create hubness problem [116–118]. Similarly, semantic to visual mapping is not optimal as one class may have several corresponding visual features [132].

Attention based approaches Majority of the existing embedding learning approaches use global visual features for ZSL and/or GZSL task [114,118,119,133–139]. On contrary, to explore local fine-grained details, a few works have applied attention mechanisms. Most of these approaches aim to improve fine-grained ZSL or GZSL tasks by discovering relation between local visual details and semantics. However, some of them do not explore proper guidance from attributes [9,120–122] and others ignore local visual details [123–125]. In this thesis, one of our sub-objective is to improve fine-grained GZSL or ZSL.

A recent attention-based work [126] for fine-grained GZSL limits the feature exploration space to the number of attributes to construct attribute embedding and requires expensive attribute selection. As the ultimate goal is to learn a visual classifier, unlike [126], we aim to construct a visual feature embedding, which retains necessary global visual features and the feature regions linked to the attributes are assigned more attention than other regions in Chapter 7.

A non-fine-grained attention-based GZSL approach, APN [140], integrates the exploration of both global and local details for GZSL tasks. The global module in APN is separated from the local module and the global module extracts channel-wise global information. This may create incompatibility in the network. The local module in APN aims to construct attributes from the local visual regions for GZSL. In this thesis (Chapter 7), we aim to preserve local region-wise global information. Consequently, for building the feature embedding, we will be able to maintain better synchronisation of global features with the attribute-weighted local region features, which we will discuss in Chapter 7. In contrary to recent attention-based approaches [126,140], we argue to place two-level of dense attention mechanism to capture and highlight finer details for fine-grained tasks.

2.7.2 Generative approach

Feature synthesising or generative approaches adversarially learn to synthesise visual features from class semantics and reduce the ZSL to a standard supervised classification task [132, 141–146]. Generative approaches can be grouped into two more groups: *unidirectional mapping* and *bidirectional mapping* approaches.

2.7.2.1 Unidirectional Mapping approaches

These approaches adversarially learns to synthesise visual features from class semantics and reduce the GZSL to a standard supervised classification task [10,132,141–143,143–145,147–149].

For generation of target class features, f-clsWGAN [141], CVAE [145, 150], SE-GZSL [143] used conditional Generative Adversarial Networks (GANs) or Variational Autoencoders (VAE). The feature synthesising approaches learn to generate global visual features conditioned on the attribute descriptions and ignore local distinctive details [141,142,144,145,151,152]. One of our sub-objective is to explore local information related to the attributes for synthesising features for improved fine-grained zero-shot recognition (discussed in Chapter 7).

The above-mentioned approaches rely on only one-directional semantic to visual features mapping, which does not guarantee strong visual-semantic interactions. Therefore, bidirectional mapping approaches are studied.

2.7.2.2 Bidirectional Mapping approaches

Bidirectional mapping approaches perform semantic-to-visual and visual-to-semantic mapping to ensure strong interaction between the visual and semantic spaces. This bond is vital for GZSL tasks.

The first effort in this field is DASCN [10], which simply employs two generative adversarial networks to construct visual features from semantics and reconstruct back semantics from the visual space for dual learning. GDAN [132] utilises a regressor instead of another GAN to map generated features back to the semantic space. For bidirectional optimisation, GDAN minimises the distance between the real and reconstructed semantics. This optimisation is not strong enough to preserve high-level semantic information [10]. Unlike GDAN, in the proposed BMCoGAN (Chapter 8), in addition to distance-based optimisation, we show that adversarial supervision to reconstruct the semantics using a coupled discriminators facilitate the network.

GZSL-AVSI [153] proposes to implement dual learning between an inference module and a generative module. GZSL-AVSI uses a Wasserstein semantic alignment loss to optimise the semantic space. On the other hand, our proposed BMCoGAN optimises Wasserstein distance-based adversarial loss for adversarially supervising the source domain's visual feature space.

In addition to dual learning, RBGN [154] incorporates adversarial attack strategies to train a more attentive discriminator. VAEs rely only on the lower bound of the log-likelihood of observed data. To address this issue, cFLOW-ZSL [155] utilises generative flow to estimate accurate likelihood during the bidirectional mapping using VAEs.

2.7.3 Reducing Bias Towards Source Domain

Since the target domain has no visual data, training only on the source data may induce bias towards the source domain samples. To overcome bias towards source domain, ZSL approaches have explored novelty detection and prediction calibration [126,139]. For transfer learning target classes, [113] relies on the reconstruction of source class semantic vectors from target classes. In this thesis (Chapter 7), for loosely smoothing out target class probabilities, we aim to measure class similarity by exploring shared information between the class semantic vectors. This is more reliable as the class semantic vectors only hold the confidence of attributes in a class.

IBZSL [156] uses the information bottleneck constraint and data uncertainty estimation technique to design a bias passing mechanism to alleviate the noises and gap between visual features and human-annotated semantics and hope to reduce source domain bias. ISE-GAN [157] proposes an integrated-classifier to be trained with the bidirectional learning scheme. The integrated-classifier is trained to reduce bias towards source classes, and without the integrated-classifier, the SE-GAN [157] is inclined towards the source classes. On the contrary, we will infuse the knowledge of smoothing out bias towards source classes in the generator and the discriminator by designing a loss optimisation function in Chapter 8. And, unlike SE-GAN [157], our generator can preserve source-target domain discrimination in the visual features.

2.8 Summary

In this chapter, we have reviewed the background of probability and information theory, machine learning, deep learning, and transfer learning settings and approaches relevant to this thesis. In the following six chapters, we will discuss our proposed models in addressing limitations of existing works and their performance evaluation in detail. In the next chapter, we will discuss our first contribution, i.e, to improve parameter fine-tuning using the category-specific CNN features.

Chapter 3

Improving Parameter Fine-tuning Using the Classification Layer of CNN

[1]In the previous chapter, we provided an overview of the related works to this thesis. In this chapter, we present our first contribution. As discussed in Section 2.5.1, it is widely believed that the classification layer of a CNN trained on a diverse dataset, which holds category-specific features may not contribute to transfer learning. However, in this chapter, we present our systematic study to show that the pre-trained classification layer features has transferablity and can assist in improving parameter fine-tuning.

CNNs [1–4] need a huge amount of labelled training data to perform optimally. Luckily, training CNNs on a large and diverse dataset (e.g., ImageNet) has been shown to enable the knowledge transfer across a wide range of tasks [6]. Parameter fine tuning is a transfer learning approach whereby learned parameters from pre-trained source network are transferred to the target network followed by fine-tuning. Prior research has shown that this approach is capable of improving task performance. However, the impact of the ImageNet pre-trained classification layer in parameter fine-tuning is mostly unexplored in the literature. In this chapter, we present our proposed fine-tuning approach with the pre-trained classification layer. We employ layer-wise fine-tuning to determine which layers should be frozen for optimal performance. Our empirical

[1] Chapter 3 is adapted from: **T. Shermin**, S. W. Teng, M. Murshed, G. Lu, F. Sohel, and M. Paul, "Enhanced Transfer Learning with ImageNet Trained Classification Layer," in Proceedings of the Pacific-Rim Symposium on Image and Video Technology (PSIVT), 2019.

analysis demonstrates that proposed fine-tuning with the pre-trained classification layer performs better than traditional fine-tuning. This finding indicates that the pre-trained classification layer holds more global information than believed earlier. Thus, we hypothesise that the presence of this layer is crucial for growing network depth to adapt better to a new task. Our study manifests that careful normalisation and scaling are essential for creating harmony between the pre-trained and new layers for target domain adaptation. We evaluate the proposed depth augmented networks for fine-tuning on several challenging benchmark datasets and show that they can achieve higher classification accuracy than contemporary transfer learning approaches.

The main contributions of this chapter are as follows.

- We propose to include the pre-trained classification layer in fine-tuning and find that the transfer learning performance with the pre-trained classification layer is higher than traditional fine-tuning approach.
- We investigate which layers should be frozen during fine-tuning for optimal performance.
- For developmental transfer learning, we propose to augment new layers beyond the pre-trained classification layer to adapt better to target task. We also investigate the best fit normalisation scheme for our proposed depth augmented networks.

3.1 Problem Setting

The inductive transfer learning setting constitutes a source domain $\mathbb{D}_s = (x_i^s, y_i^s)_{i=1}^{n_s}$ of n_s labelled instances associated with $|C_s|$ classes, which are drawn from distribution p_s and a target domain $\mathbb{D}_t = (x_j^t, y_j^t)_{j=1}^{n_t}$ of n_t labelled instances drawn from distribution p_t. The task is to learn a classifier for $\mathbb{D}_s$ first and then transfer the knowledge to learn $\mathbb{D}_t$. Note that this setting is also applicable for the contribution discussed in Chapter 4.

3.2 Background and Motivations

Parameter fine-tuning is one of the best performing transfer learning approaches used by the deep learning community. Parameter fine-tuning assists transferring learned knowledge to accomplish the target task with limited labelled data and increase the performance of the target model over random initialisation [5]. The sequence of traditional parameter fine-tuning is to replace the pre-trained classification layer of a CNN trained using a large and diverse dataset (e.g., ImageNet) with a randomly initialised new classification layer as per the target task. Then the new model undergoes forward-backward propagation to tune gradient descent on the target set. This transfer learning approach is exploited with success by a number of contemporary transfer learning research [18,25,26,158].

Transfer learning works well when the learned features are generic, which refers to having features suitable to both base and target datasets. Deep neural networks incline to learn generic features in the first layer that resemble Gabor filters and colour blobs irrespective of datasets and training objectives [1, 159, 160]. Several works in various computer vision tasks have reported significant results by transferring inner layer features of deep networks [15, 18, 161]. As the deep network architecture moves toward Fully-connected (FC) layers, the specificity increases while the generic nature of features decreases [5], i.e., the intuition is that they are highly specific to pre-trained classes and might not generalise well in transferring knowledge.

Therefore, the classification layer is not included in parameter fine-tuning. However, our research manifests that even the classification layer of CNNs trained on ImageNet has plenty of overlapping or neighbouring high-level features with the target sets containing images of natural and artificial objects, since ImageNet consists of a massive amount of labelled images of natural and human-made objects. Figure 3.1 shows the visualisation of the relative distribution of extracted features from the classification layer of pre-trained AlexNet for widely used eight transfer learning target datasets (Section 4.1) and ImageNet (source dataset). The high intermingling or neighbouring between source and target feature distribution manifests that the classification layer of

ImageNet pre-trained CNN may well assist the target network to adapt to the target domain via parameter fine-tuning. Also, by jointly adapting pre-trained classification and other representation layers for the target task, we could essentially bridge the domain shift underlying both the marginal distribution and the conditional distribution, which is pivotal for enhancing transfer learning [61]. Thus, we believe the pre-trained classification layer may be important for transfer learning and propose to include this layer in the fine-tuning procedure. In this work, we evaluate the fine-tuning approaches with our layer-wise optimal fine-tuning scheme to observe fine-tuning from which layer produces optimal performance.

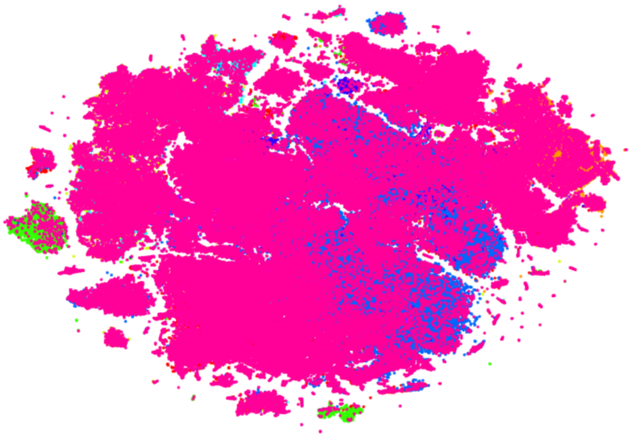


Figure 3.1: The t-SNE visualisation of extracted features from the pre-trained classification layer. Dark pink colour represents ImageNet features, and other colours represent features from eight different target sets.

A significant number of works have exploited incremental and lifelong learning [36–38]. For developmental transfer learning, in consistent with the traditional fine-tuning sequence, Wang et al. [40] and Oquab et al. [25] have discarded the pre-trained classification layer and appended new layers after the penultimate FC (FC) layer of AlexNet. However, inspired by our empirical analysis of the transferability of the pre-trained classification layer, we argue that the presence of the pre-trained classification layer is vital to increasing the network depth for transfer learning. Thus, we propose to consider this layer as the last FC layer and to append new layers beyond it for developmental transfer learning. During fine-tuning, our proposed depth augmented networks might struggle from internal covariate shift of activations across pre-trained and new layers. Thus, to establish harmony between the learning of new and pre-trained layers, and to reduce sensitivity to random initialisation, we introduce a normalisation scheme to the network. We experiment with L_2-norm normalisation [40] and batch normalisation [162] to search for the best performing normalisation scheme for proposed depth augmented networks.

3.3 Proposed Fine-tuning

We introduce the architectural and notational details of the proposed and traditional fine-tuning, the proposed depth augmented networks, and the optimal fine-tuning scheme in this section.

Let us assume χ_N be a CNN pre-trained by using a large dataset (e.g., ImageNet) with N layers including the classification layer, $L_1, ..., L_N$.

3.3.1 Traditional and proposed fine-tuning

Let χ_N^κ denote the sub-network comprising the first κ layers of χ_N, $1 \leq \kappa < N$. Let $[\chi_N^\kappa + \psi]_\nu \equiv [\chi_N^\kappa]_\nu^\psi$ denote a transfer learning network (TLN) from the first κ, $1 \leq \kappa < N$, layers of the pre-trained CNN χ_N, with parameter fine-tuning from layer L_ν onwards, $1 \leq \nu \leq \kappa$, where ψ is a classification module for new classes, which is a FC classification layer C followed by a Softmax layer. Figure. 3.2a illustrates the block diagram of a TLN ($[\chi_N^{N-1}]_\nu^\psi$) which follows traditional fine-tuning sequence where the pre-trained classification layer L_N is discarded before fine-tuning. On the contrary, we include the pre-trained classification layer L_N in the proposed fine-tuning approach, as shown in Figure 3.2b. Our proposed fine-tuning TLN which comprises of all the layers of χ_N, is denoted as $[\chi_N]_\nu^\psi$, with parameter fine-tuning from layer L_ν onwards.

3.3.2 Proposed Depth Augmented Networks for Fine-tuning

We increase the depth capacity of the network by constructing new FC layers consisting of $S \in \{512, 1024, 2048, 4096\}$ neurons on top of the classification layer L_N as shown

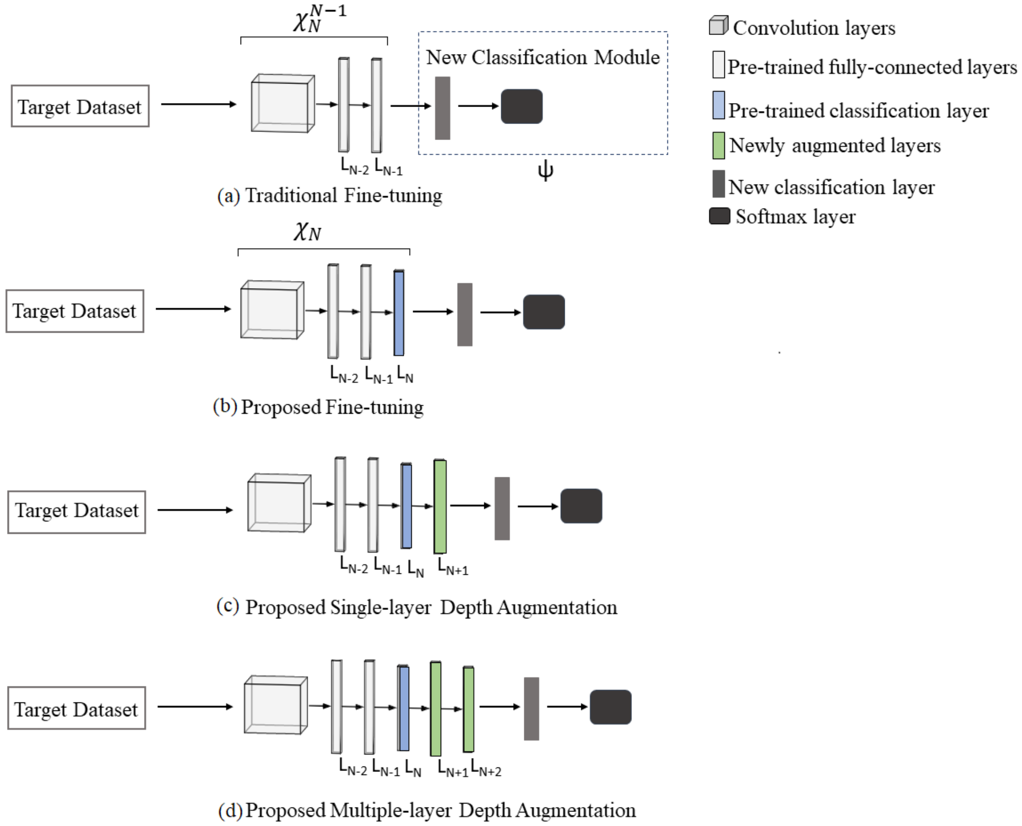

χ_N^{N-1}
Target Dataset
New Classification Module
L_{N-2} L_{N-1}
ψ
(a) Traditional Fine-tuning
χ_N
Target Dataset
L_{N-2} L_{N-1} L_N
(b) Proposed Fine-tuning
Target Dataset
L_{N-2} L_{N-1} L_N L_{N+1}
(c) Proposed Single-layer Depth Augmentation
Target Dataset
L_{N-2} L_{N-1} L_N L_{N+1} L_{N+2}
(d) Proposed Multiple-layer Depth Augmentation
Convolution layers
Pre-trained fully-connected layers
Pre-trained classification layer
Newly augmented layers
New classification layer
Softmax layer

Figure 3.2: Block diagram of the traditional fine-tuning, proposed fine-tuning and depth augmentation.

in Figure 3.2c single-layer depth augmented TLN $[\chi_N]_\nu^{1+\psi}$, and Figure 3.2d two-layer depth augmented TLN $[\chi_N]_\nu^{2+\psi}$. Let $[\chi_N + L_{N+1} + ... + L_{N+\tau} + \psi]_\nu \equiv [\chi_N]_\nu^{\tau+\psi}$ denote a depth augmented TLN from the pre-trained CNN χ_N augmented with τ, $\tau \geq 0$, additional FC layers $L_{N+1}, ..., L_{N+\tau}$, with parameter fine-tuning from layer L_ν onwards, $1 \leq \nu \leq N + \tau$, where ψ is the new classification module, which has a FC classification layer C with a Softmax layer. Appended layers are treated as adaptation layers to compensate for the different image statistics of the source and target sets. Moreover, they allow for suitable compositions of pre-existing parameters and avoid unwanted modifications to the parameters of pre-trained layers for their adaptation to the new task. To maintain learning pace, we propose to include a normalisation scheme in proposed depth augmented networks. We explore both L_2-norm normalisation and batch normalisation. For the first normalisation approach, consistent with [40], we apply L_2-norm normalisation to the input activations of new layers. In case of batch normalisation, we standardise the mean and variance of the input activations of new layers for stabilising the learning process. Finally, we employ the learnable scaling parameter to scale the normalised activations.

3.3.3 Optimal Fine-tuning Scheme

We evaluate all the approaches discussed above using a two-step layer-wise optimal fine-tuning scheme. In the first step, we initialise the transferred layers with pre-trained parameters and new layers randomly. In the second step, we start fine-tuning from the last transferred layer and freeze other layers. These two steps are repeated K times with different setups (K is the number of transferred layers), i.e., each time we unfreeze one more penultimate layer. For instance, in the second setup, we start fine-tuning from the penultimate transferred layer onwards. It is worth mentioning that the fine-tuning setups are mutually exclusive and parameters of all the different setups are initialised according to Step 1. We record transfer learning performance for each of these fine-tuning setups to determine which setup yields optimal performance.

3.4 Performance Study and Analysis

This section describes the datasets used in our experiments, the implementation details, and our evaluation outcomes for proposed approaches. In particular, we investigate the optimal fine-tuning setup for the proposed fine-tuning and depth augmented networks, best fit normalisation scheme, average performance gain of the proposed approaches, visualise learned features, and optimal learning rate for the proposed approaches.

3.4.1 Datasets and Implementation Details

We assembled eight different fine-grained and coarse target datasets, as stated in Table 3.1. Fine-grained datasets used in this work are 102 Flowers [163] with 102 categories, CUB

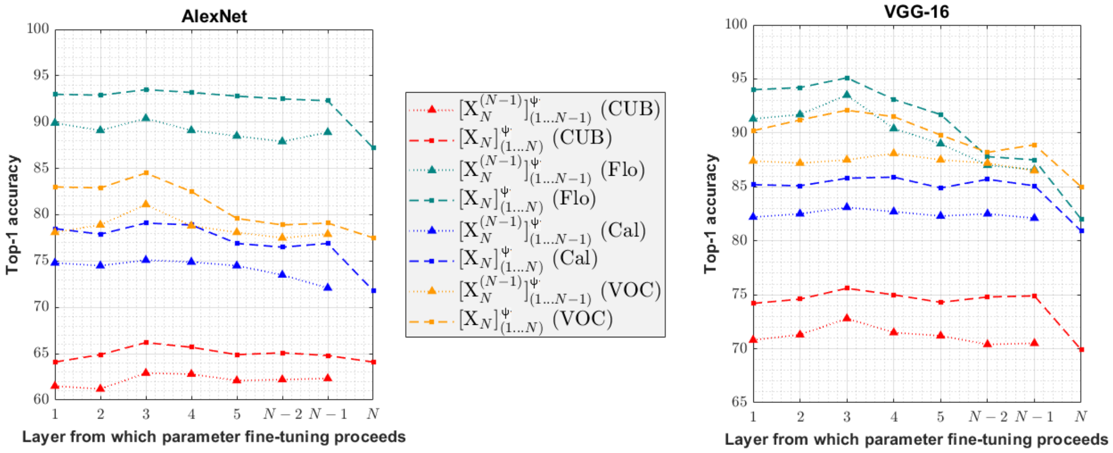
Table 3.1: Selected target datasets.

Type	Source	Images	Categories
Fine grained	102 Flowers	8189	102
	CUB 200-2011	11788	200
	Stanford Dogs	20580	120
	Oxford Pets	7400	37
Coarse	Caltech-256	30607	256
	Pascal VOC-07	9963	20
	MIT-67 scenes	15620	67
	SUN-397 scenes	108754	397

200-2011 [164] with 200 types of birds, Stanford Dogs [165] with 120 classes and Oxford Pets [166] with 37 classes. The Coarse or mixed semantic datasets are Caltech-256 [167] with 256 categories, Pascal VOC-07 [168] having 20 different classes, MIT-67 scenes [169] with 67 classes of indoor scenes and SUN-397 scenes [170] with 397 categories. Note that we have used only ImageNet as the source dataset for its diversity. We have used AlexNet [1] and VGG16 [2] pre-trained on ImageNet as source networks. Networks with two different depths, i.e., AlexNet, and VGG16 are used to observe whether proposed parameter fine-tuning has consistent performance across different architectures. For preprocessing the training dataset, input images are first randomly cropped, horizontally flipped, and then normalised. A split of 75% of target dataset is used for training, and the remaining 25% for testing. We execute 2000 iterations with a batch size of 100 and momentum 0.9 for optimal fine-tuning. A global learning rate of 0.005 is used with a piece-wise scheduler which lowers down the learning rate by 10 times less than the previous one at every 10 epochs. We have used 10 times higher learning rate in the newly appended layers of proposed depth augmented networks.

3.4.2 Evaluation and Analysis of Proposed Fine-tuning

To investigate the impact of pre-trained classification layer in parameter fine-tuning and to compare the performance of proposed fine-tuning with traditional fine-tuning, we utilise our optimal fine-tuning scheme (Section 3.3.3). Figure 3.3 presents the results



(a) (b)

Figure 3.3: Performance comparison between proposed and traditional fine-tuning.

of two coarse (Caltech-256 and Pascal VOC-07) and two fine-grained datasets (CUB 200-2011, 102 Flowers). The *x-axis* in Figure 3.3a represents 8 layers of the AlexNet as $1, 2, ..., N-2, N-1, N$ and shows the layer from which fine-tuning proceeds while earlier layers are frozen (e.g., $N-2$ denotes fine-tuning of layer L_{N-2} to L_N and other layers are frozen). The *x-axis* in Figure 3.3b shows 16 layers (5 convolution blocks and 3 FC layers) of the VGG16 as $1, 2, ..., N-2, N-1, N$. For both networks, proposed fine-tuning (dashed lines) significantly outperforms traditional fine-tuning (dotted lines) for all datasets. This finding substantiates that fine-tuning pre-trained classification layer along with other transferred layers assist better transfer learning. Note that other datasets also perform similarly.

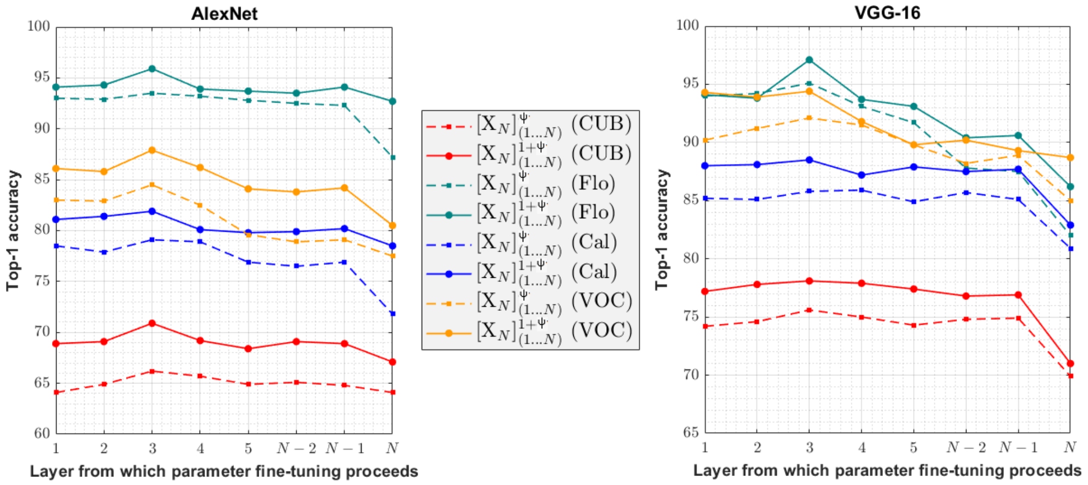
3.4.3 Layers to be Frozen for Optimal Performance

As shown in Figures 3.3a and 3.3b, the proposed and traditional fine-tuning performance increases when we continue to unfreeze and fine-tune more pre-trained layers according to different fine-tuning setups (Section 3.3.3). However, we observe a significant drop in performance when we tune initial conv. layers, more specifically, conv. layer 1 and 2 for AlexNet, and convolution block 1 and 2 for VGG16. The intuition is that fine-tuning these generic layers might introduce noisy or unwanted modifications of parameters.

That is, updating parameters would force the network to learn highly generic features of target set which are already learned from source set. The fine-tuning procedure has far fewer data and iterations than training from scratch, which might not let a vast number of pre-trained parameters of initial conv. layers find another such equilibrium to interact with next conv. layer in the same pace. Figure 3.3a portrays that optimal fine-tuning from the third convolution layer onwards of AlexNet yields the highest accuracy. Figure 3.3b manifests that VGG16 holds a similar trend for the third convolution block, which gives another perception that the first two convolution blocks of VGG16 may contain highly generic or low-level features.

3.4.4 Performance Analysis of Proposed Depth Augmented Networks

We append a new FC layer on top of the pre-trained classification layer, employ normalisation scheme to the augmented network, and perform layer-wise optimal fine-tuning. We discuss details about our normalisation scheme later in this section. The accuracy graphs in Figure 3.4 present the performance of proposed fine-tuning with the pre-trained classification layer (dashed lines), and proposed single-layer depth



(a) (b)

Figure 3.4: Performance comparison between fine-tuning with classification layer and single-layer augmentation.

augmented network (solid lines) for CUB 200-2011, 102 Flowers, Caltech-256 and Pascal VOC-07 datasets. The *x-axis* in Figure 3.4a represents 8 layers of the AlexNet as $1, 2, ..., N-2, N-1, N$ and shows the layer from which fine-tuning proceeds while earlier layers are frozen (e.g., $N-2$ denotes fine-tuning of layer L_{N-2} to L_N and other layers are frozen). The *x-axis* in Figure 3.4b shows 16 layers (5 convolution blocks and 3 FC layers) of the VGG16 as $1, 2, ..., N-2, N-1, N$. The results shown in solid lines of Figures 3.4a and 3.4b present our best performing augmented networks with 2048 neurons and signify that parameter fine-tuning with increased network capacity paves the way to learn better. Our proposed single-layer depth augmented network performes better than fine-tuning with the pre-trained classification layer. This observation verifies the effectiveness of increasing model capacity beyond the pre-trained classification layer when adapting it to both fine-grained and coarse classification task. Considering the best fine-tuning performance, CUB and 102 Flowers seem to achieve more gain than the other two.

Table 3.2: Performance analysis of proposed depth augmented networks.

Network	Dataset	S	$[\chi_N]_N^{(1+\psi)}$	$[\chi_N]_{N-1}^{(1+\psi)}$	$[\chi_N]_{N-2}^{(1+\psi)}$	$[\chi_N]_{N-5}^{(1+\psi)}$	All
AlexNet	CUB 200-2011	512	65.2	66.5	66.9	67.8	67.2
		1024	66.2	67.5	68.9	69.9	68.4
		2048	66.3	68.7	69.0	**70.9**	69.2
		4096	66.7	68.1	67.1	68.1	66.8
	Caltech-256	512	75.5	78.2	78.6	79.5	79.1
		1024	75.9	78.3	78.9	80.9	80.3
		2048	75.2	77.9	78.5	**81.9**	81.1
		4096	75.0	78.1	78.3	81.0	80.9
VGG16	CUB 200-2011	512	76.4	76.8	77.1	77.9	77.6
		1024	76.8	77.1	77.5	77.6	77.9
		2048	76.7	77.6	77.9	**78.1**	77.1
		4096	75.9	77.7	77.0	77.5	77.4
	Caltech-256	512	85.5	86.1	87.7	87.9	87.1
		1024	85.9	86.3	86.9	87.8	87.5
		2048	85.5	87.8	88.0	88.4	88.1
		4096	85.3	88.1	89.5	**88.5**	88.1

A detailed analysis of our investigation with single-layer depth augmented networks having different combinations of neurons, such as 512, 1024, 2048, and 4096, is shown in Table 3.2 where, S denotes the number of neurons in newly appended FC layer L_{N+1}.

Our empirical results indicate that the increase in performance is proportional to the increase in the magnitude of the new layer; however, for 4096 neurons, it diminishes marginally. In the proposed depth augmentation approach, the bridge between new and pre-trained layers is the layer consisting only 1000 neurons; it might suffer from an overabundance of parameters while propagating information through to four times larger neural layer. Single-layer depth augmentation with 2048 neurons and fine-tuning from layer $N - 5$ (i.e., third convolution layer/ block) yields best performance for almost all combinations. It is worth mentioning that our augmented networks also perform similarly for other datasets.

Table 3.3: Performance comparison of proposed single-layer augmentation with contemporary approaches (AlexNet).

Approach	**CUB 200-2011**	**102 Flowers**	**Stanford Dogs**	**Oxford Pets**	**Caltech-256**	**VOC07**	**MIT-67**	**SUN-397**
Normal_FT_CNN	62.3	88.9	63.8	80.0	72.1	77.9	61.2	53.9
CNN-SVM [6]	53.3	74.7	66.8	79.6	72.3	75.3	58.4	55.9
CNNAug-SVM [6]	61.8	86.8	66.6	79.9	74.8	76.8	69.0	56.2
LSVM [48]	61.4	87.1	65.0	77.6	69.7	75.2	66.7	55.8
MsML+ [48]	66.6	89.4	69.5	81.1	68.4	74.8	59.8	52.1
CombinedAlexNet [171]	63.3	83.3	64.5	76.9	69.2	77.1	58.8	54.2
WA-CNN [40]	69.0	92.8	66.9	82.4	79.5	83.4	66.3	58.3
Grow-conv [41]	66.1	91.4	67.2	82.1	78.3	77.0	70.1	50.2
Proposed network	**70.9**	**95.9**	68.7	**84.5**	**81.9**	**87.9**	69.9	**62.6**

Table 3.4: Performance comparison of proposed single-layer augmentation with contemporary approaches (VGG16).

Approach	**CUB 200-2011**	**102 Flowers**	**Stanford Dogs**	**Oxford Pets**	**Caltech-256**	**VOC07**	**MIT-67**	**SUN-397**
Normal_FT_CNN	70.5	85.6	68.2	85.2	83.9	86.5	66.5	61.8
CNN-SVM [6]	66.5	81.5	66.7	86.4	79.9	82.4	60.4	56.6
Muldip-Net [35]	71.5	81.9	65.0	86.1	80.9	87.5	68.9	63.5
Grow-conv [41]	72.5	88.7	75.1	89.1	86.1	89.1	72.1	67.5
DA-CNN [40]	76.1	93.3	72.8	88.4	84.9	91.4	73.1	67.2
Proposed network	**78.1**	**97.1**	**76.1**	**90.6**	**88.5**	**94.4**	**76.8**	**69.8**

Comparison with Contemporary Transfer Learning Works To further prove the robustness of proposed single-layer depth augmented networks, we summarise the performance comparison with different existing transfer learning approaches from literature in Tables 3.3 and 3.4. The best outcomes among various combinations of our single-layer depth augmented AlexNet and VGG16 evaluated by our optimal fine-tuning scheme are shown. For other approaches, the performance gap between our implementation and that reported by [6], [48], [171], [41], [40], [35] is due to different target sets, train-test splits, network architectures, and iterations. Note that we have used similar hyper-parameters, iterations, and train-test splits for all approaches in Tables 3.3 and 3.4 to maintain a fair comparison. Consistent superior outcomes validate that the presence of the pre-trained classification layer in increasing model capacity for parameter fine-tuning is effective for adjusting the network to a wide range of target tasks.

Table 3.5: Performance comparison between single-layer and multiple-layer augmentation.

Network	Dataset	Configuration	Accuracy (%)	Network	Dataset	Configuration	Accuracy (%)
AlexNet	CUB 200-2011	$[\chi_N]^{2+\psi}_{N-5}$	**72.1**	VGG16	CUB 200-2011	$[\chi_N]^{2+\psi}_{N-5}$	**78.9**
		$[\chi_N]^{1+\psi}_{N-5}$	70.9			$[\chi_N]^{1+\psi}_{N-5}$	78.1
	102 Flowers	$[\chi_N]^{2+\psi}_{N-5}$	**97.2**		102 Flowers	$[\chi_N]^{2+\psi}_{N-5}$	**98.2**
		$[\chi_N]^{1+\psi}_{N-5}$	95.9			$[\chi_N]^{1+\psi}_{N-5}$	97.1
	Caltech-256	$[\chi_N]^{2+\psi}_{N-5}$	**82.8**		Caltech-256	$[\chi_N]^{2+\psi}_{N-5}$	**89.4**
		$[\chi_N]^{1+\psi}_{N-5}$	81.9			$[\chi_N]^{1+\psi}_{N-5}$	88.5
	VOC-07	$[\chi_N]^{2+\psi}_{N-5}$	**88.8**		VOC-07	$[\chi_N]^{2+\psi}_{N-5}$	**95.9**
		$[\chi_N]^{1+\psi}_{N-5}$	87.9			$[\chi_N]^{1+\psi}_{N-5}$	94.4

Comparison of Single and Multiple-layer Depth Augmentation Augmenting two new layers beyond pre-trained classification layer is observed to be the cut-off point as performance starts to diminish after that. Table 3.5 shows the results of the best combination (i.e., L_{N+1}=2048 and L_{N+2}=1024) of two-layer and single-layer depth augmentation. Appending two new layers after the pre-trained classification layer facilitate network marginally over single-layer augmentation by increasing representational capacity. It is proven once again that the pre-trained classification layer holds prominent high-level features which are capable of propagating learned knowledge to multiple newly appended layers. This also manifests increasing network incrementally by augmenting depth is a stable parameterisation for improving performance.

Table 3.6: Performance comparison between different normalisation scheme.

Network	Dataset	Norm	Accuracy (%)				
			$[\chi_N]_N^{(1+\psi)}$	$[\chi_N]_{N-1}^{(1+\psi)}$	$[\chi_N]_{N-2}^{(1+\psi)}$	$[\chi_N]_{N-5}^{(1+\psi)}$	All
AlexNet	CUB 200-2011	Standardisation	66.2	67.5	68.9	**69.9**	68.4
		L_2	62.1	64.2	64.3	**64.5**	63.5
VGG16	CUB 200-2011	Standardisation	76.7	77.6	77.9	**78.1**	78.5
		L_2	71.5	72.1	73.1	**73.2**	72.7

3.4.5 Best Fit Normalisation Scheme

After exploring two types of normalisation, we observe that standardisation [162] assisted better learning for proposed single and multiple-layer depth augmented networks. We represent the results of diagnostic experiments with $S = 1024$ and standardisation in Table 3.6. Results of the single-layer depth augmented AlexNet trained on CUB 200-2011 dataset show that without normalisation followed by scaling scheme, the improvement of our depth augmented network is around 2% compared to traditional fine-tuning for L_2-norm normalisation, and more than 6% otherwise. A similar significant boost in performance is also noticed in other datasets, which are not stated in this paper for limited space. Increase in task performance states that standardisation reduces the chances of the pre-trained activations to dominate the randomly initialised ones.

3.4.6 Average Performance Gain

Tables 3.7 and 3.8 show the performance gain of proposed fine-tuning $[\chi_N]_{N-5}^{\psi}$ from traditional fine-tuning $[\chi_N^{N-1}]_{N-5}^{\psi}$, single-layer depth augmented networks $[\chi_N]_{N-5}^{1+\psi}$ from proposed fine-tuning, and double-layer $[\chi_N]_{N-5}^{2+\psi}$ from single-layer depth augmented networks for fine-grained and coarse datasets, respectively, where parameter fine-tuning proceeds from layer L_{N-5}. On average both coarse and fine-grained target datasets leverage the presence of the pre-trained classification layer in proposed fine-tuning and depth augmentation. However, coarse datasets manifest slightly more significant performance gain than fine-grained ones. This suggests that this layer possesses general information to transfer to a wide range of datasets. Results show that both the networks

Table 3.7: Performance gain for fine-grained datasets.

	CUB 200-2011		102 Flowers		Stan. Dogs		Oxf. Pets		Avg.	
	AlexNet	VGG16	AlexNet	VGG16	AlexNet	VGG16	AlexNet	VGG16	AlexNet	VGG16
$[\chi_N]^{\psi}_{N-5}$	3.3	2.8	3.1	1.6	1.4	1.6	4.2	2.0	**3.0**	**2.0**
$[\chi_N]^{1+\psi}_{N-5}$	4.7	2.5	2.0	2.9	1.8	3.7	2.1	3.7	**2.7**	**3.2**
$[\chi_N]^{2+\psi}_{N-5}$	1.2	0.8	1.3	1.1	0.4	1.2	1.0	0.9	**1.0**	**1.0**

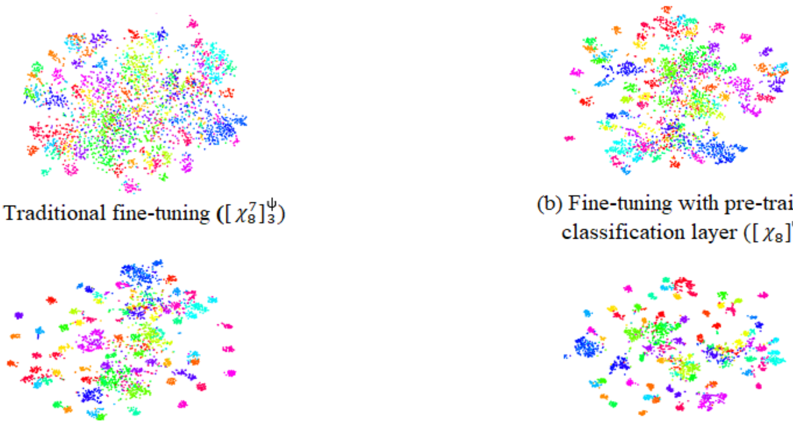
Table 3.8: Performance gain for coarse datasets.

	Caltech-256		VOC-07		MIT-67		SUN-397		Avg.	
	AlexNet	VGG16	AlexNet	VGG16	AlexNet	VGG16	AlexNet	VGG16	AlexNet	VGG16
$[\chi_N]^{\psi}_{N-5}$	4.0	2.7	3.4	4.6	5.1	6.5	6.7	4.5	**4.8**	**4.6**
$[\chi_N]^{1+\psi}_{N-5}$	2.8	2.7	3.4	2.3	3.6	3.8	2.0	3.5	**3.0**	**3.1**
$[\chi_N]^{2+\psi}_{N-5}$	0.9	0.9	0.9	1.5	1.4	1.1	1.7	1.3	**1.2**	**1.2**

gain significant improvement for single-layer augmentation while for two-layer, the increase is marginal. Moreover, less deep backbone network seems to be more benefited from depth augmentation.

3.4.7 Feature Visualisation

We plot t-SNE [172] based on the semantic categories of CUB 200-2011 dataset, to view the interrelation of the learned features. Figure 3.5 refers to the feature spaces of the new classification layer of different optimally fine-tuned networks. Figure 3.5a shows traditional fine-tuning improves a little bit of semantic separation of the pre-trained network while Figure 3.5b demonstrates significantly comprehensible semantic clusters because of the presence of pre-trained classification layer. The feature spaces of our depth augmented networks shown in Figures 3.5c and 3.5d exhibit even better semantic clusters which indicate that the proposed modules catch prominent features for classification as a result of increasing the network capacity beyond the pre-trained classification layer and that is compatible with their improved target task performance.



(a) Traditional fine-tuning $([\chi_8^7]_3^\psi)$

(b) Fine-tuning with pre-trained classification layer $([\chi_8]_3^\psi)$

(c) Single-layer depth augmented fine-tuning $([\chi_8]_3^{1+\psi})$

(d) Multiple-layer depth augmented fine-tuning $([\chi_8]_3^{2+\psi})$

Figure 3.5: t-SNE visualisations of learned features.

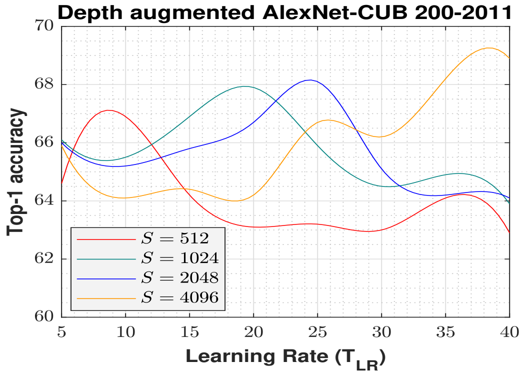


Figure 3.6: Performance of proposed depth augmented networks for various learning rates.

3.4.8 Analysing Optimal Learning Rate

Wang et al. [40] reported that they increased the learning rate of the new layer by 10 times more the than global rate. They provide no further reasoning behind choosing this value. To investigate their finding in a broader range, we applied a learning rate multiplier to the new layers, such as, $New_{LR} = T_{LR} \times Global_{LR}$, here $Global_{LR}$ = 0.005, and $T_{LR} \in \{5, 10, 15, ..., 40\}$. By varying the learning rate multiplier for new layers in the range of T_{LR}, we can measure the effect of increasing learning rate in task performance. Our experiments tend to relate the increment of learning rate to the size of the appended layer, as shown in Figure 3.6. More precisely, appending 512, 1024, 2048 and 4096 neurons tend to perform better when their learning rates are increased in the range of (5-10), (10-20), (15-25) and (25-40) times the global rate respectively. That can be attributed to the fact that the more we introduce new neurons to the network, the greater learn rate it requires to keep pace with the pre-existing ones. We also observe that fine-tuning of the target sets which have more similar categories with the source set desire for comparatively less increment of learning rate in new layers than distant target sets. Therefore, we claim there is no optimal setup for increasing the learning rate for new layers; rather, it depends on the target set and magnitude of the appended layer.

3.5 Summary

In this chapter, we have demonstrated that the pre-trained classification layer, which captures high-level features would assist parameter fine-tuning approach and propose a novel fine-tuning approach with it. We have empirically established that proposed fine-tuning approach outperforms traditional fine-tuning for all selected target datasets. Also, we have noticed on an average, the coarse target datasets with ImageNet achieve more performance gain than fine-grained ones. For evaluating traditional and proposed fine-tuning approaches, we have introduced a fine-tuning scheme. Our fine-tuning scheme manifests that freezing initial convolutional layers yield optimal fine-tuning performance for all target datasets. Being inspired by developmental transfer learning and impact of the pre-trained classification layer in fine-tuning, we have augmented new layers beyond the pre-trained classification layer for a better adaptation of the target task. Assessment of the proposed depth augmented networks on eight different datasets have shown that they outperform existing transfer learning approaches.

This chapter has presented our first contribution. It has also provided practitioners strong justification to utilise the ImageNet pre-trained classification layer for fine-tuning and depth augmentation beyond it for adapting the network to target tasks.

In the following chapter, we will investigate for suitable factors that encourage optimal performance in transfer learning using the classification layer features i.e, feature representation transfer.

Chapter 4

Analysing Suitable Factors for Feature Representation Transfer Approach

[1] In previous chapter, we show that parameter fine-tuning that utilises the ImageNet trained classification layer features can boost the performance of deep networks to learn to classify new datasets. However, the area of studying the impact of this layer in feature representation transfer and the impact of different factors such as optimal classifier, correlation between the source and the target datasets in this type of transfer learning is largely unexplored. Therefore, as our second contribution, this chapter presents an in-depth study of the influence of various target datasets (fine-grained and coarse) based on their similarity to the source dataset. We also explore the optimal machine learning classifier for this type of transfer learning. The results show that multi-layer perceptron performs better than SVMs. We observe that for coarse target datasets the source-target similarity has beneficial influence. For fine-grained datasets, the influence of inter-class similarity surpasses the source-target similarity. Finally, we utilise a feature selection algorithm based on conditional mutual information to verify our findings and prove the importance of the ImageNet trained classification layer features in transfer learning.

[1] Chapter 4 is adapted from: **T. Shermin**, S. W. Teng, M. Murshed, G. Lu, and F. Sohel, "Suitable Conditions for Transfer Learning with the ImageNet Trained Classification Layer Features," 2021 (to be submitted as a journal article).

4.1 Overview

In this chapter, we study the feature representation transfer approach with the classification layer features and the suitable factors to achieve optimal performance in transfer learning using the classification layer features. First, we explore the best performing machine learning classifier for feature representation transfer approach using the classification layer features. Second, we investigate the effect of correlation between the source (ImageNet), and the different types of target datasets (fine-grained and coarse). Finally, to verify our findings, we study the significance of the pre-trained classification layer features in transfer learning using a feature selection algorithm based on conditional mutual information.

To systematically accomplish our tasks of interest, we use the extracted features from the classification layer of ImageNet pre-trained deep networks and learn target datasets by training different machine learning algorithms. We also correlate the source and the target datasets based on the similarity between the source and target datasets by utilising a distance measure. We analyse the influence of relation between source and target datasets in transfer learning with the pre-trained classification layer features. In summary, we systematically investigate the following research questions (RQs).

RQ1: What is the optimal machine learning classifier for classification layer feature representation transfer approach?

RQ2: Does the similarity (based on distance) of target classes with the source classes influence performance?

RQ3: Does fine-grained (similar species) target classes influence the performance?

RQ4: How much the performance is influenced when the target dataset has coarse (mixed species) categories?

RQ5: Is the classification performance influenced by the number of training samples and the number of target classes?

For comparison, we explore traditional transfer learning approach that utilises the features from the layer before the pre-trained classification layer. The summary of the findings of this chapter are as follows:

We observe that non-linear machine learning classifiers perform better than linear ones. Among different classifiers, neural networks yield optimal performance. The proposed approach outperforms the traditional approach (using SVMs) for all cases. However, for neural networks, the proposed approach outperforms the traditional approach for the target datasets that possess greater source-target similarity to the source dataset, i.e., similarity between the target dataset and the source dataset (ImageNet).

For coarse-grained target datasets, transfer learning with the ImageNet trained classification layer features is influenced more by the source-target dataset similarity than the inter-target-class similarity. On the other hand, for fine-grained datasets, the inter-target-class similarity (among the classes of the target dataset) tend to have more impact than source-target similarity on the performance of this type of transfer learning.

We observe that the classification performance of the transfer learning with the pre-trained classification layer is positively influenced by the increasing number of samples per class regardless of similarity/dissimilarity.

4.2 Methodology

In this section, we formally discuss the transfer learning approaches, the procedure of determining similarity based on distance between datasets and our adopted feature selection algorithm for verifying the contribution of the classification layer features in transfer learning.

4.2.1 Transfer Learning Approaches

Figure 4.1a presents the block diagram of a pre-trained AlexNet, which has three FC layers and a softmax layer. Feature representation transfer approach extracts features from the pre-trained layers of AlexNet and uses them to learn new classifier. More specifically, the layer after the expected layer is deducted from the CNN and then the features are extracted from the expected layer. For example, in traditional way of feature representation transfer, the last FC (classification) and the softmax layers are deducted and the features are extracted from the second FC layer as shown in Figure 4.1b. However, we propose to deduct only the softmax layer and extract features from the third FC layer for transfer learning. We formally describe the traditional and our proposed approach below.

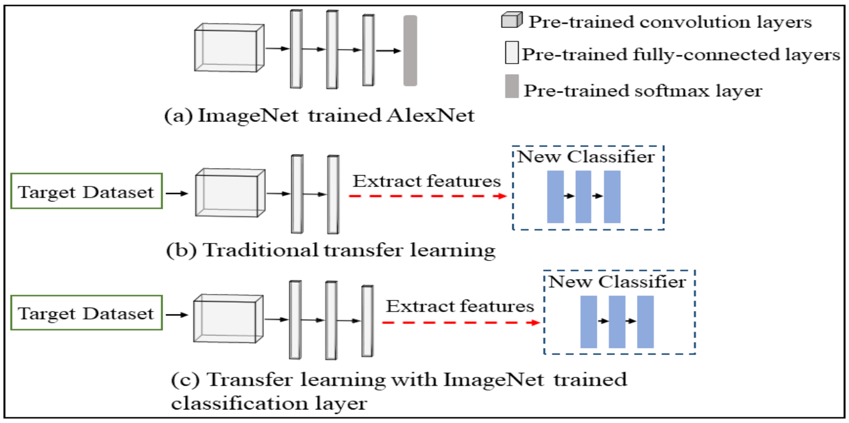

Pre-trained convolution layers
Pre-trained fully-connected layers
Pre-trained softmax layer
(a) ImageNet trained AlexNet
Target Dataset
Extract features
New Classifier
(b) Traditional transfer learning
Target Dataset
Extract features
New Classifier
(c) Transfer learning with ImageNet trained classification layer

Figure 4.1: Block diagram of feature representation transfer approach.

Traditional Transfer Learning Features from the penultimate FC layer (4096 dimensions) of the ImageNet trained network is extracted, and then the extracted features are utilised for separately training new machine learning classifiers such as SVMs, neural networks, etc (Figure 4.1b) [6, 25].

Transfer Learning with the Pre-trained Classification Layer In this approach, first, we extract the features from the classification layer (1000 dimension) of ImageNet trained deep networks. Then, the extracted features are used for training machine learning classifiers (e.g., neural networks and SVMs) for classifying target datasets separately (Figure 4.1c).

We will study the optimal classifier for our proposed approach and the impact of corelation between source and target datasets on the proposed approach. For comparison, we will utilise the traditional approach. We will also use both the approaches to show that the features from the classification layers are selected more for classification by using a feature selection algorithm (Section 4.2.3). This will verify the robustness of our approach.

4.2.2 Dataset Similarity based on Nearest Neighbours

First, we construct a feature representation (embedding) for representing each sample. The constructed feature representation for a sample is expected to hold low to high-level characteristics of the CNN features. This will ensure the comparison of similarity between datasets in all level of feature attributes. Then, we determine the similarity between the feature embedding of the target datasets to the feature embedding of source (ImageNet) dataset based on the nearest neighbours algorithm.

Deep CNN kernels are spatial filters which extract low, middle, and high-level features at different layers [5]. For example, a subset of the first convolutional layer's filter of ImageNet trained AlexNet exhibit oriented stripes resembling Gabor filters [1]. Therefore, such filters trained on the large-scale and diverse dataset, i.e. ImageNet, can be used for representing generic low, mid, and high-level image characteristics [29]. To construct the feature embedding, for each image of the source and target datasets, an average spatial pooling is performed over the output of each convolutional filter of pre-trained AlexNet for obtaining a single value per filter [173]. Each of the extracted features per convolutional filter will determine the availability of a visual pattern in the image on an average, irrespective of its location and help to reduce dimension. For the three FC layers in AlexNet, we take into account the outputs without pooling. As we wish to represent low, mid, and high-level features of an image in the feature embedding. We concatenate all spatially pooled convolutional and FC output features for constructing the final feature embedding [173]. For each feature embedding of the target dataset, we search for its K nearest neighbours in the source domain feature embedding and calculate the average Euclidean distance, where, K is equal to 100. Note that the value of K is determined empirically. We determine the similarity of the target datasets based-on the majority (above 80%) of samples located to the nearest distance with the source dataset.

4.2.3 Analysing Contribution of Classification Layer Features in Target Dataset Classification

To verify the effectiveness of the features from the pre-trained classification layer in feature representation approach, we show that the features are selected in good proportion for classification the target datasets. We adopt a feature selection algorithm [174] to perform this analysis. The algorithm depends on conditional mutual information. This approach is capable of addressing the problem of selecting the important features for the classification task based on a conditional mutual information score. The score searches for the feature, that adds the newest information to the already selected features and avoids redundancy.

First, we construct a feature embedding concatenating the features of pre-trained penultimate FC layer and classification layer. The embedding has 5096 concatenated features, sequentially representing the penultimate and classification layers of a deep network, i.e., $4096 + 1000$. Since the embedding has sequential indexing, the features belonging to a layer can be tracked down easily. Then, we apply the feature selection algorithm on the constructed feature embedding. The objective function of the feature selection approach is as follows,

$$\lim_{N \to \infty} -l = \mathbb{E}_{xy}\{\log \frac{p(y|x_\theta)}{q(y|x_\theta, t)}\} + I(X_{\widehat{\theta}}; Y|X_\theta) + H(Y|X). \tag{4.1}$$

Here, l is the conditional log-likelihood, x is a d-dimensional feature vector (i.e., 5096-dimension) and y is the target class label, drawn from the underlying distribution of random variables X and Y respectively. The first term represents a likelihood ratio between the true and the predicted class distributions given the selected features, averaged over the input space. The magnitude of this term is dependent on how well the model q can approximate p, given the selected features. The second term denotes the conditional mutual information between the class label and the unselected features, given the selected features. The value of this term will decrease as the selected feature set X_θ explains more about Y. And the term will become zero when the remaining/unselected features $X_{\widehat{\theta}}$ contain no additional information about Y in the context of the selected features. Due to chain rule, the mutual information between X_θ and Y can be formalised as,

$$I(X;Y) = I(X_\theta;Y) + I(X_{\widehat{\theta}};Y|X_\theta). \tag{4.2}$$

This indicates that minimising $I(X_{\widehat{\theta}};Y|X_\theta)$ is similar to maximising $I(X_\theta;Y)$. We, therefore, trace the importance of features of different layers by leveraging the X_θ of the second term of (4.1) to study how much knowledge the classification layer features possess to contribute to the target classification task. In particular, we keep track of the total number of times the X_θ are picked from both layers. If the classification layer features have a significant number of selection count, it will verify the effectiveness of the classification layer features in classifying target datasets. To compare, we will also show the selection counts for the penultimate layer features.

4.2.4 Study Method

First, we determine the similarity between the source and target datasets. Then, we use transfer learning with the pre-trained classification layer and traditional transfer learning to investigate the optimal classifier and address the RQs stated in Section 4.1. Finally, we utilise the feature selection algorithm to verify the importance of the classification layer features in transfer learning.

4.3 Experimental Studies

In this section, we describe our implementation details, compare the performance of the transfer learning with pre-trained classification layer with the traditional transfer learning approach, determine the best performing classifier, analyse the influence of datasets on the former type of transfer learning based on our research questions stated in Section 4.1.

4.3.1 Datasets

We used eight different fine-grained (same species) and coarse datasets (mixed species), as stated in Table.3.1. ImageNet is used as the source dataset. We determine the similarity between the source and target datasets following our similarity measure described in Section 4.2.2. Figure 4.2 shows the similarity between source and target datasets according to our similarity measure. VOC 07, and Caltech-256 among coarse datasets manifest high similarity with the Imagenet. On the other hand, ImageNet has low similarity with MIT 67 indoor scenes as it has object categories which collectively form different category in the scenes of MIT 67. SUN-397 manifests the highest distant semantic attributes to ImageNet, among others as its scenes comprise objects that have negligible semantic similarity with ImageNet categories. Among fine-grained datasets, CUB 200-2011, and 102 Flowers demonstrate high dissimilarity with ImageNet while the other two Stanford Dogs and Oxford Pets show a great deal of similarity with the ImageNet. The ranking of the target datasets in descending order of similarity to the source dataset is as VOC > CAL > MIT > DOG > PET > SUN > FLO > CUB.

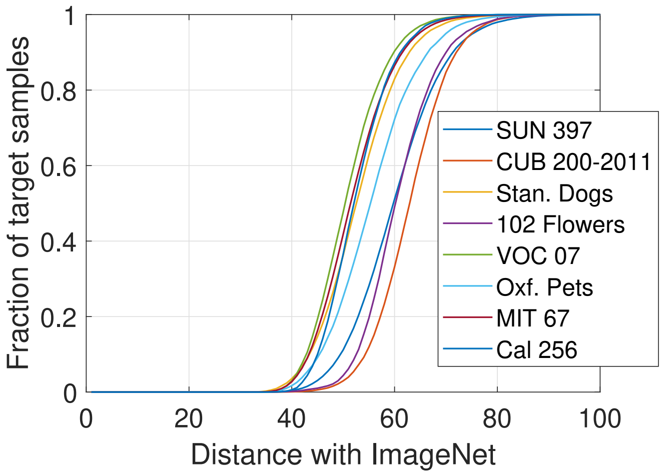


Figure 4.2: Distance between the source and target datasets-based on nearest neighbours.

4.3.2 Implementation details

For feature extraction, AlexNet and VGG16 pre-trained in ImageNet [175] are utilised. We implement three different types of machine learning algorithms (Multi-Layer Perceptron (MLP), Linear SVM, RBF-kernel SVM) for classifying the target datasets. To train the MLP classifier, stochastic gradient descent (SGD) optimisation was used with a learning rate of 0.001. For the purpose of calculating loss function, categorical cross-entropy loss was used. To train the two variants of multi-class SVMs, to follow the default setup of Scikit-learn. The retrieval of pre-trained weights and all the experiments are done in PyTorch. To evaluate the experimental results of two transfer learning approaches, we consider the test accuracy obtained on the test sets as the performance metric.

4.3.3 Best Fit Classifier (RQ1)

Table 4.1: Performance comparison of various classifiers (accuracy in %).

Dataset	AlexNet						VGG16					
	TL_N^S	TL_O^S	TL_N^R	TL_O^R	TL_N^M	TL_O^M	TL_N^S	TL_O^S	TL_N^R	TL_O^R	TL_N^M	TL_O^M
VOC	71.5	71.6	73.2	71.0	79.3	77.8	80.3	80.3	82.1	80.1	86.2	84.4
CAL	64.3	65.7	66.5	63.6	75.9	74.3	68.5	68.1	70.9	70.5	83.3	82.3
MIT	54.6	56.2	57.5	57.1	62.4	61.6	61.1	61.8	62.7	61.8	63.9	62.4
DOG	64.6	65.4	65.8	65.1	67.2	65.9	70.6	71.9	70.8	70.1	67.3	66.4
PET	78.2	77.3	78.8	77.1	80.5	78.7	78.2	78.6	84.9	82.1	86.4	85.9
SUN	50.2	51.6	52.1	50.8	56.9	57.9	54.1	54.6	55.8	55.0	60.0	60.9
FLO	79.1	81.3	82.8	79.9	84.2	85.3	72.1	79.2	78.5	78.1	86.2	86.5
CUB	46.9	50.2	49.6	46.7	64.3	64.2	57.2	57.9	58.1	57.2	67.9	68.2

Table 4.2: Performance comparison of proposed transfer learning with contemporary approaches.

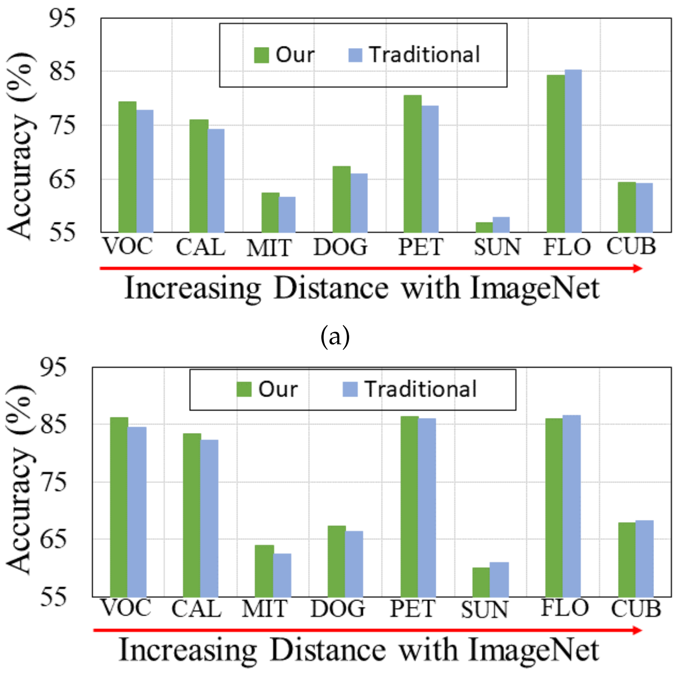
Network	Approach	**CUB 200-2011**	**102 Flowers**	**Stanford Dogs**	**Oxford Pets**	**Caltech-256**	**VOC07**	**MIT-67**	**SUN-397**
AlexNet	CNN-SVM [6]	53.3	74.7	66.8	79.6	72.3	75.3	58.4	55.9
	CNNAug-SVM [6]	61.8	86.8	66.6	79.9	74.8	76.8	69.0	56.2
	LSVM [48]	61.4	87.1	65.0	77.6	69.7	75.2	66.7	55.8
	CombinedAlexNet [171]	63.3	83.3	64.5	76.9	69.2	77.1	58.8	54.2
	TL_N^M (our)	**64.3**	84.2	**67.2**	**80.5**	**75.9**	**79.3**	62.4	**56.9**
VGGNet	CNN-SVM [6]	66.5	81.5	66.7	86.4	79.9	82.4	60.4	56.6
	TL_N^M (our)	**67.9**	**86.2**	**67.3**	**86.4**	**83.3**	**86.2**	**63.9**	**60.0**

Table 4.1 shows the classification accuracy (%) of transfer learning using the pre-trained classification layer's features (TL_N) and traditional transfer learning (TL_O) with two variants of SVM (i.e., Linear and with RBF-kernel) and MLP where, Linear SVM is denoted as S, RBF-kernel SVM as R and MLP as M. The transfer learning using the classification layer features outperform the traditional approach for majority of the cases when RBF-SVM and MLP is used (highlighted in blue). To be more specific, the MLP (TL_N^M) classifier performs the best among three classifiers and the RBF-kernel SVM (TL_N^R) outperforms linear SVM (TL_N^S). This proves that the classification layer features facilitate to boost performance when non-linearity is introduced in the target classifier (i.e., activation function of MLP and RBF-kernel of SVM introduce non-linear characteristics in the classifiers). However, for the best performing classifier (MLP), we observe that proposed approach outperforms the traditional approach when the target dataset has more similarity to the source dataset. Otherwise, the accuracy is similar to the traditional approach. For example, the least similar target datasets SUN, FLO, and CUB show similar performances in both approaches. Thus, for MLP, the proposed approach performs better than traditional approach when the target datasets has more global similarity to the source datasets. For RBF-SVM, the proposed approach outperforms the traditional approach for all cases.

Following our analysis, we compare the performance of the proposed approach using the highest performing classifier, MLP, with contemporary feature representation transfer approaches in Table 4.2. We observe that our approach outperforms all compared approaches for majority of the datasets for both AlexNet and VGGNet features. This proves that the classification layers features have potential transferable information to generalise to target datasets.

4.3.4 Analysis of Impact of Datasets

In this section, we discuss in detail about the findings from our experiments as research answers and derive the role of several target datasets in transfer learning with the ImageNet pre-trained classification layer. We perform fine-grained and coarse class classification for both the transfer learning approaches using MLP.



(a)

(b)

Figure 4.3: Performance comparison with respect to source-target similarity. (a) Pre-trained AlexNet and (b) pre-trained VGG16 is used.

RQ2: Our second RQ concerns the effect of similarity between the source and target classes on the performance of transfer learning with the ImageNet pre-trained classification layer. Figure 4.3 shows performance of transfer learning with the ImageNet trained classification layer (green bars) and traditional transfer learning (blue bars) with respect to distance between the source and target datasets. It is evident that for all coarse-grained datasets (VOC, CAL, MIT, and SUN) with the gradual diminution of similarity with the source dataset the accuracy for both approaches decrease. This observation establishes a strong relationship between the similarity of source-target (coarse) classes and the classification performance. The intuition is that the more common characteristics the source and coarse target datasets hold, the easier it is for the transfer learning model to classify better.

However, for the fine-grained datasets (DOG, PET, FLO, and CUB) no such pattern is observed. We also observe that PET and FLO outperforms DOG despite having low similarity. The higher performance of PET and FLO datasets can be explained by the fact that the datasets are comparatively easy for the machine learning classifiers due to their level of complexity (more detail on next RQ).

RQ3: Our third RQ focuses on the relationship between visual similarity among the target classes (fine-grained) and the transfer learning using the ImageNet trained classification layer features. Among the selected target datasets, DOG, PET, FLO, and CUB are fine-grained datasets. As shown in Figure 4.3, the classification performance of the fine-grained datasets with more similarity among the classes lag behind the fine-grained datasets with lower similarity among the classes. Particularly, the DOG and CUB datasets comprise similar categories of dogs and birds, respectively. These two datasets are difficult to classify as they have large number of common intra-dataset semantic attributes. On the other hand, the PET dataset has different types of pets such as cats and dogs, and FLO dataset has variety of flowers with less common attributes. Therefore, PET and FLO datasets are comparatively easier than DOG and CUB to classify. Hence, PET and FLO outperforms the nearest fine-grained target dataset DOG for both AlexNet and VGGNet. CUB also outperforms DOG despite being the farthest to the source dataset for VGGNet. Thus our study demonstrates that the transfer learning classification performance is more influenced by the visual similarity among the target classes than the source-target class similarity for fine-grained datasets. More specifically, low inter-class similarity benefits the transfer learning performance more.

RQ4: Our empirical analysis demonstrates that the impact of coarse-grained (mixed species) datasets in the transfer learning with the classification layer features is linked to the global similarity between the source and target datasets. That is, the more the coarse dataset is correlated (distance-based similarity) to the source dataset, the better the classification accuracy (Figure 4.3). More specifically, VOC performs better than CAL, CAL performs better than MIT, and MIT performs better than SUN. VOC is nearest to the source dataset while SUN is the fartest among the coarse datasets. This also indicates that this type of transfer learning benefits from the coarse dataset that has more common classes to the source dataset, for example, VOC and CAL have more overlapping classes with ImageNet. The traditional transfer learning also tends to show a similar pattern for this type of datasets. Another interesting finding is that when the visual similarity of inter-classes of a coarse target dataset increases and the target dataset has low similarity to the source dataset then the proposed approach yields comparatively less performance. For example, SUN and MIT datasets have many common inter-class scenes and objects within the datasets but have less common classes with the source (ImageNet). Consequently, the performance of SUN and MIT lags behind VOC and CAL. The intuition is that the greater global similarity than local similarity facilitates this type of transfer learning for coarse datasets more.

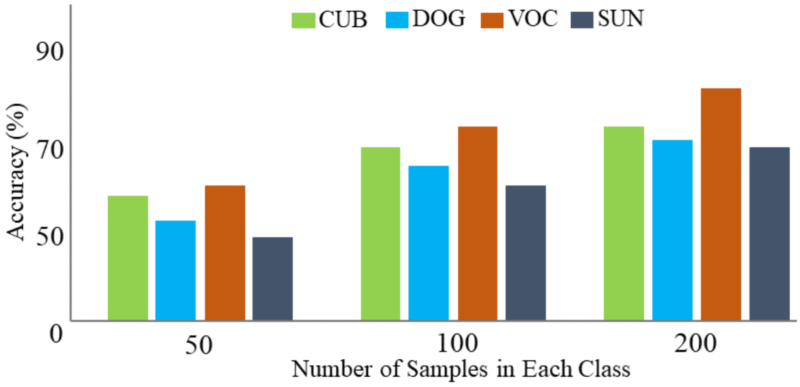


Figure 4.4: Bar-chart showing performance for various number of training samples per class.

RQ5: To investigate our fifth RQ, we select two fine-grained and two coarse datasets. We experiment with 50, 100 and 200 samples in each class. For CUB, we use data augmentation to fulfil the number of samples per class. Figure 4.4 demonstrates that classification performance is highly affected by the number of samples used during training. For target datasets with different level of similarity to the source dataset, experiments with 200 samples outperform other combinations. This indicates more training samples assist better transfer learning.

Tables 4.3 and 4.4 shows the performance comparison of transfer learning with the classification layer's features (TL_N) and traditional transfer learning (TL_O) for a different number of classes in the target datasets where, $C_1 = 10$, $C_2 = 20$ for PET and VOC, $C_2 = 20$ for other datasets, $C_3 = 50$ and *All* denotes all classes in target datasets. Note that '-' denotes not enough classes available in the dataset. As shown in

Table 4.3: Performance comparison for various number of target classes (AlexNet).

Type	Target Dataset	Classes							
		C_1		C_2		C_3		All	
		TL_O	TL_N	TL_O	TL_N	TL_O	TL_N	TL_O	TL_N
Coarse	VOC	81.4	82.6	77.8	79.3	-	-	77.8	**79.3**
	CAL	79.3	80.2	77.9	78.1	75.7	76.9	74.3	**75.9**
	MIT	64.5	64.7	62.9	63.1	61.9	62.5	61.6	**62.4**
	SUN	59.3	57.6	59.2	56.9	58.7	56.3	**57.9**	56.9
Fine-grained	DOG	70.2	71.8	70.1	69.5	67.3	68.3	65.9	**67.2**
	PET	84.0	84.2	81.6	82.5	-	-	78.7	**80.5**
	FLO	87.3	87.2	86.8	85.9	86.2	85.1	**85.3**	84.2
	CUB	66.2	65.9	66.1	64.5	65.8	63.9	64.2	**64.3**

Table 4.4: Performance comparison for various number of target classes (VGG16).

Type	Target Dataset	Classes							
		C_1		C_2		C_3		All	
		TL_O	TL_N	TL_O	TL_N	TL_O	TL_N	TL_O	TL_N
Coarse	VOC	85.1	87.8	84.4	86.2	-	-	84.4	**86.2**
	CAL	85.5	86.8	85.1	85.6	84.1	84.5	82.3	**83.3**
	MIT	66.7	67.8	65.9	65.9	65.7	64.8	62.4	**63.9**
	SUN	63.1	61.9	62.8	61.1	62.1	60.8	**60.9**	60.0
Fine-grained	DOG	72.9	73.2	69.4	70.3	68.7	68.5	66.4	**67.3**
	PET	87.8	88.9	87.9	87.7	-	-	85.9	**86.4**
	FLO	89.5	89.1	88.1	88.7	87.7	87.3	**86.5**	86.2
	CUB	69.8	69.3	68.5	68.9	68.9	68.3	**68.2**	67.9

Tables 4.3 and 4.4, from left to right, with an increase in the number of classes in the target datasets, the accuracy shows a decreasing trend. This means the proposed approach performs better when the target dataset has less number of classes. Tables 4.3 and 4.4 also demonstrate that the proposed approach outperforms the traditional approach for various number of target classes in the datasets.

4.3.5 Feature Importance Analysis

To verify the importance of classification layer features in transfer learning, we implement the feature selection algorithm for target datasets classification, as discussed in Section 4.2.3. When the classification layer features are selected in significant proportion it means the features have generalisation ability and benefits transfer learning.

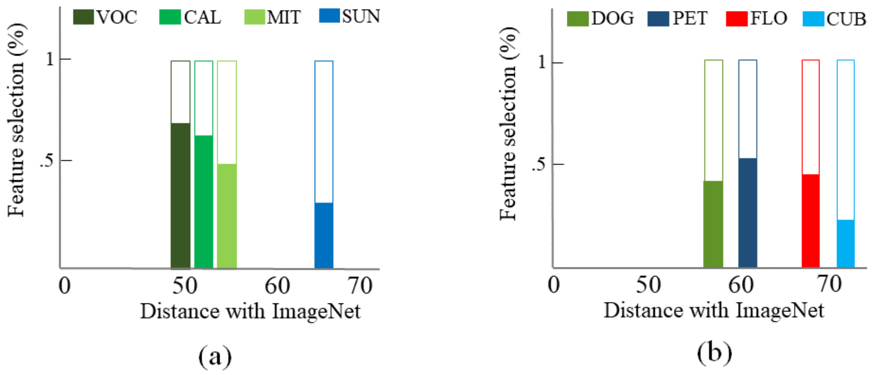


Figure 4.5: Bar-chart showing feature importance.

Figure 4.5 shows the percentage of feature selection for the classification layer and traditional approach (penultimate FC layer) features with respect to the source-target similarity. Here, the proportion of the selected classification layer features are coloured, and the hollow bars represent the proportion of selected features of traditional transfer learning approach. The x-axes show the proportion of feature selection and the y-axes show the distance between ImageNet and target datasets.

For classifying coarse-grained target datasets (VOC, CAL, SUN, and MIT), it is observed that the selection algorithm selects more classification layer features from the target datasets that have more similarity to the source dataset (see Figure 4.5a). That is, the percentage of classification layer feature selection decreases with the increasing dissimilarity between the source and target datasets. For fine-grained target datasets (DOG, PET, FLO, and CUB), the classification layer features of the datasets with less inter-class similarity has more selection rate except for the DOG dataset (see Figure 4.5b). More specifically, PET and FLO have low semantic similarity among the classes within the datasets while DOG and CUB have high semantic similarity. PET and FLO have more classification layer selection rate than DOG and CUB.

Apparently, the feature selection algorithm selects significant proportion of classification layer features for majority of the target datasets. Therefore, the classification layer features of a pre-trained CNN hold meaningful and valuable information that benefits classification of new target datasets.

4.4 Summary

In this chapter, we have proposed a transfer learning approach that uses the ImageNet trained classification layer features. We have extensively studied the optimal classifier and similarity-based correlation between the source and target datasets for the proposed approach. We have found that the neural networks perform best for the proposed approach. Our study demonstrated that the ImageNet trained classification layer features are more effective for classifying coarse-grained target datasets that hold more global similarity (inter-dataset similarity) to the target dataset. However, for fine-grained target datasets, the local similarity (inter-class similarity) influences more. In the end, we have verified the importance of the ImageNet trained classification layer features in transfer learning by utilising a feature selection algorithm.

Our second contribution discussed in this chapter has proven that the classification layer features of a pre-trained CNN on diverse dataset plays vital role in improving different types of inductive sequential transfer learning approaches. In the next chapter, we will present the third contribution, i.e, to mitigate negative transfers using adaptive weighting and improve open set domain adaptation.

Chapter 5

Reducing Negative Transfers in OSDA Using Adaptive Weighting

[1]In previous two chapters, we presented our first contribution to improve the sequential transfer learning using the category-specific CNN features. In this chapter, we will present the third contribution of the thesis.

DA aims to transfer knowledge from a domain with adequate labelled samples to a domain with scarce labelled samples. Prior research has introduced various OSDA settings in the literature to extend the applications of DA methods in real-world scenarios. In this thesis, we focus on the type of OSDA setting where the target domain has both private ('unknown classes') label space and the shared ('known classes') label space. However, the source domain only has the 'known classes' label space. This OSDA setting can handle real-world problems due to supporting the availability of unknown classes besides shared/known classes in the target domain. However, the negative transfer is a critical issue in OSDA, which stems from misalignments of known/unknown classes during adaptation. Current OSDA methods, in some cases, are vulnerable to negative transfers. In this and the following chapters, we describe in detail our proposed approaches to reduce negative transfers and thereby improve OSDA performance for image classification.

Prevalent distribution-matching DA methods are inadequate in such a setting that demands adaptation from a smaller source domain to a larger and diverse target domain

[1] Chapter 5 is adapted from: **T. Shermin**, G. Lu, S. W. Teng, M. Murshed, and F. Sohel, "Adversarial network with multiple classifiers for OSDA," IEEE Transactions on Multimedia, 2020.

with more classes. To address this specific OSDA setting, prior research introduces a domain adversarial model that uses a fixed threshold for distinguishing known from unknown target samples and lacks at handling negative transfers. In this chapter, we extend their adversarial model and propose a novel adversarial DA model with multiple auxiliary classifiers. The proposed multi-classifier structure introduces a weighting module that evaluates distinctive domain characteristics for assigning the target samples with weights which are more representative to whether they are likely to belong to the known and unknown classes to encourage positive transfers during adversarial training and simultaneously reduces the domain gap between the shared classes of the source and target domains. A thorough experimental investigation shows that our proposed method outperforms existing DA methods on several benchmark DA datasets.

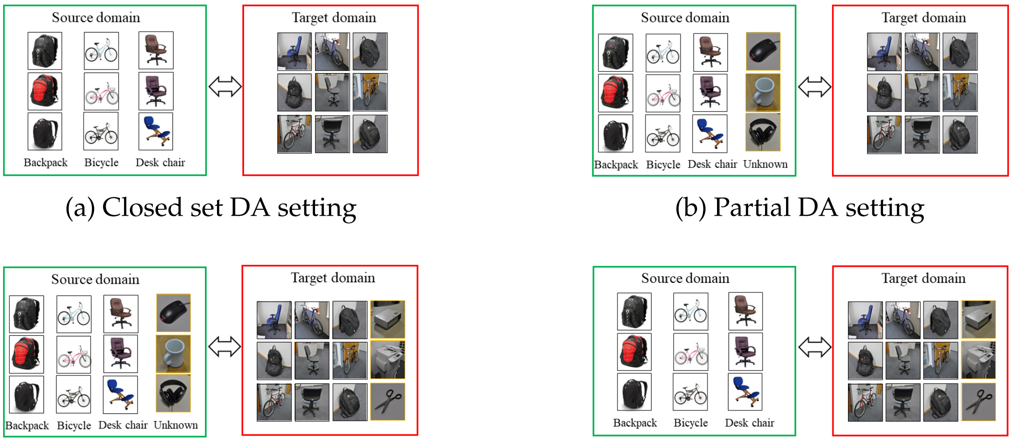
5.1 DA Setting

The OSDA setting of our focus constitutes a source domain $\mathbb{D}_s = (x_i^s, y_i^s)_{i=1}^{n_s}$ of n_s labelled instances associated with $|C_s|$ classes, which are drawn from distribution p_s and a target domain $\mathbb{D}_t = (x_j^t)_{j=1}^{n_t}$ of n_t unlabelled instances drawn from distribution p_t, where $p_s \neq p_t$. We denote the class labels of the target and source domains as C_t and C_s respectively. The shared label space is denoted as $C = C_s \cap C_t$. $\overline{C_t} = C_t \setminus C$ represents the label sets private to the target domain, which should be recognised as 'unknown'. According to the setting, we have access to a fully labelled source domain and a fully unlabelled target domain during training. The source domain has only known (C) classes, while the target domain has both known (C) and unknown (C_t) classes. The task is to correctly classify the shared classes C and target private classes $\overline{C_t}$ as 'unknown'. As the target domain is unlabelled, it is difficult to identify which part of the target label space C_t is shared with the source label space C_s because the target domain is fully unlabelled and C_t is unknown at the training time. This setting is also used in the next chapter.

5.2 Overview

As discussed earlier, CNNs usually require a massive amount of labelled data entailing highly laborious work for annotating data [3, 4, 176, 177]. An alternative is to use labelled data from a related (source) domain to boost the performance of the model in a target domain. However, as the source and target data may have domain gaps such as different illumination set-ups, and perspectives, synthesised data by using different variants of sensors, the performance of this approach may suffer. Existing DA methods aim to decrease the above-mentioned domain divergences either by using distribution matching methods [67, 73, 178, 179] or by transforming samples from one domain to another through generative models [8, 52, 73, 180–183]. Generally, it is assumed that label sets across the source and target domains are identical (closed set DA [50, 51, 55]), as shown in Figure 5.1a. However, such a simplified setting only has limited real-world applications. OSDA [81, 84], and partial DA [105, 184, 185] methods have been proposed to ease the closed set DA assumption. Open set and partial DA settings assume source or target domain private label sets besides the identical (shared) label sets. The DA models-based on these DA settings are required to recognise the samples of the target domain private label sets as 'unknown' class and the samples belonging to shared label sets as known classes. As illustrated in Figure 5.1b, partial DA [105, 184, 185] setting assumes that the source domain label space is a superspace of the target domain label space. OSDA setting proposed by [81] requires images from unknown classes both in the source and target domain besides the shared known classes. However, this is not a cost-effective setting for OSDA as it requires a collection of a large number of unknown source samples with no prior knowledge about target labels. During training, Busto et al. [81] bound the DA model to align unknown classes of the target domain towards unknown classes of the source domain. This may enforce a firm boundary for the unknown classes. During testing, samples from unknown classes other than the trained unknown classes may confuse the model.

OSDA by back-propagation (OSBP) [84] setting takes another step towards the practical DA scenario by removing unknown classes from the source domain such that



(a) Closed set DA setting

(b) Partial DA setting

(c) OSDA setting [81]

(d) OSDA by back-propagation setting [84]

Figure 5.1: (a) The closed set DA setting assumes that both source and target domains consist of images only of the same set of classes. (b) Partial DA setting assumes that the source domain label space (classes) is the superspace of the target domain label space. This adaptation setting includes images of unknown classes in the source domain only. (c) OSDA setting proposed by Busto et al. [81] requires images of unknown classes in both source and target domains, i.e., both source and target domains contain images that do not belong to the label space of interest. (d) OSDA setting introduced by Saito et al. [84] requires images of unknown classes only in the target domain besides the classes of interest. We focus on the DA setting illustrated in (d).

the source label space is a subset of the target label space (Figure 5.1d). This means that the OSBP DA setting has samples from unknown classes only in the target domain and encourages the DA model to learn to detect an unknown target sample as unknown when it does not belong to the known classes. Thus, this setting assists in training the DA model with a broader unknown class boundary and detect unknown samples during testing better than the previous setting [81]. This chapter focuses on improving the performance of DA model for the OSBP DA setting, which is realistic and challenging as the source domain has less number of classes compared to the target domain. This setting is suitable for several real-world applications. For example, if the automated car robots are capable of recognising unseen traffic-signals/street-signs during training as ‘unknown’, the chances of on-road misclassification (which may stem life-threatening accidents) can be reduced. Also, labeling them later will increase the knowledge base of the robots.

Prevalent distribution matching DA methods [67,73] cannot be applied to the OSBP

DA setting as the absence of unknown samples in the source domain does not allow the unknown samples of the target domain to be aligned. Saito et al. [84] have proposed a generative model to address the OSBP DA setting where they enable the classifier to draw a rough boundary between the source and target samples (i.e., initially all the target samples will be classified as 'unknown') and the generator has to separate the target samples into known and unknown classes adversarially. The adversarial learning between the generator and the classifier depends on the pseudo decision of the classifier. However, their proposed method does not explore any underlying domain discriminative information to assess the pseudo decision of the classifier before the adversarial training. Also, they set an empirical fixed threshold for generator-classifier adversarial training to differentiate known target samples from unknown ones. We argue that this concept of relying only on the rough decision of the classifier may encourage the negative transfer of target samples.

Pan et al. [7] stated that a DA model is prone to negative transfers when it lags behind a non-DA model (which is trained only on the source domain) in performance. The fixed threshold (0.5) for constructing boundary between known and unknown target samples leads to biased adversarial learning. That is, for a target sample, when the classifier assigns an unknown probability low than the threshold based on its pseudo decision boundary, the generator will always be encouraged to align that sample towards known classes even if they belong to unknown classes. The OSBP method struggles to perform better than non-DA classifier for some tasks discussed in Section 5.4.3 because of negative transfers by aligning unknown target samples towards known classes.

To address the limitations of the current OSBP methods, we propose to extend their domain adversarial network by integrating a new multi-classifier based weighting module to the network. We refer to the extended generative model as the multi-classifier based adversarial DA model. To reduce negative transfers, first, we evaluate the underlying discriminative domain information of the known and unknown target samples, and then assign them with distinguishable weights which are more representative to their similarities to the source domain. In particular, the underlying domain information of target samples is measured by evaluating discriminative label information based on the resemblance to each source domain classes and probable similarity with the source domain classes when measured against the probability of belonging to the unknown classes of the target domain. This ensures that the proposed weighting module provides a rough estimate of the underlying domain of the target samples. Then based on our generated weights, the generative module performs adversarial DA and aligns known target samples towards known source samples and rejects the unknown target samples. Thus, in the proposed DA model, the generative module is not forced to draw a threshold driven boundary between known and unknown target samples, which may initiate negative transfers as the baseline method. The proposed model improves performance over the previous method [84], which indicates it constructs a good boundary between known and unknown target samples by eliminating negative transfers.

Several attempts to address partial DA [184, 185], universal DA [107], and open set DA [85] by generating weights are identified in the literature. Zhang et al. [185] utilise the output of an auxiliary domain discriminator to derive the probability of the source samples belonging to the target domain and assign weights to the source samples accordingly for partial DA. For improving prior partial DA methods participating in negative transfer [105, 106], Example Transfer Network (ETN) [184] degrades the weight of source images from source private classes before integrating to the source classifier and places a discriminative domain classifier to quantify sample transferability. To generate weights for source samples belonging to the shared label sets, Universal Adaptation Network (UAN) [107] integrates domain similarity and prediction uncertainty. Unlike our proposed method, UAN does not integrate underlying label information to evaluate domain similarity between the source and target samples. The above-mentioned methods concentrate on assigning weights to the source samples, whereas the OSBP DA setting demands to assign weights to the unlabelled target samples. This is more challenging than weighting source samples as we do not know the labels of target samples and the shared label space during training. We propose a new multi-classifier based weighting scheme in an adversarial DA method for the OSBP DA setting. Separate to Adapt (STA) [85] method assigns weights to a target sample based on its highest similarity to one of the source domain classes. On the contrary, our proposed weighting module assesses domain information based on a target samples' similarity to known classes and dissimilarity to the unknown class, and then assign identifiable weights (please refer to Section 5.3.2.2 for details). The main contributions of this paper are as follows:

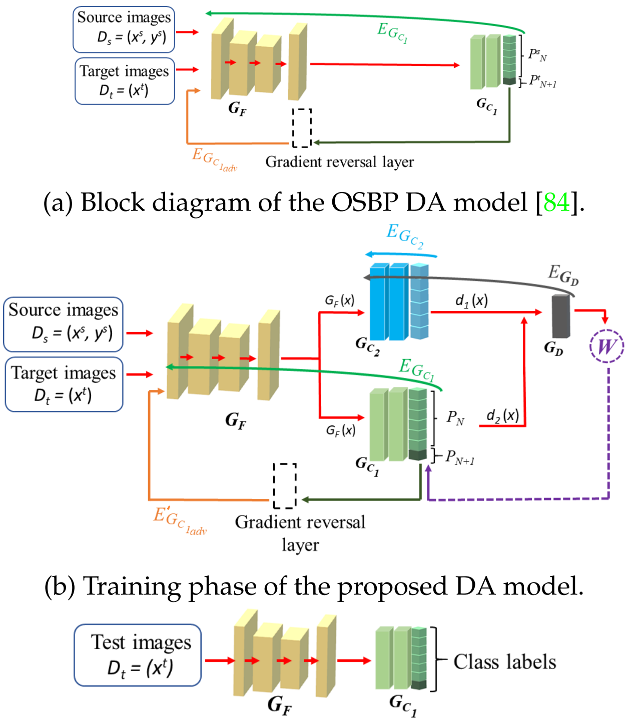
- We propose a domain adversarial model by integrating a new multi-classifier module in the OSBP domain adversarial model [84] for the OSDA setting that has access to unknown classes only in the target domain. The multi-classifier structure introduces a weighting scheme in the proposed model, that assesses fundamental domain information based on distinctive label information for assigning identifiable weights to the known and unknown target samples. This enhances the positive transfer of target samples and facilitate the adversarial training. Unlike the previous method [84], which requires an assumption of the known-unknown boundary threshold, the proposed method is capable of automatically discovering the boundary between known and unknown target samples.
- We conduct comprehensive experiments and demonstrate that our proposed model reduces the rate of negative transfer and achieves better performance than contemporary DA methods on several datasets.

5.3 Proposed Methodology

In this section, we discuss the limitations of the baseline method and present our proposed method.

5.3.1 Baseline Domain Adversarial Model

In this section, we briefly discuss the OSBP [84] model and its tendency to initiate negative transfers. The OSBP method (Figure 5.2a) is an adversarial DA model that aims to reduce divergence between the source and target domains by learning transferable features in a two-player minimax game in line with existing domain adversarial networks



(a) Block diagram of the OSBP DA model [84].

(b) Training phase of the proposed DA model.

(c) Testing phase of the proposed DA model.

Figure 5.2: (a) Block diagram of the baseline method [84]. (b) Block diagram of the training phase of our proposed adversarial DA network with a multi-classifier based weighting scheme. Here, G_F, G_{C_1}, G_{C_2} and G_D denotes the generator, domain classifier, non-adversarial supplementary source and domain classifier respectively. $E_{G_{C_1}}$ and $E_{G_{C_{1adv}}}$, $E_{G_{C_2}}$, and E_D are errors for optimising G_F and G_{C_1}, G_{C_2}, and G_D respectively. (c) A pictorial illustration of the proposed model during the testing phase. Red arrows denote forward passes while other arrows represent a backward pass.

[54, 56]. The first player of the model is a domain classifier G_{C_1} and the second player is a feature generator G_F. The final objective of the OSBP method is to correctly classify known target samples as corresponding known class and target samples belonging to unknown classes as 'unknown'.

The feature generator G_F takes inputs from both source domain D_s and target domain D_t at the same time. The domain classifier G_{C_1} takes features from G_F and outputs $N+1$ dimensional probability, where N specifies the number of known or source categories (C_s) and the probability for the unknown category is indicated by the $(N+1)^{th}$ index. During the forward pass, within G_{C_1}, the features are transformed to a $N+1$-dimensional class probability through softmax function as, $\sigma(z) = \exp(z)/(\sum_{i=1}^{N+1} \exp(z_i))$, where z is the logit vector.

The OSBP method intends to construct a pseudo decision boundary for unknown classes. As the target domain is unlabelled during training, the domain classifier G_{C_1} is weakly trained to construct a pseudo decision boundary between known source samples and target samples by putting the target samples on the side of the unknown category. The OSBP method trains the domain classifier G_{C_1} to output $P(y = N+1|x_j^t) = T$ for 'unknown class'. Then the feature generator G_F is trained to deceive the domain classifier G_{C_1} adversarially. The feature generator G_F is trained with the ability to increase or decrease the 'unknown' class probability $P(y = N+1|x_j^t)$ of the classifier G_{C_1} for maximising the error of G_{C_1} and align target samples to known or unknown classes. Also, the OSBP method assumes that the empirical threshold value $T = 0.5$ dictates the generative model to construct a good boundary between known and unknown target samples. The OSBP method trains the domain classifier and the generator on source samples first as follows,

$$E_{G_{C_1}} = \frac{1}{n_s} \sum_{i=1}^{n_s} L_{G_{C_1}}(G_F(x_i^s), y_i^s). \tag{5.1}$$

Here, $L_{G_{C_1}}$ is the standard cross-entropy loss function for minimising the error of G_{C_1}. Then a binary cross-entropy loss is used for maximising the error of G_{C_1} adversarially to separate known and unknown target samples as follows,

$$E_{G_{C_{1adv}}} = -\frac{1}{n_t} \sum_{j=1}^{n_t} T \log(P(y = N+1|x_j^t)) - \frac{1}{n_t} \sum_{j=1}^{n_t} (1-T) \log(1 - P(y = N+1|x_j^t)). \tag{5.2}$$

The overall training objective of the OSBP method is,

$$\theta_{G_{C_1}} = \underset{\theta_{G_{C_1}}}{\operatorname{argmin}} E_{G_{C_1}} + E_{G_{C_{1adv}}}, \theta_{G_F} = \underset{\theta_{G_F}}{\operatorname{argmin}} E_{G_{C_1}} - E_{G_{C_{1adv}}}. \tag{5.3}$$

The training objective indicates that the domain classifier G_{C_1} tries to set 'unknown' class probability $P(y = N+1|x_j^t)$ equal to T, on the other hand the generator G_F tries to make $P(y = N+1|x_j^t)$ different from T for maximising the value of $E_{G_{C_{1adv}}}$. For calculating the gradient of $E_{G_{C_1 adv}}$ efficiently, the OSBP method utilises a gradient reversal layer proposed by [67].

Negative Transfer in the OSBP Method: The OSBP method does not evaluate any underlying domain level discerning characteristics but relies only on the pseudo decision of G_{C_1}. The lack of such domain knowledge and enforcement of an assumed boundary threshold $T = 0.5$ value for separating known and unknown classes will harm the model as the generator will attempt to align all the target samples with $P(y = N+1|x_j^t) < 0.5$ to known classes during training. As a result, the model will be deprived of the opportunity to learn such image features which should be recognised as unknown even when $P(y = N + 1|x_j^t)$ is less than 0.5. For example, during training, the domain classifier G_{C_1} assigns $P(y = N + 1|x_j^t) = 0.4$ for a target sample and the rest 0.6 of the softmax probability is distributed over other $1, 2, ..., N$ indices of G_{C_1} with no index holding $P \geq 0.4$. This probability outcome distribution suggests the sample should be aligned towards the unknown class. However, the generator will always find it easier to decrease $P(y = N + 1|x_j^t) < 0.4$ to maximise the error of G_{C_1} and align it towards known classes as it does not explore underlying domain knowledge before adversarial training. Thus, the model will be exposed to negative transfers during training and testing, i.e., target samples from unknown classes will be aligned to known classes.

To reduce the propensity for such negative transfers in the OSBP method and improve classification performance, we propose to extend their domain adversarial model. Our proposed multi-classifier based domain adversarial model is discussed in the next section.

5.3.2 Proposed Multi-classifier Based Domain Adversarial Model

The limitations of the OSBP method is mainly due to the lack of an indicator of the likelihood of a target sample belonging to known or unknown classes. This prompted us to design a method to produce the indicator of target samples belonging to known or unknown classes before the adversarial training to facilitate DA. The proposed method is illustrated in Figure 5.2b. To investigate underlying domain information of target samples for reducing negative transfers, we propose to integrate a multi-classifier based weighting module in the baseline network. Our proposed method comprises of two modules: 1) Adversarial module; and 2) Multi-classifier based weighting module.

5.3.2.1 Adversarial module

The adversarial module of our proposed method has a similar structure to the OSBP method. The first player of the model is a domain classifier G_{C_1} which is trained to distinguish the features of the source domain from the target domain. The second player is a feature generator G_F which is simultaneously trained to reduce feature distribution divergence in the opposite direction of the domain classifier. However, our proposed method follows a different adversarial optimisation procedure for distinguishing unknown target samples from the known target samples compared to the OSBP. The ultimate goal is to train a source domain classifier that is transferred to the target domain classifier with an extra category named 'unknown'.

The proposed weighting module quantifies each target sample with a generated weight $W(x_j^t)$ (Section 5.3.2.2), which is an encoding of the underlying discriminative domain information. In particular, prior to adversarial DA, we assign identifiable weights $W(x_j^t)$ to known and unknown target samples based on their similarity to source domain to facilitate the generator in deciding whether to decrease or increase the 'unknown' class probability $P(y = N + 1|x_j^t)$ for maximising the error of G_{C_1} and eventually align the target samples to known classes or 'unknown' class. Therefore, the generator does not have to draw an empirical threshold driven boundary between known-unknown target samples by depending only on the pseudo decision of the classifier, which may encourage negative transfer.

We use the standard cross-entropy loss optimisation, as illustrated in (5.1) for minimising the error of the domain classifier G_{C_1}. To distinguish known from unknown target samples and simultaneously maximise the error of the domain classifier G_{C_1} adversarially, we use a binary cross-entropy loss. We infuse our computed weight measure for the target samples in the optimisation procedure as follows,

$$\begin{aligned} E^{'}_{G_{C_1 adv}} &= -\frac{1}{n_t}\sum_{j=1}^{n_t} W(x_j^t)\log(P(y=N+1|x_j^t)) \\ -\frac{1}{n_t}\sum_{j=1}^{n_t}&(1-W(x_j^t))\log(1-P(y=N+1|x_j^t)). \end{aligned} \tag{5.4}$$

The mini-max game between the generator and domain classifier in the adversarial model is equivalent to aligning target samples towards known classes of the source domain or 'unknown' class based on their weights.

5.3.2.2 Proposed Multi-classifier based weighting module

In this section, we present the conceptual details of the weighting scheme of our proposed DA method.

Overview The main challenge in our proposed method, as illustrated in (5.1) and (5.4), is the way of measuring the probability of each target samples resembling the source domain on the basis of which the boundary between known and unknown classes is to be constructed. We aim to develop a weight measure $W(x_j^t)$ for each target samples based on their discriminative label and domain information. We propose to integrate a supplementary source classifier G_{C_2} in the adversarial model (Figure 5.2b) to determine the similarity of target samples to individual known-source labels C_s. The combined similarity to each known classes represents the similarity to the source domain and utilise the pseudo-decision of G_{C_1} to compute the combined similarity to all known classes (i.e., similarity to source domain) measured against target private labels $\overline{C_t}$. This ensures the exploitation of probable underlying label information of target samples concerning both known and unknown classes.

We further introduce a non-adversarial supplementary domain classifier G_D in the model to evaluate underlying domain information and produce a weight for target samples. The non-adversarial supplementary domain classifier G_D assumes that the target samples belonging to the shared label space C are closer to the source domain samples than $\overline{C_t}$. Being inspired by a prior work [186] that integrated the label information into the domain discriminator, we combine the domain similarity measure of G_{C_1} and G_{C_2}, and encode in G_D. Now that the supplementary domain classifier G_D holds the encoded information required to serve our purpose, G_D is jointly trained with G_{C_2} to distinguish the source domain samples from the target domain samples by utilising the Sigmoid probability of classifying each target sample (x_j^t) to the source domain. The output ($W(x_j^t)$) of G_D for target samples gives the probability of a sample belonging to the shared label space. This constitutes our multi-classifier based weighting module. The purpose of the weighting scheme is to assign either high or low weights to the target samples depending on their similarity to the source domain and reduce the chances of negative transfer. The generated weights are back-propagated to the adversarial module for optimising the generator G_F and the domain classifier G_{C_1} to construct boundary between known and unknown classes.

Mechanism in detail: We place the supplementary source classifier G_{C_2} to predict the source class labels with a leaky-softmax function [184], which maintains the total probability of less than 1. The supplementary source classifier G_{C_2} converts features of the generator G_F to $|C_s|$-dimensional class probabilities as follows,

$$\overline{\sigma}(l) = \frac{\exp{(l)}}{|C_s| + \sum_{c=1}^{|C_s|} \exp{(l_c)}} \tag{5.5}$$

where l is the logit vector. The parameters of G_{C_2} is trained only on the source samples; therefore, unlike source samples, target samples will have smaller logits or uncertain predictions. We define the probability of each sample belonging to the source domain based on known-source label information $d_1(x)$ as follows,

$$d_1(x) = \sum_{k=1}^{|C_s|} G_{C_2}^{k}(G_F(x)) \tag{5.6}$$

where, $G_{C_2}^{k}(G_F(x))$ is the probability of a sample belonging to the k^{th} known class. The element-sum ($d_1(x)$) of the leaky-softmax outputs for samples resembling the source domain will be high or close to 1 whereas, samples dissimilar to source domain will yield low or close to 0 outputs. That is, the higher the value of $d_1(x_j^t)$ is, the higher the chance that a target sample lies in the vicinity of shared label space C. On the other hand, the smaller the value of $d_1(x_j^t)$ is, the more probable that the target sample comes from $\overline{C_t}$. We train G_{C_2} by a multiclass one-vs-rest binary loss for the $|C_s|$-class classification as,

$$E_{G_{C_2}} = -\frac{1}{n_s}\sum_{i=1}^{n_s}\sum_{k=1}^{|C_s|} y_{i,k}^s \log G_{C_2}^k(G_F(x_i^s)) + (1-y_{i,k}^s)\log(1-G_{C_2}^k(G_F(x_i^s))) \tag{5.7}$$

where $y_{i,k}^s$ denotes the ground-truth label for source example x_i^s and the probability of each sample x belonging to class k is $G_{C_2}^k(G_F(x))$. This similarity measure defined so far is still exposed to risk as to the value of $d_1(x)$ for target samples can be uncertain. To further support the similarity measure, we compute the probability of a sample belonging to C when measured against the 'unknown' class probability from the domain classifier G_{C_1} as follows,

$$d_2(x) = (1 - P(y = N+1|x)). \tag{5.8}$$

Target samples residing in the shared label set are likely to produce higher $d_2(x)$ than unknown target samples. Now, we define our final similarity measure supported by G_{C_1} and G_{C_2} as,

$$G_D(G_F(x)) = (d_2(x))(d_1(x)). \tag{5.9}$$

Now, $G_D(G_F(x_j^t))$ can be seen as the complete measure of the likelihood of target samples belonging to shared label space C, i.e., for target samples, the higher the value of $G_D(G_F(x))$ is the more probable that it belongs to the shared classes.

For the convenience of understanding, we represent the possible cases of outcomes from the two deciding factors (i.e., $d_1(x_j^t)$ and $(d_2(x_j^t))$ for computing the similarity measure of the known and unknown target samples as follows:

$A_\mathbf{H}$: For the target samples belonging to the shared label space C, it is highly likely the output of $d_1(x_j^t)$ will be high.

$A_\mathbf{L}$: For the target samples belonging to the target private label space $\overline{C_t}$, it is highly likely the output of $d_1(x_j^t)$ will be low.

$B_\mathbf{H}$: For the target samples belonging to the shared label space C, it is highly likely the output of $d_2(x_j^t)$ will be high.

$B_{\mathbf{L}}$: For the target samples belonging to the target private label space $\overline{C_t}$, it is highly likely the output of $d_2(x_j^t)$ will be low. Note that, here, high means close to 1 and low means close to 0. In the cases below, (A, B) denotes the occurrence of A and B for measuring the similarity of a target sample to the source domain.

Case 1 (A_L, B_L**):** For this case, the computed similarity measure (5.9) will be low and this will assist the generator in deciding to increase the value of $P(y = N+1|x_j^t)$ for maximising the error of the domain classifier G_{C_1} and align the sample towards the 'unknown' class.

Case 2 (A_L, B_H**):** For this case, the computed similarity measure (5.9) will be low and this will assist the generator in deciding to increase the value of $P(y = N+1|x_j^t)$ for maximising the error of the domain classifier G_{C_1} and align the sample towards the 'unknown' class.

Case 3 (A_H, B_L**):** For this case, the computed similarity measure (5.9) will be low and this will assist the generator in deciding to increase the value of $P(y = N+1|x_j^t)$ for maximising the error of the domain classifier G_{C_1} and align the sample towards the 'unknown' class.

Case 4 (A_H, B_H**):** For this case, the computed similarity measure (5.9) will be high and this will assist the generator in deciding to decrease the value of $P(y = N+1|x_j^t)$ for maximising the error of the domain classifier G_{C_1} and align the sample towards the known classes.

We train the supplementary domain classifier G_D as follows,

$$E_{G_D} = -\frac{1}{n_s}\sum_{i=1}^{n_s}\log(G_D(G_F(x_i^s)) - \frac{1}{n_t}\sum_{j=1}^{n_t}\log(1 - G_D(G_F(x_j^t))). \tag{5.10}$$

Equations (5.9) and (5.10) indicate that the outputs of G_D are dependent on the output of the supplementary source classifier G_{C_2} and the output of the domain classifier G_{C_1}. This verifies that G_D is trained to evaluate target samples based on the discriminative known classes and unknown class label information, which will assist G_D to assign meaningful and identifiable weights to target samples belonging to C and $\overline{C_t}$. Thus, we obtain weights to quantify the similarity of target samples to the source domain from G_D as,

$$W(x_j^t) = G_D(G_F(x_j^t)). \tag{5.11}$$

During the early phase of training, if either one of the classifiers (G_{C_1}, G_{C_2}) produces uncertain similarity measure ($d_2(x)$, $d_1(x)$), it will be supported by the other ones' decision for assigning weights to the target samples. However, over the training epochs, both the classifiers will converge to their optimal value for the feature extractor G_F. In such an advanced phase of training, the target samples belonging to the shared label space will surely get close to 1 similarity score from G_{C_2} as it is trained on source samples only. Similarly, the similarity score for that target sample from G_{C_1} will also be high as G_{C_1} learns to yield low 'unknown' class probability for target samples belonging to C. Thus, the combined weight $W(x_j^t)$ will be high, and the sample will be aligned to the known classes. On the other hand, if a target sample comes from outside the shared label space, then the G_{C_2} will produce close to 0 similarity score. The G_{C_1} will produce low score as well for such sample, and eventually, the weight $W(x_j^t)$ will be low, and the sample will be aligned to the 'unknown' class. Section 5.4.6 provides pictorial representation of learned weights.

Considering all the above-discussed derivations, we present our proposed adversarial DA model with multi-classifiers. We denote the parameters of the supplementary source classifier G_{C_2} as $\theta_{G_{C_2}}$. The overall objectives of our proposed method are:

$$\begin{aligned} &\theta_{G_{C_1}} = \underset{\theta_{G_{C_1}}}{\operatorname{argmin}} E_{G_{C_1}} + E'_{G_{C_1 adv}}, \theta_{G_F} = \underset{\theta_{G_F}}{\operatorname{argmin}} E_{G_{C_1}} - E'_{G_{C_1 adv}}, \\ &\theta_{G_{C_2}} = \underset{\theta_{G_{C_2}}}{\operatorname{argmin}} E_{G_{C_2}} + E_D. \end{aligned} \tag{5.12}$$

During back-propagation, we use a gradient reversal layer [67] to calculate the gradient of $E'_{G_{C_1 adv}}$ efficiently. Unlike prior work [84], our proposed model does not need any prior training on the source dataset. We optimise all the objectives simultaneously in an end-to-end fashion.

5.3.2.3 Constraining positive transfer

In this section, we discuss how the proposed method limits the tendency of negative transfer in the OSBP model based on the case discussed in Section 5.3.1, which explains negative transfers in the OSBP model. In contrary to OSBP model, for example, when the domain classifier assigns $P(y = N + 1|x_j^t) = 0.4$ for a target sample, the G_D in our proposed method will generate a weight $W(x_j^t)$ for the sample after evaluating its similarity to the source domain based on (5.5 - 5.11). In short, we compute the value of $d_2(x_j^t)$ (5.6) and $d_1(x_j^t)$ (5.6). The former one is 0.6 in this case (we consider this value as a high as it is closer to 1 than 0), and the latter one can be either high and yield high weight $W(x_j^t)$ or low leading to low weight $W(x_j^t)$ based on (5.11). If the weight $W(x_j^t)$ is high, the proposed model will assist the generator in aligning the sample to known classes by decreasing $P(y = N+1|x_j^t)$. Otherwise, if the weight $W(x_j^t)$ is low, the sample will be aligned to 'unknown' class by maximising the value of $P(y = N + 1|x_j^t)$. Thus, our DA model does not participate in negative transfer by aligning unknown samples to known classes.

5.3.2.4 Testing Phase

During the training phase, we fulfill our goal to transform the domain classifier G_{C_1} from source domain classifier to target domain classifier, including the category 'unknown' by utilising G_{C_1} and G_D classifiers. In the testing phase, we omit the supplementary classifiers and utilise only the trained feature generator G_F, and G_{C_1} to classify test images correctly, as shown in Figure 5.2c.

5.4 Experimental Studies

In this section, we describe the datasets, our evaluation details, and the results. We conduct experiments to evaluate our proposed method with contemporary DA methods on four standard datasets.

5.4.1 Datasets

Office-31 [57] has 31 categories in three visually distinct domains, namely: amazon (**A**), DSLR (**D**) and webcam (**W**). This dataset comprises a collection of samples from **amazon.com**, captured samples from DSLR and web camera for DA. We have chosen the first 10 classes as C and the last 10 classes as 'unknown' samples in the target domain $\overline{C_t}$ for accomplishing six open set DA tasks: A→ W, D→ W, W→ D, A→ D, D→ A and W→ A.

VisDA2017 [187] poses a special DA setting by focusing on a simulation (rendered 3D images) to real-world DA setting. Game engines generate the samples of source domain while the target domain samples are actual images. This dataset comprises 12 categories. Inline with [84], we have chosen six classes (bicycle, bus, car, motorcycle, train, and truck) as the shared classes C and the remaining six classes as 'unknown' classes in the target domain.

Office-Home [188] consists of 65 classes in four different domains: Artistic images (Ar), Clip-Art images (Cl), Product images (Pr), and Real-World images (Rw). The first 10 classes in alphabetical order are used as the shared classes C. Leaving the next five classes private to the source domain, the rest classes are considered as 'unknown' or private to the target domain. For this dataset, we have designed 12 open set DA tasks: Ar→ Cl, Ar→ Pr, Ar→ Rw, Cl→ Ar, Cl→ Pr, Cl→ Rw, Pr→ Ar, Pr→ Cl, Pr→ Rw, Rw→ Ar, Rw→ Cl and Rw→ Pr.

ImageNet-Caltech is a combination of ImageNet-1K [175] consisting 1000 categories and Caltech-256 with 256 categories. In line with previous works [105,184], we have used the common 84 classes as the known or shared classes C and have used the remaining classes as the 'unknown' class in the target domain. We have performed two open set DA tasks I→ C and C→ I for this dataset.

5.4.2 Evaluation Details

Evaluation Protocols. In this paper, we have followed the evaluation protocol of the Visual DA (VisDA 2018) Open-Set Classification Challenge. This protocol assumes all the target domain private classes $|\overline{C_t}|$ as a unified 'unknown' class and the average per-class accuracy for all the $|C|+1$ classes is the final result. Also, being inspired by [57], we present the normalised classification accuracy measured on all the known classes and the 'unknown' classes ($|C|+1$) as OS, and the normalised classification accuracy only on the shared classes ($|C|$) as OS*.

Implementation Details. We have used ImageNet pre-trained ResNet-50 and ResNet-152 [4] with new fully-connected and batch normalisation layers as the feature generator. We have used SGD with a learning rate of 0.001 for pre-trained layers, 10 times higher than that for new layers, and momentum of 9. Note, while the original papers show results on a variety of backbone networks such as VGG, AlexNet, and ResNet-50, for the sake of fairness and consistency we tested them all using ResNet-50. Also, we have executed up to 1000 epochs for training, but if any of the contemporary methods converged earlier than that, we stopped the training. The performance difference between our implementation of the contemporary DA methods and the original papers is mainly due to different backbone networks and the number of iterations. Note that the results cannot be directly compared against other publicly reported results due to different train-test data split and versions of PyTorch.

Compared DA Methods. For the sake of thoroughness, we have compared the performance of the proposed method with: **1)** Classifier without DA: ResNet [4] (Note that Negative transfer is calculated against this non-DA classifier.); **2)** Closed-set DA methods: Domain-Adversarial Neural Networks (DANN) [54] and Residual Transfer Networks (RTN) [50]; **3)** Partial DA methods: Importance Weighted Adversarial Nets (IWAN) [185] and Example Transfer Network (ETN) [184]; **4)** Open set DA methods: Unsupervised DA by back-propagation (BP) [67] with unknown source samples, Assign-and-Transform-Iteratively (ATI) [81], OSDA by Back-Propagation (OSBP) [84] and Separate to Adapt (STA) [85]; **5)** Universal DA method: Universal Adaptation Network (UAN) [107].

5.4.3 Classification results

The classification results on the 12 tasks of Office-Home, 6 tasks of Office-31, the task of VisDA and 2 large-scale tasks of ImageNet-Caltech are shown in Tables 5.1, 5.2, 7.3

Table 5.1: Classification accuracy (%) of proposed and contemporary DA methods on Office-Home dataset tasks.

Approach	Accuracy (%)																									
	Ar→ Cl		Ar→ Pr		Ar→ Rw		Cl→ Ar		Cl→ Pr		Cl→ Rw		Pr→ Ar		Pr→ Cl		Pr→ Rw		Rw→ Ar		Rw→ Cl		Rw→ Pr		Avg	
	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*
ResNet [4] (2016)	54.4	54.8	69.5	70.1	78.7	78.1	62.0	61.3	60.8	62.3	71.6	72.5	64.2	64.4	58.9	58.6	75.5	76.1	70.3	69.2	52.5	51.5	74.5	75.3	66.1	66.2
DANN [54] (2016)	-	*44.8*	-	*68.5*	-	79.5	-	65.5	-	*57.9*	-	*67.4*	-	*56.9*	-	*40.2*	-	77.5	-	*68.5*	-	*45.2*	-	77.6	-	*62.4*
RTN [50] (2016)	-	*50.9*	-	75.6	-	82.9	-	66.5	-	73.4	-	85.7	-	65.6	-	*47.9*	-	84.5	-	78.1	-	56.9	-	77.6	-	70.4
IWAN [185] (2018)	*53.1*	*52.1*	79.4	78.5	86.1	86.4	70.2	69.7	70.9	71.3	86.8	85.1	74.9	74.5	*55.6*	*55.7*	85.1	84.2	77.9	78.7	60.8	59.4	77.2	76.8	73.1	72.7
ETN [184] (2019)	59.3	59.0	77.1	76.8	85.6	85.7	63.1	62.9	65.6	65.1	75.3	75.6	68.3	67.8	*55.4*	*55.6*	86.4	85.9	78.7	77.5	62.3	61.5	84.4	84.2	71.8	71.4
UAN [107] (2019)	63.0	62.5	82.8	82.4	86.8	85.9	76.8	76.9	78.7	79.1	84.4	84.8	78.2	77.4	*58.6*	*57.8*	86.8	85.9	83.4	82.5	63.2	63.0	79.1	78.1	76.8	76.6
BP [67] (2015)	*53.6*	*51.0*	*69.1*	*65.9*	*75.9*	*74.1*	59.5	57.3	65.2	62.5	73.2	*72.0*	*47.2*	*45.0*	*43.9*	*40.2*	78.7	76.4	70.6	*65.3*	*45.6*	*42.1*	77.5	*74.2*	*63.3*	*60.5*
ATI [81] (2017)	*53.8*	*51.3*	80.4	77.9	86.1	85.0	71.2	67.8	72.3	70.5	85.1	83.2	74.3	72.5	*57.9*	*55.1*	85.6	84.7	76.1	75.2	60.2	58.7	78.3	77.0	73.4	71.5
OSBP [84] (2018)	*48.5*	*48.6*	70.9	70.6	*75.2*	*74.2*	59.5	58.2	61.6	*59.9*	75.1	74.5	*61.9*	*62.2*	*43.5*	*43.2*	79.9	80.4	*70.1*	70.2	53.9	54.1	75.7	75.4	*64.6*	*64.3*
STA [85] (2019)	57.8	58.1	71.3	70.1	84.9	85.5	*61.4*	61.9	68.1	67.9	75.2	75.8	64.3	*63.2*	*51.8*	*52.2*	80.2	79.1	73.9	74.2	53.6	54.5	80.5	81.4	68.6	68.7
Our(ResNet-50)	64.8	64.9	84.6	84.9	**88.1**	**88.2**	79.6	79.9	81.6	82.9	85.6	86.8	77.9	78.5	61.8	63.1	85.9	87.5	85.4	85.6	67.5	65.2	80.9	80.1	78.6	79.0
Our(ResNet-152)	**66.1**	**66.9**	**85.9**	**86.7**	87.6	87.9	**80.5**	**80.8**	**83.5**	**85.6**	**86.9**	**87.9**	**79.9**	**81.1**	**63.1**	**64.8**	**87.9**	**88.7**	**86.9**	**88.2**	**68.8**	**69.7**	**82.1**	**82.8**	**80.0**	**80.9**

Table 5.2: Classification accuracy (%) of proposed and other DA methods on Office-31 tasks.

Approach	Accuracy (%)													
	A→ W		D→ W		W→ D		A→ D		D→ A		W→ A		Avg	
	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*
ResNet [4] (2016)	83.2	83.8	93.8	94.7	95.8	94.6	84.6	84.7	72.3	71.9	75.5	75.3	84.2	84.2
DANN [54] (2016)	-	*80.2*	-	*79.9*	-	*87.5*	-	*80.4*	-	74.9	-	80.9	-	*80.6*
RTN [50] (2016)	-	88.1	-	*89.8*	-	*84.8*	-	*73.1*	-	85.1	-	84.7	-	84.2
IWAN [185] (2018)	86.5	84.9	*89.9*	*87.1*	*91.1*	*90.3*	*82.3*	*79.6*	82.2	80.6	85.7	83.1	86.3	84.3
ETN [184] (2019)	85.7	84.4	*93.6*	*92.9*	96.9	96.0	84.9	85.1	84.9	85.6	85.2	85.0	88.5	88.1
UAN [107] (2019)	85.6	84.3	94.7	*93.9*	97.9	97.5	86.5	84.9	85.5	85.6	85.1	84.6	89.2	88.4
BP [67] (2015)	*75.9*	*74.1*	*89.7*	*87.2*	*94.4*	*93.2*	*78.4*	*76.8*	*56.8*	*55.3*	*62.9*	*62.7*	*76.3*	*74.8*
ATI [81] (2017)	*78.4*	*74.9*	*92.6*	*90.6*	97.1	95.6	*78.9*	*77.5*	*71.6*	*70.1*	76.8	*74.2*	*82.5*	*80.4*
OSBP [84] (2018)	*67.4*	*66.9*	*83.7*	*83.5*	*95.1*	94.8	*82.6*	*82.0*	76.6	76.5	79.5	78.9	*80.8*	*80.4*
STA [85] (2019)	88.6	90.1	97.1	95.2	97.3	97.5	**91.9**	**93.1**	88.3	88.6	84.1	84.2	91.2	91.4
Our(ResNet-50)	88.3	88.4	97.3	97.8	98.1	98.4	87.8	88.6	89.9	89.6	84.9	85.8	91.1	91.5
Our(ResNet-152)	**90.2**	**90.8**	**97.9**	**98.8**	**98.6**	**98.8**	89.5	89.7	**91.0**	**91.7**	**86.8**	**87.6**	**92.4**	**92.9**

and 5.4 respectively ('-' indicates that results could not be regenerated because of closed set DA setting and negative transfer for DA classifiers against the non-DA classifier ResNet [4] (shown in gray background) are indicated by showing the classification accuracy in italic). Our proposed method outperforms all the compared methods in terms of the average per-class accuracy except for Office-31 dataset, where STA [85] leads by 0.1%. However, we have better OS* than STA for Office-31 dataset. We observe that all contemporary partial, universal, and open set DA methods lag behind ResNet classifier [4] on some tasks because of negative transfer during adaptation. This negative transfer is the effect of the difference in source and target domain label space introduced by unknown classes. In addition, the accuracy of the OS* is lower than that of OS for the majority of the tasks, which means a large number of unknown images are misclassified.

Table 5.3: Classification accuracy (%) of proposed and other DA methods on ImageNet-Caltech tasks.

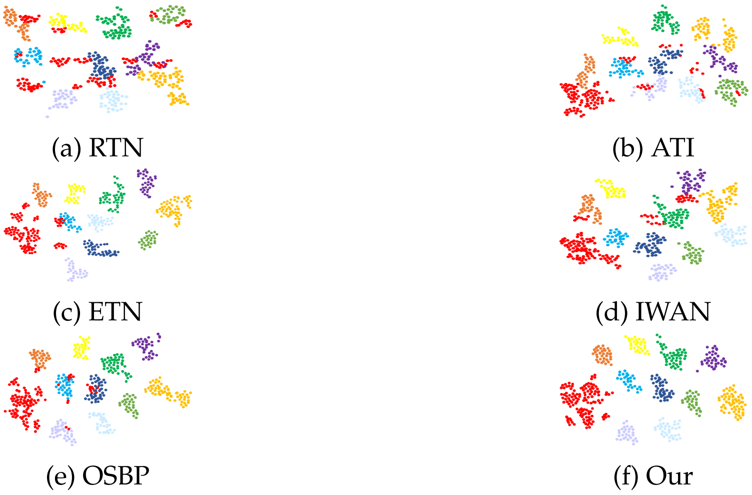
Approach	Accuracy (%)					
	I→ C		C→ I		Avg	
	OS	OS*	OS	OS*	OS	OS*
ResNet [4]	75.7	75.1	67.1	67.8	71.4	71.5
DANN [54] (2016)	-	*71.1*	-	*65.9*	-	*68.5*
RTN [50] (2016)	-	*71.9*	-	*66.2*	-	*69.1*
IWAN [185] (2018)	*74.1*	*72.9*	68.7	*65.3*	71.4	*69.1*
ETN [184] (2019)	74.9	*74.8*	69.8	69.9	72.4	72.4
UAN [107] (2019)	*75.3*	76.3	70.2	70.8	72.7	73.6
BP [67] (2015)	*68.9*	*67.3*	*61.2*	*59.0*	*65.0*	*63.2*
ATI [81] (2017)	*71.6*	*65.9*	67.4	*65.1*	*69.5*	*65.5*
OSBP [84] (2018)	*63.1*	*63.4*	*54.8*	*53.6*	*58.9*	*58.5*
STA [85] (2019)	75.3	*74.2*	68.1	68.3	71.7	*71.3*
Our(ResNet-50)	77.4	77.8	69.8	70.1	73.6	74.0
Our(ResNet-152)	**78.9**	**79.7**	**71.9**	**71.7**	**75.4**	**75.7**

Table 5.4: Classification accuracy (%) of proposed and other DA methods on VisDA2017 tasks.

Approach	Accuracy (%)								
	bicycle	bus	car	motorcycle	train	truck	unknown	OS	OS*
ResNet [4] (2016)	40.2	55.4	63.5	70.8	74.1	35.2	45.6	54.9	56.5
DANN [54] (2016)	*32.4*	*51.6*	65.1	71.3	85.1	*23.1*	-	-	*52.1*
RTN [50] (2016)	*31.6*	63.6	*54.2*	76.9	**87.3**	*21.5*	-	-	*51.1*
IWAN [185] (2018)	*30.6*	69.8	*58.3*	76.8	*65.5*	*30.8*	69.7	57.3	*55.3*
ETN [184] (2019)	*31.6*	66.8	*61.7*	77.8	*70.8*	*30.8*	70.7	58.6	56.6
UAN [107] (2019)	42.6	67.8	65.7	76.9	*69.8*	*31.8*	70.7	60.9	59.1
BP [67] (2015)	*31.8*	66.5	*50.5*	70.1	86.9	*21.8*	*38.5*	*52.3*	*54.6*
ATI [81] (2017)	*33.6*	*51.6*	64.2	78.1	85.3	*22.5*	*42.5*	*54.8*	*52.6*
OSBP [84] (2018)	*35.6*	59.8	*48.3*	76.8	*55.5*	*29.8*	81.7	55.4	*50.9*
STA [85] (2019)	50.1	69.1	*59.7*	**85.7**	84.7	*25.1*	**82.4**	65.3	62.4
Our(ResNet-50)	50.6	74.8	66.7	80.6	75.9	38.8	73.9	65.9	64.6
Our(ResNet-152)	**52.1**	**77.7**	**67.7**	81.4	80.8	**39.8**	75.5	**67.8**	**66.6**

Our proposed method yields better results for OS* than OS which indicates a better separation of known and unknown target samples.

DA methods such as BP (with unknown classes in the source domain) [67] and ATI [81] are trained to align target domain towards source domain employing distribution matching methods. During this process, the unknown source or target samples disturb the known class feature alignment leading to such performance degradation. The performance lag in OSBP [84] method compared to ResNet backbone for some tasks supports our claim. That is, the OSBP incurs negative transfer because it tends to align some unknown target samples to known classes by enforcing empirical boundary threshold, and not exploring underlying domain information of target samples. However, because of adapting sample-level transferability, IWAN [185], ETN [184], UAN [107], and STA [85] have lower negative transfer rate compared to other existing methods.


(a) RTN

(b) ATI

(c) ETN

(d) IWAN

(e) OSBP

(f) Our

Figure 5.3: Learned features of our proposed method for **A** → **W** task from **Office-31** dataset show a better separation of unknown samples (red dots) from known classes than contemporary DA methods. We select 10 shared classes, 10 source domain private classes, and 10 target domain private classes for the task. Visualisation prepared by the t-SNE algorithm [172] with the Perplexity parameter set to 50.

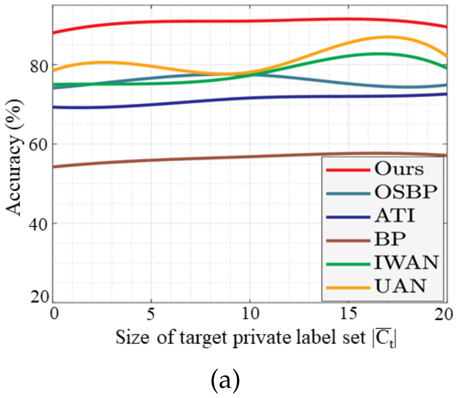
In Figure 5.3, we plot the t-SNE [172] embeddings of the features learned by RTN [50], ATI [81], IWAN [185], ETN [184], OSBP [84] and proposed method on **A** $\rightarrow$ **W** task with 10 shared classes, 10 source domain private classes and 10 target domain private classes as per respective DA settings. The proposed method demonstrates significantly comprehensible and well-segregated clusters for all known and unknown classes than other DA methods. This distinct separation of unknown target samples from the known ones is because of the supplementary source classifier, which is trained only on the source samples to learn discriminative features for known classes. Unlike OSBP [84], we do not need any prior training on the source domain to learn discriminative known class features to support better classification. On the other hand, RTN [50] and ATI [81] methods which utilise distribution matching techniques such as MMD only tries to align target samples with the source ones and do not separate well among known and

Table 5.5: Accuracy (%) of proposed method on VisDA2017, Office-31, and ImageNet-Caltech tasks for ablation study.

	Accuracy (%)										
Approach	VisDA	A→ W	D→ W	W→ D	A→ D	D→ A	W→ A	Avg	I→ C	C→ I	Avg
w/o d_2	63.3	86.1	93.1	96.2	84.2	86.5	82.1	88.0	75.3	66.4	70.9
w/o d_1	61.1	84.8	93.8	94.7	85.1	84.7	82.3	87.6	74.6	65.0	69.8
Our(ResNet-50)	65.9	88.3	97.3	98.1	87.8	89.9	84.9	91.1	77.4	69.8	73.6

unknown classes. Though ETN [184] shows better clusters than IWAN [185] for initiating less negative transfers, it certainly lags behind our method.

5.4.4 Analysis on Various Open Set DA Settings



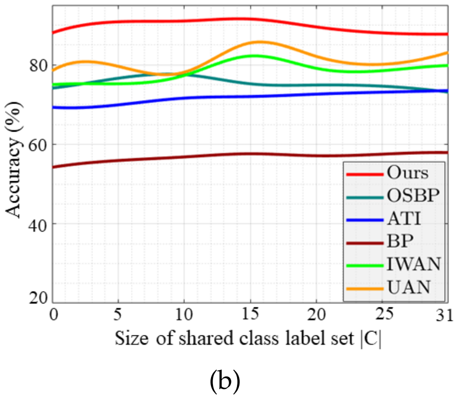


Figure 5.4: (a) The proposed method consistently performs better than contemporary DA methods for all cases of $|\overline{C_t}|$. The performance of our proposed method increases with an increase of the number of unknown classes in the target domain $|\overline{C_t}|$. (b) Our proposed method outperforms contemporary DA methods with different DA settings for different sizes of the shared label set. We observe that when $|C|$ reaches beyond 20, the performance of our method decreases slightly. This indicates our proposed method is more suitable for tasks that have more 'unknown' classes in the target domain.

Varying Size of $\overline{C_t}$. We compare the performance of our proposed method with other methods by varying the number of unknown samples in the target domain for D→ A task. Figure 5.4a shows that our proposed method maintains moderate increment of performance with no significant drop in-between transitions of varying $|\overline{C_t}|$ and outperforms all compared DA methods for all cases. This indicates a larger number of unknown classes in the target domain $|\overline{C_t}|$ compared to the shared classes assists both the source G_{C_1} and supplementary source G_{C_1} classifiers, by initiating less distraction and can lead to solving more realistic tasks where unknown source samples are unavailable. Note that our proposed method does not take advantage of any prior knowledge about the label sets like OSBP [84] and IWAN [185].

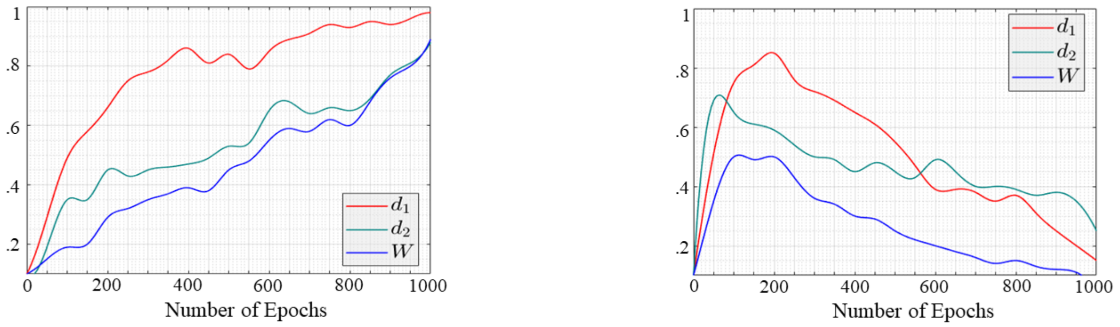
Varying Size of *C*. We further explore the performance of the proposed method by varying the number of shared classes for the same task D→ A and compare it with other methods (Figure 5.4b). The proposed method maintains high accuracy with a slight drop when $|C| > 20$. The evaluation of the task shown here is a sub-task of the Office-31 dataset. The office-31 dataset has 31 categories when the shared label set C is beyond 20 the target private label set $|\overline{C_t}|$ decreases to less than 10. The proposed method is designed to handle well the tasks which have a large number of unknown classes in the target private label space. Therefore, the increment of shared label space, which causes decrement of target private label space in a large proportion, harms the model. However, the proposed method substantially outperforms other compared methods for all cases of $|C|$.

5.4.5 Ablation Study

We execute an ablation study for evaluating two variants of the proposed DA method with the multi-classifier based weighting module to investigate deeper into its effectiveness. **w/o d$_2$** is the variant of proposed method without integrating the outcome of $d_2(x)$ (domain information) in the procedure of weighting target samples, i.e., in (5.9) and (5.10). **w/o d$_1$** is the variant without integrating the known-source label information into the weighting mechanism by omitting G_{C_2} and deploying G_D to depend only on the value of $d_2(x)$ for source and target samples. To execute this variant, we need to omit (5.5) and (5.7), omit $d_1(x)$ in (5.9) and (5.10).

Table 5.5 presents the results for the variants of the proposed method, as mentioned above. The performance of both **w/o d$_2$** and **w/o d$_1$** lag behind the proposed method. This is because both the deciding factors $d_1(x_i^t)$ and $d_2(x_i^t)$ are required for defining meaningful weights $W(x_j^t)$ to the target samples. We also observe that the variant without **w/o d$_2$** achieves better average per-class accuracy than the other one for all three tasks, which indicates integrating the supplementary source classifier G_{C_2} in the weighting module for exploiting label information is more effective.

5.4.6 Weight Visualisation



(a)

(b)

Figure 5.5: Pictorial representation of learned weights of (a) known target samples and (b) unknown target samples for the task D→ A (Office-31). Here, W represents the weights assigned to target samples during training by G_D after assessing the two similarity measures d_1 and d_2.

To analyse the trend of generated weights, we plot the learned similarity measures (d_1 and d_2) and final weights (W) of known and unknown target samples against the training epochs in Figure 5.5. Figures 6.7a and 6.7b show that our proposed method assigns sufficiently high and low weights (W) to known and unknown target samples, respectively. It is evident that such weights will assist the adversarial module in enhancing positive transfer by better separation of known target samples from unknowns. During the early stage of learning, both d_1 and d_2 increases for known and unknown samples which means the classifiers give uncertain decisions. However, after some initial epochs, for known samples, d_1 seems to increase at a larger pace and learns to yield a much higher value than d_2. This is because the classifier G_{C_2} is trained only on source samples. On the other hand, for unknown samples, both the similarity measures start to decrease. When both G_{C_2} and G_{C_1} converge to their optimal value for G_F, d_1, d_2 and W for known target samples reaches near 0.98, 0.88 and 0.86 respectively. Which means the target samples get very high weights as the training proceeds and dictate

G_F to decrease the 'unknown' class probability for aligning them to known classes. For unknown target samples, d_1, d_2 and W decreases to 0.15, 0.25 and 0.04 respectively. This indicates the target samples get very low weights and assists G_F to increase the 'unknown' class probability and align them to unknown classes. It is worth mentioning, the final value of d_1, d_2 and hence W may differ a little based on the dataset or task.

5.5 Summary

In this chapter, we have proposed a multi-classifier based domain adversarial network for an OSDA setting, where the target domain has a larger number of classes than the source domain. The multi-classifier structure incorporates a weighting module that explores discriminative label and domain information, and assigns distinguishable scores to the known and unknown target samples for enhancing positive transfer and eventually, assisting the feature generator and the domain classifier to separate known and unknown target samples. Another noteworthy attribute of our proposed method is the ability to discover the boundary between shared label space and target private label space automatically. A thorough experimental evaluation has demonstrated that the proposed method consistently outperforms the existing DA methods. We have further shown through the ablation study that the two deciding factors for generating weights for target samples are crucial for maintaining the integrity of the model and initiating positive sample transfer.

This chapter has discussed in detail our third contribution. In the next chapter, we discuss our fourth contribution, i.e., utilise mutual information concept from the information theory to mitigate negative transfers from OSDA.

Chapter 6

Reducing Negative Transfers in OSDA by Mutual Information

[1]In the previous chapter, we introduced our proposed multi-classifier based adaptive weighting scheme to reduce negative transfers and improve OSDA performance. This chapter examines how to mitigate negative transfers from OSDA by utilising mutual information (I), thereby fulfilling the fourth contribution of this thesis.

In some cases, current OSDA methods are vulnerable to negative transfers due to deficient known-unknown target sample separation modules. In this chapter, we propose another novel approach to OSDA, Domain Adaptation based on Mutual Information (DAMI) to address this problem. For learning a better separating module, we propose optimising *Mutual Information* to increase shared information between source and known target samples and decrease shared information between source and unknown target samples. Then, we propose to use *Point-wise Mutual Information* to compute the similarity between the source and target samples. A weighting module in DAMI utilises the *mutual information* optimisation and *point-wise mutual information* to execute *coarse-to-fine* separation of the known and unknown target samples. An adversarial DA module in DAMI uses the weighting module for adapting known target samples towards the source domain. The weighting module limits negative transfer by step-wise evaluation and verification. We demonstrate that DAMI is robust to various openness levels, performs

[1] Chapter 6 is adapted from: **T. Shermin**, G. Lu, S. W. Teng, M. Murshed, and F. Sohel, "Adversarial Open Set Domain Adaptation based on Mutual Information," Revision Submitted to IEEE Transactions on Image Processing, 2021 [arXiv: 2007.00384v1].

well across significant domain gaps, and remarkably outperforms contemporary DA methods on several benchmark datasets.

6.1 Overview

Like in Chapter 5, we focus on OSBP DA setting (Figure 5.1d) in this chapter. To mitigate negative transfer, the OSDA model must first separate the unknown and known target samples and then only align the known target samples to the source domain. Many authors have relied only on the domain confidence of a source trained classifier for separating known-unknown target samples [85–87, 189]. Such separating classifiers do not have knowledge about the target domain and misclassify unknown target samples as known target samples. Consequently, DA performance degrades. Clearly, an OSDA model with a faulty known-unknown target samples separation module is vulnerable to negative transfer and DA performance.

To address the issue of negative transfer and improve OSDA performance, we propose an adversarial OSDA model. We refer to our proposed OSDA model as Domain Adaptation based on Mutual Information (DAMI). DAMI has a new three-step *coarse-to-fine* known-unknown separation weighting module and a domain adversarial network (DAN). First, the weighting module assigns distinguishable weights to known and unknown target samples by exploiting distinctive domain information and mutual dependence between domains. Then based on the generated weights, the DAN performs adversarial domain adaptation by aligning known target samples towards known source samples and recognises the unknown target samples as 'unknown'. Unlike existing works [85–87, 189], our weighting module do not rely only on the domain confidence. First, it coarsely separates known-unknown target samples based on underlying discriminative domain information. Then it simultaneously maximises *mutual information* I between the source samples and the preliminary known target samples and minimises I between the source samples and the preliminary unknown target samples. This optimisation reduces the uncertain predictions for out-of-the-source distribution samples. Specifically, it forces the preliminary unknown target samples to move away from the source domain and pull the preliminary known target samples closer to the source domain. After this optimisation, the weighting module gains better knowledge about target samples similar and dissimilar to the source domain. Consequently, the module learns richer known-unknown discriminative features. Finally, the weighting module uses the richer features for fine separation and assigns distinguishable weights to the known-unknown target samples. Fine separation is performed by evaluating the *pointwise mutual information* (*pmi*) between each target sample and the source domain. The known target samples will exhibit strong statistical relation/dependence with the source samples compared to the unknown target samples. Thus, *pmi* score for known target samples will be higher than unknown samples making the known-unknown separation more accurate. The three-step assessment ensures that our model encourages improved separation of known-unknown target samples. As a result, DAMI delivers a better adaptation of known target samples to the source domain compared to contemporary works. DAMI also outperforms the non-DA classifiers, which indicates the mitigation of negative transfers.

For different DA settings, existing DA works based on I either minimises or maximises I [95–99]. However, DAMI simultaneously maximises and minimises I for OSDA. For optimising I, existing works use adversarial approximation of the variational upper bound and contrastive lower bound [97], Kullback–Leibler divergence [98], Jensen-Shannon I estimator [99, 190], and Barber & Agakov lower bound [191] of I [100]. DAMI adopts a neural estimator (MINE [192]) compatible with GANs for optimising I. MINE produces tractable estimators, which have tighter bounds than [191] with the optimal critic [193]. MINE has not been exploited for simultaneous minimisation and maximisation of I. *pmi* is a measure of co-occurrence. In prior research, *pmi* is used for sentiment classification, speech tagging, and other DA tasks in Natural Language Processing (NLP) [101, 104]. However, to the best of our knowledge, we are the first to exploit *pmi* in OSDA for image recognition.

The key contributions of this chapter can be summarised as follows:

- We propose a novel OSDA model that has a three-step *coarse-to-fine* weighting module for separating known-unknown target samples to limit negative transfer and an adversarial DA module.

- The proposed weighting module does not only rely on domain confidence. It optimises *I* between known-unknown target samples and the source domain to learn better known-unknown discriminative features and ensure better separation. For fine separation, it uses *pmi* score between the source and target samples. This facilitates in generating distinguishable instance-level weights for known and unknown target samples.

- We present an extensive empirical evaluation on several benchmark datasets to demonstrate the superior state-of-the-art performance of DAMI compared to contemporary methods. We also show that DAMI maintains desirable performance on different levels of openness and domain gaps.

6.2 Proposed Method

The proposed method DAMI shown in Figure 6.1 comprises of two modules: 1) Weighting module to separate known-unknown target samples, and 2) Domain Adversarial Network (DAN). The weighting module uses domain confidence and mutual domain dependence to generate identifiable weights for the known-unknown target samples. This will encourage correct separation and mitigate negative transfers. Then, the generative adversarial DA module, DAN, uses the weights to align the known target samples to the source domain and recognises the unknown target samples as 'unknown'. A conceptual overview of DAMI is shown in Figure 6.2.

6.2.1 Weighting Module to Separate Known-Unknown Target Samples

In this section, we present the weighting module of DAMI in detail. The weighting module aims to assign an instance-level weight $w(x_j^t)$ to every target sample in three steps. The three steps of the *coarse-to-fine* weighting module are (1) Coarse Separation

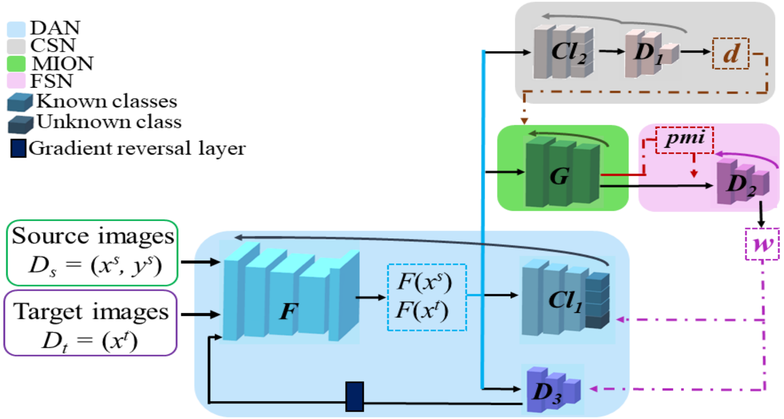


Figure 6.1: Block diagram of DAMI. The Domain Adversarial Network (DAN) comprises the feature extractor F, main classifier Cl_1, and adversarial domain discriminator D_3. The Weighting module is constructed with a classifier Cl_2, two non-adversarial binary discriminators D_1 and D_2, and a feature discriminator G. The distinguishable weights w for known and unknown target samples are generated after three-step evaluation through (1) Coarse Separation Network (CSN); (2) Mutual Information Optimisation Network (MION); and (3) Fine Separation Network (FSN). The DAN uses w to perform the OSDA task. The right arrows denote forward passes; the dotted arrows represent the similarity measures in different steps, and the left arrows represent backward passes.

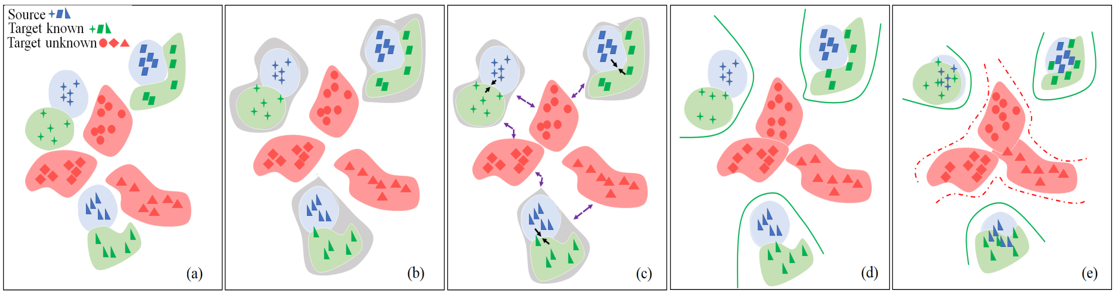


Figure 6.2: A conceptual overview of the proposed DAMI. (a) A view of the initial state of the OSDA setting. (b) CSN roughly determines the known-unknown target samples. (c) MION pushes the unknown classes away from known classes (purple arrows) and pulls known target classes towards the known source classes(black arrows). (d) FSN utilises the latent space created between the known and unknown domain in the MION to finely separate known-unknown target samples using *pmi* and generate distinguishable weights w. (e) Finally, DAN uses w to align known target samples to source domain and unknown target samples as 'unknown'.

Network (CSN), (2) Mutual Information Optimisation Network (MION), and (3) Fine Separation Network (FSN).

Overview: First, the CSN uses domain confidence to coarsely separate known-unknown target samples. The domain confidence is computed by evaluating the underlying domain similarity of the target samples to the source domain. Since this rough separation is dependent only on the domain confidence of a classifier, it is not satisfactory for reducing negative transfers. Then, to learn better discriminative information for known and unknown target samples, we impose the knowledge of the similarity and dissimilarity of the target samples to the source domain in the MION. MION uses the decision of CSN to simultaneously maximise *I* between preliminary known target samples and source samples and minimise *I* between preliminary unknown target samples and source samples. Hence, MION drives preliminary unknown target samples away from the source domain and pulls preliminary known target samples closer to the source domain (see Figure 6.2c). Consequently, MION better understands discriminative information for dissimilar source-target classes and common information for similar source-target classes. Finally, FSN uses the richer features of MION for fine separation and generates distinguishable weights to the known-unknown target samples. Fine separation is performed by evaluating the *pmi* score between every target sample and the source domain. Since MION optimises *I*, computing the *pmi* score depending on MION will lead to finer separation (see Figure 6.2d). The target sample of a source-target pair with a high *pmi* score will be assigned higher weights and vice-versa.

CSN: For coarse separation, we determine the underlying domain similarity between the source and target samples. To accomplish this, we place a multi-class one-vs-rest source classifier Cl_2 (Figure 6.1).We aim to estimate the similarity of a target sample to the source classes based on the domain confidence of Cl_2. The Cl_2 takes features from the feature generator F and determines the similarity of target samples to individual source labels C_s. We utilise a leaky softmax [184] and binary cross-entropy loss to optimise Cl_2 as follows,

$$E_{Cl_2} = -\frac{1}{n_s}\sum_{i=1}^{n_s}\sum_{c=1}^{|C_s|} y^s_{i,c}\log(Cl_2^c(F(x_i^s))) + (1-y^s_{i,c})\log(1-Cl_2^c(F(x_i^s))), \tag{6.1}$$

where $y^s_{i,c}$ denotes the ground-truth label for the i^{th} source sample, which belongs to the c^{th} known class. $Cl_2^c(F(x_i^s))$ is the probability of a sample belonging to the

c^{th} known class. As Cl_2 is trained only on the source domain $Cl_2^c(F(x))$ represents the similarity between the target sample and the c^{th} known class. The element-sum ($E_{sum}(x) = \sum_{c=1}^{|C_s|} Cl_2^c(F(x))$) of the leaky-softmax outputs will be high or close to 1 for target samples resembling the source domain. However, it will low or close to 0 for samples dissimilar to the source domain. Thus, known target samples are prone to producing a higher $E_{sum}(x)$.On the other hand, the unknown target samples produce low $E_{sum}(x)$.

We then utilise $E_{sum}(x)$ to train a binary discriminator D_1 to generate coarse similarity scores. The binary discriminator assumes that known target samples belong to the shared label space. Therefore, D_1 trains itself to produce high output for samples with high $E_{sum}(x)$ and low output for samples with low $E_{sum}(x)$ as follows,

$$E_{D_1} = -\frac{1}{n_s}\sum_{i=1}^{n_s} \log D_1(E_{sum}(x_i^s)) - \frac{1}{n_t}\sum_{j=1}^{n_t} \log(1 - D_1(E_{sum}(x_j^t))). \tag{6.2}$$

Now, we represent the output probability of D_1 as $d(x_j^t) = D_1(E_{sum}(x_j^t))$. $d(x_j^t)$ represents a probabilistic domain confidence, which is a preliminary similarity score for coarse separation of source-like target samples and source-unlike target samples at this step. For a source-like target sample $d(x_j^t)$ is high while for a source-unlike target sample $d(x_j^t)$ is a low probability value (for more details see Section 6.3.4).

MION: Now, we introduce a feature discriminator G to minimise *I* between preliminary unknown target samples and the source samples and maximise *I* between preliminary known target samples and the source samples. Maximising *I* will increase the amount of uncertainty removed in G for the preliminary known target samples. On the other hand, minimising *I* will introduce more uncertainty in G for the preliminary unknown target samples. In particular, simultaneously knowing how to group target classes similar to the source domain and separating target classes dissimilar to the source domain will assist G to learn better known-unknown discrimination for the target domain. We rank $d(x_j^t)$ for the target samples to identify m preliminary known and m preliminary unknown target samples, where, $m = b/4$ and $b =$ batch size. We use the selected m preliminary known, m preliminary unknown target samples and the source samples for optimising *I*. Note that the value of $m = b/4$ is determined empirically and is suitable for a wide range of tasks.

We adopt MINE [192] to calculate the *I*. In DAMI, MINE exploits the bound $I(X^s, X^t) \geq I_\Theta(X^s; X^t)$, where X represents the output features of G and $I_\Theta(X^s; X^t)$ denotes the *neural information measure*. The unbiased estimation of *I* on *l i.i.d* samples is measured by using a neural network T_θ as,

$$\widehat{I(X^s; X^t)_l} = \sup_{\theta \in \Theta} \mathbb{E}_{P^{(l)}_{X^s X^t}}[T_\theta] - \log(\mathbb{E}_{P^{(l)}_{X^s}) \otimes \widehat{P}^{(l)}_{X^t}}[\exp^{T_\theta}]. \tag{6.3}$$

Here, $P_{X^s X^t}$ is the joint probability distribution of $(G(F(\mathbb{D}_s)), G(F(\mathbb{D}_t)))$, and P_{X^s}=$\int_{X^t} P_{X^s X^t}$ and P_{X^t}=$\int_{X^s} P_{X^s X^t}$ are the marginals. The objective of maximising and minimising *I* is fulfilled by gradient ascent and descent, respectively. The expectations are estimated by shuffling the samples from the joint distribution within a batch (during training). This establishes dependency through marginalisation. G is trained to optimise *I* as,

$$E_{MI_k} = \lambda_1 \widehat{I(X^s; X^t_k)_l}, E_{MI_{uk}} = \lambda_2 \widehat{I(X^s; X^t_{uk})_l}. \tag{6.4}$$

Here, E_{MI_k} and $E_{MI_{uk}}$ represent the computed *I* of preliminary known X^t_k and preliminary unknown X^t_{uk} target samples with the source samples X^s, respectively. The hyper-parameters (λ_1 and λ_2) are easy to choose empirically and work well across multiple tasks. Setting $\lambda_1 > \lambda_2$ ensures that *I* of known samples are given higher attention than unknowns. After optimising (6.4), G learns a function that partitions the data such that the features of source and known target samples are closer compared to the unknown target samples.

FSN: For fine separating known-unknown target samples, first, for source and target samples, we convert the output features X of G to probability distribution using a softmax layer. Then, we compute the *pmi* scores between softmax outputs of the target and source samples. The softmax outputs for all target samples (Z^t) and source samples (Z^s) can be treated as a distribution of discrete random variables. The *pmi* score between two instances (Z^s_i and Z^t_j) of two random variables (Z^s and Z^t) represents the probability of their co-occurrence compared to their independent occurrence. The *pmi* score is formalised as,

$$pmi(Z_i^s; Z_j^t) = \log \frac{P(Z_i^s, Z_j^t)}{P(Z_i^s)P(Z_j^t)}. \tag{6.5}$$

The dependency between Z^s and Z^t is ensured by marginalisation over all samples. The joint probability distribution is constructed as $M = Z^s \cdot {Z^t}^\top$ ($n_s \times n_t$ tensor M), where Z^s and Z^t matrices hold the dimension of $n \times X$. The marginals are computed from the summation of rows and columns of M. In the OSDA setting, the target samples from the shared classes possess a great deal of similarity to the source samples. Thus, for G and image feature pairs ($F(x_i^s)$, $F(x_j^t)$), when each image feature contains the similar object of its pair but has a domain gap, the random variable constructed by the first of each pair, Z_i^s, will have a powerful statistical influence on the random variable for the second one, Z_j^t. On the other hand, image feature pair with dissimilar images will pose a weak statistical relation. Therefore, the known target samples will produce higher *pmi* than unknown samples. We compute the *pmi* of a target sample to every source sample and consider the highest *pmi* value.

For generating the final instance-level weights, we introduce the binary discriminator D_2 in FSN. The goal of D_2 is to distinguish known and unknown target samples depending on the computed *pmi* scores. To be specific, D_2 learns to generate higher weights for target samples with highest *pmi* than samples with the lowest *pmi*. We optimise D_2 as follows,

$$E_{D_2} = -\frac{1}{n_t}\sum_{j=1}^{n_t}(\log(D_2^h(G(F(x_j^t)))) + \log(1 - D_2^l(G(F(x_j^t))))), \tag{6.6}$$

where $D_2^h(G(F(x_j^t)))$ and $D_2^l(G(F(x_j^t)))$ denote the probability of target samples with highest and lowest *pmi* scores, respectively. After optimising (6.6), D_2 can generate fine distinctive weights ($w(x_j^t) = D_2(G(F(x_j^t)))$) for the target samples. The value of $w(x_j^t)$ is close to 1 for known target samples and close to 0 for unknown target samples. This fine separation by D_2 has high confidence because only extremely similar and dissimilar samples are used. The target samples with high $w(x_j^t)$ value will be aligned to the source domain by DAN and the target samples with low $w(x_j^t)$ value will be recognised as 'unknown' (6.10).

6.2.1.1 Domain Adversarial Network

DAN aims to classify known target samples to corresponding source (known) classes and recognises unknown target samples as 'unknown'. Known target samples are first aligned to the source domain adversarially in a two-player minimax game and then classified to corresponding known classes. Note that the weighting module identifies known and unknown target samples.

Adversarial adaptation of known target samples: In line with [54, 84], we exploit the generative adversarial principle in DAN. The first player of DAN is a domain discriminator D_3. D_3 distinguishes the features of the source domain from the target domain. The second player is a feature generator F. F simultaneously reduces the feature distribution divergence in the opposite direction of D_3. In particular, the objective of D_3 is to adversarially assist F to learn transferable features for the known target samples and align them towards the source domain. F takes inputs from both source domain $\mathbb{D}_s$ and target domain $\mathbb{D}_t$. D_3 takes input from F. For adversarially aligning the distributions of source and known target samples, we optimise the objective below,

$$E_{D_3} = -\frac{1}{n_s}\sum_{i=1}^{n_s} \log(D_3(F(x_i^s))) - \frac{1}{n_t}\sum_{j=1}^{n_t} w(x_j^t) \log(1 - D_3(F(x_j^t))). \tag{6.7}$$

where, $D_3(F(x_i^s))$ and $D_3(F(x_j^t))$ denote the probability of source and target samples, respectively. D_3 uses the generated weights $w(x_j^t)$ (Section 6.2.1) in the adversarial optimisation. This optimisation is equivalent to aligning known target samples towards the source domain.

Now, to perform the final classification of known and unknown classes, we place a source classifier Cl_1 with $|C| + 1$ indices in DAN, where $|C|$ indices represent the known classes in the source domain and the $(|C| + 1)^{th}$ index denotes the 'unknown' class for $\overline{C_t}$.

Classifying known classes: The classifier Cl_1 takes features from F and produces $|C|+1$ dimensional output. Softmax function transforms the output into a $|C|+1$-dimensional class probability as, $\sigma(z) = \exp(z)/(\sum_{i=1}^{|C|+1} \exp(z_i))$, where z is the logit vector. We define the classification loss on the known classes as follows,

$$E^s_{Cl_1} = \frac{1}{n_s} \sum_{i=1}^{n_s} L_{Cl_1}(Cl_1(F(x^s_i)), y^s_i). \tag{6.8}$$

Here, L_{Cl_1} is the standard cross-entropy loss function for minimising the error of the classifier Cl_1 and y^s_i is the ground truth label of i^{th} source sample.

To preserve cluster assumption characteristics during adaptation, we further minimise the conditional entropy (CE) for the known classes of the target domain as follows,

$$E^t_{ce} = -\mathbb{E}_{x_j \sim \mathbb{D}_t}[w(x^t_j) Cl_1(F(x^t_j))^\top \log(Cl_1(F(x^t_j)))]. \tag{6.9}$$

Here, $Cl_1(F(x^t_j))$ denotes the probability of j^{th} target sample from Cl_1. Minimising CE will enforce confidence in Cl_1 for unlabelled target samples, and the decision boundaries will occur far away from the data-dense regions. We incorporate $w(x^t_j)$ to the CE minimisation so that only the entropy of known target samples is minimised. However, if Cl_1 is not locally-Lipschitz, only minimising CE may over-fit the training samples. To prevent this, we use virtual adversarial training (VAT) [194] to smooth the surface of Cl_1 around the unlabelled points [194, 195]. In line with [195], we optimise the VAT loss for the source and target domain as follows,

$$\begin{aligned} E^t_v &= -\mathbb{E}_{x \sim \mathbb{D}_t}[w(x^t_j) \max_{\|\gamma\| < \epsilon} \mathcal{D}_{KL}(Cl_1(F(x^t_j)) \| Cl_1(F(x^t_j + \gamma)))], \\ E^s_v &= -\mathbb{E}_{x_i \sim \mathbb{D}_s}[\max_{\|\gamma\| < \epsilon} \mathcal{D}_{KL}(Cl_1(F(x^s_i)) \| Cl_1(F(x^s_i + \gamma)))]. \end{aligned} \tag{6.10}$$

where, $\mathcal{D}_{KL}(.)$, $Cl_1(F(x^t_j))$, $Cl_1(F(x^s_i))$, γ, and ϵ represent the Kullback–Leibler divergence, probability of j^{th} target sample, probability of i^{th} source sample, adversarial perturbation, and noise, respectively. Note that we have used $w(x^t_j)$ in VAT losses for enforcing Cl_1 to be consistent within the norm-ball neighbourhood of each sample belonging to the label space C.

Classifying unknown class: Based on our weighting module, target samples with low weights ($w(x_j^t)$) have more probability of belonging to the unknown classes. We use the target samples with the lowest weights to recognise unknown target samples. We define the weighted loss for recognising the 'unknown' class by Cl_1 as,

$$E_{Cl_1}^t = \frac{1}{n_t}\sum_{j=1}^{n_t}(1 - w(x_j^t))L_{Cl_1}(Cl_1(F(x_j^t)), y_u^t), \tag{6.11}$$

where y_u^t denote the label 'unknown'. Note that $\mathbb{D}_t$ is unlabelled and the label y_u^t is not provided by the DA setting. We force Cl_1 to classify the target samples with lowest $w(x_j^t)$ as 'unknown'. This loss is minimised only for the $(|C|+1)^{th}$ index of the classifier Cl_1.

6.2.2 Training phase

We denote the parameters of the DAMI components: F; G; Cl_1; D_3; Cl_2; D_1; and D_2 as θ_F; θ_G; θ_{Cl_1}; θ_{D_3}; θ_{Cl_2}; θ_{D_1}; and θ_{D_2}, respectively. The overall training objectives of DAMI are as follows,

$$\begin{aligned}
\theta_{Cl_2}, \theta_{D_1} &= \underset{\theta_{Cl_2}, \theta_{D_1}}{\operatorname{argmin}}\, E_{Cl_2} + E_{D_1}, \\
\theta_G &= \underset{\theta_G}{\operatorname{argmin}}\, E_{MI_{uk}} - E_{MI_k}, \theta_{D_2} = \underset{\theta_{D_2}}{\operatorname{argmin}}\, E_{D_2}, \\
\theta_{Cl_1}, \theta_{D_3} &= \underset{\theta_{Cl_1}, \theta_{D_3}}{\operatorname{argmin}}\, E_{Cl_1}^s + E_{Cl_1}^t + E_{D_3} + E_{ce}^t + E_v^s + E_v^t, \\
\theta_F &= \underset{\theta_F}{\operatorname{argmin}}\, E_{Cl_1}^s + E_{Cl_1}^t - E_{D_3} + E_{ce}^t + E_v^s + E_v^t.
\end{aligned} \tag{6.12}$$

During forward propagation, the values of the loss functions shown in (6.12) are computed. During backward propagation, the gradients are calculated. We use a gradient reversal layer (GRL) [67] to calculate the gradient of E_{D_3} efficiently during back-propagation. The gradient reversal phenomenon confirms that the feature distributions over the shared classes C of the source and target domains are made as indistinguishable as possible in F and Cl_1. Then, the parameters of the weighting module are updated by the gradients of the first three equations of (6.12). Finally, the parameters of DAN components are updated by the gradients of the last two equations of (6.12). The training procedure is outlined in Algorithm 1.

Algorithm 1 DAMI Training Procedure

Input: labelled $\mathbb{D}_s$; unlabelled $\mathbb{D}_t$; Feature generator F; classifier Cl_1; adversarial domain discriminator D_3; feature discriminator G; auxiliary source classifier Cl_2; two non-adversarial discriminators: D_1 and D_2. F, Cl_1 and Cl_2 are trained on D_s.

Output: Trained F and Cl_1.

```
while not converged do
    Sample mini-batch from $(x_i^s, y_i^s)_{i=1}^{n_s}$ and $(x_j^t)_{j=1}^{n_t}$;
    Update $F$, $Cl_2$, and $D_1$ by (6.1) + (6.2);
    Select $m$ preliminary known and $m$ preliminary unknown target samples using $d(x_j^t)$;
    Compute mutual information by (6.3);
    Update $G$ by (6.4) ;
    for j = 1:mini-batch do
        Compute pmi between every source sample $(x_i^s, y_i^s)_{i=1}^{n_s}$ and $(x_j^t)$ by (6.5);
        Record the highest pmi;
    end for
    Record target samples with highest and lowest pmi;
    Update $D_2$ by (6.6);
    Compute losses by (6.7) – (6.11) and update $Cl_1$, $F$, and $D_3$ jointly by computed losses;
end while
return Trained $Cl_1$ and $F$.
```

6.3 Experimental Studies

This section describes the datasets, evaluation details, experimental outcomes (classification results, visualisation of learned features, impact of openness, loss convergence analysis and hyper-parameters setups), analyse visualisation of learned weights, features, and present a detailed ablation study. Datasets used are discussed in Chapter 5 and we follow the similar evaluation protocols as discussed in Section 5.4.2.

6.3.1 Implementation Details

We have used ImageNet pre-trained ResNet-50 [4] with customised new layers as the feature generator F for all datasets. For VisDA, additionally, we use ImageNet pre-trained VGG-16 and VGG-19. The classifier Cl_2 has a 1024-dimensional hidden layer and discriminator D_1 has no hidden layer. The feature discriminator G, discriminator D_2, and classifier Cl_1 have a hidden layer of 256 dimensions each. The final discriminator D_3 has a hidden layer of 1024 dimensions. We have used RELU in F and Cl_2 structures, and Leaky RELU in Cl_1, D_3, G, and D_2. We have used SGD with a learning rate of 0.001, a weight decay of 0.005, and momentum of 0.9. We found the value of λ_1 as 10 times greater than λ_2 optimal for all tasks. A batch size greater than or equal to 16 is suitable for the model's effective computation of I. We have applied data augmentation over the m selected samples to make the computation of I robust—for example, different transformations such as random crops, central crops and horizontal flips. The images within each batch are repeated to ensure that every source sample is paired with the original target sample and their transformations. We have set the values of $\lambda_1 = 1$ and $\lambda_2 = 0.1$ for I optimisation. For optimising (6.12), we have used 0.001 as the coefficient of E^t_{ce}, E^s_v, and E^t_v, 0.3 as the coefficient of E_{D_3}, and 0.3 as the coefficient of $E^t_{Cl_1}$. Depending on the tasks of interest, the values of the hyper-parameters and the coefficients are to be adjusted empirically.

6.3.2 Compared DA Methods

For thorough comparison, we have compared the performance of DAMI with a number of contemporary methods. They include **1)** Classifier without DA: ResNet [4] (Note that Negative transfer is calculated against this non-DA classifier.); **2)** Closed-set DA methods: Domain-Adversarial Neural Networks (DANN) [54] and Residual Transfer Networks (RTN) [50]; **3)** OSDA methods: Assign-and-Transform-Iteratively (ATI) [81], OSDA by Back-Propagation (OSBP) [84], Separate to Adapt (STA) [85], DA with Multiple

Classifiers (DAMC) [189], Towards Inheritable Models (TIM) [87], Mutual to Separate (MTS) [86], Progressive Graph Learning (PGL) [89], Joint Partial Optimal Transport (JPOT) [196], and Factorised Representations for Open-set DA (FRODA) [90]. DAMI follows the same setting and evaluation protocols.

6.3.3 Classification results

In this section, we compare the performance of DAMI against contemporary works. Tables 6.1, 6.2, 6.3, and 6.4 present the results on Office-31, Office-Home, ImageNet-Caltech, and VisDA datasets, respectively. The results of the compared methods on Office-31, Office-Home, and VisDA datasets are adopted from [85–87, 89, 90, 189, 196]. Similar to [85] and [87], we report only OS accuracy in Table 6.2. As some of the results of compared methods for the ImageNet-Caltech tasks are not available in the literature, for a fair comparison, we produce the results of all the compared methods except FRODA [90] in Table 6.3. We use ResNet-50 as the backbone. Because of the number of iterations and data split, the results may vary from other papers. The results of FRODA [90] on ImageNet-Caltech is adopted from the original paper. The contemporary OSDA methods have several different backbone networks for VisDA tasks. We experiment with ResNet-50, VGG-16, and VGG-19 as the backbone networks for a fair comparison with the contemporary works and report the results in Table 6.4. For DAMI, we report the mean of OS and OS* over three separate runs in all the tables. Note that negative transfers (performance lower than ResNet) are highlighted in blue colour in the tables.

It is evident from Tables 6.4, 6.2, 6.3, and 6.4 that DAMI outperforms all the compared methods on the majority of the tasks for all the datasets. For closed set methods [54] and [50], we observe that the performance lags behind ResNet on some tasks when evaluated following the OSDA setting (Table 6.1). The lag in the performance compared to the non-DA classifier results from the negative transfer. Negative transfers are initiated by aligning the whole target domain with the source domain.

The majority of existing OSDA methods perform better than the non-DA classifier ResNet and show no negative transfer in the Office-31 tasks (Table 6.1) as the dataset

Table 6.1: Classification accuracy (%) of DAMI and compared DA methods on Office-31 tasks (ResNet-50). Here, OS denotes the average per-class accuracy measured on all known and unknown classes and OS* represents the average per-class accuracy computed on only the known classes.

Method	Accuracy (%)													
	A→ W		A→ D		D→ W		W→ D		D→ A		W→ A		Avg	
	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*
ResNet [4] (2016)	82.5	82.7	85.2	85.5	94.1	94.3	96.6	97.0	71.6	71.5	75.5	75.2	84.2	84.4
DANN [54] (2016)	85.3	87.7	86.5	87.7	97.5	98.3	**99.5**	**100.0**	75.7	76.2	74.9	75.6	86.6	87.6
RTN [50] (2016)	85.6	88.1	89.5	90.1	94.8	96.2	97.1	98.7	72.3	72.8	73.5	73.9	85.4	86.8
OpenMax [80] (2016)	87.4	87.5	87.1	88.4	96.1	96.2	98.4	98.5	83.4	82.1	82.8	82.8	89.0	89.3
ATI [81] (2017)	87.4	88.9	84.3	86.6	93.6	95.3	96.5	98.7	78.0	79.6	80.4	81.4	86.7	88.4
OSBP [84] (2018)	86.5	87.6	88.6	89.2	97.0	96.5	97.9	98.7	88.9	90.6	85.8	84.9	90.8	91.3
FRODA [90] (2018)	78.8	-	88.0	-	94.6	-	98.0	-	73.7	-	76.5	-	84.9	-
STA [85] (2019)	89.5	92.1	93.7	96.1	97.5	96.5	**99.5**	99.6	89.1	93.5	87.9	87.4	92.9	94.1
DAMC [189] (2020)	90.2	90.8	89.5	89.7	97.9	98.8	98.6	98.8	91.0	91.7	86.8	87.6	92.4	92.9
TIM [87] (2020)	91.3	93.2	94.2	97.1	96.5	97.4	**99.5**	99.4	90.1	91.5	88.7	88.1	93.4	94.5
MTS [86] (2020)	92.4	96.8	94.7	98.2	97.9	99.5	98.9	**100.0**	89.6	92.0	89.7	91.9	93.8	96.4
JPOT [196] (2020)	92.8	92.2	95.2	96.0	98.1	96.2	99.5	98.6	93.0	94.1	88.9	88.4	94.6	94.3
DAMI	**96.3**	**98.6**	**98.2**	**100.0**	**100.0**	**100.0**	98.7	99.9	**95.7**	**97.8**	**93.4**	**95.8**	**97.0**	**98.6**

Table 6.2: Average per-class accuracy (%) of DAMI and contemporary DA methods on Office-Home dataset tasks for all known and unknown classes. Here, ResNet-50 is used as the backbone network.

Method	Accuracy (%) (OS)												
	Ar→ Cl	Pr→ Cl	Rw→ Cl	Ar→ Pr	Cl→ Pr	Rw→ Pr	Cl→ Ar	Pr→ Ar	Rw→ Ar	Ar→ Rw	Cl→ Rw	Pr→ Rw	Avg
ResNet [4] (2016)	53.4	52.7	51.9	69.3	61.8	74.1	61.4	64.0	70.0	78.7	71.0	74.9	65.3
ATI [81] (2017)	55.2	52.6	53.5	69.1	63.5	74.1	61.7	64.5	70.7	79.2	72.9	75.8	66.1
OpenMax [80] (2016)	56.5	52.9	53.7	69.1	64.8	74.5	64.1	64.0	71.2	80.3	73.0	76.9	66.7
OSBP [84] (2018)	56.7	51.5	49.2	67.5	65.5	74.0	62.5	64.8	69.3	80.6	74.7	71.5	65.7
STA [85] (2019)	58.1	53.1	54.4	71.6	69.3	81.9	63.4	65.2	74.9	85.0	75.8	80.8	69.5
DAMC [189] (2020)	64.8	63.1	67.5	**84.6**	81.6	80.9	**79.6**	77.9	85.4	88.1	85.6	85.9	78.7
TIM [87] (2020)	60.1	54.2	56.2	70.9	70.0	78.6	64.0	66.1	74.9	83.2	75.7	81.3	69.6
MTS [86] (2020)	63.7	58.4	64.4	80.6	74.2	83.3	68.4	71.1	78.0	86.0	79.5	82.7	74.2
PGL [89] (2020)	61.6	58.4	65.0	77.1	72.0	83.0	68.8	72.2	78.6	85.9	82.8	82.6	74.0
JPOT [196] (2020)	59.6	54.2	54.6	72.3	70.1	82.1	62.9	68.3	75.1	84.8	77.4	81.2	70.2
DAMI	**68.1**	**63.2**	**70.5**	83.7	**83.8**	**86.3**	76.7	**80.9**	**86.3**	**91.2**	**87.8**	**88.9**	**80.6**

has easily adaptable domain differences. However, these methods lag behind ResNet on some tasks of Office-Home and ImageNet-Caltech as reported in Tables 6.2 and 6.3. These datasets are more challenging as they have a larger domain gap than the Office-31 dataset and more samples per class. The effect of negative transfer causes the performance sacrifice in the contemporary OSDA methods. Though OSDA methods do not blindly align all target samples to source samples, the negative transfer happens during the known-unknown target sample separation stage before DA.

The issue of negative transfer in OSDA methods can further be explained by the

Table 6.3: Classification performance of DAMI and compared DA methods on ImageNet-Caltech tasks (ResNet-50). Here, OS and OS* denote the average per-class accuracy measured on all known and unknown classes and the average per-class accuracy computed only on the known classes, respectively.

Method	Accuracy (%)					
	Im→ Cal		Cal→ Im		Avg	
	OS	OS*	OS	OS*	OS	OS*
ResNet [4]	75.6	75.1	67.3	67.9	71.4	71.5
ATI [81] (2017)	71.9	66.9	67.6	65.7	69.8	66.3
OSBP [84] (2018)	64.2	64.5	54.9	53.7	59.6	59.1
FRODA [90] (2018)	79.9	-	74.5	-	77.2	-
STA [85] (2019)	75.4	74.1	67.9	68.1	71.7	71.1
MTS [86] (2020)	78.7	80.5	68.1	67.2	73.4	73.9
DAMC [189] (2020)	77.1	77.9	69.9	67.8	73.5	72.9
DAMI	**81.3**	**83.8**	**73.3**	**75.4**	**77.3**	**79.6**

Table 6.4: Classification accuracy (%) of DAMI (ResNet-50, VGG-16, and VGG-19) and other DA methods on VisDA2017 tasks. Here, we represent average per-class accuracy for every known and unknown classes separately, combined per-class accuracy combining all known and unknown classes as OS, and combined per-class accuracy for all known classes as OS* for better analysis.

Method	Accuracy (%)								
	bicycle	bus	car	motorcycle	train	truck	unknown	OS	OS*
ResNet [4] (2016)	40.2	55.4	63.5	70.8	74.1	35.2	45.6	54.9	56.5
DAMC [189] (ResNet-50) (2020)	50.6	74.8	66.7	80.6	75.9	38.8	73.9	65.9	64.6
DAMI (ResNet-50) (Our)	60.1	81.9	89.7	96.1	90.8	40.4	81.5	77.2	76.5
STA [85] (VGGNet) (2019)	52.4	69.6	59.9	87.8	86.5	27.2	84.1	66.8	63.9
TIM [87] (VGG-16) (2020)	53.5	69.2	62.2	85.7	85.4	32.5	**88.5**	68.1	64.7
DAMI (VGG-16) (Our)	58.1	78.9	88.7	96.8	87.8	**42.6**	83.1	76.5	75.4
OSBP [84] (VGG-19) (2018)	51.1	67.1	42.8	84.2	81.8	28.0	85.1	62.9	59.2
PGL [89] (VGG-19) (2020)	**93.5**	**93.8**	75.7	98.8	96.2	38.5	68.6	80.7	**82.8**
DAMI (VGG-19) (Our)	74.1	90.9	**89.9**	**98.9**	**97.1**	40.6	85.5	**82.4**	81.9

comparison of OS* and OS results. It is worth mentioning that VisDA dataset is way more challenging than the other datasets as the adaptation has to happen from synthetic to real images. In Table 6.4, we observe that there is a considerable gap between the accuracy of the OS* and OS for the majority of the contemporary methods (shown in purple colour). The OS* lags behind OS, which means a large number of known images are classified as unknown images. PGL [89] shows less negative transfer compared to other methods. However, PGL lags far behind DAMI and other methods in recognising the unknown class, which shows the approach is not robust enough for OSDA tasks.

DAMI (VGG-16) and DAMI (VGG-19) shows increased per-class accuracy compared to other methods and decreased the gap between OS* and OS. This means optimisation of I facilitates the weighting module of DAMI by imposing better known-unknown discrimination.

Table 6.5: AUC-ROC of DAMI and other compared methods. Here, DAMI w/ D_2 and DAMI w/ multi-binary D_2 denote the binary discriminator D_2 and one-vs-rest multi-binary discriminator D_2, respectively.

	AUC-ROC						
Method	A→ W	A→ D	D→ W	W→ D	D→ A	W→ A	Avg
STA [85] (2019)	82.7	83.6	84.1	72.1	74.2	78.5	79.2
ROS [197] (2020)	**89.0**	85.1	89.8	95.9	83.5	80.8	87.3
DAMI w/ D_2	87.5	**97.3**	**100**	**99.8**	**89.1**	87.1	**93.4**
DAMI w/ multi-binary D_2	87.1	96.8	99.0	99.5	88.2	**88.2**	93.1

Table 6.6: Average per-class accuracy for all the known (OS) and unknown (UNK) classes of DAMI and compared open set DA methods.

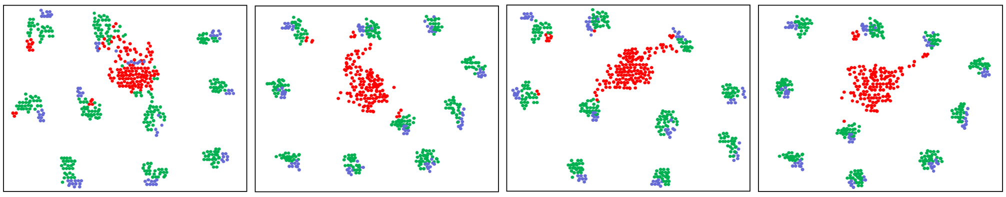
Method	Accuracy (%)													
	A→ W		A→ D		D→ W		W→ D		D→ A		W→ A		Avg	
	OS*	UNK	OS*	UNK	OS*	UNK	OS*	UNK	OS*	UNK	OS*	UNK	OS*	UNK
OSBP [8] (2018)	87.8	**75.2**	90.7	75.2	96.7	95.7	98.9	84.0	78.1	70.3	73.5	**74.0**	87.6	79.0
STA [85] (2019)	92.4	57.9	95.9	45.7	96.1	48.7	96.6	48.5	94.1	55.0	91.1	47.3	94.3	50.5
DAMI	**98.6**	73.3	**100.0**	**80.2**	**100.0**	**100.0**	**99.9**	**95.9**	**97.8**	**74.7**	**95.8**	69.4	**98.6**	**82.2**

AUC-ROC Analysis. To further analyse the effectiveness of the separation module (CSN + MION + FSN) of DAMI, we compute the area under the receiver operating characteristic curve (AUC-ROC) over the final probabilistic weights w in two settings. First, we compute AUC-ROC using the one-vs-rest (i.e., unknown-vs-all known) binary discriminator D_2. Second, we replace D_2 with $|C|+1$ one-vs-rest multi-binary classifiers. For every target sample in a batch, we compute the *pmi* score with all source samples. Then, for a target sample, if the *pmi* score is closer to 1, we record the source class yielding the highest *pmi* score and channel the target sample to that one-vs-rest classifier. If the *pmi* score is closer to 0, the target sample is channelled to the $(|C|+1)^{th}$ one-vs-rest classifier. We perform the second experiment to observe the robustness of the weighting module for multi-class weight generation. The results are reported in Table 6.5.

To compare the AUC-ROC of DAMI with contemporary works, we reproduce the AUC-ROC for ROS [197], and STA [85]. We follow the setting in [197] for reproducing the results. For ROS, we compute the AUC-ROC using the multi-rotation classifier over the normality score. For STA, we calculate the AUC-ROC using the multi-binary classifier over the instance-level weights. For all three methods, we report an average of three runs. Table 6.5 demonstrates that both variants of DAMI achieve better AUC-ROC than ROS and STA for the majority of individual tasks. As expected, the performance of the multi-binary D_2 is lower than the binary D_2. This is because the unknown class may have some similarities to some of the known classes, and that will degrade the binary classification performance of those known classes. This observation again verifies the robustness of the separation module of DAMI with binary D_2 for separating known and unknown samples. Moreover, to compare the unknown class and known classes accuracy separately, we compute the OS* and UNK (average unknown-class accuracy) for DAMI and three contemporary methods. The results in Table 6.6 provide an insight into how well DAMI recognises the unknown class compared to other methods.

Feature Visualisation. We use the t-SNE [172] algorithm for analysing the learned features. The t-SNE algorithm reduces the divergence between two distributions by preserving the closely related clusters of high dimension in converted low dimension. Figure 6.3 shows the t-SNE visualisation of the extracted features from the last layer of STA, DAMC, TIM, and DAMI for task A→ D. The learned features of some unknown classes in the STA method lie in the near vicinity of the known classes. In contrast, others are intermingled with known class features. This proves that STA confuses known and unknown samples because of depending only on the multi-binary classifier. However, DAMC suffers from less negative transfer than STA due to not depending only on a source trained classifier. For some cases, TIM improves over the STA and DAMC method and shows better separation. For other cases, TIM lags behind STA and DAMC. This is due to having no access to the source domain during adaptation. DAMI features show a better known-unknown separation and well-segregated clusters of aligned shared classes.

Robustness to Openness. To further substantiate the robustness of DAMI in differ-



(a) (b) (c) (d)

Figure 6.3: The t-SNE [172] visualisation of the last-layer features of the adapted classifiers for the task A→ D. Source, target-known and target-unknown samples are shown in green, blue and red colour respectively.

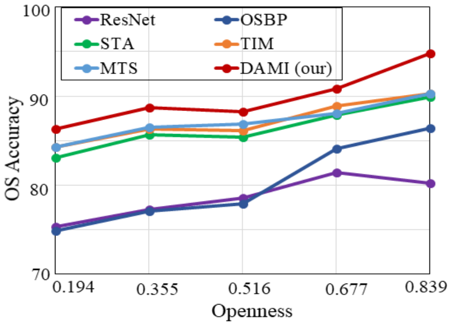


Figure 6.4: OS accuracy with respect to different openness levels in the target domain for the task A→ D of Office-31 dataset.

ent degrees of openness, we execute experiments on the Office-31 dataset. In Figure 6.4, we present the OS accuracy on different levels of Openness, $\mathbb{O} = 1 - |C_s|/|C_t|$ [198]. The superior performance of DAMI demonstrates that our method is robust for a wide range of openness settings. The improved known-unknown separation and proper adaptation achieved the ability to perform well in different openness settings.

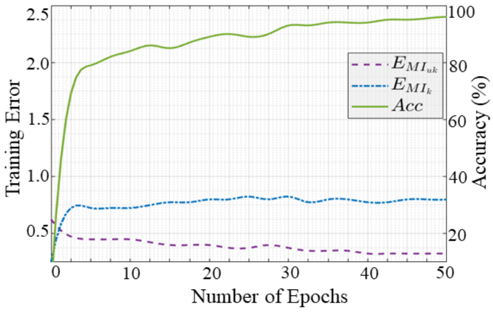


Figure 6.5: *Mutual information* optimisation training errors $E_{MI_{uk}}$, E_{MI_k}, and accuracy with respect to the training epochs for the A→ W task.

Convergence Analysis. We plot the I optimisation losses $E_{MI_{uk}}$, E_{MI_k} (6.4), and the accuracy with respect to the training epochs in Figure 6.5. The graph shows that I between preliminary known target samples and the source domain is maximised, and I between preliminary unknown target samples and the source domain is minimised over the epochs. At the same time, the accuracy steadily increases.

Hyper-parameters Setups. We analyse different hyper-parameters (λ_1 and λ_2) setups and accuracy in Figure 6.6. We show four different combinations starting from $\lambda_1 = 0.01$ and $\lambda_2 = 0.001$. It is apparent that as the value of λ_1 and λ_2 increases from the starting combination, the accuracy also increases up to $\lambda_1 = 1$ and $\lambda_2 = 0.1$. However, after that, the accuracy decreases sharply. The intuition is that very high attention (i.e., the value of hyper-parameters) to $E_{MI_{uk}}$ and E_{MI_k} causes greater updates in the parameters of G and harms the stability of the network. This lack of stability later harms the FSN and DAN. Thus, the performance decreases.

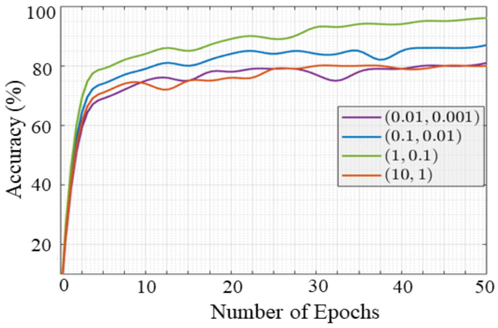
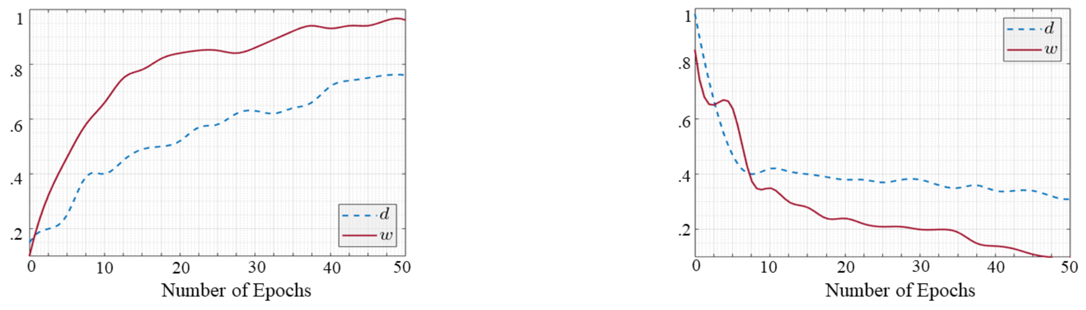


Figure 6.6: Classification accuracy using different combinations of hyper-parameters (λ_1 and λ_2) with respect to the training epochs for the A→ W task. Here, $(0.01, 0.001)$ represents (λ_1, λ_2) and so on.

6.3.4 Visualising Learned Weights

In this section, we visualise the learned weights generated in different steps of the weighting module and present a comparison between the weights distributions. Figure 6.7 shows that the final weights w for known target samples are consistently greater than the primary weight d. Similarly, for unknown samples, w is smaller than d across



(a) (b)

Figure 6.7: A pictorial illustration of learned weights of (a) known target samples and (b) unknown target samples for the task A→ W (Office-31), where d represents the weights generated by discriminator D_1, and w denotes the final instance-level weights assigned to target samples by discriminator D_2 for distinguishing known-unknown target samples and DA.

all epochs. The generated weights d from D_1 struggles to reach near 1 and near 0 values due to the leaky softmax layer in Cl_2. The leaky-softmax assigns a high probability for a logit that has a large probability for a known class. This helps in determining rough domain similarity. However, the leaky-softmax tends to generate an element-sum of its output smaller than 1. Due to this tendency, d lie in the mid-high and mid-low range. On the other hand, D_2 has no such constraint and is trained on the features of G, which has more discriminative knowledge about known-unknown samples. Thus, the weights w generated from D_2 have high confidence. This facilitates D_3 to better adapt known samples with high w values and Cl_1 to recognise unknown samples with low w values.

6.3.5 Ablation Study

We present the ablation study in a building block manner to justify the role of different components in DAMI. The results of our ablation studies are reported in Table 6.7.

Table 6.7: Classification accuracy (%) of different variants of DAMI on Office-31 tasks (ResNet-50).

Method	Accuracy (%)													
	A→ W		A→ D		D→ W		W→ D		D→ A		W→ A		Avg	
	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*
DAN	80.1	72.3	82.3	76.9	79.1	75.0	89.2	84.5	81.2	76.9	82.1	78.1	82.3	77.2
DAN + CSN	91.1	90.1	92.5	92.0	96.9	95.7	95.5	94.3	88.7	87.9	89.1	91.5	92.8	91.9
DAN + CSN + MION	93.3	93.2	94.5	94.7	97.4	97.6	95.7	94.8	91.1	92.3	88.3	88.1	93.3	93.4
DAMI (DAN + CSN + MION + FSN)	**96.3**	**98.6**	**98.2**	**100.0**	**100.0**	**100.0**	**98.7**	**99.9**	**95.7**	**97.8**	**93.4**	**95.8**	**97.0**	**98.6**

DAN: We start the ablation analysis with the base DAN. This variant has the main feature discriminator F, classifier Cl_1, and discriminator D_3 in the structure. Due to having no known-unknown separation module, DAN depends on the pseudo decision of Cl_1 for rejecting unknown target samples and suffers from a huge amount of negative transfer.

DAN + CSN: To reduce negative transfer by separating known-unknown samples before DA, we place Cl_2 and D_1. Though this variant outperforms the previous one, the generated weights d harm the adversarial DA for the leaky effect of Cl_2 (discussed in section 6.3.4).

DAN + CSN + MION: This variant is implemented without the FSN. Though MION gains the ability to generate well discriminative features, it still lacks at generating suitable weights. In this variant, we use the highest output of G as instance-level soft weights and observe the DAN struggles to differentiate known-unknown as the weights are not smooth yet. Such roughness in the weights reflects in the performance of the task W$\rightarrow$ A compared to the previous variant (Table 6.7).

DAMI (DAN + CSN + MION + FSN): DAMI outperforms the variants mentioned above. DAMI is capable of finely separating known-unknown target samples due to integrating FSN. FSN evaluates *pmi* for verifying similarity/dissimilarity of target samples and generates smooth weights through D_2. To summarise, DAMI has access to discerning knowledge of known and unknown samples, which assists better in the fine separation of known-unknown target samples.

6.3.6 Robustness of learned features

To further analyse the robustness of DAMI, we compare the divergence between the source and target domains before DA (raw input) and after DA (DAMI features). We compare the *Proxy $\mathcal{A}$-distance (PAD)* distances of DAMI representations and raw input. The PAD distance is formulated as $\hat{d}_{\mathcal{A}} = 2(1 - 2\beta)$, where β is the generalisation error [199]. The PAD distance is similar to $\mathcal{H}$-divergence (a measure of domains divergence [199]). If we select $\mathcal{A} = \mathcal{A}_{\eta} \in \mathcal{H}$, the $\mathcal{A}$-distance and the $\mathcal{H}$-divergence are similar [54].

Here, $\mathcal{A}_\eta$ is the set represented by η and η denotes the function of the DA model. We follow [54] to compute the PAD value using Amazon Reviews dataset. The dataset has four domains (books, DVD disks, electronics, and kitchen appliances). Each domain has reviews of a specific type of product. 5000-dimensional feature vectors of unigrams and bigrams represent the reviews. A product is labelled '1' if it has got up to 2 stars and '0' otherwise. The 3 stars were treated as 'unknown'. We perform 12 DA tasks. We used 2000 labelled sources and 2000 unlabelled target samples, among which 500 were from unknown classes. We replace F with a fully connected network. F and G have only one hidden layer with 1024 and 512 dimensions, respectively. Cl_1, Cl_2, D_1, D_2, and D_3 are implemented without hidden layers. Other implementation setups are kept the same as discussed in Section 6.3.1. Figure 6.8 shows that the representations of DAMI

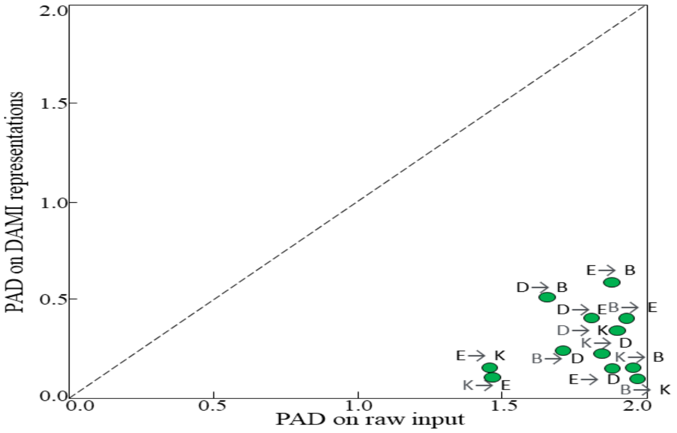


Figure 6.8: Comparison of Proxy $\mathcal{A}$-distances of DAMI feature representations and raw input on the dataset Amazon Reviews for the shared label space (known classes). Here, x-axis and y-axis denotes the PAD value of DAMI features and raw input, respectively.

achieve much less PAD value than raw inputs for the known classes. This shows DAMI performs improved DA.

6.4 Summary

In this chapter, we have proposed a new DA model for addressing the challenge of handling negative transfer in OSDA. DAMI reduces negative transfers by correctly separating known from unknown target samples. The separation module has three*coarse-to-fine* steps. First, it explores domain confidence. Then it optimises I in between similar classes and dissimilar classes simultaneously. Finally, it uses *pmi* score for fine separation. After known-unknown target separation, DAMI aligns features of known target samples to the source domain and recognises unknown target samples as 'unknown'. It has been evident from the comprehensive analysis on several benchmark datasets that DAMI is robust to different levels of openness, large domain gaps, and outperforms contemporary works.

The fourth contribution discussed in this chapter demonstrated that the proposed weighting module associated with the mutual information and domain confidence (Chapter 6) outperforms the proposed weighting module depending only on domain confidence (Chapter 5). In the next chapter, we will present the fifth contribution of the thesis.

Chapter 7

Attention Model for Fine-grained Generalised Zero-Shot Learning

[1]In Chapters 3, 4, 5, and 6 we have discussed the first four contributions of this thesis. We now turn our attention to the fifth contribution of this thesis in this chapter. This contribution focuses on solving the limitations of existing GZSL approaches.

Embedding learning (EL) and feature synthesising (FS) are two of the popular categories of GZSL methods. EL or FS using global features cannot discriminate fine details in the absence of local features. On the other hand, EL or FS methods exploiting local features either neglect direct attribute guidance or global information. Consequently, neither method performs well. In this chapter, we propose to explore global visual features and direct attribute-supervised local visual features for both EL and FS categories in an integrated manner for fine-grained GZSL. The proposed integrated network has an EL sub-network and a FS sub-network. Consequently, the proposed integrated network can be tested in two ways. We propose a novel two-step dense attention mechanism to discover attribute-guided local visual features. We introduce new mutual learning between the sub-networks to exploit mutually beneficial information for optimisation. Moreover, we propose to compute source-target class similarity based on mutual information and transfer-learn the target classes to reduce bias towards the source domain during testing. We demonstrate that our proposed method outperforms contemporary methods on benchmark datasets.

[1] Chapter 7 is adapted from: **T. Shermin**, S. W. Teng, F. Sohel, M. Murshed, and G. Lu, "Integrated Generalised Zero-Shot Learning for Fine-Grained Classification," Pattern Recognition, 2021.

7.1 Problem Setting.

The GZSL problem setting has a source domain $\mathbb{D}_s$ with $\mathcal{Y}^s$ label space and a target domain $\mathbb{D}_t$ with $\mathcal{Y}^t$ label space having C_s and C_t classes, respectively, where $\mathcal{Y}^s \cap \mathcal{Y}^t = \emptyset$. The source and target classes are indexed as $\{1, \ldots, C_s\}$ and $\{C_s + 1, \ldots, C_s + C_t\}$ respectively. A dataset of N labelled images are available in the source domain, $\mathbb{D}_s = \{(x_i, y_i) \mid x_i \in \mathcal{X}, y_i \in \mathcal{Y}^s\}_{i=1}^{N}$, $\mathcal{X}$ denotes the visual feature space. The target domain classes have no training samples or features. The source *class semantic vectors* for $c \in \mathcal{Y}^s$ are $\mathcal{A}^s = \{a_c\}_{c=1}^{C_s}$. The target *class semantic vectors* for $c \in \mathcal{Y}^t$ are $\mathcal{A}^t = \{a_c\}_{c=C_s+1}^{C_t}$. The semantic vector of class c is $a_c = [a_c^1, \ldots, a_c^A]$, where a_c^A represents the score of the presence of the A^{th} attribute in the class. Similar to [126], we assume *attribute semantic vectors* $\{\boldsymbol{v}_i\}_{i=1}^{A}$ are provided. Here, $\boldsymbol{v}_i$ denotes the average GloVe [200] representation of words in the i^{th} attribute, e.g., 'throat colour blue'. The goal of ZSL is to learn visual classifiers of unseen classes only $h_{zsl} : \mathcal{X} \rightarrow \mathcal{Y}^t$. The objective of GZSL is to train visual classifiers of all source and target classes $h_{gzsl} : \mathcal{X} \rightarrow \mathcal{Y}^s \cup \mathcal{Y}^t$.

7.2 Overview

Conventional supervised deep learning classifiers require a large amount of labelled training data and the training and testing data must be drawn from the same distribution. Although ordinary object images are easily accessible, there are many object categories with scarce visual data, such as endangered species of plants and animals [201]. To address the issues, *Zero-shot learning* (ZSL) methods are studied. ZSL methods aim to exploit the visual-semantic relationship of source (seen) classes to train a visual classifier on source classes and test the classifier on target classes only. Though, the underlying distribution of source and target domains is disjoint, the ZSL setting assumes that the trained visual classifier knows whether a test sample belongs to a source or target class. To alleviate such an unrealistic assumption, the ZSL setting is extended to a more realistic setting called Generalised Zero-Shot Learning (GZSL) [202–204], where the classifier has to classify test images from both source and target classes.

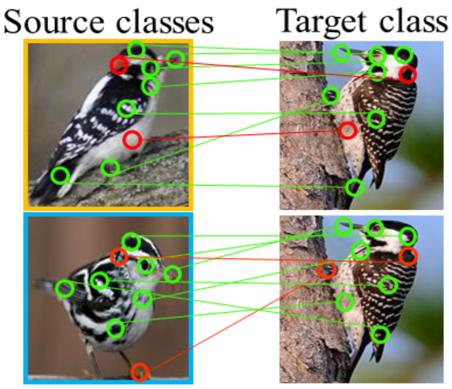


Figure 7.1: Samples from CUB dataset showing only a few dissimilar attributes between the source and target classes. Red and green indicators denote dissimilar and similar attributes, respectively. Best viewed in colour.

The ultimate aim of this work is to improve GZSL for fine-grained recognition. Unlike the coarse-grained datasets (classes, e.g., Animal, Table, and Bus, with no sub-ordinate classes), classification of fine-grained datasets (with sub-ordinate classes, e.g., different types of birds (Blue jay, Florida jay, and Green jay)) demands more local discriminative properties. The region-based local features capture more fine distinctive information and relevance to the semantic attributes than the global features (Figure 7.1). On the other hand, global features hold the generic structure of the deep neural network's visual representation, which is vital for generalisation. Therefore, besides exploring local details for improved fine-grained GZSL, we argue to preserve global details for constructing a better visual GZSL classifier.

Embedding learning (EL) and feature synthesising (FS) methods are two popular approaches for GZSL methods. Most existing EL [205] [121] and FS [144,151] methods only use global features for fine-grained GZSL tasks. Some EL methods [9,121,122] focus only on the local features. However, these EL methods do not relate individual attributes to the local features; they relate a combination of all attributes. Consequently, they do not fully explore the discriminative local information linked to the attributes.

We aim to address the aforementioned limitations in both EL and FS methods. As such, we propose an integrated network, which has an EL sub-network (Attribute Guided Attention Network (AGAN)) and a FS sub-network (Adversarial Feature Generation Network (AFGN)). In the proposed method, first, we divide a sample into local regions. Then, we preserve the global representation of the local regions. After that, we propose a two-step dense attention mechanism to explore the relation between the semantic attributes and the local regions for discovering fine-discriminative information. Next, we combine explored global and local information to construct a feature embedding used by both sub-networks. Finally, we propose a mutual learning-based optimisation so that both sub-networks can assist each other and learn better features for the GZSL task.

The proposed two-step dense attention mechanism uses direct attribute supervision to construct a visual feature embedding that holds *attribute-weighted local visual* information. In particular, to assign the first-level of attention to the region features, we explore two general questions, i.e., 'Is the region related to any attribute?' and 'Which attribute has the most relevance to the region?'. Thus, a region's attention has information about the presence of attributes and the most relevant attribute in the region. This will encourage only the most relevant attribute to a region to be attended and assist in learning fine distinction. In the second-level, we infuse the confidence score of having that attribute in the class so that the attention of a region containing an attribute that has a greater class score is higher weighted than others. This knowledge will encourage a better focus on common intra-class information, thereby facilitating improved class decisions. The dense-attention mechanism is placed in AGAN. We design the connection between AGAN and AFGN in such a way that they both can leverage the attribute-weighted features constructed by the attention mechanism.

To reduce bias towards source classes, we explore mutual information-based source-target class similarity and loosely learn target classes in AGAN. Mutual learning explores mutually useful information between AGAN and AFGN. Thus, the source class bias in AFGN is also partially smoothed out. Moreover, AFGN is flexible as it can be replaced with any sophisticated FS network to learn attribute guided local features. Since the proposed integrated network has both EL (AGAN) and FS sub-network (AFGN), the proposed method can test in two ways, following the test sequence of both EL and FS GZSL methods (Section 7.3.4). Thus, the proposed integrated network will contribute to the GZSL field in two different fine-grained classification methods.

The main contributions of this paper are as follows:

- We propose to integrate an embedding learning sub-network and a feature generation sub-network to an integrated network. We introduce mutual learning to optimise both sub-networks. This is the first work to apply mutual learning in this domain to the best of our knowledge. The integration also enables two different ways of testing capability.
- We propose a novel two-step attention mechanism, which discovers fine distinctive local visual information directly supervised by the attributes. In addition, unlike existing fine-grained GZSL methods, we propose to preserve global visual information for developing a better GZSL visual classifier.
- For fine-grained GZSL tasks, we introduce the exploration of attribute guided fine-distinctive visual features in both embedding learning and feature synthesising networks in a unified way.
- To reduce the bias towards source classes during testing, we propose to transfer-learn a target class from the most similar source class based on the *pointwise mutual information* (*pmi*) score.
- We present an extensive empirical evaluation on several fine-grained datasets to demonstrate the superior state-of-the-art performance of the proposed method compared to contemporary GZSL and ZSL methods.

7.3 Proposed Method

7.3.1 Proposed GZSL

The proposed method addresses the limitations of existing embedding learning and feature synthesising methods that ignore individual attributes for guiding feature embedding construction. The method shown in Figure 7.2 comprises two networks: 1)

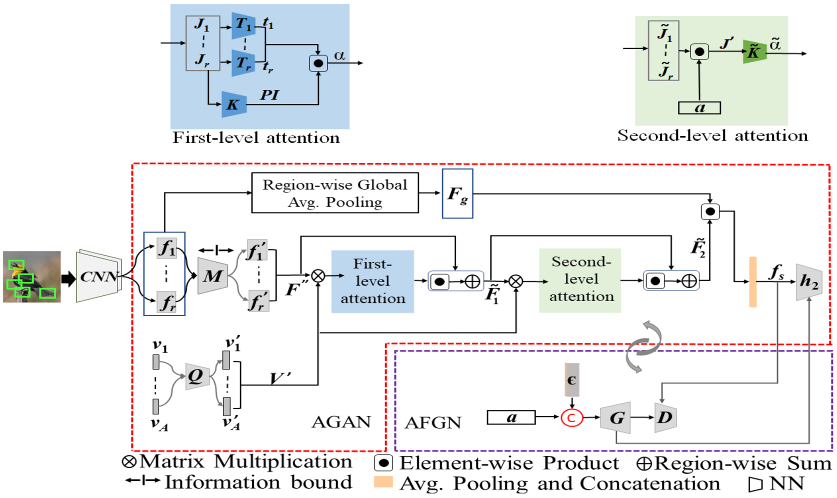


Figure 7.2: Block diagram of the proposed GZSL method. The red and purple blocks in dashed lines represent the Attribute Guided Attention Network (AGAN) and the Adversarial Feature Generation Network (AFGN), respectively. Best viewed in colour.

Attribute Guided Attention Network (AGAN); and 2) Adversarial Feature Generation Network (AFGN). AGAN is the embedding learning part of the proposed method. The attention mechanism is placed in AGAN. First, AGAN constructs feature embedding using the attention mechanism and leverages the feature embedding for the GZSL task. Then, the feature synthesising part, AFGN, uses the constructed feature embedding to learn the generation of attribute-weighted visual features adversarially. Furthermore, AGAN and AFGN are mutually optimised to improve each other's performance, i.e., AGAN takes supervision from AFGN for optimising the constructed feature embedding, and AFGN takes supervision from AGAN to generate visual features. This optimisation is performed by minimising our designed losses.

7.3.1.1 Attribute Guided Attention Network

First, we select local visual regions and preserve region-wise global information. Then we statistically bound the local regions to filter out irrelevant information. Then, the two-step dense attention mechanism constructs an attribute-weighted features. In Figure 7.2, the blue and green shaded parts on AGAN show the two levels of attention mechanism, respectively. Then AGAN constructs the feature embedding utilising the output of the attention mechanism. The feature embedding holds global representation, redundancy-free, and attribute-weighted local visual information (the probability of the most relevant attribute to the visual regions and the likelihood of having that attribute in the class). Finally, a classifier utilises the feature embedding to infer class decisions. To reduce the source class bias while learning the classifier, we optimise a transfer learning loss.

Constructing Visual Regions For simplicity and consistency, in line with [206] and [126], we divide an image I into r equal sized regions, $I_1, \ldots, I_r$. We use a CNN to extract features for the r regions. For example, the feature vector of the i^{th} region is $f_i = f_\theta(I_i)$, where θ denotes parameters of the CNN. Note that the CNN is frozen.

Exploring Global Information To learn global discriminative features compatible with the local region features, we apply region-wise global average pooling on the local feature vectors $F = \{f_i\}_{i=1}^r$. This operation provides us with a feature vector $F_g \in \mathbb{R}^r$, where F_{g_i} represents the average global information of the i^{th} local region feature f_i.

Learning Relevant Information We want to reduce highly irrelevant information from the extracted local feature space $F = \{f_i\}_{i=1}^r$ by restricting the information propagation from F to F'. This will reduce the interruption of redundant information in the attribute-weighted feature embedding.

As shown in Figure 7.2, $F = \{f_i\}_{i=1}^r$ is the input to M and $F' = \{f'_i\}_{i=1}^r$ is the output from M. Therefore, we aim to bind irrelevant information propagation from the inputs of M to the outputs of M. To execute this, we have to place a information propagation bound in M. We use Mutual information (I) to bound M network to filter irrelevant information. Mutual information between two random variables F and F' can be related to the marginal $H(F')$ and conditional $H(F'|F)$ entropy as $I(F; F') = H(F') - H(F'|F)$. We want $I(F; F')$ to be less than an upper bound so that only relevant information in F is passed to F' through M to help reduce noise. The upper bound is found empirically and we train M to learn to hold $I(F; F')$ less than the bound.

Since the extracted feature space is high dimensional, the estimation of mutual information may be difficult. Therefore, we adopt a variational upper bound of I [92] to compute $I(F; F')$ as,

$$I(F'; F) \leq \mathbb{E}_{p(f)} \left[D_{KL} \left[p_M(f'|f) \| r(f') \right] \right], \tag{7.1}$$

where $p_M(f'|f)$ is the conditional probability of the region features f', which holds only important information conditioned on the extracted real region-features f. D_{KL} and $r(f')$ denote the Kullback-Leibler divergence and variational approximation of the marginal probability distribution of f', respectively. Note that we do not reduce feature regions or filter out redundancy from global features [151], which may lose important visual information and harm the image's visual feature representation. We remove redundancy from the feature regions to use only the relevant information within a region.

First-level Dense Attention. Now, we aim to construct a dense connection i.e, every attribute is to be connected to every visual region to explore the relevance between every attribute and every visual region. Therefore, we form a matrix F''. The rows of F'' represent the bounded features of each region. The corresponding attribute semantic vectors $\boldsymbol{v}$ are converted to V' matrix by using Q network, where the A^{th} row represents the $\boldsymbol{v}'^{th}_A$ attribute. Both M and Q are neural networks with non-linear activation function ReLU.

F'' and V' are fused as $J = F'' \otimes V'$, where $\otimes$ denotes matrix multiplication, $F'' \in \mathbb{R}^{r \times m}$, r is the number of regions and m is the dimension of region features f'. Similarly, $V' \in \mathbb{R}^{n \times A}$, A denotes the number of attributes and n denotes the dimension of attribute vectors v', where $m = n$ is ensured by M and Q networks. The matrix multiplication ensures a dense connection between F'' and V' as the product contains information of every attribute (columns) in every regional feature (rows). The output of the matrix multiplication is $J \in \mathbb{R}^{r \times A}$ and we denote the i^{th} region as J_i, where $J_i \in \mathbb{R}^A$.

Existing attention-based GZSL works [9,126,207], have adopted soft attention [208] to only predict. On the other hand, we propose to use soft attention to predict the most relevant attribute to every region besides predicting the presence of attributes in the regions. A conceptual view of assigning attention to a region is shown in Figure 7.3.

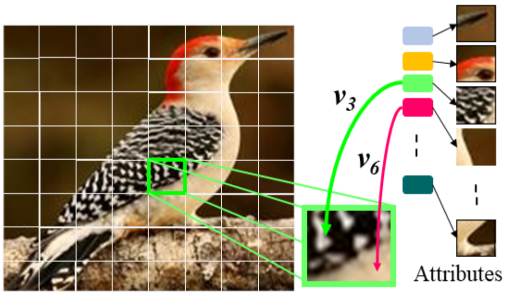


Figure 7.3: A conceptual illustration of the first-level attention mechanism of the proposed attribute-weighted visual feature embedding. Note that if a region has more than one attributes, then the attention of the attribute having highest confidence is assigned to the region, e.g., attribute $\boldsymbol{v}_3$ ('wing pattern stripped') wins over $\boldsymbol{v}_6$ ('belly colour white'). Thus, presence of attribute and the most relevant attribute to a region is attended.

The K network takes J matrix and applies a soft-attention normalisation to set different degrees of attention to the r regions. The attentions indicate the confidence of the presence of attributes in a region. We learn individual soft-attentions for every one of the r regions using $\{T_i\}_{i=1}^r$ neural networks, which encourages to learn to attend only the most relevant attribute to a region. This definitive attention assignment facilitates the embedding to hold fine distinctive information. The attention assignment in K and $\{T_i\}_{i=1}^r$ networks are performed as follows,

$$\begin{aligned} PI &= \text{softmax}(\tanh(J^\top W_B)W_A), \\ t_i = \text{softmax}(\tanh(J_i W_{TA_i})W_{TB_i}), \alpha_i &= \lambda_\alpha PI_i H_{t_i}, \end{aligned} \tag{7.2}$$

where K is a neural network with learned parameters W_A and W_B and output $PI \in \mathbb{R}^r$. W_{TA_i} and W_{TB_i} are learned parameters of T_i^{th} network. The softmax outputs of T_i is $t_i \in \mathbb{R}^A$, which can be treated as soft attentions of the attributes on the i^{th} feature region. The attribute yielding the highest softmax probability is most likely to have greater relevance to the i^{th} feature region than others. Thus, we consider only the highest softmax probability H_{t_i}. It also helps the attention module to focus and learn only one attribute per visual region for better discriminative learning. Similarly, we compute the soft attentions for each of the r regions using $\{T_i\}_{i=1}^r$. α_i denotes the attention and the parameter λ_α helps to avoid negligible attention. Note, to handle $\{T_i\}_{i=1}^r$ networks simultaneously, we use depth-wise (grouped) convolution; please see Section 7.4.3 for more details.

We obtain the weighted feature regions by applying the inferred soft attention as $\widehat{F''}_i = \alpha_i F''_i$. To preserve both noise-free and semantic guided visual information, we combine the attribute-weighted region features with the redundancy-free region features as follows,

$$\tilde{F}_{1_i} = F''_i \oplus \widehat{F''}_i, \tag{7.3}$$

where, $\oplus$ denotes region-wise summation.

Second-level Dense Attention. To further infuse the probability of the presence of an attribute in the class in $\tilde{F}_{1_i}$ and boost the weighted feature regions for handling more sophisticated cases, we apply another level of attention mechanism. This assists the attention mechanism in learning to assign a higher weight to a region that may contain an attribute which is more likely to be present in the class samples and helps in making a better class decision.

First, we construct the visual-semantic matrix as $\tilde{J} = \tilde{F}_1 \otimes V'$, which has the similar dimensional properties as J matrix. Then, we combine the class semantic vector a as $J' = \tilde{J}a$, where a vector is multiplied to each row of $\tilde{J}$ matrix element-wise and $J' \in \mathbb{R}^{r \times A}$. The second-level soft attention $\tilde{\alpha}$ is computed by using J' matrix and $\tilde{K}$ network similar to the first part (PI) of (7.2), i.e., $\tilde{\alpha} = \text{softmax}(\tanh(J'^{\top} W'_B) W'_A)$, where W'_B and W'_A are learned parameters of $\tilde{K}$ network. The feature embedding $\tilde{F}_2 \in \mathbb{R}^{r \times m}$ is constructed by summation of $\tilde{F}_1$ and $\tilde{F}'_1 = \tilde{\alpha}\tilde{F}_1$ as (7.3). Note that, since the information of the most relevant attribute to a region is propagated into the second-level and beyond through the embedding $\tilde{F}_1$, we do not use $\{T_i\}_{i=1}^{r}$ neural networks in the second-level. Besides, we empirically found that using $\{T_i\}_{i=1}^{r}$ in the second-level does not facilitate the attention mechanism significantly.

Feature Embedding. To hold the global information in the feature embedding, we apply a region-wise product between $\tilde{F}_2$ and F_g, i.e., the feature vector F_g is element-wise multiplied to all the column vectors of $\tilde{F}_2$ matrix and the dimension of $\tilde{F}_2$ is preserved as is. This operation infuses the region-based global information to $\tilde{F}_2$. To retain all the extracted information in the final embedding, we apply an average pooling over the m-dimension of $\tilde{F}_2$. Then, we concatenate the pooled features and form the final feature embedding fs, where $fs \in \mathbb{R}^m$.

AGAN-GZSL Task. Finally, f_s is fed into the classifier h_2, which is a neural network with one hidden fully-connected layer and a softmax layer. The classifier takes f_s as input and produces $|C_s + C_t|$-dimensional output, where the first $|C_s|$ indices represent the source classes and the remaining indices represent the target classes. The class scores are computed as $p(s_i) = \exp(s_i)/\sum_{c \in |C_s|} \exp(s_i^c)$, where $s = h_2(f_s)$, $h_2(f_s) \in \mathbb{R}^{|C_s + C_t|}$, $|C_s + C_t|$ is the total number of source and target classes.

7.3.1.2 Adversarial Feature Generation Network

In this section, we present the proposed AFGN, which utilises final feature embedding f_s from AGAN to learn to generate features that are highly related to the attributes for fine-grained classification.

The AFGN can adopt any adversarial feature synthesising GZSL method. In this work, we adopt a feature generation method f-WGAN [141], which has a visual feature generator G and a discriminator D. The f-WGAN takes random Gaussian noise ϵ and the class semantic vector a as inputs and learns to generate a visual feature $\tilde{x} \in \mathcal{X}$ of class y. The idea is to train G to generate features of the source class images x^s conditioned on a_c^s so that during testing, the generator G can repurpose its learned knowledge to generate target class features only from a_c^t. In f-WGAN [141], the global features (i.e., x^s) are used as the real features to guide G. On the contrary, we propose to utilise our attribute-weighted features for the guidance. Not only our features are associated with attribute attention, they also hold redundancy-free information. For converting the semantic vectors to visual features, the usage of f_s inferred from AGAN will assist the AFGN to follow the underlying dependency between the semantic and visual feature spaces. Therefore, we optimise,

$$\mathcal{L}_{WGAN} = E[D(f_s, a)] - E[D(\tilde{x}, a)] - \lambda E\left[(\|\nabla_{\hat{x}} D(\hat{x}, a)\|_2 - 1)^2\right], \tag{7.4}$$

where $\tilde{x} = G(\epsilon, a)$, λ denotes the penalty coefficient, and $\hat{x} = \eta f_s + (1-\eta)\tilde{x}$ with $\eta \sim U(0,1)$. To further ensure the learned features hold discriminative properties suitable for classification and less bias towards source classes, we use the h_2 from AGAN as follows,

$$\mathcal{L}_{cls} = -E_{\tilde{x} \sim p_{\tilde{x}}(\tilde{x})}[\log P(y \mid \tilde{x}; \theta)], \tag{7.5}$$

where y is the true class label of $\tilde{x}$ and $P(y \mid \tilde{x}; \theta)$ denotes the probability of $\tilde{x}$ being predicted as y by h_2.

7.3.2 Optimisation

In this section, we present the loss optimisation details of the proposed method.

7.3.2.1 Mutual Learning

Since both AGAN and AFGN use the attribute-weighted feature embedding f_s to learn their tasks, we utilise both networks to assist one-another through mutual learning. We define the mutual learning losses for AGAN and AFGN networks as follows,

$$\mathcal{L}_{m1} = \frac{1}{2}||f_s - \tilde{x}||_2^2, \mathcal{L}_{m2} = \frac{1}{2}||\tilde{x} - f_s||_2^2, \tag{7.6}$$

where, $\tilde{x} = G(\epsilon, a)$. By optimising $\mathcal{L}_{m1}$, AGAN utilises the construction power of G in AFGN to facilitate its embedding learning. On the other hand, AFGN uses the learned embedding in AGAN to improve its construction ability by optimising $\mathcal{L}_{m2}$..

For mutual training we need to optimise both L_{m1} and L_{m2} in every training iteration. In every iteration we first optimise L_{m1} and then L_{m2} (Algorithm 2). To optimise L_{m1} (7.6), first, we feed a batch of samples to AGAN to get f_s, then we pass the same batch through G in AFGN (in eval mode) to get $\tilde{x}$, and finally, compute L_{m1}. Similarly, to optimise L_{m2} (7.6), first, we feed a batch of samples to AFGN to get $\tilde{x}$, then we pass the same batch through AGAN (in eval mode) to get f_s, and compute L_{m2}.

7.3.2.2 Loss optimisation in AGAN

For the source classes, we optimise the standard cross-entropy loss as,

$$\mathcal{L}_{ce} = \frac{1}{ns} \sum_{i=1}^{ns} \mathcal{L}(h_2(f_{s_i}), y_i), \tag{7.7}$$

where y_i is the true class label of f_{s_i} and ns denotes the number of samples.

To smooth out bias towards source classes, we hope to loosely learn a target class from the knowledge of its closest source class. Thus, we propose to optimise the following loss over the target class indices in h_2 in one-vs-rest fashion,

$$\mathcal{L}_u = \sum_{i=1}^{ns} \sum_{j=C_s+1}^{C_s+C_t} pmi_{ij} \log P(y=j|f_{s_i}) - (1 - pmi_{ij}) \log(1 - P(y=j|f_{s_i})). \tag{7.8}$$

Here, pmi_{ij} is the similarity measure between the class of i^{th} source sample and j^{th} target class and $P(y=j|f_{s_i})$ means the probability of j^{th} index given the feature of its closest source class.

To measure class similarity, we adopt pointwise mutual information (pmi). In information theory, pmi measures association and co-occurrence between two events of two discrete random variables. In the fine-grained GZSL setup, the target classes share many attributes with the source classes. Therefore, the random variables of the class semantic vectors of source classes a_c^s will pose significant statistical dependence with that of target classes a_c^t. This implies that the target classes will produce higher pmi for similar source classes in the class semantic vector space. We convert a_c of the source and target classes to probability distributions by applying a softmax function.

Let, $Z_{\mathcal{A}^s}$ and $Z_{\mathcal{A}^t}$ represent the converted probability distributions of the source and target classes. The pmi between two individual events $Z^i_{\mathcal{A}^s}$ and $Z^j_{\mathcal{A}^t}$ of the two discrete random variables $Z_{\mathcal{A}^s}$ and $Z_{\mathcal{A}^t}$ can be computed as,

$$pmi(Z^i_{\mathcal{A}^s}; Z^j_{\mathcal{A}^t}) = \log \frac{P(Z^i_{\mathcal{A}^s}, Z^j_{\mathcal{A}^t})}{P(Z^i_{\mathcal{A}^s}) P(Z^j_{\mathcal{A}^t})}. \tag{7.9}$$

We construct the joint probability distribution as $Jn = Z_{\mathcal{A}^t} \cdot {Z_{\mathcal{A}^s}}^\top$ (tensor Jn has a dimension of $C_t \times C_s$), and $Z_{\mathcal{A}^t}$ and $Z_{\mathcal{A}^s}$ matrices hold the dimension of $C_t \times a_c^t$ and $C_s \times a_c^s$. The marginals are computed from the summation of rows and columns of Jn. The final objective for the AGAN network becomes,

$$\begin{aligned} &\mathcal{L}_{ce} + \lambda_p \mathcal{L}_u + \lambda_{m1} \mathcal{L}_{m1} \\ &\text{s.t. } \mathbb{E}_{p(f)} \left[D_{KL} \left[p_M(f' \mid f) \| r(f') \right] \right] \leq \gamma, \end{aligned} \tag{7.10}$$

where λ_p and λ_{m1} are a hyper-parameters to weight the losses for target classes and mutual learning respectively, and γ is the MI bound.

7.3.2.3 Loss optimisation in AFGN

The final objective of AFGN is as follows,

$$\min_G \max_D \mathcal{L}_{WGAN} + \lambda_{cls} \mathcal{L}_{cls} + \lambda_{m2} \mathcal{L}_{m2}. \tag{7.11}$$

Here, λ_{cls} and λ_{m2} are hyper-parameters for weighting the contribution of $\mathcal{L}_{cls}$ in the optimisation and mutual learning respectively. We train AGAN and AFGN in an end-to-end fashion. During each iteration, first, we sample a mini-batch from $(x_i^s, y_i^s)_{i=1}^{n_s}$ and Gaussian noise ϵ. Then we update the learnable components of AGAN by (7.10). Finally, we update the learnable components of AFGN by (7.11).

7.3.3 Training Phase

The training procedure of the proposed method is summarised in Algorithm 2, where e denotes the number of steps to train discriminator D. We have used 5 steps. Please note that in our experiments, we extract CNN region features $f_{i i=1}^{r}$ for training images prior to training the proposed method.

Algorithm 2 Training Procedure

Input: $\mathcal{D}^s$; a_c^s; $\boldsymbol{v}$; a_c^t; AGAN components; ϵ; AFGN components.
Output: Trained AGAN and G from AFGN.

```
while not converged do
    Sample mini-batch from extracted CNN region features, $a_c^s$, and $\epsilon$;
    Compute $F_g$ by region-wise avg. pooling;
    Compute $I(F; F')$ between inputs ($f_{i\,i=1}^{\,r}$) and outputs ($f'^{\,r}_{i\,i=1}$) of $M$ (7.1);
    Form $F''$ matrix using $f'^{\,r}_{i\,i=1}$;
    Convert $\boldsymbol{v_i}_{i=1}^{A}$ to $V'$ matrix using $Q$;
    Perform matrix multiplication between $F''$ and $V'$ to get $J$;
    Compute soft attentions through $K$ and $T_{i\,i=1}^{\,r}$ networks by using (7.2);
    Compute $\tilde{F_1}$ by (7.3);
    Perform matrix multiplication between $\tilde{F_1}$ and $V'$ to get $\tilde{J}$ ;
    Perform element-wise product between $\tilde{J}$ and $a$ to get $J'$;
    Compute $\tilde{\alpha}$ through $\tilde{K}$ network similar to first part of (7.2);
    Compute $\tilde{F_2}$ by summation of $\tilde{F_1}$ and $\tilde{\alpha}\tilde{F_1}$ similar to (7.3);
    Perform avg. pooling and concatenation on $\tilde{F_2}$ to get $f_s$;
    Feed $f_s$ to $h_2$ for class probabilities;
    Use $a_c^s$ and $\epsilon$ to get $\tilde{x}$ from $G$;
    Compute losses by (7.7)–(7.10) and $\mathcal{L}_{m1}$ (7.6);
    Update learnable components of AGAN;
    Sample mini-batch, $\epsilon$, $a_c^s$, and compute $f_s$ from AGAN;
    for $e$ steps do
        Update $D$ by (7.4);
    end for
    Sample mini-batch, $\epsilon$, $a_c^s$, and compute $f_s$ from AGAN;
    Update $G$ by (7.4)–(7.6);
end while
```

7.3.4 Testing Phase

Once the AGAN is trained, we formulate the classification score of a test instance as $P_{GZSL}\,(x_i) = \max_i \{s_i\}_{i=1}^{C_t}$ and $P_{ZSL}\,(x_i) = \max_i \{s_i\}_{i=C_s+1}^{C_t}$. For AFGN, we use the trained generator G and re-sampled ϵ to generate multiple synthetic features for every source and target class. Then, we learn a separate supervised classifier, which produces $|C_s + C_t|$ and $|C_t|$ dimensional outputs for GZSL and ZSL. We define the classification loss as $\mathcal{L}_{h_{AFGN}} = -E_{x' \sim p'}[\log P(y \mid x'; \theta_{h_{AFGN}})]$, where x', y, and p' denote samples of the newly formed training dataset, the true class label of x', and distribution of the new training dataset respectively. $P(y \mid x'; \theta_{h_{AFGN}})$ represents the probability of x' being recognised as y.

7.4 Experimental Studies

In this section, we describe the datasets, evaluation protocol, implementation details, experimental outcomes, hyper-parameter settings, ablative analysis, and learned attention visualisation.

7.4.1 Datasets

In line with fine-grained GZSL method [126], we conduct our experiments on three popular fine-grained datasets, Caltech-UCSD Birds-200-2011 (CUB) [209], SUN Attribute (SUN) [210], and Animals with Attributes 2 (AWA2) [211]. We further extend our experiments to Animals with Attributes 1 (AWA1) [108] dataset, which is a version of AWA2 dataset. We follow [201], to split the total classes into source and target classes on each dataset.

CUB contains a total of 11,788 images of 200 classes of fine-grained bird species, among them, 150 are selected as source classes, and the remaining 50 classes are treated as the target or unseen classes. **SUN** is composed of 14,340 images with 717 categories of scenes. This dataset is widely used for fine-grained scene recognition and GZSL. The number of source and target classes used for GZSL are 645 and 72, respectively. **AWA1** consists of 30,475 images of 50 different sub-ordinate classes of animals. For GZSL, 40 classes are used as source, and 10 are used as target classes. **AWA2** has 40 source and 10 target classes comprising 37,322 images in total.

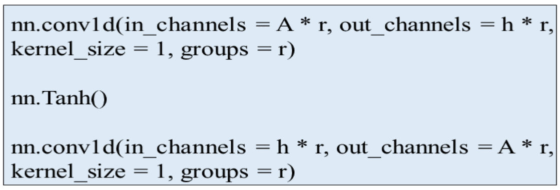
7.4.2 Evaluation Metrics

We evaluate the performance of our method by per-class Top-1 accuracy. For the source domain, we will evaluate the Top-1 accuracy on source classes denoted as S. For the target domain, the Top-1 accuracy on the target classes is represented as T. For evaluating the total performance of GZSL, we compute the harmonic mean in line with [201] as, $H = (2 \times S \times T)/(S + T)$.

7.4.3 Implementation Details

In our experiments, we extract a feature map of size $7 \times 7 \times 2048$ from the last convolutional block of pre-trained ResNet-101 and use it as a set of features from 7×7 local regions. It is worth mentioning that the pre-trained ResNet-101 model is only used for feature extraction and not fine-tuned in the training procedure. In AGAN, the networks M, Q, and h_2 are fully-connected neural networks with no hidden layers. The networks K and $\tilde{K}$ have only one hidden layer.

Grouped Convolution Attention We replace $\{T_i\}_{i=1}^{r}$ fully-connected neural networks with grouped 1D convolutional block. Everyone of the r T_i networks has two linear layers, one is followed by *tanh* function and the other has a *softmax* function after it. We replace the linear layers, as shown in Figure 7.4. We use a kernel size of 1 in the convolutional block to mimic the linear or fully-connected neural layers. The input of the convolution block has $b \times (A * r) \times 1$-dimension, where b, A, r, and $*$ denote the batch size, number of attributes, number of regions, and multiplication respectively. In Figure 7.4, h denotes the size of the hidden layer of T_i. Note that in the first conv layer, ${r_i}^{th}$ group will be connected to only A input channels and in the second conv layer ${r_i}^{th}$ group will be connected to h input channels (hidden layer neurons). Thus, the weights of different groups in the convolution block are not shared, which supports our goal to learn separate attentions for r regions parallelly. After the second conv layer we obtain an output of $b \times (A * r) \times 1$-dimension which is reshaped to $b \times r \times A$-dimension for applying softmax over the A attributes of r regions.



```
nn.conv1d(in_channels = A * r, out_channels = h * r,
kernel_size = 1, groups = r)

nn.Tanh()

nn.conv1d(in_channels = h * r, out_channels = A * r,
kernel_size = 1, groups = r)
```

Figure 7.4: Grouped convolution pseudo-code.

In AFGN, since the generator has to produce fully-connected features from conditional input, we maintain a full fully-connected structure of the generator for efficiency, i.e., the generator has only one hidden fully-connected layer. The discriminator has no hidden layers in the structure.

The threshold γ for MI bound in the region features is cross-validated between $[0.01, 0.05]$. The attribute semantic vectors $\boldsymbol{v}$ for all datasets are extracted from Wikipedia articles trained GloVe model [200]. The attention balancing hyper-parameter λ_α is set to 10. Adam solver with $\beta_1 = 0.5$, $\beta_2 = 0.999$ and learning rate 0.0001 is used for optimisation. The suitable hyper-parameters setting across all datasets is as follows, $\lambda_p \in [0.1, 0.2, 0.3, 0.4]$, $\lambda_{m1} = 0.1$, $\lambda_{cls} = 0.1$, and $\lambda_{m2} = 0.2$.

Computation time The proposed method is trained using an NVIDIA Quadro P5000 GPU for 100 epochs with a batch size of 32. Each epoch takes approximately 50 seconds to execute. Thus, the total training time is approximately 5000 seconds. We compute the inference or testing time of AGAN in two ways: 1) include CNN (ResNet-101) feature extraction in the process and 2) exclude the CNN (ResNet-101) feature extraction from the process. For Case 1, the inference time for a sample is approximately 0.81 seconds and for Case 2, it is approximately 0.01 seconds. For AFGN, the inference time for a sample is approximately 0.006 seconds.

7.4.4 Results and Analysis

In this section, we analyse the evaluation of the proposed and contemporary GZSL methods. The ZSL results of LATEM [137], DEM [212], and SGMAL [122] are adopted from SGMAL [122], GZSL results of LATEM [137] and DEM [212] are taken from ASPN [146], and the results of other compared methods are obtained from their corresponding published articles. For a fair comparison, we compare both AGAN and AFGN with only inductive methods and synthesise 400 features per class for comparing AFGN's performance. In Tables 7.1 and 7.2, △ and □ denote embedding learning and feature synthesising methods, respectively, and '-' represents that the results are not reported.

7.4.4.1 Generalised Zero-Shot Learning

Table 7.1 shows that both AGAN and AFGN achieves more Harmonic mean H compared to contemporary methods. H the main indicator of how well a GZSL method

Table 7.1: Performance comparison. T and S are the Top-1 accuracies tested on target classes and source classes, respectively, in GZSL. H is the harmonic mean of T and S.

Approach	Model	GZSL											
		CUB			SUN			AWA1			AWA2		
		T	*S*	*H*	*T*	*S*	*H*	*T*	*S*	*H*	*T*	*S*	*H*
△	LATEM [137] (2016)	15.2	57.3	24.0	-	-	-	7.3	71.7	13.3	11.5	77.3	20.0
	DEM [212] (2017)	19.6	57.9	29.2	-	-	-	32.8	84.7	47.3	30.5	86.4	45.1
	DCN [139] (2018)	28.4	60.7	38.7	25.5	37.0	30.2	25.5	84.2	39.1	-	-	-
	AREN [121] (2019)	38.9	78.7	52.1	19.0	38.8	25.5	-	-	-	15.6	92.9	26.7
	CRnet [213] (2019)	45.5	56.8	50.5	34.1	36.5	35.3	58.1	74.7	65.4	-	-	-
	TCN [113] (2019)	52.6	52.0	52.3	31.2	37.3	34.0	49.4	76.5	60.0	61.2	65.8	63.4
	DVBE [214] (2020)	53.2	60.2	56.5	45.0	37.2	40.7	-	-	-	63.6	70.8	67.0
	DAZLE [126] (2020)	56.7	59.6	58.1	52.3	24.3	33.2	-	-	-	60.3	75.7	67.1
	VSG-CNN [203] (2020)	52.6	62.1	57.0	30.3	31.6	30.9	-	-	-	60.4	75.1	67.0
	APN [140] (2020)	65.3	69.3	67.2	41.9	34.0	37.6	-	-	-	56.5	78.0	65.5
	AGAN (Ours)	67.9	71.5	**69.7**	40.9	42.9	**41.8**	65.1	83.2	**73.0**	64.1	80.3	**71.3**
□	SE-GZSL [143] (2018)	41.5	53.3	46.7	40.9	30.5	34.9	56.3	67.8	61.5	58.3	68.1	62.8
	f-CLSWGAN [141] (2018)	43.7	57.7	49.7	42.6	36.6	39.4	57.9	61.4	59.6	-	-	-
	f-VAEGAN-D2 [144] (2019)	48.4	60.1	53.6	45.1	38.0	41.3	-	-	-	57.6	70.6	63.5
	LisGAN [152] (2019)	46.5	57.9	51.6	42.9	37.8	40.2	52.6	76.3	62.3	-	-	-
	GMN [142] (2019)	56.1	54.3	55.2	53.2	33.0	40.7	61.1	71.3	65.8	-	-	-
	RFF-GZSL (softmax) [151] (2020)	52.6	56.6	54.6	45.7	38.6	41.9	59.8	75.1	66.5	-	-	-
	TF-VAEGAN [215] (2020)	52.8	64.7	58.1	45.6	40.7	43.0	59.8	75.1	66.6	-	-	-
	ASPN [146] (2020)	50.7	61.5	55.6	-	-	-	58.0	85.7	69.2	46.2	87.0	60.4
	E-PGN [216] (2020)	52.0	61.1	56.2	-	-	-	62.1	83.4	71.2	52.6	83.5	64.6
	APN [140] + f-VAEGAN-D2 [144] (2020)	65.7	74.9	70.0	49.4	39.2	43.7	-	-	-	62.2	69.5	65.6
	AFGN (Ours)	69.8	77.1	**73.2**	53.1	45.9	**49.2**	67.5	83.8	**74.7**	68.1	82.9	**74.7**

Table 7.2: Performance comparison of ZSL tasks.

Approach	Model	CUB	SUN	AWA1	Approach	Model	CUB	SUN	AWA1
△	LATEM [137]	49.4	-	78.4	□	SE-GZSL [143]	60.3	64.5	83.8
	DEM [212]	51.8	-	80.3		cycle-CLSWGAN [145]	58.6	59.9	66.8
	S^{2}GA (2-attention layer) [9]	68.9	-	-		LisGAN [152]	58.8	61.7	70.6
	S^{2}GA (3-attention layer) [9]	68.5	-	-		GMN [142]	64.3	63.6	71.9
	SGMAL [122]	70.5	-	83.5		f-CLSWGAN [141]	57.3	60.8	68.2
	TCN [113]	59.5	61.5	70.3		SABR [217]	65.2	62.8	-
	DAZLE [126]	67.8	-	-		f-VAEGAN [144]	72.9	65.6	-
	APN [140]	72.0	61.6	68.4		APN [140] +f-VAEGAN-D2 [144]	73.8	65.7	71.7
	AGAN (Our)	**74.9**	**66.5**	**88.7**		AFGN (our)	**78.5**	**69.8**	**89.1**

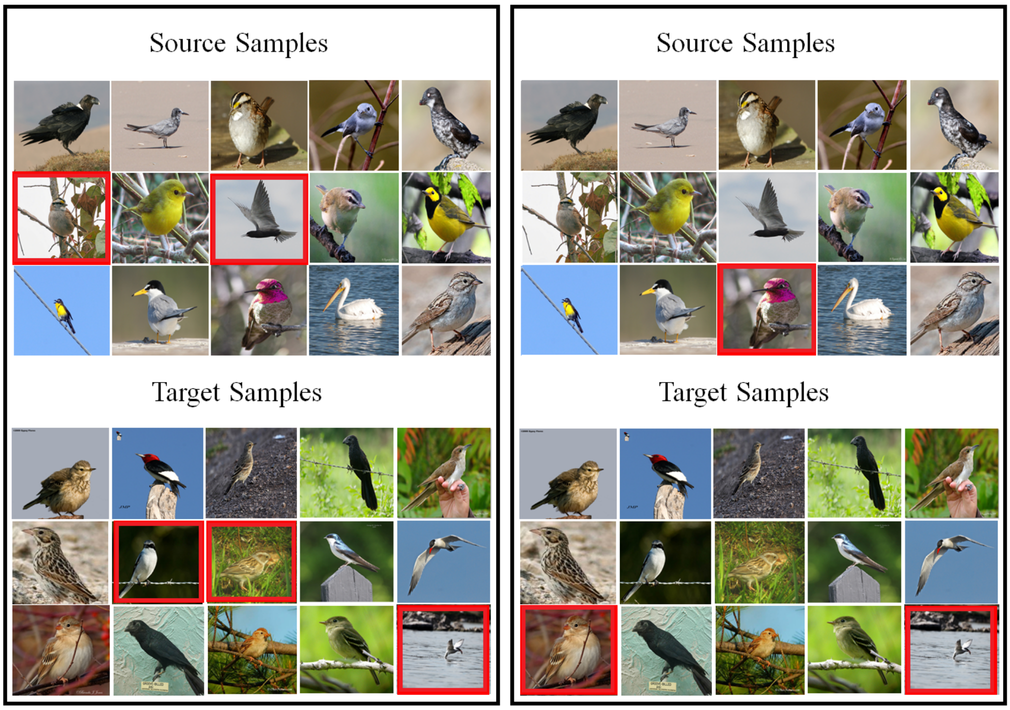
performs. AGAN and AFGN also significantly outperform the contemporary methods for the majority of the GZSL tasks. Unlike embedding learning methods, feature synthesising methods leverage supervised training on synthesised data during testing and outperform embedding learning methods. Similarly, AFGN outperforms AGAN. AGAN outperforms all the compared embedding learning methods, which either use local or global feature embedding. This indicates that the proposed method's feature embedding holds finer discriminative information required for fine-grained tasks. The improved performance of AGAN also proves that both global and local information plays a vital role in fine-grained GZSL.

APN [140] is the closest competitor, which has a global feature learning module (BaseMod) along-with a local feature learning module (ProtoMod). AGAN outperforms APN significantly. AFGN increases the accuracy of GZSL by a large margin compared to APN + f-VAEGAN-D2 [144]. This means the proposed method is more effective for GZSL tasks.Considering fine-grained attention-based GZSL methods, DAZLE [126] is the closest competitor, which leverages only local region-based features. However, DAZLE restricts the embedding space to the number of selective attributes. In comparison, we preserve all local region features highlighted by the most relevant attributes to the regions and the global information corresponding to the local regions. The improved performance of AGAN and AFGN verifies the effectiveness of our feature embedding.

Concerning irrelevant information removing GZSL methods, RFF-GZSL [151] filters out redundant information from global features. On the other hand, the proposed method preserves global information on an average to hold the generic trend of deep classifier features and removes redundancy from local regions to reduce the interruption of irrelevant information. The higher performance of AGAN and AFGN validates that the proposed feature embedding holds better distinctive and necessary information. Compared to other methods, AGAN reduces the source domain bias by optimising the target loss $\mathcal{L}_u$ and makes better knowledge transfer from source to target classes. AFGN follows the same trend as it uses the discriminative knowledge of h_2. We present some qualitative results of AGAN and AFGN on the CUB dataset's GZSL task in Figure 7.5. The samples shown in the figure are selected from the test set. The results show both AGAN and AFGN have minimal misclassifications.

7.4.4.2 Zero-Shot Learning

The performance on ZSL tasks (CUB, SUN, AWA1) of different methods is shown in Table 7.2. As expected, the results show that the target class accuracy of all ZSL methods



(a) AGAN GZSL results.

(b) AFGN GZSL results.

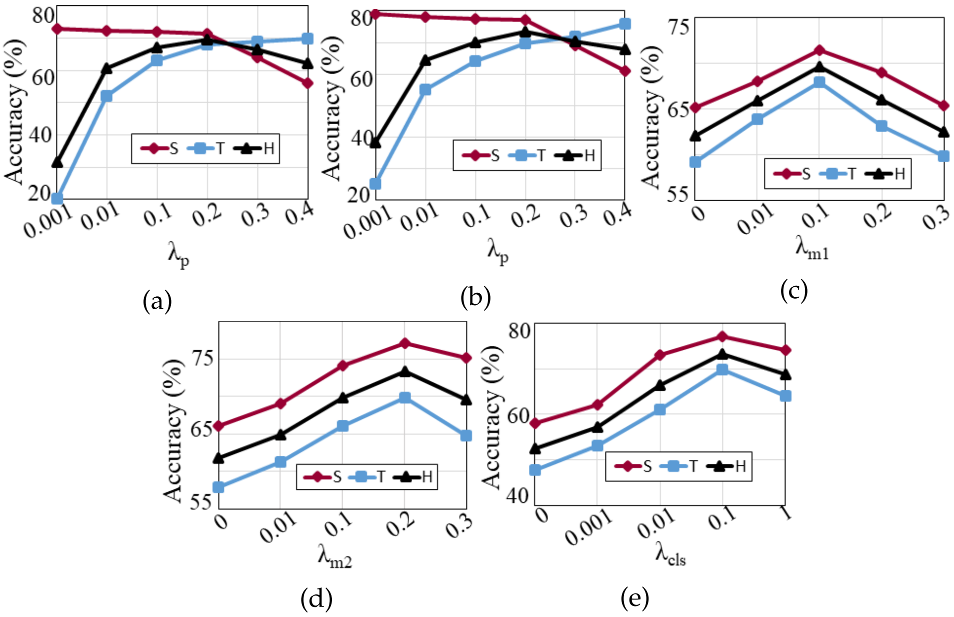
Figure 7.5: Qualitative results on GZSL task of the CUB dataset. Images with red boxes show misclassifications by AGAN and AFGN.

is higher than the GZSL tasks. The proposed AGAN and AFGN perform better than contemporary methods. The improved performance of the networks for ZSL tasks shows that the trained networks gain the ability to generalise well to unseen target classes even in the conventional ZSL setup, which is encouraged by the optimisation of $\mathcal{L}_u$ based on the pmi similarity.

7.4.4.3 Hyper-parameters Analysis

For studying the trend of GZSL accuracy of AGAN and AFGN in different hyper-parameters (λ_P, λ_{m1}, λ_{m2}, and λ_{cls}) settings, we plot the graphs shown in Figure 7.6.

Figures 7.6a and 7.6b show the performance of AGAN and AFGN with various λ_P setups respectively. In case of both the networks, we observe that the source accuracy



(a) (b) (c)

(d) (e)

Figure 7.6: Effect of varying the hyper-parameters in the GZSL performance on the CUB dataset.

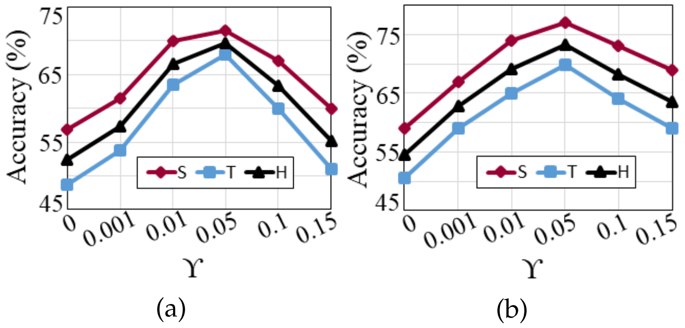
depicts a sharp decreasing pattern after $\lambda_P = 0.2$ while the target accuracy starts to surpass the source accuracy a little after that point. This means the networks gradually lose the capability to recognise the source domain samples correctly. The harmonic mean H achieves the optimal performance at $\lambda_P = 0.2$ and decreases soon after that. Thus, we find the value of $\lambda_P = 0.2$ optimal for the task. Note that for other datasets we cross-validate λ_P in the range $[0.001, 0.01, 0.1, 0.2, 0.3, 0.4]$.

The effect of different settings of the hyper-parameters weighting the mutual loss λ_{m1} in AGAN and λ_{m2} in AFGN are shown in Figures 7.6c and 7.6d respectively. We observe that the optimal performance in AGAN is achieved when $\lambda_{m1} = 0.1$, and the source and target classes performances are harmed when the value of λ_{m1} is greater than that. On the other hand, the AFGN network has low accuracies for fewer values of λ_{m2} and achieves optimal performance when λ_{m2} is 0.2. This means the AFGN is more facilitated by mutual learning compared to AGAN.

Figure 7.6e illustrates the performance of AFGN in different settings of λ_{cls}. Note that AFGN has a very low source and target accuracy for near-zero values of λ_{cls}, which indicates the importance of the discriminative feedback of h_2 in the network. We demonstrate that the performance increases for greater values of λ_{cls}, however, decreases slightly after $\lambda_{cls} = 0.1$. Therefore, we set the value of λ_{cls} to 0.1 for optimal performance in AFGN.

7.4.4.4 I Bound Analysis

Figure 7.7 shows the change in performances of AGAN and AFGN on different values of γ. Both networks show low accuracy near zero I bound, which indicates interruption of redundant information in the features.



(a) (b)

Figure 7.7: Performance comparison for different I bounds γ of AGAN (a) and AFGN (b) on GZSL task of the CUB dataset.

Note that the performance of both networks depicts an increasing trend with the increasing values of γ. However, after $\gamma = 0.05$, the performance starts to decrease, which means the necessary information flow is harmed. Both the networks achieve optimal performance when the I bound is $\gamma = 0.05$. Note that for some datasets we observe better performance at $\gamma = 0.01$, therefore we mentioned earlier $\gamma \in [0.01, 0.05]$.

7.4.4.5 Ablation Study

To highlight the impact of different vital components on the performance of the proposed method, we perform an ablative analysis by removing those components from AGAN and AFGN. The results of the ablative analysis are shown in Table 7.3.

First, we omit the I bound from the proposed method and study its importance. The variants AGAN w/o γ and AFGN w/o γ show the performance without the I bound.

Approach	T	S	H	Approach	T	S	H
AGAN w/o γ	48.7	56.8	52.4	AFGN w/o γ	50.5	59.1	54.4
AGAN (f_s w/o $\mathcal{L}_u$)	20.1	72.5	31.4	AFGN (f_s w/o $\mathcal{L}_u$)	25.2	78.9	38.1
AGAN (f_s w/ $\tilde{F}_1$)	58.1	61.9	59.9	AFGN (f_s w/ $\tilde{F}_1$)	60.9	70.1	65.1
-	-	-	-	AFGN w/o $\mathcal{L}_{cls}$	47.8	58.1	52.4
AGAN w/o F_g	59.9	65.3	62.4	AFGN w/o F_g	62.4	68.7	65.3
AGAN w/o $\mathcal{L}_{m1}$	59.2	65.2	62.0	AFGN w/o $\mathcal{L}_{m2}$	57.9	66.1	61.7
AGAN w/o m	59.1	64.4	61.6	AFGN w/o m	61.1	71.3	65.8
AGAN	**67.9**	**71.5**	**69.7**	AFGN	**69.8**	**77.1**	**73.2**

Table 7.3: Ablative analysis for GZSL on the CUB dataset.

The accuracy of AGAN decreases drastically without the I bound. AFGN without the I bound shows a similar trend of inferior results. The existence of irrelevant information in the local regions while constructing the feature embedding harms AGAN and AFGN for fine-grained GZSL recognition. Thus, the I bound is crucial for the proposed method.

Second, we omit the target loss optimisation represented by the variants f_s w/o $\mathcal{L}_u$ as feature embedding without the target loss. We observe that both AGAN and AFGN variants show high S accuracy and very low T accuracy. This indicates that without $\mathcal{L}_u$, AGAN and AFGN struggle to generalise to target classes, which demonstrates the importance of the target loss based on pmi similarity.

Third, we omit the second step attention. The variants of AGAN and AFGN where the feature embedding is formed with only one-step dense attention (f_s w/ $\tilde{F}_1$) show a large decrease in performance. This justifies that only one level of dense attention mechanism is not sufficient enough to yield satisfactory performance.

Fourth, we remove $\mathcal{L}_{cls}$ from AFGN optimisation and observe that the performance of AFGN decreases as the discriminative property of the generated features is not monitored during training.

Fifth, for analysing the influence of global features in the proposed method, we omit F_g from the two variants AGAN w/o F_g and AFGN w/o F_g. We observe that the performance of both networks decreases to a large extent. This demonstrates the impact of the global features besides local features in the performance of GZSL tasks.

Sixth, to investigate whether AGAN or AFGN is more facilitated by the mutual training, we omit $\mathcal{L}_{m1}$ from AGAN in one variant (AGAN w/o $\mathcal{L}_{m1}$) and $\mathcal{L}_{m2}$ from

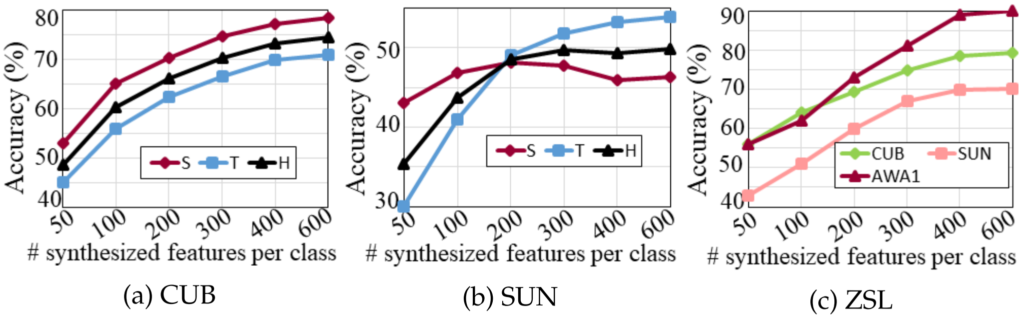
AFGN in the other variant (AFGN w/o $\mathcal{L}_{m2}$). AGAN and AFGN are trained jointly in both variants. The results indicate that AFGN is more facilitated than AGAN by mutual learning.

Finally, to study the impact of mutual learning in the proposed method, we remove mutual training, i.e., first, we train AGAN separately and then use the feature embedding from AGAN to train AFGN. These two variants are denoted by AGAN w/o m and AFGN w/o m. The degrading performance of the two variants shows the impact of the interaction between AGAN and AFGN during optimisation.

7.4.4.6 Analysing Number of Generated Features

To analyse the effect of the number of generated features per class during testing, we plot the graphs in Figures 7.8a, 7.8b and 7.8c.

The graphs (Figures 7.8a and 7.8b) show the performance comparison of CUB and SUN datasets with respect to a various number of generated features per class for GZSL. In general, we demonstrate that with the increasing number of features per class, the H increases. For CUB dataset, S and T significantly increase till 400, and after that the increment is marginal. For SUN dataset, S marginally decreases after 200; however, T increases with the increasing number of features per class. Notice that after 400, the value of H plateaus as both S and T depict no significant change. We demonstrate that AFGN can generalise well to unseen target classes besides seen source classes. For



(a) CUB (b) SUN (c) ZSL

Figure 7.8: (a) and (b) Increasing the number of synthesised features wrt GZSL performance in CUB and SUN datasets. (c) Increasing the number of synthesised features wrt ZSL performance in CUB, SUN, and AWA1 datasets.

ZSL (Figure 7.8(c)), the performance of all the datasets significantly increases with the increasing number of synthesised features per class. More number of features per class helps the final classifier learn better and generalise more to unseen target classes. Similar to GZSL tasks, we observe that the increment in performance is marginal after 400. The improved generalisation to target classes in GZSL and ZSL tasks validates that AFGN reduces source domain bias.

7.4.4.7 Analysing Two-level of Attentions

The first row of Figure 7.9 presents some examples of the class 'Mallard' from the CUB dataset. To study the learned attention, we visualise the learned attention maps for an image of the class 'Mallard' in the second row of Figure 7.9. We visualise the output of the first level of attention in the second column of the second row, which shows that the local regions linked to the attributes are assigned more weights than the other regions. This assists in focusing better on the possible distinctive attributed regions. The third column of the second row shows the visualisation of second-level attention. Compared to the output of one-level attention, two-level attention shows more weight assignment on the regions having intra-class common attributes to assist in better class decisions. In particular, notice that the region shown in green circles in the third column achieve more

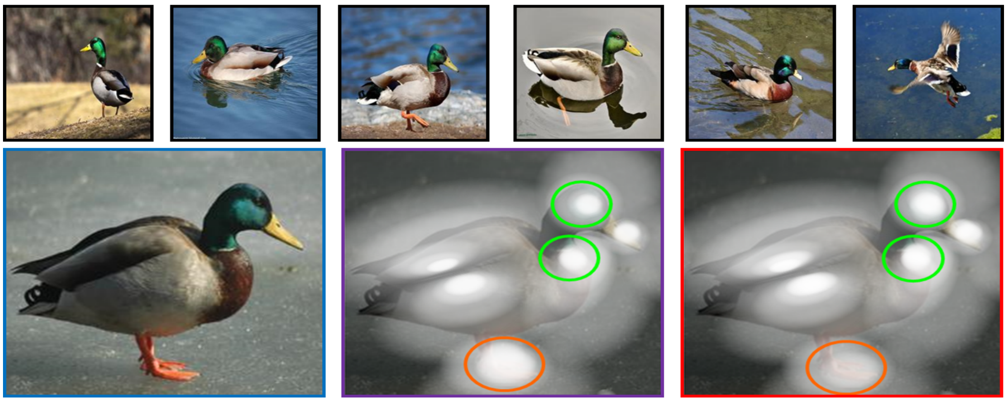


Figure 7.9: Samples from 'Mallard' class of CUB dataset (first row). Visualisation of the learned attention maps (second row), where first, second, and third columns show the original images, images after applying one-step attention (α), and two-step attention (α and $\tilde{\alpha}$), respectively. Best viewed in colour.

attention compared to that of the second column as the attributes 'forehead colour green' and 'breast colour grey' have a greater score of presence in the samples of the class. On the other hand, the region shown in orange circle in the third column receives less attention than the second column as the attribute 'leg colour orange' has less visibility in the samples of the class. This visualisation verifies the importance of our two-step attention mechanism to learn better attribute-weighted features for fine-grained GZSL.

7.5 Summary

Existing EL and FS GZSL methods use either local or global details to accomplish fine-grained classification. However, in this chapter, we have argued that both global and local details are crucial. Local features are necessary to capture fine distinctive information related to the semantic attributes, and global features are required to preserve generic visual feature representation structure. To utilise local and global features in EL and FS approaches, we have proposed to integrate an EL network (AGAN) and a FS network (AFGN) into a unified GZSL network. In the proposed GZSL network, we have introduced a new two-step dense attention mechanism to discover local details linked to the attributes. The global details are preserved region-wise. We have then introduced a mutual learning optimisation between the two networks to exploit mutually beneficial information. To reduce bias towards the source domain, we have transfer learned the target classes depending on their shared information with the source classes. The integration avails two-way testing capability. We have presented a thorough evaluation of the proposed method on benchmark datasets for GZSL and ZSL tasks and demonstrate that it outperforms contemporary works. The improved performance of the proposed method evinced that both global and local information are essential for fine-grained classification. Although the network has many hyper-parameters, a saddle point can be easily found with a moderate hyper-parameter tuning or cross-validation. Once the saddle point is located, it works for a wide range of tasks. This chapter has presented our fifth contribution. In the following chapter, we will discuss the final contribution of this thesis, i.e., to improve bidirectional GZSL and reduce bias.

Chapter 8

Bidirectional Mapping Coupled GAN for Generalised Zero-Shot Learning

[1] In the previous chapter, we discussed the proposed attribute weighted attention mechanism to improve embedding learning and generative ZSL. In this chapter, we present the sixth contribution of this thesis, i.e., to improve bidirectional mapping for strong coupling between visual and semantic space and reduce source classes bias to improve GZSL.

Bidirectional mapping-based GZSL methods rely on the quality of synthesised features to recognise source and target data. Therefore, learning a joint distribution of source-target classes and preserving domain distinctive information is crucial for these methods. However, existing methods only learn the underlying distribution of source data, although target class semantics are available in the GZSL problem setting. Most methods neglect retaining domain (source-target) distinction and use the learned distribution to recognise source and target data. Consequently, they do not perform well. In this chapter, we utilise the available target class semantics alongside source class semantics and learn joint distribution through a strong visual-semantic coupling. We propose a bidirectional mapping coupled generative adversarial network (BMCoGAN) by using the principle of coupled generative adversarial network. We further integrate

[1] Chapter 8 is adapted from: **T. Shermin**, S. W. Teng, F. Sohel, M. Murshed, and G. Lu, "Bidirectional Mapping Coupled GAN for Generalised Zero-Shot Learning," Revision submitted to IEEE Transactions on Image Processing, 2021 [arXiv: 2012.15054].

a Wasserstein generative adversarial optimisation to supervise the joint distribution learning. We design a loss optimisation for retaining source-target distinctive information in the synthesised features and reducing bias towards source classes, which pushes synthesised source features towards real source features and pulls synthesised target features away from real source features. We evaluate BMCoGAN on benchmark datasets and demonstrate its superior performance against contemporary methods.

8.1 Overview

Feature synthesising or generative ZSL approaches only rely on the class semantics for adversarially generating visual features [141, 143, 145, 147, 151, 152]. The generative methods capture the visual distribution only via a unidirectional alignment from the class semantics to the visual feature. This leads to weak visual-semantic coupling, which is vital for zero-shot tasks [10, 132, 153], and harms the performance. To handle this issue and improve performance, recent methods [10, 132, 154–157] use the bidirectional mapping between visual and semantic domains.

For GZSL, during testing, the bidirectional mapping models have to recognise both source and target classes. Thus, integrating the knowledge of the joint distribution of source-target classes will enhance the GZSL recognition performance. However, existing models ignore learning source-target classes joint distribution [10, 132, 154–157]. In addition, preserving source-target discriminative information in the model is necessary for improved classification and limiting bias towards source classes during testing. However, except for a few existing bidirectional mapping methods [156, 157], they [10, 132, 153–155] ignore the challenge of reducing bias towards source classes while performing zero-shot recognition (for more details see Section 2.7).

In this chapter, we propose a new generative bidirectional mapping GZSL method to address the above-mentioned issues. We refer to our proposed method as Bidirectional Mapping Coupled Generative Adversarial Network (BMCoGAN). Our ultimate aim is to learn two broad tasks: 1) learn the joint distribution of source-target classes by establishing strong coupling between the semantic-visual spaces through bidirectional mapping; and 2) preserve source-target discriminative characteristics in the generated feature space. Task 1 will enhance source-target data classification performance, and Task 2 will mitigate bias towards source classes.

As we follow the inductive setting, we do not have access to target image data. However, the GZSL problem setting states the availability of target class semantics [10, 151]. Therefore, we aim to learn the joint distribution using source and target class semantics. The Coupled GAN (CoGAN) [218] has a generative and weight sharing structure. So, it can learn dual-domain joint distribution with just samples drawn from the marginal distributions by restricting the network capacity. Therefore, to learn Task 1, we utilise the concept of CoGAN [218] to learn the joint distribution of source-target classes in our proposed bidirectional mapping method. We partially adopt the weight sharing properties of CoGAN in BMCoGAN to learn the joint distribution and integrate the bidirectional mapping property in the model by introducing regressors (Figure 8.3). To the best of our knowledge, ours is the first attempt to utilise the principle of CoGAN for GZSL recognition.

There are significant fundamental and architectural differences between CoGAN and the proposed BMCoGAN. In CoGAN, the joint distribution is learned in a one directional generative adversarial way for domain adaptation tasks. In particular, CoGAN learns to generate fake images of two domains from noise and adversarially distinguishes between the real and fake images of those two domains. On the other hand, the proposed BMCoGAN learns the joint distribution of the source and target classes for GZSL in a bidirectional way. For source and target classes, BMCoGAN generates visual features from class semantics, reconstructs class semantics back from generated visual features, and adversarially assesses reconstructed class semantics against the real class semantics. In BMCoGAN, we simultaneously learn the joint distribution and establish strong coupling between the visual and semantic space through bidirectional mapping. Note that the strong coupling between the class semantics and the synthetic visual features is crucial for improved GZSL performance as the target classes do not have available visual features. Through this bidirectional mapping and adversarial semantic assessment, BMCoGAN learns to generate synthetic visual features for the target classes.

For source classes, in addition to bidirectional mapping and semantic assessment, visual feature generation is adversarially supervised by available training images. To preserve discrimination between source and target classes (Task 2), we propose to push generated source visual features towards real source visual features and pull generated target visual features away from real source visual features. This mitigates the bias towards source classes. For details on the structural differentiation of CoGAN and BMCoGAN please see Section 8.3.1.

The main contributions are as follows:

- We propose a generative adversarial GZSL network, which learns source-target classes joint-distribution through bidirectional mapping and generates better source-target discriminative features.
- To capture the joint distribution of source and target set of classes, we propose to use the joint distribution learning concept of CoGAN in the proposed BMCoGAN for GZSL tasks. In particular, we relate the visual-semantic spaces by imposing bidirectional mapping in the principle of CoGAN and modify the weight sharing properties of CoGAN. We also support the joint distribution learning with a supervisory Wasserstein generative adversarial optimisation.
- Unlike existing methods, in addition to distance-based optimisation for bidirectional mapping, the proposed BMCoGAN adversarially ensures that the underlying distribution of the synthesised class semantics is aligned with the real semantics for both source and target classes.
- To reduce bias towards source classes, we encourage the proposed method to maintain more similarity between real source and synthesised source features than real source and synthesised target features. This helps in retaining the required discrimination between the features of source and target classes.
- We present an extensive empirical evaluation of BMCoGAN on several datasets to demonstrate its superior performance compared to contemporary GZSL methods.

8.2 Coupled Generative Adversarial Networks

The Coupled Generative Adversarial Networks (CoGAN) [218] can learn the joint distribution of two domains. In CoGAN, there are two GANs with shared layers to handle high-level semantic features and separate layers to deal with low-level features for different domains. CoGAN also has two discriminators with shared layers, as shown in Figure 8.1.

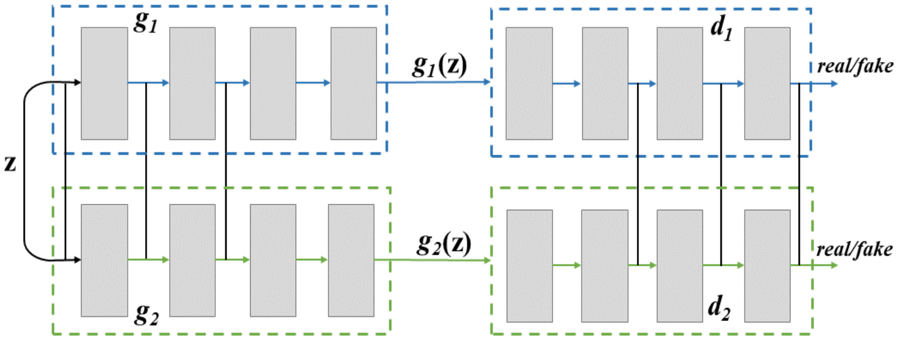


Figure 8.1: Block diagram of CoGAN.

This setting allows CoGAN to capture the joint distribution and generate images of multiple domains. Therefore, CoGAN is used for performing domain adaptation tasks. CoGAN can learn the joint distribution without relying on the correspondence between data samples in the two domains. The minimax objective function for CoGAN is as follows,

$$\begin{aligned} \max_{g_1,g_2} \min_{d_1,d_2} \mathcal{L} \equiv \mathbb{E}_{f_1 \sim p_{f_1}} \left[\log d_1\left(f_1\right)\right] \\ +\mathbb{E}_{z\sim p_z}\left[\log\left(1-d_1\left(g_1(z)\right)\right)\right] \\ + \mathbb{E}_{f_2 \sim p_{f_2}} \left[\log d_2\left(f_2\right)\right] \\ +\mathbb{E}_{z\sim p_z}\left[\log\left(1-d_2\left(g_2(z)\right)\right)\right]. \end{aligned} \tag{8.1}$$

Here, GAN$_i$ consists of generator g_i and discriminator d_i, and $i = 1, 2$. f_1 and f_2 denote two samples from different domains, and ϵ represents the Gaussian noise distribution. The CoGAN optimisation is subjected to two constraints as follows,

$$\begin{array}{ll} \theta_{g_1^j} = \theta_{g_2^j} & 1 \leq j \leq s_g \\ \theta_{d_1^{l_1-k}} = \theta_{d_2^{l_2-k}} & 0 \leq k \leq s_d - 1 \end{array}$$

where $\theta_{g_i^j}$ denotes the parameter of the j^{th} layer from the top of the generator g_i, $\theta_{d_i^{l_i-k}}$ represents the parameter of the $(k+1)^{th}$ layer from the final layer of the discriminator d_i, l_i denotes the number of layers in d_i. In the following section, we discuss how the concept of CoGAN for learning the joint distribution is used in the proposed BMCoGAN for GZSL tasks and highlight the architectural differences between CoGAN and BMCoGAN.

8.3 Proposed Method

In this section, we formally present our proposed method. We follow the GZSL setting described in Section 7.1.

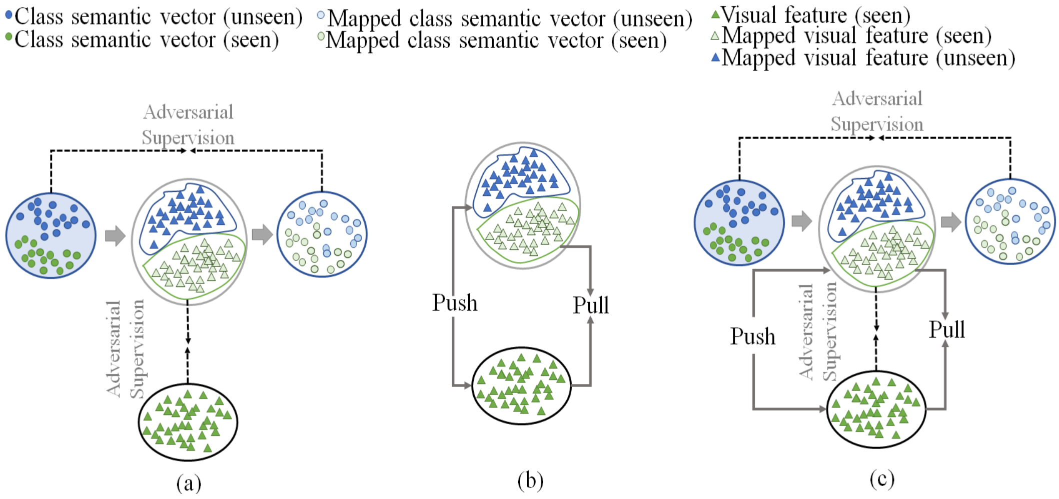

●Class semantic vector (unseen)
●Class semantic vector (seen)
○Mapped class semantic vector (unseen)
○Mapped class semantic vector (seen)
▲Visual feature (seen)
△Mapped visual feature (seen)
▲Mapped visual feature (unseen)
Adversarial Supervision
Adversarial Supervision
(a)
Push
Pull
(b)
Adversarial Supervision
Push
Adversarial Supervision
Pull
(c)

Figure 8.2: A conceptual overview of the proposed method. (a) To learn a joint distribution of source and target classes, we learn to construct visual features from class semantics and reconstruct class semantics back from the generated visual features. This bidirectional mapping ensures a strong relation between semantic-visual space. This compensates for the absence of supervision from target real features during synthetic target feature generation. (b) To retain source-target discrimination in the feature generation procedure, we encourage the real source features to attract the generated source features and repel the generated target features. (c) The combination of (a) and (b) constructs the final model.

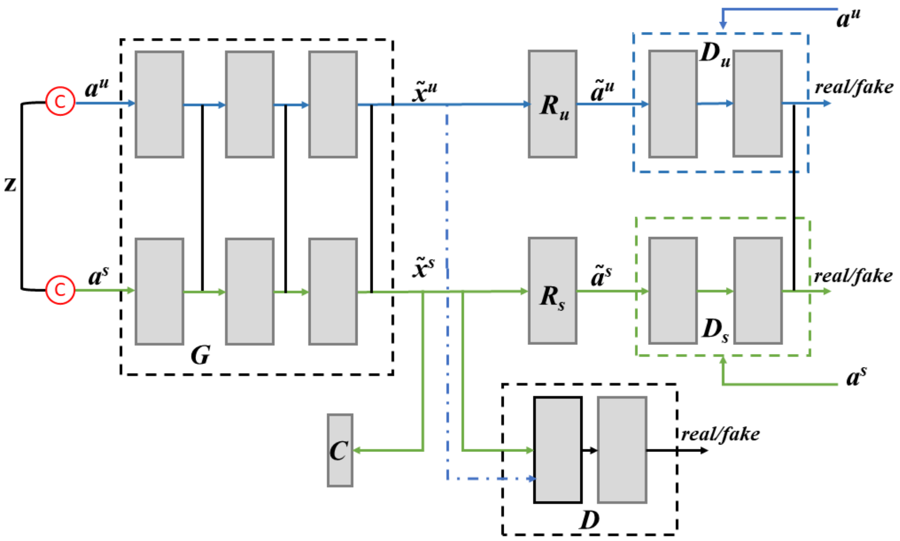


Figure 8.3: Block diagram of our proposed BMCoGAN. The BMCoGAN has a full weight shared coupled generator G, a source classes regressor R_s, an target classes regressor R_u, partial weight sharing coupled discriminators D_s and D_u, a feature discriminator D, and a pre-trained classifier C. The generator G takes noise z as input conditioned on the class semantics a^s and a^t. G generates synthetic visual features and the regressors reconstructs the features back to class semantics. The coupled discriminators adversarially assess the real ans synthetic class semantics. The joint distribution is learned by the bidirectional mapping performed by G, R_u, R_s, D_s, and D_u. The distinctive characteristics required for classification is imposed in the synthetic features by the adversarial discriminator D and classifier C. The source-target discrimination is learned by our designed loss using D.

8.3.1 Proposed GZSL

As mentioned earlier, the proposed BMCoGAN addresses two broad tasks: 1) Learning joint distribution through bidirectional mapping and 2) Learning to preserve discriminative information in synthesised features.

Figure 8.2a shows an illustration of the concept for Task 1. For both source and target classes, our aim is to learn the underlying semantic and visual distributions. We plan to construct visual features of both classes using the available class semantics. The quality of constructed source class visual features can be evaluated by the available real features. However, for target classes, we do not have access to real visual features. Therefore, to evaluate, we utilise bidirectional mapping. This means we reconstruct the class semantics back from the generated features and then compare real semantics with reconstructed semantics. This bidirectional mapping not only helps to supervise target feature learning, also establishes a strong coupling between semantic and visual spaces. To fulfill our goal of learning the joint distribution, besides the target classes, we extend the bidirectional mapping to the source classes.

For Task 2, compared to the constructed target features, we encourage the constructed source features to remain closer to the real source feature space. As shown in Figure 8.2b, this objective is fulfilled by pushing the constructed source features towards the real source features and pulling the constructed target features away from the real source features. The ultimate concept of the proposed model (Figure 8.2c) is a combination of both Tasks.

Architecture Overview. The proposed BMCoGAN is illustrated in Figure 8.3. In CoGAN, the generators share only the bottom layers so that the GANs learn to deal with high-level concepts in the same manner. This also helps in learning the joint distribution of data samples. Unlike CoGAN, in BMCoGAN, we propose to use a fully shared generator for source-target classes. So, the main component of BMCoGAN, the shared generator G modifies the first constraint of CoGAN. We demonstrate that the fully shared generator is optimal than partially weight shared generators for GZSL tasks in the ablative section.

Moreover, contrary to CoGAN, our shared generator G is conditioned on the class semantic vectors for generating visual features. The shared generator helps to learn to handle low to high-level features for source-target classes in a similar manner, which is crucial for our problem setting and encourages the joint distribution learning paradigm. The generator G plays an important role in accomplishing Tasks 1 and 2. To introduce bidirectional mapping within the CoGAN structure, we place two separate regressors (R_s and R_u). The regressors map visual features to their corresponding class semantic vectors. Unlike CoGAN, we introduce coupled discriminators (D_s and D_u) with only one shared layer in BMCoGAN. The coupled discriminators learn to distinguish between the real and reconstructed class semantic vectors. The generator and the regressors combinedly generate class semantic vectors through bidirectional mapping and try to deceive the coupled discriminators. On the other hand, the coupled discriminators try to recognise real and generated class semantic vectors adversarially. The purpose of the partially shared architecture of the coupled discriminators is to simultaneously synchronise with the shared G to learn joint distribution and to learn discriminative attributes of source-target class semantics. The synchronisation is ensured by the final shared layer of the coupled discriminators. The separate layer learns discriminative information of source-target class semantics. This helps G to roughly hold source-target discrimination in the learned feature space. The final weight-shared layer also reduces the number of parameters in the network.

The proposed method also has a feature discriminator D, which learns to separate real and generated visual features for the source classes. The generator G and discriminator D interacts with each other adversarially. We adopt Wasserstein generative adversarial optimisation for this supervision.

To learn source-target discriminative information for handling bias towards source classes, the generator G and the discriminator D learn to attract source classes generated features to real source classes features and repels target classes generated features away from real source classes features. The classifier C is placed to assess and enrich the discriminative characteristics in the generated features. The proposed model provides a unified framework to learn the joint distribution of source-target classes simultaneously and retain source-target distinctive information and can be trained in an end-to-end fashion.

8.3.1.1 Learning joint distribution using bidirectional mapping

We want to utilise the class semantic vectors of both source and target classes and generate synthetic visual features. We follow [141] to construct the conditional Generator $G : \mathcal{Z} \times \mathcal{A} \rightarrow \mathcal{X}$ that uses random Gaussian noise $z \in \mathcal{Z}$ and class semantic vector $a_c \in \mathcal{A}$, and generates visual feature $x \in \mathcal{X}$ of class c. Now, the feature Generator G can generate source and target classes visual features conditioned on their class semantic vectors.

On the other hand, the regressors have to perform the reverse task of constructing the class semantic vectors from the generated visual features. We aim to train the regressors to generate class semantic vectors as similar as possible to the real class semantic vectors. This will ensure strong coupling between *semantic→ visual* and *visual→ semantic*. The semantic and visual spaces are better related through this bidirectional mapping. Note that we place separate regressors for the source and target classes to encourage solid source-target bidirectional mapping. We train the regressors with supervised loss as follows,

$$\mathcal{L}_{Reg} = ||a - \tilde{a}||_2^2. \tag{8.2}$$

We compute $\mathcal{L}_{Reg}^s$ and $\mathcal{L}_{Reg}^t$ for R_s and R_u, respectively.

The ultimate aim of this task is to learn the joint distribution of source and target classes. We optimise the shared generator G to learn the joint distribution by adversarially interacting with coupled discriminators D_s and D_u. The generator G generates source-target features such that the generated class semantics from the features are indistinguishable to the real class semantics. The regressors generate class semantics using the generated features of G. On the other hand, the coupled discriminators distinguish the generated class semantics from the available real class semantics for source and target classes. The adversarial optimisation between the generator and the coupled discriminators is as follows,

$$\begin{aligned}\min_{G} \max_{D_s, D_u} \mathcal{L}_{G1} = {} & \mathbb{E}_{a^s \sim p(a^s)}[\log D_s(a^s)] + \mathbb{E}_{\tilde{a}^s \sim p(\tilde{a}^s)}[\log(1 - D_s(\tilde{a}^s))] \\ & + \mathbb{E}_{a^t \sim p(a^t)}[\log D_u(a^t)] + \mathbb{E}_{\tilde{a}^t \sim p(\tilde{a}^t}[\log(1 - D_u(\tilde{a}^t))].\end{aligned} \tag{8.3}$$

Here, $\tilde{a}$ denotes the constructed class semantics by the regressors. This adversarial optimisation encourages joint distribution learning.

To impose the knowledge of the true underlying visual distribution in the generated visual space, we further adversarially supervise the generator with the real source features. To be specific, we train the discriminator D to distinguish between a real feature x^s and a synthetic feature $\tilde{x}^s$ conditioned on a^s. Simultaneously the feature generator is trained to produce synthetic features indistinguishable to the real features.

The adversarial supervision is performed by using Wasserstein distance as follows,

$$\begin{aligned}\min_{G}\max_{D}\mathcal{L}_{G_2} =&E_{p(x^s)}[D(x^s)] - E_{p(\tilde{x}^s)}[D(\tilde{x}^s)]- \\ &\beta E_{p(\hat{x}^s)}\left[(\|\nabla_{\hat{x}}D(\hat{x}^s)\|_2 - 1)^2\right],\end{aligned} \tag{8.4}$$

where $\tilde{x}^s = G(z, a^s)$ and $\hat{x}^s = \alpha x + (1-\alpha)\tilde{x}^s$ with $\alpha \sim U(0,1)$. β is the penalty coefficient. Wasserstein distance is calculated using the first two terms. The gradient penalty is computed by the final term i.e., the gradient of the discriminator D is forced to maintain a unit norm along the straight line between real and generated source visual feature pairs. Note that unlike [141], we do not integrate the class semantics to D.

8.3.1.2 Learning to preserve discriminative information in synthesised features

To preserve a distinction between the generated features of source classes from that of the target classes, we explicitly design an optimisation function, which encourages the generator G to generate source visual features closer to real source visual features and generate target visual features away from real source visual features. This phenomenon will help maintain a suitable gap between generated source and target features for better classification and reduce the bias towards source classes. We optimise the following loss for holding discrimination,

$$\mathcal{L}_d = ||D(x^s) - D(\tilde{x}^s)||_2^2 - ||D(x^s) - D(\tilde{x}^t)||_2^2. \tag{8.5}$$

We hypothesise that during testing, using the trained discriminator D will help to retain the source-target distinctive information in the synthesised features. Thus, along with the Generator G, we extend the discriminative learning to the discriminator D. In particular, we optimise both G and D with the above loss with the hope to utilise the partial layers of D in test time feature synthesis.

To further support the discrimination among source and target classes in the feature learning procedure, we constrain the early layers of the discriminator D as follows,

$$\mathcal{L}(D, \mathbb{C}) = \mathbb{E}_{p(x^s,y^s)} \left[\mathbb{E}_{p_D(k|x^s)} \left[\mathcal{L}_{\mathbb{C}} \left(k, y, y' \right) \right] \right] \tag{8.6}$$

where, $\mathcal{L}_{\mathbb{C}}(k, y, y') = \max \left(0, \Delta + \|k - \mathbb{C}_y\|_2^2 - \left\|k - \mathbb{C}_{y'}\right\|_2^2 \right)$ [219], k is the early layer output of D for the source classes, y and y' are the true class label of x^s and random class label other than y, respectively. The $\mathbb{C}_y$ denotes the y^{th} source class center of deep features. The distributions of source classes are grouped according to their labels by the center loss. This in turn facilitates the learned feature space to maintain distance among source and target classes by concentrating the source classes near their corresponding centers.

Finally, to make the generated features well suited for learning a discriminative target classifier, in line with [141], we utilise the decision of a source classes trained Classifier C as,

$$\mathcal{L}_{cls} = -E_{\tilde{x}^s \sim p_{syn}}[\log P(y \mid \tilde{x}^s; \theta)], \tag{8.7}$$

where y is the true class label of $\tilde{x}$ and $P(y \mid \tilde{x}; \theta)$ denotes the probability of $\tilde{x}$ being predicted as y conditioned on its semantic descriptor a.

Our ultimate objective becomes,

$$\begin{gathered} \min_{G,R_s,R_u} \max_{D_s,D_u} \lambda_1 \mathcal{L}_{G1} + \mathcal{L}^s_{Reg} + \mathcal{L}^t_{Reg}, \\ \min_{G} \max_{D} \lambda_2 \mathcal{L}_{G2} + \lambda_{cls}\mathcal{L}_{cls} + \lambda_d \mathcal{L}_d + \lambda_{cen}\mathcal{L}(D, \mathbb{C}). \end{gathered} \tag{8.8}$$

Here, λ_1, λ_2, λ_d, λ_{cls} and λ_{cen} are hyper-parameters for weighting the losses. More detail on setting the hyper-parameters will be discussed in the results section. The training procedure of BMCoGAN is outlined in Algorithm 3, where where e denotes the number of steps to train discriminators D_s, D_u, and D. We have used 5 steps..

Testing Phase The test phase of BMCoGAN is shown in Figure 8.4. During the test phase our aim is to train a final visual classifier for both source and target classes and use

Algorithm 3 Training Procedure

Input: labelled source dataset $\mathcal{Y}^s$; source class semantic vectors $\mathcal{A}^s$; target class semantic vectors $\mathcal{A}^t$; generator G; regressors R_s and R_u; coupled discriminators D_s and D_u; discriminator D; source classifier C pre-trained on $\mathcal{Y}^s$.

Output: Trained generator G and discriminator D.

```
while not converged do
    Sample mini-batch from $(x_i^s, y_i^s)_{i=1}^{n_s}$ and their corresponding class semantic vectors $\mathcal{A}^s$;
    Sample mini-batch of class semantic vectors from $\mathcal{A}^t$;
    Update $R_s$ and $R_u$ by (8.2);
    for $e$ steps do
        Update $D_s$, and $D_u$ by (8.3);
        Update $D$ by (8.4) + (8.5) + (8.6);
    end for
    Sample mini-batch from $(x_i^s, y_i^s)_{i=1}^{n_s}$ and their corresponding class semantic vectors $\mathcal{A}^s$;
    Sample mini-batch of class semantic vectors from $\mathcal{A}^t$;
    Update $R_s$ and $R_u$ by (8.3);
    Update $G$ by (8.3) + (8.4) + (8.5) + (8.7);
end while
return Trained $G$ and $D$.
```

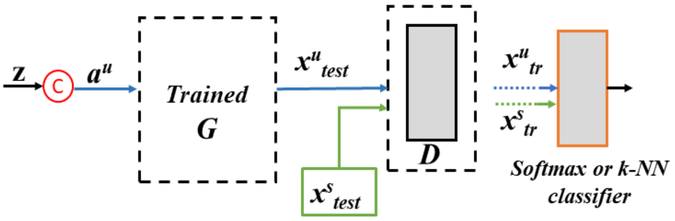


Figure 8.4: Testing sequence of our proposed GZSL method.

the classifier for GZSL task. We exploit two kinds of classifiers: softmax and k-nearest neighbor (k-NN) as the final classifier.For final GZSL task, we use the trained softmax or k-NN classifier.

We need visual features of both source and target classes to train the final classifier. We have access to test features of the source classes x^s_{test}. Note that the test features of the source classes are split from the source class test set. However, we do not have access to visual features of target classes. So, we generate multiple synthetic features x^t_{test} for every target class by using the trained generator G, class semantics a^t, and noise

z (re-sampling). We have trained the discriminator D to learn the discriminative features of the source and target classes during training phase. We want to use this distinctive knowledge in the final GZSL classification to reduce source bias. Therefore, we pass both x^s_{test} and x^t_{test} through the discriminator D and obtain the final features x^s_{tr} and x^t_{tr}, respectively. Note that only the early layer of the discriminator D is used to preserve the feature dimensions, i.e., x^t_{test} and x^s_{test} will have dimensions similar to the extracted features from the pre-trained ResNet-101. Once we have the final features of the source x^s_{tr} and target x^t_{tr} classes, we train the final classifier separately.

After receiving final features, the softmax classifier (h) produces $|C^s{+}C^t|$ dimensional class probability through log softmax function, i.e., $|C^s|$ for source classes and $|C^t|$ for target classes. We define the classification loss as follows,

$$\mathcal{L}_h = -E_{x_{tr} \sim p_{tr}}[\log P(y \mid x_{tr}; \theta_h)] \tag{8.9}$$

where, x_{tr}, y, and p_{tr} denote the final features of the source and target classes, the ground-truth of x_{tr}, and distribution of the final features, respectively. $P(y \mid x_{tr}; \theta_h)$ denotes the probability of the final features (x_{tr} being recognised as y. For the k-NN classifier, we only evaluate using 1-NN classifier.

8.4 Experimental Studies

In this section, we describe the implementation details, discuss our experimental outcomes (classification results, hyper-parameters setups, and number of synthesised features per target class) and present a detailed ablation study. Note that we follow the same evaluation metrics and datasets discussed in Sections 7.4.2 and 7.4.1, respectively.

8.4.1 Implementation Details

We extract 2048 dimensional features from pre-trained ResNet-101 for all source classes for our experiments. Since the generator has to produce fully-connected features from conditional input, we maintain a total fully-connected structure of the generator for efficiency i.e., the generator has one hidden fully-connected layer of dimension 4096. The fully-connected structure of the generator also helps in learning the joint distribution of the source-target classes. Both the Regressors have a hidden layer with 1024 fully-connected neurons. The coupled discriminators reduce the class semantic vectors to 256 dimension through separated fully-connected layers and then share the final fully-connected layer.

We observed that the discriminator D co-operates more in the discriminative learning and supervision when it has only one hidden layer. We have used 1024-dimensional fully-connected hidden layer for the discriminator D in our experiments. We follow [220] for improved Wasserstein GAN training. Adam solver with $\beta_1 = 0.5$ and $\beta_2 = 0.999$ is used for optimisation. A learning rate of 0.0001 is used for the generator, the discriminator D, and the center loss (8.6), and 0.0002 is used for the regressors and the coupled discriminators. For our final objective, we find setting $\lambda_1 = 2$, $\lambda_2 = 0.8$, $\lambda_d = 1$, $\lambda_{cls} = 0.2$, and $\lambda_{cen} = 0.1$ optimal.

8.4.2 Results and Analysis

We compare BMCoGAN with state-of-the-art methods on generalised zero-shot learning, and the results are shown in Table 8.1. The results of LATEM [137], DEM [212], and SGMAL [122] are adopted from SGMAL [122] and the results of other compared methods are obtained from their corresponding published articles. We compare the performance with only inductive methods for a fair comparison, which do not use target images during training. The results with 400 synthesised features per class are shown in Table 8.1. Though we present results from embedding learning and feature synthesising methods in Table 8.1, for comparison, our main focus is on the bidirectional methods. We perform GZSL tasks with the proposed BMCoGAN using both 1-NN and softmax classifier. The Harmonic mean (H) is the main indicator of how well a GZSL method performs.

Table 8.1 shows that BMCoGAN with softmax classifier outperforms all the contemporary bidirectional mapping GZSL methods significantly in terms of the Harmonic

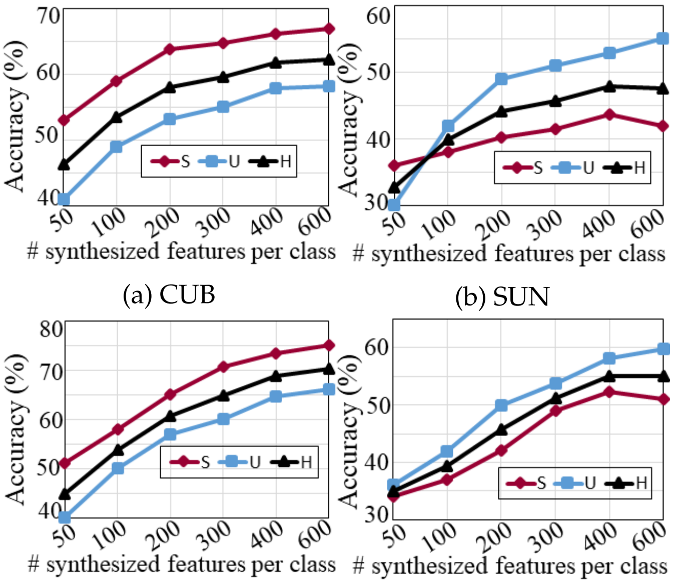
Table 8.1: Performance comparison. T and S are the Top-1 accuracies tested on target classes and source classes, respectively, in GZSL. H is the harmonic mean of T and S. △, □, and □□ denote embedding learning, generative, and bidirectional mapping methods, respectively, and '-' represents that the results were not reported.

Approach	Model	GZSL											
		CUB			SUN			AWA1			AWA2		
		T	*S*	*H*	*T*	*S*	*H*	*T*	*S*	*H*	*T*	*S*	*H*
△	DCN [139] (2018)	28.4	60.7	38.7	25.5	37.0	30.2	25.5	84.2	39.1	-	-	-
	CRnet [213] (2019)	45.5	56.8	50.5	34.1	36.5	35.3	58.1	74.7	65.4	-	-	-
	TCN [113] (2019)	52.6	52.0	52.3	31.2	37.3	34.0	49.4	76.5	60.0	61.2	65.8	63.4
	DVBE [214] (2020)	53.2	60.2	56.5	45.0	37.2	40.7	-	-	-	63.6	70.8	67.0
	DAZLE [126] (2020)	56.7	59.6	58.1	52.3	24.3	33.2	-	-	-	60.3	75.7	67.1
	VSG-CNN [203] (2020)	52.6	62.1	57.0	30.3	31.6	30.9	-	-	-	60.4	75.1	67.0
□	SE-GZSL [143] (2018)	41.5	53.3	46.7	40.9	30.5	34.9	56.3	67.8	61.5	58.3	68.1	62.8
	f-CLSWGAN [141] (2018)	43.7	57.7	49.7	42.6	36.6	39.4	57.9	61.4	59.6	-	-	-
	cycle-CLSWGAN [145] (2018)	45.7	61.0	52.3	49.4	33.6	40.0	56.9	64.0	60.2	-	-	-
	CADA-VAE [147] (2019)	51.6	53.5	52.4	47.2	35.7	40.6	57.3	72.8	64.1	55.8	75.0	63.9
	f-VAEGAN-D2 [144] (2019)	48.4	60.1	53.6	45.1	38.0	41.3	-	-	-	57.6	70.6	63.5
	LisGAN [152] (2019)	46.5	57.9	51.6	42.9	37.8	40.2	52.6	76.3	62.3	-	-	-
	GMN [142] (2019)	56.1	54.3	55.2	53.2	33.0	40.7	61.1	71.3	65.8	-	-	-
	RZSL-CVCP [221] (2019)	47.4	47.6	47.5	36.6	42.8	39.3	62.7	77.0	69.1	56.4	81.4	66.7
	RFF-GZSL (softmax) [151] (2020)	52.6	56.6	54.6	45.7	38.6	41.9	59.8	75.1	66.5	-	-	-
	LsrGAN [222] (2020)	48.1	59.1	53.0	44.8	37.7	40.9	54.6	74.6	63.0	-	-	-
	TF-VAEGAN [215] (2020)	52.8	64.7	58.1	45.6	40.7	43.0	59.8	75.1	66.6	-	-	-
	TI-GZSL(Res) [223] (2020)	44.8	42.2	43.5	31.5	20.3	24.7	61.5	67.7	64.4	72.1	63.9	67.7
	ASPN [146] (2020)	50.7	61.5	55.6	-	-	-	58.0	85.7	69.2	46.2	87.0	60.4
□□	DASCN [10] (2019)	45.9	59.0	51.6	42.4	38.5	40.3	59.3	68.0	63.4	-	-	-
	GDAN [132] (2019)	39.3	66.7	49.5	38.1	89.9	53.4	-	-	-	32.1	67.5	43.5
	RBGN [154] (2020)	47.0	54.3	50.4	46.0	37.2	41.2	57.5	67.1	61.9	57.2	71.4	63.6
	GZSL-AVSI [153] (2020)	61.2	57.7	59.4	-	-	-	60.5	71.9	65.7	59.4	74.2	66.0
	ISE-GAN [157] (2020)	52.4	55.4	53.8	51.3	34.7	41.4	58.7	74.4	65.6	55.9	79.3	65.5
	SE-GAN+SM [157] (2020)	48.4	57.6	52.6	44.7	37.0	40.5	53.9	68.3	60.3	55.1	61.9	58.3
	IBZSL [156] (2020)	52.2	56.2	54.1	43.8	37.8	40.6	-	-	-	56.0	80.0	65.9
	cFlow-ZSL [155] (2020)	50.8	54.9	52.8	46.7	39.5	42.8	57.1	68.1	62.1	56.7	74.8	64.5
	BMCoGAN (softmax)	57.9	66.1	61.7	52.9	43.7	47.8	61.5	78.2	68.8	61.9	76.9	68.5
	BMCoGAN (1-NN)	64.6	73.5	**68.7**	58.1	52.4	**55.1**	66.1	86.1	**74.7**	66.9	81.3	**73.4**

mean H. This variant also achieves better T accuracy than the bidirectional and contemporary embedding learning and feature synthesising methods for most tasks. BMCoGAN with 1-NN classifier achieves better H accuracy than all compared bidirectional and other methods for all tasks. This variant also outperforms the majority of contemporary methods in both T and S accuracy for all four datasets. We observe that the 1-NN classifier variant of BMCoGAN performs better than the softmax classifier variant, which means the learned feature space is more suitable for nearest neighbour classifier.

Note that, in general, BMCoGAN maintains a good balance between the S and T accuracies. That is, besides achieving higher T accuracy than other bidirectional mapping methods, it does not show a significant drop in S accuracy such as [153] (CUB). This means the proposed method can generate well discriminative features for improved classification. Both versions of BMCoGAN achieve improved performance than ISE-GAN [157] and IBZSL [156], and verify better discriminative learning, which reduces bias towards source classes.

8.4.3 Analysing Number of Generated Features



(a) CUB

(b) SUN

(c) CUB

(d) SUN

Figure 8.5: (a) and (b) Increasing the number of synthesised features wrt GZSL performance in CUB and SUN datasets using **softmax classifier**. (c) and (d) Increasing the number of synthesised features wrt ZSL performance in CUB and SUN datasets using **1-NN classifier**.

For analysing the effect of the number of generated features per class during testing, we plot the graphs in Figure 8.5. The graphs show the performance comparison of CUB and SUN datasets for various generated features per class. In general, we demonstrate that with the increasing number of features per class, the harmonic mean H of both datasets increases.

For the softmax version, in the CUB dataset, S and T significantly increase till 400, and after that, the increment is marginal (Figure 8.5a). For the SUN dataset, S

decreases after 400; however, T increases with the increasing number of features per class (Figure 8.5b). Notice that after 400, the value of harmonic mean plateaus as both S and T show no significant changes.

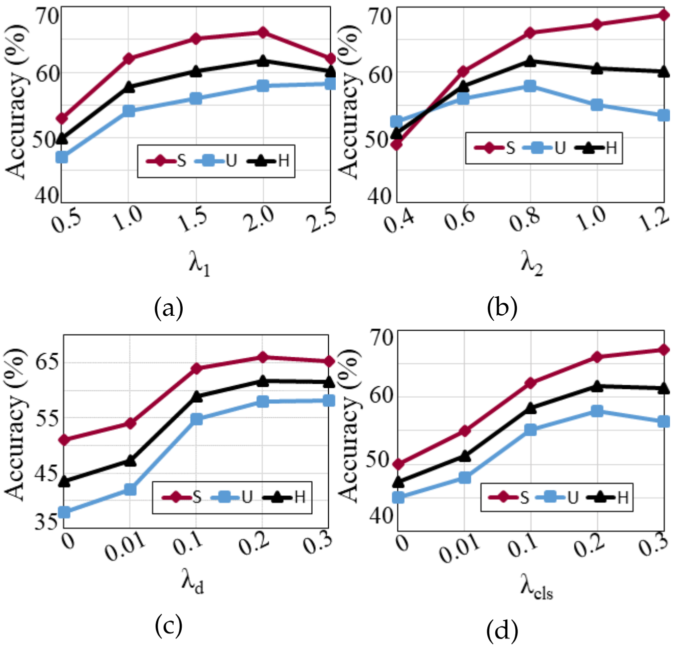
To demonstrate the trend of GZSL performance of 1-NN variant of BMCoGAN on a different number of generated features per class, we plot the graph shown in Figures 8.5c and 8.5d. The CUB dataset shows consistently increasing performance wrt an increasing number of synthesised features per class. On the other hand, the SUN dataset shows plateaued H accuracy with slightly increasing S and decreasing T accuracies. Overall, we observe that synthesising 400 to 600 features per class is suitable for BMCoGAN, depending on the dataset.

8.4.4 Hyper-parameters Analysis

For studying the trend of GZSL accuracy of BMCoGAN in different hyper-parameters (λ_1, λ_2, λ_{cls}, and λ_{cen}) settings, we plot the graphs shown in Figure 8.6 for CUB dataset. We present the analysis of the hyper-parameters setting for the softmax classifier variant of the proposed method.

The hyper-parameter λ_1 weights the contribution of bidirectional adversarial learning in the overall optimisation and joint distribution learning. Figure 8.6a shows the performance of BMCoGAN with various λ_1 setups. We observe that source accuracy increases with the increase of λ_1 till $\lambda_1 = 2$ and decreases after that. On the other hand, the increment rate of target accuracy plateaus after $\lambda_1 = 2$. However, it does not show a decreasing pattern. Though the gap between source and target accuracy decreases after $\lambda_1 = 2$, the H accuracy decreases. Thus, we find the value of $\lambda_1 = 0.2$ optimal as the H accuracy is optimal at that value. Overall, the graph indicates that a significant contribution of $\mathcal{L}_{G1}$ reflects better GZSL recognition and verifies the robustness of the proposed BMCoGAN network for learning better source-target joint distribution.

The supervisory WGAN optimisation is weighted by λ_2. As shown in Figure 8.6b, the source accuracy shows an increasing pattern with higher λ_2 values. However, the target accuracy depicts a sharp decrement after $\lambda_2 = 0.8$. It is worth noting that the



(a) (b) (c) (d)

Figure 8.6: Effect of varying the hyper-parameters in the GZSL performance on the CUB dataset.

target accuracy slightly surpasses the source accuracy at a significantly low λ_2 value. However, with the increasing contribution of $\mathcal{L}_{G2}$, the network becomes more biased towards source classes. The network shows optimal H accuracy at $\lambda_2 = 0.8$. The optimum supervision of $\mathcal{L}_{G2}$ is important to maintain a proper balance between S and T accuracies in the network.

To study the effect of discriminative loss $\mathcal{L}_d$, we plot the graph shown in Figure 8.6c. We observe that the target accuracy is significantly low, and the gap between source and target accuracy is huge at near-zero λ_d value. This means the source classes are dominating the classification performance. However, the high bias towards source classes reduces with the increasing value of λ_d, and the gap between source and target accuracies also reduces. This evaluates the network is capable of preserving source-target distinctive information with an optimal contribution of $\mathcal{L}_d$.

Figures 8.6d and 8.6e show a similar trend. The source accuracy depicts an increasing pattern, while the target accuracy decreases after a certain point. We demonstrate that up to a certain value, λ_{cls} and λ_{cen} serve their purpose of supporting the discrimination by preserving distinctive source classes information in the network. Therefore, we set the value of λ_{cls} and λ_{cen} to 0.2 and 0.1, respectively for the optimal H accuracy.

8.4.5 Ablative Analysis

To justify the role of different components and optimisation in BMCoGAN separately, we present the ablative analysis in Table 8.2. We perform the ablative analysis by deducting vital components from BMCoGAN and introducing alternative components in BMCoGAN, and justify their impact on the performances.

Table 8.2: Ablative analysis for GZSL on the CUB dataset.

	CUB		
Approach	U	S	H
BMCoGAN w/o $\mathcal{L}_{G2}$	48.3	44.2	46.1
BMCoGAN w/o $\mathcal{L}_{d}$	38.7	51.9	44.3
BMCoGAN w/o $\mathcal{L}_{cls}$	45.8	50.7	48.1
BMCoGAN w/o $\mathcal{L}_{cen}$	44.8	50.8	47.6
BMCoGAN w/ R	53.4	64.1	58.2
BMCoGAN w/ sep. D_s and D_u	56.1	65.2	60.3
BMCoGAN w/ sep. G_s and G_u	45.1	54.9	49.5
BMCoGAN w/o D (test)	52.1	65.2	57.9
BMCoGAN	**57.9**	**66.1**	**61.7**

The variant BMCoGAN w/o $\mathcal{L}_{G2}$ represents optimising the BMCoGAN without the WGAN loss or supervision for the source visual features. We observe that both source and target accuracies significantly decrease, which means the supervision from real visual features is crucial for learning to generate visual features. Compared with target accuracy, the source accuracy drastically decreases. This indicates the network loses the capacity to learn the true underlying distribution of the source classes without supervision.

BMCoGAN w/o $\mathcal{L}_{d}$ denotes the variant without the optimisation for retaining source-target discriminative information in the synthesised features. The performance of this variant shows that without $\mathcal{L}_{d}$ optimisation, the learned visual distribution is dominated by the source classes. This creates towards source classes. This justifies the importance of $\mathcal{L}_d$ in the network.

We demonstrate that the variants without $\mathcal{L}_{cls}$ and $\mathcal{L}_{cen}$ also show a significant drop in performance especially, for source classes. This proves that these losses help the network learn distinctive knowledge about the source classes, confine the generated source features to their corresponding centers, and eventually reduce bias towards source classes.

BMCoGAN w/ R is the variant in which we replace separate regressors with a shared regressor R. The reduced performance indicates that learning separate bidirectional mapping for source-target classes enhances joint distribution learning and GZSL recognition. The variant with separate class semantic discriminators is denoted as BMCoGAN w/ sep. D_s and D_u. The slight decrement in the performance indicates that separate discriminators do not harm the network to a great extent, like other variants. However, the coupled discriminators encourage improved performance and reduce the number of parameters in the network.

BMCoGAN w/ sep. G_s and G_u, this variant has coupled generators for source-target classes similar to CoGAN [218]. We observe that the performance drastically decreases. This means, only learning the underlying high-level concept in a similar way for source-target classes is not sufficient for the GZSL recognition task. Thus, the proposed structure of the shared generator is optimal for the task.

To verify the contribution of partial layers of D in retaining the source-target discrimination, we test the network without passing generated target features through early layers of D in the variant BMCoGAN w/o D. We notice that not only the performance of GZSL decrease, the gap between source and target accuracy significantly increases. This proves increased bias towards source classes and less discrimination in the generated visual features.

8.5 Summary

In this chapter, we have proposed a new bidirectional mapping generative model for the GZSL tasks. In particular, we have proposed to incorporate the concept of bidirectional mapping into the coupled generative adversarial network for learning source-target joint distribution and design a loss optimisation for preserving source-target discrimination. We have presented and discussed the evaluation of our proposed methods on different benchmark datasets and the performance comparison with existing GZSL methods. We have demonstrated that the proposed method outperforms contemporary methods. In addition, we have provided detailed ablative analysis to discuss the importance of different components in the proposed network.

In this chapter we have presented the final contribution of this thesis. In the following chapter, we will highlight and summarise insights from the approaches proposed in this thesis, and provide future research directions.

Chapter 9

Conclusion

This chapter will highlight our contributions in the three areas of transfer learning (sequential transfer learning, DA, and GZSL), summarise our findings, and discuss future research directions.

9.1 Synopsis

In this thesis, we have studied the issue of automatically learning transferable feature representations for performing improved image classification across different domains.

In Chapter 1, we introduced the motivations and research objectives of this thesis. Chapter 2 provided a brief discussion on the basics and a comprehensive overview of the transfer learning areas related to the thesis.

As the first and second contributions, Chapters 3 and 4 presented approaches that use high-level (category-specific) CNN parameters and features for sequential transfer learning. Specifically, Chapter 3 presented a novel parameter fine-tuning approach and a new developmental transfer learning approach. Chapters 4 presented a new feature representation transfer approach. All the proposed approaches fall under the supervised learning category and use the category-specific high-level features/parameters of a CNN trained on a large-scale, diverse dataset (ImageNet). The feature representation transfer approach was adopted to perform comprehensive analyses of the factors affecting transfer learning using high-level features and the importance of high-level features in transfer learning. We evaluated the proposed methods on eight diverse benchmark datasets and reported superior results compared to the contemporary approaches. These contributions have addressed the first two research objectives stated in Chapter 1.

To present the third and fourth contributions, Chapters 5 and 6 discussed open set adversarial DA approaches that overcome the divergence between domains by reducing negative transfers. These approaches follow unsupervised settings as the target domain is fully unlabelled. More specifically, Chapter 5 presented the novel approach that reduces negative transfers by using a multi-classifier structure that introduces an adaptive weighting module. The weighting module assesses domain confidence to differentiate between known-unknown target samples and facilitates adversarial adaptation. The new approach proposed in Chapter 6 takes another step forward and evaluates mutual information along with domain confidence to distinguish between known-unknown target samples. We evaluated both the proposed methods on four benchmark datasets and showed they outperform the contemporary works. Chapters 5 and 6 have addressed the third research objective stated in Chapter 1.

To present the fifth and sixth contributions, in Chapters 7 and 8, we studied the problems of zero-shot learning and bias towards the source domain. Chapter 7 presented a novel approach to discovering the local visual details linked to the semantics besides the global details through a dense attention mechanism to improve zero-shot learning. The approach proposed in Chapter 8 is a bidirectional mapping method to improve generative zero-shot learning. For reducing bias towards the source domain, the first proposed GZSL model computes source-target class similarity based on mutual information and transfer-learn the target classes. The second one optimises a new loss to learn and preserve domain discriminative information. We evaluated the proposed methods on four benchmark ZSL datasets and demonstrated their superior performance compared to other approaches. The fifth and sixth contributions have addressed the last two research objectives stated in Chapter 1.

9.2 Main Findings and Insights

Throughout the thesis, we have presented multiple novel approaches for three transfer learning scenarios. We have evaluated the proposed approaches across diverse domains and datasets and demonstrated their superior performance compared to contemporary works. We now discuss how our proposed approaches addressed the research objectives in Chapter 1 and provide a summary of our contributions and findings.

1. **Improve the parameter fine-tuning approach using category-specific features of CNNs** It is widely accepted that the classification layer of a pre-trained CNN holds category-specific features. Thus, the classification layer is discarded before performing any type of parameter fine-tuning. Our goal was to investigate the influence of the classification layer in parameter fine-tuning. We know that ImageNet consists of a massive amount of labelled images of natural and human-made objects. In Section 3.2, we observed that even the classification layer of CNNs trained on large-scale datasets such as ImageNet has plenty of neighbouring features with the target domains containing images of natural and artificial objects. This encouraged us to include the classification layer in parameter fine-tuning (Section 3.3.1). We demonstrated that the proposed fine-tuning approach outperforms contemporary fine-tuning setups (Section 3.4.2). The improved performance motivated us to fulfil the next goal, i.e. use the classification layer for developmental transfer learning (Section 3.3.2). The proposed incremental fine-tuning scheme has shown that freezing the early conv. layers boost performance. In Section 3.4.4, we demonstrated the proposed depth augmented networks' superior performance and observed single-layer depth augmentation having 2048-neurons outperform other combinations. Double-layer augmentation only marginally improves over single-layer augmentation. Finally, we have investigated the best-fit normalisation (Section 3.4.5), learned feature quality (Section 3.4.7), and learn rate (Section 3.4.8) for the proposed models. We found the proposed depth augmentation is facilitated by normalisation, the learned features are well segregated, and the learning rate of the layers depends on the target datasets.

2. **Comprehensively analyse suitable conditions for a feature representation transfer approach using the category-specific features** Feature representation transfer is a popular sequential transfer learning approach, which uses the pre-trained CNN features off-the-shelf to learn other domains or datasets. Similar to traditional parameter fine-tuning, the classification layer features are considered too specific to the source categories and not utilised for transferring knowledge. However, we have proposed to use the classification layer features off-the-shelf for transfer learning (Section 4.2.1). We have constructed feature embedding using the CNN extracted features for source and target datasets and used the embedding to measure the k-nearest neighbour similarity between the datasets (Section 4.2.2). We found that the classification layer features generalise better for target datasets with more similarity (Section 4.3.1) to the source dataset and outperforms traditional transfer learning (Section 4.3.3). To investigate the best classifier, we have exploited two variants of SVMs and MLP and demonstrated that MLP outperforms others (Section 4.3.3). We have observed the influence of similarity between source-target datasets, fine-grained target datasets, coarse target datasets, and the number of target samples and classes in transfer learning with the classification layer features (Section 4.3.4). We verify the importance of classification layer features in transfer learning by employing a mutual information-based feature selection algorithm (Section 4.3.5).

3. **Reduce negative transfer in OSDA** OSDA is an unsupervised transductive transfer learning approach. This approach aims to be able to correctly classify known classes across domains and recognise unknown classes as 'unknown'. The DA task is directly linked to the unknown class separation part. Incorrect known-unknown separation fuels incorrect adaptation. In some cases, the DA model's performance lags behind the non-DA model for the same task. This phenomenon is referred to as negative transfer. Negative transfers occur because of faulty known and unknown target sample separation. Our goal was to reduce negative transfers by designing robust known-unknown separation modules and improve the performance of OSDA. To achieve this goal, our contribution was two-fold. First,

we proposed a novel architecture to improve an existing model (Chapter 5) and then we proposed another new architecture (Chapter 6). In both proposed models, we first focus on improved known-unknown separation and then on adversarial DA. The first proposed model is an improvement of an existing adversarial DA model. The existing model relies on a fixed threshold to separate known from unknown. However, we proposed an adaptive weighting module that improves the known-unknown separation by utilising underlying domain characteristics (Section 5.3.2). In this model, we handled negative transfers by introducing an adaptive weighting module (Section 5.3.2). The module utilises multiple classifiers to explore the domain confidence, i.e. probability of similarity between target samples and source classes. The module evaluates fundamental domain information based on distinctive label information for assigning identifiable weights to the known and unknown target samples. We have demonstrated that our weighting module encourages better separation between known-unknown target samples and reduce negative transfers compared to other approaches (Section 5.4).

In the quest for mitigating negative transfers from OSDA, we have proposed another new adversarial DA model with a weighting module based on underlying domain characteristics and mutual information between domains (Section 6.2). In this model, we tackled negative transfers by introducing a three-step coarse-to-fine known-unknown target sample separation weighting module. Mutual information between domains and underlying domain similarity lies at the core of the proposed module (Section 6.2). Coarse separation based on domain confidence takes place in the first step. Then, shared information between source and target samples are utilised towards a better separation. Finally, pointwise mutual information is used for fine separation. Our extensive empirical evaluation demonstrated that the proposed module reduces negative transfers to a large extent and facilitate DA (Section 6.3).

4. **Improving GZSL** Zero-shot learning is a promising transfer learning approach for target tasks with no available visual training data in the target domain. GZSL is the

advanced variant of ZSL, which requires the final classifier to classify both source and target classes. The objective that we set was to improve GZSL performance by introducing novel architectures. The first sub-objective was to improve GZSL for fine-grained recognition. Fine-grained datasets demand more local discriminative properties. Thus, we have proposed a new two-step dense attention mechanism in Section 7.3, which discovers fine distinctive local visual information directly supervised by the semantic attributes. In addition, we have preserved global information not to harm the usual visual feature representation structure. We have also proposed integrating an embedding learning sub-network and a feature generation sub-network into an integrated network and introducing mutual learning between the sub-networks. We have thoroughly discussed in Section 7.4 that compared to contemporary works, the proposed model improves GZSL and ZSL performance by attending regions that relate to semantic attributes. The second sub-objective was to improve bidirectional mapping GZSL. The generative GZSL approaches learn to construct visual features from class semantics, which encourages weak visual-semantic coupling. Strong visual-semantic coupling is crucial for GZSL, and bidirectional mapping between visual and semantic space can ensure that. We have proposed a new bidirectional mapping approach by extending CoGAN in Section 8.3.1. The proposed model simultaneously learns joint domain distribution and discriminative domain information. We have evaluated the proposed model on benchmark datasets and demonstrated that it outperforms contemporary works (Section 8.4).

5. **Reduce source domain bias for GZSL** As the target domain in GZSL does not have any visual training data, the trained classifier tends to show bias towards the source domain classes. Our goal was to reduce this bias and balance the source and target domain prediction during testing. To smooth out bias towards source classes in the first proposed model, we have loosely learned target classes from the knowledge of the closest source classes (Section 7.3). To measure class similarity, we have used pointwise mutual information. In Section 7.4, we have shown that the proposed approach relaxes the bias towards the source domain.

In the second model, we have designed a new loss optimisation to reduce the bias (Section 8.3.1). The designed loss forces the real source features to attract synthesised source features but repel synthesised target features. This assisted in retaining the required discrimination between both domains and reduce bias. In Section 8.4, we have demonstrated that the proposed model shows improved performance on source and target classes during the test. The results evince a reduction in the bias towards the source domain.

9.3 Future works

In this section, we will briefly discuss the future research directions for each of the respective contribution areas of transfer learning of this thesis.

Sequential transfer learning The first and second contributions of this thesis introduced new parameter fine-tuning, developmental transfer learning, and feature representation transfer approaches using the classification layer of CNNs pre-trained on the ImageNet dataset. Our study revealed many essential insights about the classification layer features. This study will assist the researchers in investigating further the trends of classification layer features of other large-scale source datasets in transfer learning. Using more domain-specific source datasets may provide new insights. Besides, CNN structures other than AlexNet and VGGNet may be explored to investigate sequential transfer learning schemes presented in the thesis.

OSDA The third and fourth contributions were to design models to reduce negative transfers and improve OSDA. The proposed known-unknown separation modules still partially rely on a classifier trained on the source dataset for domain confidence. This creates room for improvement. To further reduce negative transfers, the source trained classifiers may be improved by infusing out-of-source-distribution knowledge.

GZSL Finally, this thesis proposed two new GZSL models in the fifth and sixth contributions. The objective was to reduce bias towards the source domain during testing and improve performance. The contributions open new avenues for research. For example, researchers may implement the proposed dense attention mechanism in medical imagery for disease analysis and anomaly detection, integrate a more sophisticated feature synthesising network in the first proposed model to investigate the change in performance, and improve bidirectional mapping through other types of GANs.

Bibliography

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2012. 1, 21, 28, 31, 43, 45, 50, 64

[2] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *Asian Conference on Pattern Recognition*, 2015. 1, 22, 28, 43, 50

[3] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015. 1, 28, 43, 78

[4] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 1, 23, 24, 28, 43, 78, 94, 95, 96, 115, 117, 118

[5] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Advances in Neural Information Processing Systems*, 2014. 1, 3, 28, 29, 32, 45, 64

[6] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "Cnn features off-the-shelf: an astounding baseline for recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2014. 3, 28, 31, 43, 54, 55, 63, 68

[7] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2009. 4, 25, 80

[8] K. Saito, K. Watanabe, Y. Ushiku, and T. Harada, "Maximum classifier discrepancy for unsupervised domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 4, 33, 78, 119

[9] Z. Ji, Y. Fu, J. Guo, Y. Pang, Z. M. Zhang *et al.*, "Stacked semantics-guided attention model for fine-grained zero-shot learning," in *Advances in Neural Information Processing Systems*, 2018. 4, 38, 39, 129, 134, 145

[10] J. Ni, S. Zhang, and H. Xie, "Dual adversarial semantics-consistent network for generalized zero-shot learning," in *Advances in Neural Information Processing Systems*, 2019. 4, 35, 40, 155, 156, 169

[11] C. E. Shannon, "A mathematical theory of communication," *The Bell system technical journal*, vol. 27, no. 3, pp. 379–423, 1948. 14

[12] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009. 21

[13] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010. 26

[14] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998. 28

[15] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "A deep convolutional activation feature for generic visual recognition." *Proceedings of the International Conference on Machine Learning*, 2014. 28, 45

[16] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Advances in Neural Information Processing Systems*, 2014. 28

[17] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587. 28, 31

[18] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, “Overfeat: Integrated recognition, localization and detection using convolutional networks,” *Proceedings of the International Conference on Learning Representations*, 2014. 28, 31, 45

[19] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, “Long-term recurrent convolutional networks for visual recognition and description,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015. 28

[20] A. Karpathy and L. Fei-Fei, “Deep visual-semantic alignments for generating image descriptions,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015. 28

[21] J. Carreira, P. Agrawal, K. Fragkiadaki, and J. Malik, “Human pose estimation with iterative error feedback,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 28

[22] J. Dai, K. He, and J. Sun, “Instance-aware semantic segmentation via multi-task network cascades,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 28

[23] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid, “Deepflow: Large displacement optical flow with deep matching,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2013. 28

[24] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *nature*, vol. 521, no. 7553, p. 436, 2015. 28

[25] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, “Learning and transferring mid-level image representations using convolutional neural networks,” in *Proceedings of the*

IEEE Conference on Computer Vision and Pattern Recognition, 2014. 29, 45, 46, 63

[26] S. Yang and D. Ramanan, "Multi-scale recognition with dag-cnns," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015. 29, 45

[27] N. Tajbakhsh, J. Y. Shin, S. R. Gurudu, R. T. Hurst, C. B. Kendall, M. B. Gotway, and J. Liang, "Convolutional neural networks for medical image analysis: Full training or fine tuning?" *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1299–1312, 2016. 29

[28] X. Li, H. Xiong, H. Wang, Y. Rao, L. Liu, Z. Chen, and J. Huan, "Delta: Deep learning transfer using feature map with attention for convolutional networks," *Proceedings of the International Conference on Learning Representations*, 2019. 29

[29] W. Ge and Y. Yu, "Borrowing treasures from the wealthy: Deep transfer learning through selective joint fine-tuning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 29, 64

[30] Z. Zhou, J. Shin, L. Zhang, S. Gurudu, M. Gotway, and J. Liang, "Fine-tuning convolutional neural networks for biomedical image analysis: actively and incrementally," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 29

[31] Z. Huang, Z. Pan, and B. Lei, "Transfer learning with deep convolutional neural network for sar target classification with limited labeled data," *Remote Sensing*, vol. 9, no. 9, p. 907, 2017. 29

[32] D. George, H. Shen, and E. Huerta, "Deep transfer learning: A new deep learning glitch classification method for advanced ligo," *Physical Review D*, 2018. 30, 32

[33] C. J. Holder, T. P. Breckon, and X. Wei, "From on-road to off: transfer learning within a deep convolutional neural network for segmentation and classification of off-road scenes," in *Proceedings of the European Conference on Computer Vision*, 2016. 30

[34] Z. Li and D. Hoiem, "Learning without forgetting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017. 30

[35] Y. Tamaazousti, H. L. Borgne, C. Hudelot, M. E. A. Seddik, and M. Tamaazousti, "Learning more universal representations for transfer-learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. 30, 54, 55

[36] O. Sigaud and A. Droniou, "Towards deep developmental learning," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 8, no. 2, pp. 99–114, 2016. 30, 46

[37] C. Tessler, S. Givony, T. Zahavy, D. J. Mankowitz, and S. Mannor, "A deep hierarchical approach to lifelong learning in minecraft," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2017. 30, 46

[38] M. Pickett, R. Al-Rfou, L. Shao, and C. Tar, "A growing long-term episodic & semantic memory," *arXiv:1610.06402*, 2016. 30, 46

[39] H. Azizpour, A. S. Razavian, J. Sullivan, A. Maki, and S. Carlsson, "Factors of transferability for a generic convnet representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 9, pp. 1790–1802, 2015. 30, 31

[40] Y.-X. Wang, D. Ramanan, and M. Hebert, "Growing a brain: Fine-tuning by increasing model capacity," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 30, 31, 46, 47, 49, 54, 55, 58

[41] H. Azizpour, A. Sharif Razavian, J. Sullivan, A. Maki, and S. Carlsson, "From generic to specific deep representations for visual recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2015. 30, 31, 54, 55

[42] N. Becherer, J. Pecarina, S. Nykl, and K. Hopkinson, "Improving optimization of convolutional neural networks through parameter fine-tuning," *Neural Computing and Applications*, pp. 1–11, 2017. 31

[43] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "Decaf: A deep convolutional activation feature for generic visual recognition," in *Proceedings of the International Conference on Machine Learning*, 2014. 31, 32

[44] A. Mahmood, M. Bennamoun, S. An, and F. Sohel, "Resfeats: Residual network based features for image classification," in *Proceedings of the IEEE International Conference on Image Processing*, 2017. 32

[45] R. Mormont, P. Geurts, and R. Marée, "Comparison of deep transfer learning strategies for digital pathology," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018. 32

[46] D. Garcia-Gasulla, F. Parés, A. Vilalta, J. Moreno, E. Ayguadé, J. Labarta, U. Cortés, and T. Suzumura, "On the behavior of convolutional nets for feature extraction," *Journal of Artificial Intelligence Research*, vol. 61, pp. 563–592, 2018. 32

[47] Y.-D. Kim, T. Jang, B. Han, and S. Choi, "Learning to select pre-trained deep representations with bayesian evidence framework," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 32

[48] Q. Qian, R. Jin, S. Zhu, and Y. Lin, "Fine-grained visual categorization via multi-stage metric learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015. 32, 54, 55, 68

[49] M. Long, Z. Cao, J. Wang, and M. I. Jordan, "Conditional adversarial domain adaptation," in *Advances in Neural Information Processing Systems*, 2018. 33

[50] M. Long, H. Zhu, J. Wang, and M. I. Jordan, "Unsupervised domain adaptation with residual transfer networks," in *Advances in Neural Information Processing Systems*, 2016. 33, 78, 94, 95, 96, 97, 115, 116, 117

[51] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell, "Deep domain confusion: Maximizing for domain invariance," *Proceedings of the IEEE International Conference on Computer Vision*, 2014. 33, 78

[52] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 33, 78

[53] M. Long, Y. Cao, J. Wang, and M. I. Jordan, "Learning transferable features with deep adaptation networks," *Proceedings of the International Conference on Machine Learning*, 2015. 33

[54] Y.Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, "Domain-adversarial training of neural networks," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 2096–2030, 2016. 33, 83, 94, 95, 96, 111, 115, 116, 117, 124, 125

[55] P. Haeusser, T. Frerix, A. Mordvintsev, and D. Cremers, "Associative domain adaptation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017. 33, 78

[56] J. Hoffman, E. Tzeng, T. Darrell, and K. Saenko, "Simultaneous deep transfer across domains and tasks," in *Domain Adaptation in Computer Vision Applications*, 2017, pp. 173–187. 33, 83

[57] K. Saenko, B. Kulis, M. Fritz, and T. Darrell, "Adapting visual category models to new domains," in *Proceedings of the European Conference on Computer Vision*, 2010. 33, 93, 94

[58] X. Wang and J. Schneider, "Flexible transfer learning under support and model shift," in *Advances in Neural Information Processing Systems*, 2014. 33

[59] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE Transactions on Neural Networks*, vol. 22, no. 2, pp. 199–210, 2010. 33

[60] B. Gong, Y. Shi, F. Sha, and K. Grauman, "Geodesic flow kernel for unsupervised domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012. 33

[61] K. Zhang, B. Schölkopf, K. Muandet, and Z. Wang, "Domain adaptation under target and conditional shift," in *Proceedings of the International Conference on Machine Learning*, 2013. 33, 46

[62] L. Duan, I. W. Tsang, and D. Xu, "Domain transfer multiple kernel learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 3, pp. 465–479, 2012. 33

[63] M. Long, H. Zhu, J. Wang, and M. I. Jordan, "Deep transfer learning with joint adaptation networks," in *Proceedings of the International Conference on Machine Learning*, 2017. 33

[64] W. Zellinger, T. Grubinger, E. Lughofer, T. Natschläger, and S. Saminger-Platz, "Central moment discrepancy (cmd) for domain-invariant representation learning," *Proceedings of the International Conference on Learning Representations*, 2017. 33

[65] B. Sun and K. Saenko, "Deep coral: Correlation alignment for deep domain adaptation," in *Proceedings of the European Conference on Computer Vision*, 2016. 33

[66] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems*, 2014. 33

[67] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *Proceedings of the International Conference on Machine Learning*, 2015. 33, 78, 79, 85, 91, 94, 95, 96, 113

[68] R. Volpi, P. Morerio, S. Savarese, and V. Murino, "Adversarial feature augmentation for unsupervised domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 33

[69] S. Sankaranarayanan, Y. Balaji, C. D. Castillo, and R. Chellappa, "Generate to adapt: Aligning domains using generative adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 33

[70] Z. Murez, S. Kolouri, D. Kriegman, R. Ramamoorthi, and K. Kim, "Image to image translation for domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 33

[71] Y.-C. Liu, Y.-Y. Yeh, T.-C. Fu, S.-D. Wang, W.-C. Chiu, and Y.-C. Frank Wang, "Detach and adapt: Learning cross-domain disentangled deep representation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 33

[72] S.-W. Huang, C.-T. Lin, S.-P. Chen, Y.-Y. Wu, P.-H. Hsu, and S.-H. Lai, "Auggan: Cross domain adaptation with gan-based data augmentation," in *Proceedings of the European Conference on Computer Vision*, 2018. 33

[73] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan, "Unsupervised pixel-level domain adaptation with generative adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 33, 78, 79

[74] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 34

[75] P. Russo, F. M. Carlucci, T. Tommasi, and B. Caputo, "From source to target and back: symmetric bi-directional adaptive gan," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 34

[76] J. Hoffman, E. Tzeng, T. Park, J.-Y. Zhu, P. Isola, K. Saenko, A. A. Efros, and T. Darrell, "Cycada: Cycle-consistent adversarial domain adaptation," *Proceedings of the International Conference on Machine Learning*, 2018. 34

[77] J. Li, E. Chen, Z. Ding, L. Zhu, K. Lu, and Z. Huang, "Cycle-consistent conditional adversarial transfer networks," in *Proceedings of the ACM International Conference on Multimedia*, 2019. 34

[78] F. M. Cariucci, L. Porzi, B. Caputo, E. Ricci, and S. R. Bulò, "Autodial: Automatic domain alignment layers," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017. 34

[79] Y. Li, N. Wang, J. Shi, J. Liu, and X. Hou, "Revisiting batch normalization for practical domain adaptation," *Proceedings of the International Conference on Learning Representations Workshop*, 2017. 34

[80] A. Bendale and T. E. Boult, "Towards open set deep networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 34, 117

[81] P. P. Busto and J. Gall, "Open set domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 34, 35, 78, 79, 94, 95, 96, 97, 115, 117, 118

[82] Z. Ge, S. Demyanov, Z. Chen, and R. Garnavi, "Generative openmax for multi-class open set classification," *Proceedings of the British Machine Vision Conference*, 2017. 34

[83] L. P. Jain, W. J. Scheirer, and T. E. Boult, "Multi-class open set recognition using probability of inclusion," in *Proceedings of the European Conference on Computer Vision*, 2014. 34

[84] K. Saito, S. Yamamoto, and Y. U. T. Harada, "Open set domain adaptation by backpropagation," in *Proceedings of the European Conference on Computer Vision*, 2018. 34, 78, 79, 80, 81, 82, 83, 91, 93, 94, 95, 96, 97, 99, 111, 115, 117, 118

[85] H. Liu, Z. Cao, M. Long, J. Wang, and Q. Yang, "Separate to adapt: Open set domain adaptation via progressive separation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 35, 81, 94, 95, 96, 97, 103, 115, 116, 117, 118, 119, 120

[86] D. Chang, A. Sain, Z. Ma, Y.-Z. Song, and J. Guo, "Mind the gap: Enlarging the domain gap in open set domain adaptation," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 35, 103, 116, 117, 118

[87] J. N. Kundu, N. Venkat, A. Revanur, and R. V. Babu, "Towards inheritable models for open-set domain adaptation," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 35, 103, 116, 117, 118

[88] K. Lee, H. Lee, K. Lee, and J. Shin, "Training confidence-calibrated classifiers for detecting out-of-distribution samples," *Proceedings of the International Conference on Learning Representations*, 2017. 35

[89] Y. Luo, Z. Wang, Z. Huang, and M. Baktashmotlagh, "Progressive graph learning for open-set domain adaptation," in *Proceedings of the International Conference on Machine Learning*, 2020. 35, 116, 117, 118

[90] M. Baktashmotlagh, M. Faraki, T. Drummond, and M. Salzmann, "Learning factorized representations for open-set domain adaptation," in *Proceedings of the International Conference on Learning Representations*, 2019. 35, 116, 117, 118

[91] N. Tishby, F. C. Pereira, and W. Bialek, "The information bottleneck method," *Proceedings of the Annual Allerton Conference on Communication, Control and Computing*, 2000. 36

[92] A. A. Alemi, I. Fischer, J. V. Dillon, and K. Murphy, "Deep variational information bottleneck," *Proceedings of the International Conference on Learning Representations*, 2017. 36, 133

[93] N. Friedman, O. Mosenzon, N. Slonim, and N. Tishby, "Multivariate information bottleneck," *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence*, 2013. 36

[94] Y. Song, L. Yu, Z. Cao, Z. Zhou, J. Shen, S. Shao, W. Zhang, and Y. Yu, "Improving unsupervised domain adaptation with variational information bottleneck," *Proceedings of the European Conference on Artificial Intelligence*, 2019. 36

[95] X. Peng, Z. Huang, X. Sun, and K. Saenko, "Domain agnostic learning with disentangled representations," *Proceedings of the International Conference on Machine Learning*, 2019. 36, 104

[96] Y. Shi and F. Sha, "Information-theoretical learning of discriminative clusters for unsupervised domain adaptation," in *Proceedings of the International Conference on Machine Learning*, 2012. 36, 104

[97] Z. Feng, C. Xu, and D. Tao, "Self-supervised representation learning from multi-domain data," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019. 36, 104

[98] K. Wang, J. Liu, and J.-Y. Wang, "Learning domain independent deep representations by mutual information minimization," *Computational Intelligence and Neuroscience*, 2019. 36, 104

[99] Y. Pan, T. Yao, Y. Li, C.-W. Ngo, and T. Mei, "Exploring category-agnostic clusters for open-set domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 36, 104

[100] B. Gholami, P. Sahu, O. Rudovic, K. Bousmalis, and V. Pavlovic, "Unsupervised multi-target domain adaptation: An information theoretic approach," *IEEE Transactions on Image Processing*, vol. 29, pp. 3993–4002, 2020. 36, 104

[101] X. Cui, F. Coenen, and D. Bollegala, "Effect of data imbalance on unsupervised domain adaptation of part-of-speech tagging and pivot selection strategies," in *Proceedings of the International Workshop on Learning with Imbalanced Domains: Theory and Applications*, 2017. 36, 104

[102] D. Bollegala, D. Weir, and J. Carroll, "Cross-domain sentiment classification using a sentiment sensitive thesaurus," *IEEE Transactions on Knowledge and Data engineering*, vol. 25, no. 8, pp. 1719–1731, 2012. 36

[103] X. Cui, F. Coenen, and D. Bollegala, "Tsp: Learning task-specific pivots for unsupervised domain adaptation," in *Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2017. 36

[104] X. Cui, "Learning transferable features for unsupervised domain adaptation in natural language processing," Ph.D. dissertation, University of Liverpool, 2020.

36, 104

[105] Z. Cao, M. Long, J. Wang, and M. I. Jordan, "Partial transfer learning with selective adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 36, 37, 78, 81, 93

[106] Z. Cao, L. Ma, M. Long, and J. Wang, "Partial adversarial domain adaptation," in *Proceedings of the European Conference on Computer Vision*, 2018. 37, 81

[107] K. You, M. Long, Z. Cao, J. Wang, and M. I. Jordan, "Universal domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 37, 81, 94, 95, 96, 97

[108] C. H. Lampert, H. Nickisch, and S. Harmeling, "Learning to detect unseen object classes by between-class attribute transfer," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009. 37, 38, 142

[109] ——, "Attribute-based classification for zero-shot visual object categorization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2013. 37

[110] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth, "Describing objects by their attributes," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009. 37, 38

[111] Y. Fu, T. M. Hospedales, T. Xiang, and S. Gong, "Transductive multi-view zero-shot learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2015. 37

[112] Y. Yu, Z. Ji, X. Li, J. Guo, Z. Zhang, H. Ling, and F. Wu, "Transductive zero-shot learning with a self-training dictionary approach," *IEEE Transactions on Cybernetics*, 2018. 37

[113] H. Jiang, R. Wang, S. Shan, and X. Chen, "Transferable contrastive network for generalized zero-shot learning," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019. 37, 41, 145, 169

[114] Z. Akata, S. Reed, D. Walter, H. Lee, and B. Schiele, "Evaluation of output embeddings for fine-grained image classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015. 38, 39

[115] E. Kodirov, T. Xiang, and S. Gong, "Semantic autoencoder for zero-shot learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 38

[116] Y. Shigeto, I. Suzuki, K. Hara, M. Shimbo, and Y. Matsumoto, "Ridge regression, hubness, and zero-shot learning," in *Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2015. 38

[117] G. Dinu, A. Lazaridou, and M. Baroni, "Improving zero-shot learning by mitigating the hubness problem," *Proceedings of the International Conference on Learning Representations*, 2015. 38

[118] L. Zhang, T. Xiang, and S. Gong, "Learning a deep embedding model for zero-shot learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 38, 39

[119] B. Romera-Paredes and P. Torr, "An embarrassingly simple approach to zero-shot learning," in *Proceedings of the International Conference on Machine Learning*, 2015. 38, 39

[120] Z. Akata, M. Malinowski, M. Fritz, and B. Schiele, "Multi-cue zero-shot learning with strong supervision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 38, 39

[121] G.-S. Xie, L. Liu, X. Jin, F. Zhu, Z. Zhang, J. Qin, Y. Yao, and L. Shao, "Attentive region embedding network for zero-shot learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 38, 39, 129, 145

[122] Y. Zhu, J. Xie, Z. Tang, X. Peng, and A. Elgammal, "Semantic-guided multi-attention localization for zero-shot learning," in *Advances in Neural Information Processing Systems*, 2019. 38, 39, 129, 144, 145, 168

[123] Y. Guo, G. Ding, J. Han, and S. Tang, "Zero-shot learning with attribute selection," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018. 38, 39

[124] Y. Liu, J. Guo, D. Cai, and X. He, "Attribute attention for semantic disambiguation in zero-shot learning," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019. 38, 39

[125] L. Huang, W. Wang, J. Chen, and X.-Y. Wei, "Attention on attention for image captioning," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019. 38, 39

[126] D. Huynh and E. Elhamifar, "Fine-grained generalized zero-shot learning via dense attribute-based attention," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2020. 38, 39, 41, 128, 133, 134, 142, 145, 146, 169

[127] Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid, "Label-embedding for attribute-based classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013. 38

[128] P. Morgado and N. Vasconcelos, "Semantically consistent regularization for zero-shot recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 38

[129] S. Changpinyo, W.-L. Chao, and F. Sha, "Predicting visual exemplars of unseen classes for zero-shot learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3476–3485. 38

[130] Y. Long, L. Liu, F. Shen, L. Shao, and X. Li, "Zero-shot learning using synthesised unseen visual data with diffusion regularisation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 10, pp. 2498–2512, 2017. 38

[131] X. Wang, Y. Ye, and A. Gupta, "Zero-shot recognition via semantic embeddings and knowledge graphs," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018. 38

[132] H. Huang, C. Wang, P. S. Yu, and C.-D. Wang, "Generative dual adversarial network for generalized zero-shot learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 38, 40, 155, 169

[133] A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, M. Ranzato, and T. Mikolov, "Devise: A deep visual-semantic embedding model," in *Advances in Neural Information Processing Systems*, 2013. 39

[134] Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid, "Label-embedding for image classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2015. 39

[135] Y.-H. Hubert Tsai, L.-K. Huang, and R. Salakhutdinov, "Learning robust visual-semantic embeddings," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017. 39

[136] S. Changpinyo, W.-L. Chao, B. Gong, and F. Sha, "Synthesized classifiers for zero-shot learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 39

[137] Y. Xian, Z. Akata, G. Sharma, Q. Nguyen, M. Hein, and B. Schiele, "Latent embeddings for zero-shot classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 39, 144, 145, 168

[138] H. Zhang and P. Koniusz, "Zero-shot kernel learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 39

[139] S. Liu, M. Long, J. Wang, and M. I. Jordan, "Generalized zero-shot learning with deep calibration network," in *Advances in Neural Information Processing Systems*, 2018, pp. 2005–2015. 39, 41, 145, 169

[140] W. Xu, Y. Xian, J. Wang, B. Schiele, and Z. Akata, "Attribute prototype network for zero-shot learning," in *Advances of the Neural Information Processing Systems*, 2020. 39, 145, 146

[141] Y. Xian, T. Lorenz, B. Schiele, and Z. Akata, "Feature generating networks for zero-shot learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 40, 137, 145, 155, 162, 164, 165, 169

[142] M. B. Sariyildiz and R. G. Cinbis, "Gradient matching generative networks for zero-shot learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 40, 145, 169

[143] V. Kumar Verma, G. Arora, A. Mishra, and P. Rai, "Generalized zero-shot learning via synthesized examples," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 40, 145, 155, 169

[144] Y. Xian, S. Sharma, B. Schiele, and Z. Akata, "f-vaegan-d2: A feature generating framework for any-shot learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 40, 129, 145, 146, 169

[145] R. Felix, V. B. Kumar, I. Reid, and G. Carneiro, "Multi-modal cycle-consistent generalized zero-shot learning," in *Proceedings of the European Conference on Computer Vision*, 2018. 40, 145, 155, 169

[146] Z. Lu, Y. Yu, Z.-M. Lu, F.-L. Shen, and Z. Zhang, "Attentive semantic preservation network for zero-shot learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2020. 40, 144, 145, 169

[147] E. Schonfeld, S. Ebrahimi, S. Sinha, T. Darrell, and Z. Akata, "Generalized zero-and few-shot learning via aligned variational autoencoders," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 40, 155, 169

[148] R. Keshari, R. Singh, and M. Vatsa, "Generalized zero-shot learning via over-complete distribution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 40

[149] Y. Yu, Z. Ji, J. Guo, and Z. Zhang, "Zero-shot learning via latent space encoding," *IEEE Transactions on Cybernetics*, 2018. 40

[150] A. Mishra, S. Krishna Reddy, A. Mittal, and H. A. Murthy, "A generative model for zero shot learning using conditional variational autoencoders," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018. 40

[151] Z. Han, Z. Fu, and J. Yang, "Learning the redundancy-free features for generalized zero-shot object recognition," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2020. 40, 129, 134, 145, 146, 155, 156, 169

[152] J. Li, M. Jing, K. Lu, Z. Ding, L. Zhu, and Z. Huang, "Leveraging the invariant side of generative zero-shot learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 40, 145, 155, 169

[153] S. Chandhok and V. N. Balasubramanian, "Two-level adversarial visual-semantic coupling for generalized zero-shot learning," *Proceedings of the Workshop on Applications of Computer Vision*, 2020. 41, 155, 169, 170

[154] Y. Xing, S. Huang, L. Huangfu, F. Chen, and Y. Ge, "Robust bidirectional generative network for generalized zero-shot learning," in *Proceedings of the IEEE International Conference on Multimedia and Expo*, 2020. 41, 155, 169

[155] Y.-C. Gu, L. Zhang, Y. Liu, S.-P. Lu, and M.-M. Cheng, "Generalized zero-shot learning via vae-conditioned generative flow," *arXiv:2009.00303*, 2020. 41, 155, 169

[156] Y. Liu, L. Zhou, X. Bai, L. Gu, T. Harada, and J. Zhou, "Information bottleneck constrained latent bidirectional embedding for zero-shot learning," *arXiv:2009.07451*, 2020. 41, 155, 169, 170

[157] A. Pambala, T. Dutta, and S. Biswas, "Generative model with semantic embedding and integrated classifier for generalized zero-shot learning," in *The IEEE Winter Conference on Applications of Computer Vision*, 2020. 41, 42, 155, 169, 170

[158] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik, "Hypercolumns for object segmentation and fine-grained localization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015. 45

[159] Q. V. Le, A. Karpenko, J. Ngiam, and A. Y. Ng, “Ica with reconstruction cost for efficient overcomplete feature learning,” in *Advances in Neural Information Processing Systems*, 2011. 45

[160] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, “Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations,” in *Proceedings of the International Conference on Machine Learning*, 2009. 45

[161] M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks,” in *Proceedings of the European Conference on Computer Vision*, 2014. 45

[162] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in *Proceedings of the International Conference on Machine Learning*, 2015. 47, 56

[163] M.-E. Nilsback and A. Zisserman, “Automated flower classification over a large number of classes,” in *Proceedings of the Indian Conference on Computer Vision, Graphics & Image Processing*, 2008. 49

[164] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, “The caltech-ucsd birds-200-2011 dataset,” 2011. 50

[165] A. Khosla, N. Jayadevaprakash, B. Yao, and L. Fei-Fei, “Novel dataset for fine-grained image categorization,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshop*, 2011. 50

[166] O. M. Parkhi, A. Vedaldi, A. Zisserman, and C. V. Jawahar, “Cats and dogs,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012. 50

[167] G. Griffin, A. Holub, and P. Perona, “Caltech-256 object category dataset,” 2007. 50

[168] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results.” 50

[169] A. Quattoni and A. Torralba, "Recognizing indoor scenes," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2009, pp. 413–420. 50

[170] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba, "Sun database: Large-scale scene recognition from abbey to zoo," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2010. 50

[171] A. Joulin, L. van der Maaten, A. Jabri, and N. Vasilache, "Learning visual features from large weakly supervised data," in *Proceedings of the European Conference on Computer Vision*, 2016. 54, 55, 68

[172] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008. 57, 97, 120, 121

[173] D. Garcia-Gasulla, A. Vilalta, F. Parés, E. Ayguadé, J. Labarta, U. Cortés, and T. Suzumura, "An out-of-the-box full-network embedding for convolutional neural networks," in *Proceedings of the IEEE International Conference on Big Knowledge*, 2018. 64

[174] G. Brown, A. Pocock, M.-J. Zhao, and M. Luján, "Conditional likelihood maximisation: a unifying framework for information theoretic feature selection," *The journal of machine learning research*, vol. 13, no. 1, pp. 27–66, 2012. 65

[175] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015. 68, 93

[176] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2017. 78

[177] T. Shermin, S. W. Teng, M. Murshed, G. Lu, F. Sohel, and M. Paul, "Enhanced transfer learning with imagenet trained classification layer," in *Proceedings of the Pacific-Rim Symposium on Image and Video Technology (PSIVT)*, 2019. 78

[178] J. Li, K. Lu, Z. Huang, L. Zhu, and H. T. Shen, “Heterogeneous domain adaptation through progressive alignment,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 5, pp. 1381–1391, 2018. 78

[179] J. Li, M. Jing, K. Lu, L. Zhu, and H. T. Shen, “Locality preserving joint transfer for domain adaptation,” *IEEE Transactions on Image Processing*, vol. 28, no. 12, pp. 6103–6115, 2019. 78

[180] X. Ma, T. Zhang, and C. Xu, “Deep multi-modality adversarial networks for unsupervised domain adaptation,” *IEEE Transactions on Multimedia*, vol. 21, no. 9, pp. 2419–2431, 2019. 78

[181] M. Liu, T. Breuel, and J. Kautz, “Unsupervised image-to-image translation networks,” in *Advances in Neural Information Processing Systems*, 2017. 78

[182] Y. Taigman, A. Polyak, and L. Wolf, “Unsupervised cross-domain image generation,” *Proceedings of the International Conference on Learning Representations*, 2016. 78

[183] J. Li, K. Lu, Z. Huang, L. Zhu, and H. T. Shen, “Transfer independently together: A generalized framework for domain adaptation,” *IEEE Transactions on Cybernetics*, vol. 49, no. 6, pp. 2144–2155, 2018. 78

[184] Z. Cao, K. You, M. Long, J. Wang, and Q. Yang, “Learning to transfer examples for partial domain adaptation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 78, 81, 88, 93, 94, 95, 96, 97, 98, 107

[185] J. Zhang, Z. Ding, W. Li, and P. Ogunbona, “Importance weighted adversarial nets for partial domain adaptation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 78, 81, 94, 95, 96, 97, 98, 99

[186] A. Odena, C. Olah, and J. Shlens, “Conditional image synthesis with auxiliary classifier gans,” in *Proceedings of the International Conference on Machine Learning*, 2017. 88

[187] X. Peng, B. Usman, N. Kaushik, D. Wang, J. Hoffman, and K. Saenko, "Visda: A synthetic-to-real benchmark for visual domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018. 93

[188] H. Venkateswara, J. Eusebio, S. Chakraborty, and S. Panchanathan, "Deep hashing network for unsupervised domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 93

[189] T. Shermin, G. Lu, S. W. Teng, M. Murshed, and F. Sohel, "Adversarial network with multiple classifiers for open set domain adaptation," *IEEE Transactions on Multimedia (early access)*, 2020. 103, 116, 117, 118

[190] S. Nowozin, B. Cseke, and R. Tomioka, "f-gan: Training generative neural samplers using variational divergence minimization," *Advances in Neural Information Processing Systems*, 2016. 104

[191] D. B. F. Agakov, "The im algorithm: a variational approach to information maximization," *Advances in Neural Information Processing Systems*, vol. 16, p. 201, 2004. 104

[192] M. I. Belghazi, A. Baratin, S. Rajeswar, S. Ozair, Y. Bengio, A. Courville, and R. D. Hjelm, "Mine: mutual information neural estimation," *Proceedings of the International Conference on Machine Learning*, 2018. 104, 109

[193] B. Poole, S. Ozair, A. v. d. Oord, A. A. Alemi, and G. Tucker, "On variational bounds of mutual information," *Proceedings of the Machine Learning Research*, 2019. 104

[194] T. Miyato, S.-i. Maeda, M. Koyama, and S. Ishii, "Virtual adversarial training: a regularization method for supervised and semi-supervised learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 8, pp. 1979–1993, 2018. 112

[195] R. Shu, H. H. Bui, H. Narui, and S. Ermon, "A dirt-t approach to unsupervised

domain adaptation," *Proceedings of the International Conference on Learning Representations*, 2018. 112

[196] R. Xu, Y. Z. Pelen Liu, F. Cai, J. Wang, S. Liang, H. Ying, and J. Yin, "Joint partial optimal transport for open set domain adaptation," in *Proceedings of the International Joint Conferences on Artificial Intelligence Organization*, 2020. 116, 117

[197] S. Bucci, M. R. Loghmani, and T. Tommasi, "On the effectiveness of image rotation for open set domain adaptation," in *Proceedings of the European Conference on Computer Vision*, 2020. 119, 120

[198] W. J. Scheirer, A. de Rezende Rocha, A. Sapkota, and T. E. Boult, "Toward open set recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 7, pp. 1757–1772, 2012. 121

[199] S. Ben-David, J. Blitzer, K. Crammer, F. Pereira *et al.*, "Analysis of representations for domain adaptation," *Advances in Neural Information Processing Systems*, 2007. 124

[200] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2014. 128, 144

[201] Y. Xian, B. Schiele, and Z. Akata, "Zero-shot learning-the good, the bad and the ugly," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 128, 142

[202] H. Zhang, L. Liu, Y. Long, Z. Zhang, and L. Shao, "Deep transductive network for generalized zero shot learning," *Pattern Recognition*, vol. 105, p. 107370, 2020. 128

[203] C. Geng, L. Tao, and S. Chen, "Guided cnn for generalized zero-shot and open-set recognition using visual and semantic prototypes," *Pattern Recognition*, 2020. 128, 145, 169

[204] Z. Li, L. Yao, X. Chang, K. Zhan, J. Sun, and H. Zhang, “Zero-shot event detection via event-adaptive concept relevance mining,” *Pattern Recognition*, vol. 88, pp. 595–603, 2019. 128

[205] M. Xing, Z. Feng, Y. Su, W. Peng, and J. Zhang, “Ventral & dorsal stream theory based zero-shot action recognition,” *Pattern Recognition*, vol. 116, p. 107953, 2021. 129

[206] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, “Show, attend and tell: Neural image caption generation with visual attention,” in *Proceedings of the International Conference on Machine Learning*, 2015. 133

[207] D. Huyn and E. Elhamifar, “A shared multi-attention framework for multi-label zero-shot learning,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 134

[208] D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” *Proceedings of the International Conference on Learning Representations*, 2015. 134

[209] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona, “Caltech-ucsd birds 200,” 2010. 142

[210] G. Patterson and J. Hays, “Sun attribute database: Discovering, annotating, and recognizing scene attributes,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012. 142

[211] Y. Xian, C. H. Lampert, B. Schiele, and Z. Akata, “Zero-shot learning—a comprehensive evaluation of the good, the bad and the ugly,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018. 142

[212] H. Zheng, J. Fu, T. Mei, and J. Luo, “Learning multi-attention convolutional neural network for fine-grained image recognition,” in *Proceedings of the IEEE international conference on computer vision*, 2017. 144, 145, 168

[213] F. Zhang and G. Shi, "Co-representation network for generalized zero-shot learning," in *Proceedings of the International Conference on Machine Learning*, 2019. 145, 169

[214] S. Min, H. Yao, H. Xie, C. Wang, Z.-J. Zha, and Y. Zhang, "Domain-aware visual bias eliminating for generalized zero-shot learning," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2020. 145, 169

[215] S. Narayan, A. Gupta, F. S. Khan, C. G. Snoek, and L. Shao, "Latent embedding feedback and discriminative features for zero-shot classification," *Proceedings of the European Conference on Computer Vision*, 2020. 145, 169

[216] Y. Yu, Z. Ji, J. Han, and Z. Zhang, "Episode-based prototype generating network for zero-shot learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020, pp. 14 035–14 044. 145

[217] A. Paul, N. C. Krishnan, and P. Munjal, "Semantically aligned bias reducing zero shot learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 145

[218] M.-Y. Liu and O. Tuzel, "Coupled generative adversarial networks," in *Advances in neural information processing systems*, 2016. 156, 158, 174

[219] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, "A discriminative feature learning approach for deep face recognition," in *Proceedings of the European Conference on Computer Vision*, 2016. 165

[220] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," in *Advances in Neural Information Processing Systems*, 2017. 168

[221] K. Li, M. R. Min, and Y. Fu, "Rethinking zero-shot learning: A conditional visual classification perspective," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019. 169

[222] M. R. Vyas, H. Venkateswara, and S. Panchanathan, “Leveraging seen and unseen semantic relationships for generative zero-shot learning,” in *Proceedings of the European Conference on Computer Vision*, 2020. 169

[223] L. Feng and C. Zhao, “Transfer increment for generalized zero-shot learning,” *IEEE Transactions on Neural Networks and Learning Systems*, 2020. 169