

**FEATURE-INSTANCE BASED FINE-TUNING IN  
TRANSFER LEARNING MODEL**

**RAPHAEL NGIGI WANJIKU**

**DOCTOR OF PHILOSOPHY  
(Information Technology)**

**JOMO KENYATTA UNIVERSITY  
OF  
AGRICULTURE AND TECHNOLOGY**

**2024**

# **Feature-Instance Based Fine-Tuning In Transfer Learning Model**

**Raphael Ngigi Wanjiku**

**A Thesis Submitted in Partial Fulfillment of the Requirements for  
the Degree of Doctor of Philosophy in Information Technology of the  
Jomo Kenyatta University of Agriculture and Technology**

**2024**

## DECLARATION

This research thesis is my original work and has not been presented for a degree at any other university.

Signature.....Date.....

**Raphael Ngigi Wanjiku**

This thesis has been submitted for examination with our approval as the University Supervisors

Signature.....Date.....

**Dr. Lawrence Nderu, PhD**  
**JKUAT, Kenya**

Signature.....Date.....

**Dr. Michael Kimwele, PhD**  
**JKUAT, Kenya**

## **DEDICATION**

To my dear wife Anne and my children Caleb and Eliana, I appreciate their time and sacrifice to make this possible.

## **ACKNOWLEDGEMENT**

I want to thank God for the opportunity to pursue my Doctoral studies. I appreciate the DAAD for the scholarship funding and the management of the Jomo Kenyatta University of Agriculture and Technology (JKUAT) for allowing me to study and complete my PhD program. I want to thank my supervisors, Dr. Lawrence Nderu and Dr. Michael Kimwele, who guided me throughout the program. Their high professionalism and commitment enabled me to complete the study within an acceptable period. My other gratitude goes to Dr Lawrence Nderu, Chairman of the Computing Department, Dr Kimwele, Prof. Waweru, and Dr Rimiru, who inspired and mentored me in the field of Artificial Intelligence, more so their guidance and support in the publication phase. My sincere gratitude goes to the director of the School of Computing and Information Technology (SCIT), Dr Agnes Mindila, and including Prof. Stephen Kimani, Dr Dennis Kaburu, Dr Petronilla Muriithi, Dr Jael Wekesa, Dr Anne Kibe, Prof. George Okeyo, Prof. Muliaro Wafula, Dr Ogada Kennedy, Dr Tobias Mwalili, Dr Karanja Mwangi, and other staff of JKUAT who actively participated in providing feedback, comments, and guidance during the coursework, project thesis, and seminar presentations.

## TABLE OF CONTENTS

<b>DECLARATION</b> .....	<b>ii</b>
<b>DEDICATION</b> .....	<b>iii</b>
<b>ACKNOWLEDGEMENT</b> .....	<b>iv</b>
<b>TABLE OF CONTENTS</b> .....	<b>v</b>
<b>LIST OF TABLES</b> .....	<b>xii</b>
<b>LIST OF FIGURES</b> .....	<b>xiv</b>
<b>LIST OF APPENDICES</b> .....	<b>xvii</b>
<b>ACRONYMS AND ABBREVIATIONS</b> .....	<b>xviii</b>
<b>ABSTRACT</b> .....	<b>xx</b>
<b>CHAPTER ONE</b> .....	<b>1</b>
<b>INTRODUCTION</b> .....	<b>1</b>
1.1 Background Information .....	1
1.2 Transfer Learning Process.....	3
1.3 Image Data Features .....	4
1.4 Problem Statement .....	5
1.5 Justification .....	6
1.6 Research Objectives .....	7
1.6.1 General Objective.....	7

1.6.2 Specific Objectives .....	7
1.7 Research Questions .....	7
1.8 Proposed Solutions .....	8
1.9 Research Contributions .....	8
1.10 Research Scope .....	9
<b>CHAPTER TWO.....</b>	<b>10</b>
<b>LITERATURE REVIEW.....</b>	<b>10</b>
2.1 Introduction .....	10
2.2 Convolutional Process.....	10
2.3 Transfer Learning Methods.....	12
2.3.1 Label-based Categorization.....	12
2.3.2 Feature Space Categorization .....	15
2.4 Feature-Based Transfer learning .....	21
2.4.1 Label Efficient Learning of Transferable Representations across Domains and Tasks .....	21
2.4.2 Training SVMs Using Deep Features .....	22
2.4.3 Borrowing Treasures from the Wealthy (Selective Fine-Tuning).....	23
2.4.4 CNNs Texture Classification Using Filter Banks.....	23
2.4.5 Maximum-Likelihood Principle and DCNNs in Approximate Nearest Neighbour Search .....	24

2.4.6 EEG-Based Human Intention Recognition Using Spatio-Temporal Pre-Serving Representations .....	25
2.4.7 Video Semantic Recognition Using Adaptive Semi-Supervised Feature Analysis.....	25
2.4.8 Human Activity Recognition Using Semi-Supervised Recurrent Convolutional Attention Model.....	26
2.5 Instance-based Transfer learning .....	27
2.5.1 Adaptive Fine-Tuning .....	27
2.5.2 Adaptive Filter Fine-Tuning for Deep Transfer Learning (AdaFilter) .....	27
2.5.3 Flex-Tuning .....	28
2.5.4 Using Genetic Algorithm in Transfer Learning Layer Selection .....	29
2.5.5 Adaptive Fine-Tuning in Transfer Learning .....	29
2.6 Distance Measures Literature .....	30
2.6.1 Similarity Distance Measures .....	30
2.6.2 Divergence Distance Measures .....	33
2.7 Summary .....	36
<b>CHAPTER THREE .....</b>	<b>37</b>
<b>METHODOLOGY.....</b>	<b>37</b>
3.1 Introduction .....	37
3.2 Methodology .....	37
3.3 Data Collection.....	39



3.3.1 Source Dataset.....	39
3.3.2 Target datasets.....	40
3.4 Modelling Architectures.....	43
3.4.1 VGG16.....	43
3.4.2 DenseNet169.....	44
3.4.3 MobileNetV2.....	44
3.4.4 InceptionV3.....	45
3.4.5 ResNet50.....	45
3.5 Conceptual New Model.....	46
3.6 Proposed Methods.....	47
3.6.1 Features conflation Approach.....	47
3.6.2 Dynamic Layer Selection Approach.....	55
3.7 Experimental Methods.....	59
3.7.1 Phase 1: Textural Features Extraction and Analysis Methods.....	59
3.7.2 Phase 2: Weights Comparison methods in Dynamic layer selection.....	64
3.7.3 Baseline (Conventional) Methods.....	65
3.8 Experimental Setups.....	66
3.8.1 Epochs.....	66
3.8.2 Feature Extraction Settings.....	67

3.8.3 Optimizer.....	67
3.8.4 Loss Function .....	67
3.8.5 Other settings.....	68
3.9 Summary .....	71
<b>CHAPTER FOUR .....</b>	<b>72</b>
<b>RESEARCH RESULTS AND DISCUSSION.....</b>	<b>72</b>
4.1 Introduction .....	72
4.2 Investigation of Parameter Changes in the Dynamically Fine-Tuned Model..	72
4.3 Investigation of Textural Features Conflation on the Dynamically Fine-Tuned Model.....	76
4.3.1 Performance on GLCM and LBP Properties.....	76
4.4 Investigation of Dynamic Layer selection Approach on the New Model .....	82
4.4.1 Comparison between Distance Measures .....	82
4.4.2 Comparison between the Selected Methods and Conventional Baseline Methods .....	86
4.4.3 Evaluation of Dynamic Layer Selection Approach on New MobileNets ...	91
4.5 Investigation of New Model Pipeline (Textural Features Conflation and Dynamic Layer Selection).....	93
4.5.1 Evaluation of the Proposed Methods against Conventional Baseline Methods .....	99
4.6 Investigations on Computational Complexities of the New Model .....	101

4.6.1 Evaluation of Conflation Approach Computational Complexity.....	101
4.6.2 Evaluation of Dynamic selection Approach Computational Complexity	103
4.6.3 Evaluation of New Model’s Pipeline (Merged Proposed Methods) Computational Complexities .....	103
4.7 Discussion on the Selection of Target Samples using Textural Features Conflation .....	105
4.8 Discussion on the Dynamic Selection of New Model Layers for Fine-Tuning .....	106
4.9 Comparison to Existing Methods.....	108
4.10 Discussion on the Computational Complexity of the New Model.....	111
4.11 Summary .....	112
<b>CHAPTER FIVE .....</b>	<b>113</b>
<b>SUMMARY, CONCLUSIONS AND FUTURE WORK .....</b>	<b>113</b>
5.1 Dynamically Fine-Tuned Model Summary .....	113
5.2 Objectives Review .....	113
5.3 Knowledge Contributions .....	115
5.4 Conclusion .....	117
5.5 Future Work.....	118
<b>REFERENCES .....</b>	<b>120</b>
<b>APPENDICES .....</b>	<b>150</b>



## LIST OF TABLES

<b>Table 3.1:</b> Distribution of Samples for the Experimental Datasets.....	40
<b>Table 4.1:</b> Comparison of Accuracy Performance (%) on Selected Datasets on New VGG16 .....	77
<b>Table 4.2:</b> New VGG16 Precision Performance (%) on Selected Datasets .....	79
<b>Table 4.3:</b> New MobileNetV2 Recall Performance (%) on Selected Datasets .....	81
<b>Table 4.4:</b> Comparison of Fashion-MNIST Dataset Accuracy Performance (%) of Selected New Models.....	83
<b>Table 4.5:</b> Comparison of MNIST Dataset Accuracy Performance (%) of Selected Models.....	83
<b>Table 4.6:</b> Comparison of CIFAR-10 Dataset Accuracy Performance (%) of Selected New Models .....	84
<b>Table 4.7:</b> Comparison of CIFAR-100 Dataset Accuracy Performance (%) of Selected New Models .....	84
<b>Table 4.8:</b> Comparison of DKL Accuracy (%) and Loss Using Selected Fine-Tuning Methods on the New InceptionV3 Model .....	87
<b>Table 4.9:</b> Comparison of DKL Accuracy (%) and Loss Using Selected Fine-Tuning Methods on the New VGG16 Model .....	88
<b>Table 4.10:</b> Comparison of DKL Accuracy (%) and Loss Using Selected Fine-Tuning Methods on the DenseNet169 Model.....	88
<b>Table 4.11:</b> Comparison of DKL Accuracy (%) and Loss Using Selected Fine-Tuning Methods on the New MobileNet Model.....	92
<b>Table 4.12:</b> New ResNet50 Accuracy Performance (%) Using Selected DKL Methods	

.....	95
<b>Table 4.13:</b> New InceptionV3 Accuracy Performance (%) Using Selected DKL Methods .....	96
<b>Table 4.14:</b> New DenseNet169 Accuracy Performance (%) Using Selected DKL Methods .....	97
<b>Table 4.15:</b> New MobileNetV2 Accuracy Performance (%) Using Selected DKL Methods .....	98
<b>Table 4.16:</b> New MobileNetV2 Precision Performance (%) Using Selected DKL Methods .....	98
<b>Table 4.17:</b> Proposed Methods and Commonly Used Methods Accuracy Performance (%) on ChestX-Ray8 (Homogeneity).....	100
<b>Table 4.18:</b> Comparison of Complexity Distances on New VGG16 Using LBP ...	102
<b>Table 4.19:</b> Comparison of Complexity Distances on New MobileNet Using GLCM Correlation.....	102
<b>Table 4.20:</b> DKL Complexity Comparison to Other Divergence Measures .....	103
<b>Table 4.21:</b> Divergence Measures Computational Complexity Using Conflation Approach on a Sample Caltech256 Data Point .....	104
<b>Table 4.22:</b> Divergence Measures Computational Complexity Using Dynamically Selected Layers Approach on a Sample Caltech 256 Data Point.....	104

## LIST OF FIGURES

<b>Figure 1.1:</b> Transfer Learning Process .....	2
<b>Figure 2.1:</b> Convolutional Neural Network .....	11
<b>Figure 2.2:</b> Convolutional Layer Filtering Components.....	11
<b>Figure 2.3:</b> Transductive vs Inductive Learning .....	14
<b>Figure 3.1:</b> CRISP-DM Illustration.....	37
<b>Figure 3.2:</b> Transfer Learning Conceptual Model .....	47
<b>Figure 3.3:</b> Conflation Approach Conceptual Diagram .....	49
<b>Figure 3.4:</b> Example of the Conflation Process. (a) Source Images Conflation Steps and (b) Target Images Conflation with Selected Samples .....	52
<b>Figure 3.5:</b> Pipeline of the Dynamic Layer Selection Approach: (a) Layer(s) and Its Filter Weights Identification, (b) Correlation and Ranking of the Layers Using DKL and (c) Use of the Selected Layers in the Fine-Tuning Process .....	58
<b>Figure 3.6:</b> A Flowchart of the New Approach .....	59
<b>Figure 3.7:</b> LBP Decimal Value Extraction on an Image .....	63
<b>Figure 4.1:</b> Illustration of Parameter Changes of the ResNet50 Pre-Trained Model and the New ResNet50 Model .....	73
<b>Figure 4.2:</b> Illustration of Parameter Changes of VGG16 Pre-Trained Model and the New VGG16 Model .....	74
<b>Figure 4.3:</b> An Illustration of the Pre-Trained VGG16 Model and the New VGG16 Model Showing the ‘T’ (Trainable) and ‘NT’ (Non-Trainable) Layers	75

<b>Figure 4.4:</b> GLCM Correlation Using Caltech256 on the New VGG Model .....	78
<b>Figure 4.5:</b> LBP Using MIT Indoor on the New MobileNetV2 Model .....	78
<b>Figure 4.6:</b> LBP Precision Plot of the New MobileNetV2 on the Caltech256 Dataset .....	80
<b>Figure 4.7:</b> Correlation Recall Plot on Caltech256 Dataset Using New VGG16 .....	81
<b>Figure 4.8:</b> Homogeneity Recall Plot on New MobileNetV2 on Stanford Dogs’ Dataset .....	82
<b>Figure 4.9:</b> New VGG16 Accuracy Plot on CIFAR-100 .....	86
<b>Figure 4.10:</b> Loss on New ResNet50 on CIFAR-10 Dataset .....	89
<b>Figure 4.11:</b> Loss of the New ResNet50 on the Fashion-MNIST Dataset.....	90
<b>Figure 4.12:</b> Loss of the New ResNet50 on the MNIST Dataset.....	91
<b>Figure 4.13:</b> CIFAR-10 on the New MobileNetV2 .....	93
<b>Figure 4.14:</b> New VGG16’s Layers 1 and 9 Weights Visualization.....	94
<b>Figure 4.15:</b> New VGG16 Accuracy Comparison Plots between Conflated Samples Methods and Conflated Samples with Dynamically Selected Layers Methods .....	96
<b>Figure 4.16:</b> New MobileNetV2 Confusion Matrix on ISIC 2016 .....	99
<b>Figure 4.17:</b> New VGG16 Accuracy Comparison between Methods with and without Transfer Learning (TL) on ISIC 2016 .....	100
<b>Figure 4.18:</b> New VGG16 Accuracy Comparison between Proposed Methods and Commonly Used Methods on ISIC 2016 .....	101
<b>Figure 4.19:</b> MNIST Dataset Accuracy Performance in Various Methods .....	108



<b>Figure 4.20:</b> CIFAR10 Dataset Accuracy Performance in Various Methods.....	109
<b>Figure 4.21:</b> CIFAR100 Dataset Accuracy Performance in Various Methods.....	110
<b>Figure 4.22:</b> Caltech256 Dataset Accuracy Performance in Various Methods .....	110
<b>Figure 4.23:</b> MITIndoor 67 and Stanford Dogs 120 dataset Accuracy Performance in Various Methods .....	111

## LIST OF APPENDICES

<b>Appendix I:</b> Author's Journal Publications during PhD Study .....	150
<b>Appendix II:</b> Author's Conference Publications during PhD Study .....	151

## ACRONYMS AND ABBREVIATIONS

<b>MSCOCO</b>	Microsoft Common Objects in Context
<b>MRI</b>	Magnetic Resonance Imaging
<b>PCA</b>	Principal Component Analysis
<b>ELMO</b>	Embeddings from Language Model
<b>HMCA</b>	Heterogeneous Max-margin Classifier Adaptation
<b>WbFTL</b>	Weight-based Feature-transfer Learning
<b>ANOVA</b>	Analysis of Variance
<b>SDM</b>	Sequential Decision-Making
<b>MDP</b>	Markov Decision Problem
<b>GAN</b>	Generative Adversarial Network
<b>DKL</b>	Kullback-Leibler Divergence
<b>DCT</b>	Discrete Cosine Transform
<b>LBP</b>	Linear-Binary Pattern
<b>GLCM</b>	Grey-level Co-occurrence Matrix
<b>GMM</b>	Gaussian Mixture Model
<b>EEG</b>	Electroencephalography
<b>HAR</b>	Human Activity Recognition
<b>POMDP</b>	Partially Observable Markov Decision Process
<b>PAC</b>	Probably Approximately Correct
<b>SGLD</b>	Spatial Grey Level Dependence
<b>BLW</b>	Below Average
<b>ABV</b>	Above Average

<b>NER</b>	Named Entity Recognition
<b>UTL</b>	Unsupervised Transfer Learning
<b>SLU</b>	Spoken Language Understanding
<b>HetOTLMS</b>	Heterogeneous Online Transfer Learning Algorithm
<b>OSCC</b>	Oral Squamous Cell Carcinoma
<b>ELM</b>	Extreme Learning Machine
<b>SAR</b>	Synthetic Aperture Radar
<b>DCNN</b>	Deep Convolutional Neural Network
<b>kMMD</b>	Kernel Maximum Mean Discrepancies
<b>PDF</b>	Probability Density Function
<b>BERT</b>	Bidirectional Encoder Representations from Transformer
<b>ViT</b>	Vision Transformer
<b>FTAP</b>	Feature Transferring Autonomous Machine Learning Pipeline
<b>TPOT</b>	Tree-based Pipeline Optimisation Tool

## ABSTRACT

Transfer learning is an in-depth learning approach that uses an existing model to classify a new task. The approach involves using some target data on a fine-tuned model. Fine-tuning involves freezing and tuning layers within the network to reduce the model's poor adaptation to the target task. Poor domain adaptation is a significant issue in transfer learning due to differences in the distributions between the source and the target domains. There have been various approaches to tackling the problem using fine-tuning approaches, the most common being instance and feature-based approaches. Despite many documented fine-tuning approaches in transfer learning, achieving higher accuracy performance due to poor domain adaptation is still challenging. This research looks at a feature-instance-based approach where a subset of data objects with similar low-level characteristics is selected as the target dataset. The signed weight instances are used as a routing decision network to determine the filtering layers that must be frozen during training. The approach leads to developing a model that confers textural features in selecting target samples and the convolutional layers with the most positive number of feature map elements in the transfer learning process. The study uses image datasets from nine datasets: ChestX-ray8, CIFAR-10, MNIST, CIFAR-100, Fashion-MNIST, Stanford Dogs 120, Caltech 256, ISIC 2016 and MIT Indoor Scenes on five pre-trained models: VGG16, DenseNet169, MobileNetV2, InceptionV3 and ResNet50. The experimental approach provides better convolutional networks in the transfer learning process, with improvements of between 3.12% and 7.69% in selecting quality data points and 0.24% to 13.04% for selecting suitable layers. These improvements perform well compared to the standard accuracies and some previous studies.

**Keywords:** Transfer Learning, Domain Adaptation, Distance Measures, Features Conflation, and Layer Selection.

## CHAPTER ONE

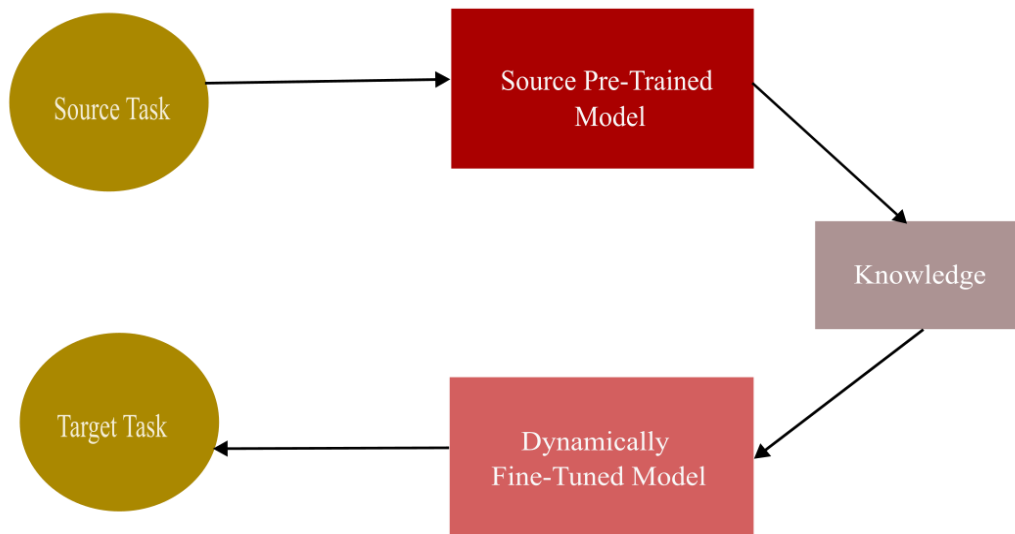
### INTRODUCTION

#### 1.1 Background Information

Transfer learning (TL) is a deep learning technique that involves reusing learned knowledge in a pre-trained model in a new related task (Hung & Chang, 2021). The reused model is the source model, while the applied model is the target model. By default, both domains, target and source, are somehow similar. A domain  $D$  consists of two components:  $Y$  and  $P(X)$ , where  $Y$  refers to the feature space,  $X$  refers to a random variable that could take any value of  $x$  and  $P(X)$  refers to the marginal probability distribution where  $x$  belongs to the set  $x_1, \dots, x_n$ . The element  $x_i$  is the feature instance at position  $i$ .

Task  $T$  consists of two components:  $Z$  and  $f(\cdot)$ , where  $Z$  is the label space, and  $f(\cdot)$  is the conditional probability  $P(Z|X)$ . The  $X$  is still a random variable, taking the value of  $x$ . By formal notation, transfer learning aims to improve the predictive function  $f(\cdot)$  in the target domain ( $DT$ ) by using the learned knowledge in the source domain ( $DS$ ) where the  $\neq DT$ , or source task ( $TS$ )  $\neq$  target task ( $TT$ ), and therefore, the marginal probability distribution between the domain data is different.

The technology behind transfer learning streams from psychology (psychologist CH Judd) suggests human beings can transfer previously known skills to new tasks that need similar or closely related acquired skills. It is a domain generalisation technique alongside multi-task learning, domain adaptation, zero-shot learning, life-long learning, meta-learning, and multiple-domain learning (Jiang et al., 2024). A practical, real-life example is riding a motorcycle, having acquired skills in riding a bicycle, or playing baseball, having played softball: it is much easier to master a new skill based on the previous experience, although a negative transfer is also possible. However, in situations with few settings (combination of task, domain, and language) and similarities in the domains, the transfer may be unsuccessful; for example, learning how to drive a car may not extend to playing the piano. As shown in Figure 1.1 below, knowledge acquired through a learning task in the pre-trained domain is usable by the application domain's learning task.



**Figure 1.1: Transfer Learning Process**

Transfer learning has been used in many fields, especially in image recognition tasks, since it allows faster training (no need to retrain the model fully), requires fewer data points in the target domain, and low computational power for use in the data labelling (fraction of the source domain). For example, the enabling factor in convolutional neural networks (CNNs) is that the low-level features, e.g., colour blobs and Gabor filters, apply to many datasets and tasks; hence, they can be reused or extended to new tasks. The problems transfer learning addresses have been noted in large datasets such as the MSCOCO (Liu et al., 2024) and ImageNet (Nayman et al., 2024), which are often unfeasible. Furthermore, using traditional classifiers with insufficient data leads to model overfitting due to their large number of hyperparameters instead of transfer learning (Mecheter et al., 2024).

With advancements in deep learning, CNNs have been at the forefront of model extensions, enabling working with smaller datasets (Abdulazeez et al., 2024). Some of the notable pre-trained models include Inception utilising the inception blocks (Srinivas et al., 2024), ResNets utilising the ResNet blocks (Xu et al., 2023), MobileNets and VGG models (Kandhro et al., 2024).

Transfer learning is broadly classified based on features and labels based on shareable features and labels in both source and target domains. The methods can also be classified based on instances, features, parameters and even the relationship between

the domains, as Zhuang et al. (2021) noted. This study will address two of the four approaches: instance-based transfer learning by looking at the shareable weights between two tasks – thereby picking the layers that would fit well in a given task and feature-based transfer learning that addresses the shareable features between the two domains. The methods in the two broad approaches suffer from poor adaptability in the target domain due to unique features that may not be in the source domain, high dimensionality between the domain spaces and the assumption of feature spaces similarity (Yang et al., 2020) as a result of task mismatch which leads to poor transfer of relevant knowledge in the downstream tasks and may also erase some of the learned knowledge during pre-training (Han et al., 2021). The introduced methods will be introducing improvements to the existing methods:

- a) Feature-based transfer learning provides a confident approach to selecting closely related features in the target domain using feature conflation. The process uses Kullback-Leibler divergence (DKL) to compare the conflated feature distributions.
- b) Instance-based transfer learning allows dynamic selection of suitable layers for fine-tuning. The process compares the layers using Kullback-Leibler divergence on signed weights.

## **1.2 Transfer Learning Process**

The transfer learning process occurs in a feed-forward network where a subset of layers is selected and fine-tuned. A threshold moves away from the output and towards the input layers as more data becomes available in the target domain. The process also involves the fine-tuning or the adjustment of hyperparameters until the best fit characterizes the model. Transfer learning tasks in CNNs initially utilized pre-trained CNN ImageNet models. Computer vision models don't necessarily train from scratch (Hong et al., 2024).

The most adapted practice by practitioners is to download an open-source model and either use it to initialize a similar architecture before learning on limited labelled data, a process involving fine-tuning a subset of the layers or use it in feature extraction. The process reuses the pre-trained model weights used in the source domain (specific



task) to address another closely related task (Vrbančič & Podgorelec, 2020). These weights are used partly or entirely in model initialization (Bhatia et al., 2020). The transfer learning process works on defining data point features, labels, and the sampling distribution of data points, as well as the dependency between labels and features (Zhang et al., 2020).

### **1.3 Image Data Features**

In image data processing, features are extracted and analyzed for general purposes for specific applications (Archana & Jeevaraj, 2024). The image features belong to three levels: low-level (explores the pixels), mid-level (explores the image descriptors) and high-level features that look at the image interpretation. This research work investigates the domain datasets' low-level textural features. The dataset's detection of low-level feature properties makes other tasks, such as image analysis and classification, possible. Low-level features can be explored from colour, texture and shape properties (Dewan & Thepade, 2020). The pixel analysis looks into the geometric design of grey levels (Sferrazza, 2023).

Texture refers to the pixel's spatial brightness variations (Kim & Lee, 2024). Texture affects human visual perception (Okada & Motoyoshi, 2021) and classification accuracy (Zhang et al., 2024). It also gives the pixel's illumination dispersion in the image and is essential in measuring the pixel's neighborhood spatial grey tones arrangement (Haq et al., 2022; Wei et al., 2024).

Textural analysis has been used in many applications, including image classification and content-based image retrieval (CBIR) systems (Bu et al., 2023). The analysis involves extracting the features using methods in seven categories: model-based, structural, transform-based, learning-based and entropy-based (Cote et al., 2023). The textural analysis methods are further categorized into structural, statistical, transform and model-based. In textural features analysis, the extracted features are quantified through descriptors depending on the statistics: first-order (entropy, standard deviation), second-order statistics (correlation, sum entropy) and high-order (mean, variance). These values are then subjected to statistic-relevant indices such as information gain for selection.

## 1.4 Problem Statement

Transfer learning suffers from poor domain adaptation due to the differences in the distributions between the source and target models: data points and fine-tunable layer selection. Poor domain adaptation results from the distribution divergence between the source and target domains (Li et al., (2022); H. Xu et al. (2021); Z. Xu et al. (2021)) and failure to address the divergence in local information (Peng et al., 2022). Various methods have been introduced to reduce this divergence, including discriminative adversarial domain adaptation (Tang, 2020), structural risk minimisation (Cherkassky & Lee, 2024) and manifold dynamic distribution adaptation (Wang et al., 2020), with most methods addressing class imbalances, marginal and or conditional distributions between the domains. However, the methods assume the data is balanced in both domains and do not address dataset imbalances in real-life scenarios. Moreover, it becomes more unrealistic in unsupervised cases. It is, therefore, important to introduce or embed sample target samples in the source classifier to create more dense neighbourhoods in the feature space (Saito et al., 2021).

In the standard fine-tuning process, the last few layers of the network undergo fine-tuning while the parameters in the initial layers of the source model become frozen. However, this does not mean that features in the latent layers would adapt to the target task (Vandikas et al., 2024). Some layers influence the number of data samples and training time in the target domain, and it is vital to use a dynamic, fine-tunable layer selection method.

Dynamic layer selection has increasingly gained attention among researchers to traditional manual selection (Vrbančić & Podgorelec, 2020; Ismail et al., 2024) but has not focused on using signed weights. Many previously used manual methods assume that the models' initial or last layers are the main ones that can participate in fine-tuning. The manual layer selection methods involve many trials and errors, as noted by Nagae et al. (2020); they are time-consuming since there are no standard procedures for fine-tunable layer selection (Guo et al., 2020). Therefore, the dynamic fine-tunable layer selection methodology becomes reliable in addressing this optimisation problem.

This research utilises a subset of data points similar in low-level characteristics

(textural features) in both the source and the target domains in addressing conditional distribution and divergence in local information and the signed weights in the latent layers to fine-tune the pre-trained models. It aims to develop new target models with reduced domain distribution divergence and dynamic layer selection for faster convergence during the transfer learning process.

### **1.5 Justification**

Fine-tuning involves the selection of a pre-trained model's layers that are used to transfer the acquired knowledge to a new task. The various fine-tuning methods, including standard fine-tuning, last-k layers, feature extraction, and target layers, are in different positions of a pre-trained model. The most suitable layers must be used for a good positive transfer, and dynamic layer selection has recently become an active research area. Many methods have been proposed, including differential algorithms (Vrbančič & Podgorelec, 2020), AdaFilter (Guo et al., 2020), and even genetic algorithms (Nagae et al., 2020). However, the methods do not address the weights – a key component alongside the inputs in a network's learning process (DeepAI, 2020). In fine-tuning, dynamic layer selection is needed to replace the manual trial and error experimentation layer selection for optimal positive transfer learning.

It is also important to use quality datasets if a model is to perform well. In the transfer learning process, as much as the domains are considered similar, there is a divergence in their marginal or conditional distributions, often leading to poor domain adaptation. Some methods have been introduced to address these divergences, including using sample target data points in the source model (Luo et al., 2017). For convolutional neural networks, which are used in tasks such as image classification, low-level features such as texture are essential since they affect visual perception (Okada & Motoyoshi, 2021), and their descriptors can be used as a divergence measure in both domains. Since reducing domain divergence and selecting suitable layers are important in the transfer learning process, this research focuses on using the least textural features, divergent data points and the least divergent layers with positively signed weights. These allow better domain adaptation and faster convergence, improving positive transfer.

## **1.6 Research Objectives**

### **1.6.1 General Objective**

The general objective of this research was to develop a new model that uses conflated textural features from the target dataset that closely matches the source dataset and dynamic fine-tunable layer selection to enhance the transfer learning model's performance.

### **1.6.2 Specific Objectives**

The specific objectives of this research were to:

- 1) Analyze the existing feature and instance-based transfer learning approaches using various pre-trained models, models' parameters, and datasets.
- 2) Determine a suitable distance metric for comparison of suitable textural features and fine-tuning layers.
- 3) Develop a fine-tuneable transfer learning model that utilises conflated textural features, data points, and source model weights: a feature-instance-based transfer learning model.
- 4) Assess the effectiveness of using the developed model with various datasets and pre-trained CNN models, comparing it with several standard transfer learning baselines.

## **1.7 Research Questions**

- 1) What are the various existing feature and instance-based transfer learning approaches, and what pre-trained models, models' parameters and datasets did they use?
- 2) When selecting a suitable distance metric to compare textural features and fine-tunable layers, what factors do we consider?
- 3) Which parameters should we consider when developing the fine-tuned transfer learning model?
- 4) How does the developed fine-tuned transfer learning model perform on various image datasets, baselines and pre-trained CNN models?

## **1.8 Proposed Solutions**

This research study has developed a dynamically fine-tuned model to address the challenge of poor domain adaptation, which often leads to poor transfer learning performance. This model, which facilitates the comparison of the existing textural features analysis and layer selection strategies, has been validated using existing transfer learning baselines. The promising results show that the newly introduced methods lead to improved domain data adaptation and layer selection in the transfer learning process, offering an alternative process for conducting transfer learning.

## **1.9 Research Contributions**

In addressing the selection of closely related transferable features in the source domain to the target domain and the dynamic choice of fine-tunable layers, the study introduces the following knowledge contributions: Firstly, the conflation of image feature probability distributions contributes to an extension of methods used in the overall image feature description using the conflation process (primarily used in geographic information systems (GIS)). Conflation allows users to derive positive transfer by selecting quality data points, which leads to a better adaptation of the pre-trained learning models, providing a simple technique for advancing feature selection, a key component in transfer learning.

Secondly, The Kullback-Leibler divergence method for comparing layers based on signed weights introduces a novel approach to layer selection, deviating from the conventional manual selection process. This innovative method empowers users of pre-trained models by providing a quantified assessment of each layer's contribution to the model's convergence, enabling more confident and informed layer selections. This approach represents a significant advancement in research, underlining the pivotal role of this concept.

Thirdly, a new method in pipelining the transfer learning process has been introduced by developing the two-step approach (the conflation of features and dynamic layer selection). While used in CNNs, this approach applies to other deep neural network-developed pre-trained models, such as recurrent neural networks (RNNs). The simplicity of this approach allows the application of this method to new transfer

learning application cases, reassuring users that it can be easily implemented since their models use features in their input data and their layers contain weights.

Finally, Introducing new methods that use the various pre-trained models and datasets plays a crucial role in building user confidence in using transfer learning. This confidence significantly shifts from the previous trial-and-error process in selecting suitable layers and models for a target dataset. The new approach encourages users to apply pre-trained models, knowing it will achieve good results with minimal negative transfer due to selecting quality data points and suitable fine-tunable layers.

### **1.10 Research Scope**

This research focused on transfer learning, addressing the adaptation of domain textural data features and fine-tuning involving dynamic layer selection. The study proposed developing a model for domain data adaptation and dynamic layer selection methods to improve accuracy performance. The adaptation process involved transfer learning methods that involved fine-tuning parameters, divergence measures and data adaptation procedures. The study used CNN pre-trained models and image data with data acquired from public repositories and pre-trained models from the TensorFlow Hub. This acquisition process reduced proprietary and time challenges.

## CHAPTER TWO

### LITERATURE REVIEW

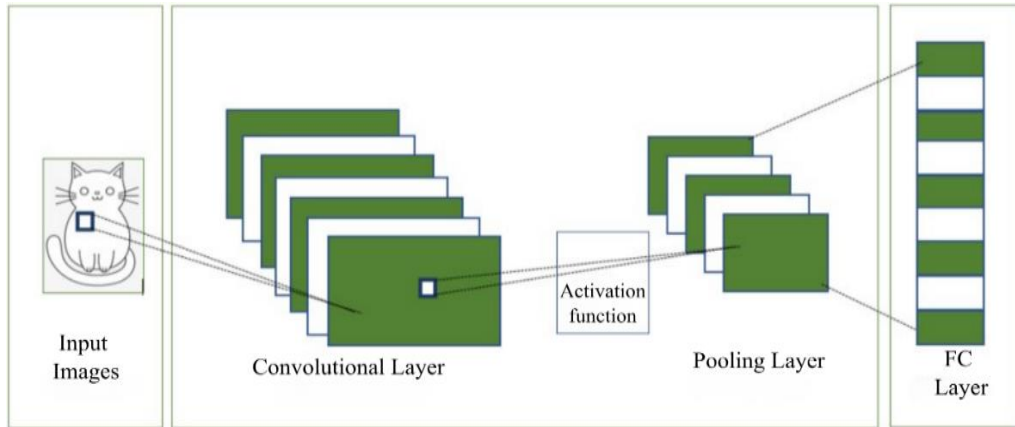
#### 2.1 Introduction

Artificial neural networks can model non-linear relationships, which can model complex relationships (Zhang et al., 2024). For a neural network to classify data correctly, the inputs go through the layers until the network gives an output. However, the output may differ from the expected output due to errors, and the output has to be taken through a backward propagation process, updating each weight in the layers. The process then reiterates until the output is satisfying.

This chapter addresses the literature on transfer learning and the distance metrics considered in the research.

#### 2.2 Convolutional Process

Convolutional neural networks have been on the frontline in advancing deep learning; therefore, more transfer learning work has been done in this area. The CNN pre-trained model filters will be used to evaluate the weights in this research. Figure 2.1 illustrates a CNN. The convolution layer receives the input image and convolves on it, extracting features using the kernel matrix (filter). Low-level features are extracted from the receptive field (area or matrix under the filter's focus). This convolutional process produces filters, mostly stacked depending on the channels in the image forwarded to the pooling layer after applying the activation function. The process ends by classifying the samples in the fully connected (FC) layer, which will be followed to identify the knowledge learned in the pre-trained models.



**Figure 2.1: Convolutional Neural Network**

Given a Kernel matrix (weight filter)  $H$  with  $2 \times 2$  dimensions and an input image  $G$  of  $3 \times 3$ ,  $H$  extracts the low-level features on  $G$  and the extraction process, and the weight sharing introduced by  $H$  reduces the parameters in the convolutional process.  $G$  and  $H$  are illustrated in Figure 2.2. The output  $O$  is an array (map) of filters with lower dimensions than the original input image pixels. The output  $O$  can be expressed as follows:

$$O(1,1) = H(1,1)G(1,1) + H(1,2)G(1,2) + H(2,1)G(2,1) + H(2,2)G(2,2) \quad (2.1)$$

$G(1,1)$	$G(1,2)$	$G(1,3)$
$G(2,1)$	$G(2,2)$	$G(2,3)$
$G(3,1)$	$G(3,2)$	$G(3,3)$

a) Input Image

$H(1,1)$	$H(1,2)$
$H(2,1)$	$H(2,2)$

b) Kernel matrix (filter)

**Figure 2.2: Convolutional Layer Filtering Components**

$O(1,1)$  then becomes the weighted average of the filters in  $G$  (receptive field); the convolutional process is a multiplicative operation of the weight filter and the input.



## **2.3 Transfer Learning Methods**

The transfer learning process can be broadly classified based on the label and feature space settings. The methods address the label or feature knowledge transfer from the source to the target task. These methods have been described in detail in sections 2.3.1 and 2.3.2.

### **2.3.1 Label-based Categorization**

This category uses label information and is classified into transductive, inductive, and unsupervised. Transductive transfer learning uses label information from the source domain, inductive transfer learning uses label information from the target domain, and unsupervised transfer learning uses unknown label information for source and target domains.

#### **2.3.1.1 Transductive**

This category uses label information from the source domain. The source and the target tasks are similar. The target data does not have labels and has different domains, as illustrated in Figure 2.3. Transductive learning learns a specific function, while inductive learning addresses a general function (Lavine et al., 2024). An example of a Transductive learning algorithm is the k-nearest neighbour algorithm since it uses the training data directly each time when conducting prediction.

Transductive transfer learning has been employed in various domain adaptations, such as brain magnetic resonance imaging (MRI) segmentation, to mitigate the impact of domain shift (Kushibar et al., 2021). Rezaei et al. (2018) developed an unsupervised transductive transfer learning method that finds the shared and specific features across the source and target domains. The technique maps both domains into sub-spaces with minimal conditional and marginal distribution divergence. Chen et al. (2023) used the Transductive method (incremental learning algorithm) using a Support Vector Machine (SVM) and gave the fastest method in training transductive SVM.

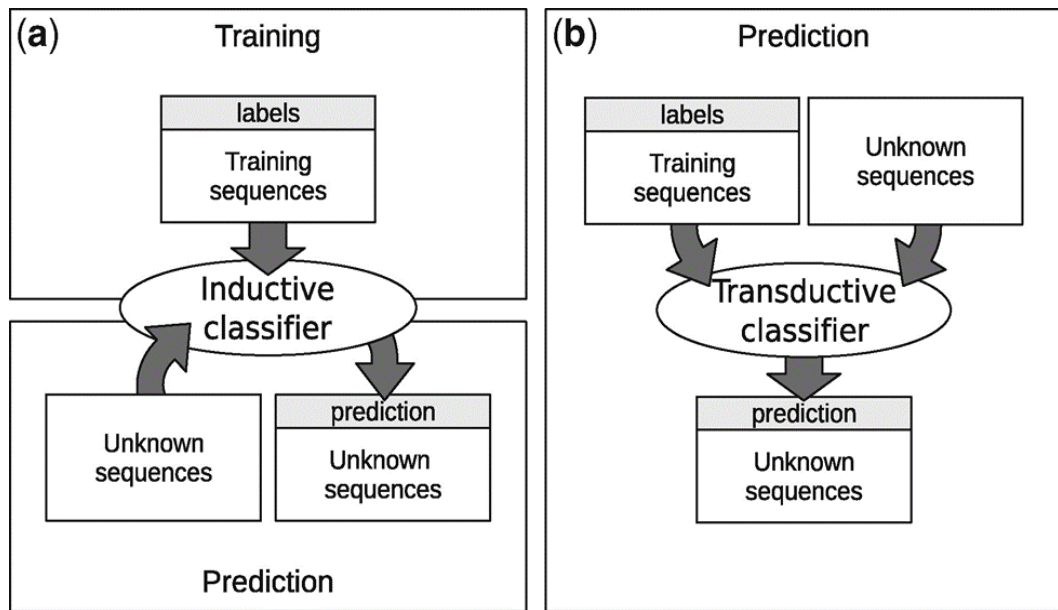
### **2.3.1.2 Inductive**

The label information comes from the target domain for inducing an objective model for the training, as illustrated in Figure 2.3. The source task is also different from the target task. However, the target data could have been labelled data despite the relationship status in the domains. An example of inductive transfer learning is a team of robots in a competitive game where they need to keep the ball away from the opponents, with the need to pass the ball to their teammates and maintain a safe distance from their opponents. The main goal of inductive learning is to create a model using training examples. These examples can be used for classification or modelling the probability distributions between interconnected variables. When creating model algorithms, this type of learning relies on assumptions about the actual distribution of the training data. These assumptions, also known as bias, are based on a space encompassing possible models. The inductive bias for the target task is derived from knowledge obtained from the source task. Some methods narrow this space, eliminating specific search steps, while others expand.

The learning method has been used to tackle the issue of drug response prediction by utilising pharmacogenomics data (from a single or multiple sources). Sharifi-Noghabi et al. (2020) employed inductive transfer learning in multitask learning, gene expression analysis and adversarial domain adaptation. In a further application of the method, Myung et al. (2022) used the PAC Net (Prune, Allocate and Calibrate) to prune model weights and enhance transfer learning using inductive transfer learning, while Meli et al. (2021) researched encoding and reasoning task knowledge in surgery using inductive transfer learning - a challenging yet crucial aspect by using logic programming and a set of answers for a limited number of basic surgical tasks.

Inductive transfer learning can optimize learning performance in computing while minimizing the need for user training samples. This approach tailors user training examples by combining them with training examples. It can be considered an all-round development learning system that involves using the samples as training data and evaluating the hypothesis. At the same time, the transductive method is observed as an examination-oriented education system that uses both the learning data and the unlabeled samples in the learning algorithm. These differences are clearly illustrated

in Figure 2.3, adapted from Yones et al. (2017).



**Figure 2.3: Transductive vs Inductive Learning**

### 2.3.1.3 Unsupervised transfer learning

In this label transfer learning, the label information is unknown for both domains. However, the target task is somehow related to the source task. Michau and Fink (2021) presented unsupervised transfer learning (UTL) for anomaly detection. They suggested incorporating UTL into learning to preserve the natural variations within datasets by aligning the distribution and loss functions of different units using a class-based approach. This method distinguishes itself from UTL methods that typically identify dataset structures through clustering or dimensionality reduction techniques.

Yin et al. (2024) used Unsupervised Transfer Learning (UTL) in question answering and has also been applied alongside k-means clustering to classify material image data (Cohn, 2021) that was tested on a VGG16-trained model and reducing feature dimensionality through principal component analysis (PCA). The method gained over 99% accuracy.

Further works on UTL has been done in spoken language understanding (SLU) tasks using the Embeddings from the Language Model (ELMO) to obtain word representations of unlabeled data using a simplified ELMO Light method (Siddhant et

al., 2018) while Jahanian et al. (2024) conducted a study exploring the use of unsupervised transfer learning in biomedicine for image clustering in chest radiography. The method has also been used to enhance person re-identification tasks, as Chen et al. (2023) noted, addressing the challenges of poor domain adaptation.

### **2.3.2 Feature Space Categorization**

This category looks at the feature space knowledge transfer between the source and the target domains in transfer learning. The various feature-space transfer learning methods can be further classified into either homogeneous or heterogeneous;

#### **2.3.2.1 Homogeneous**

This feature space classification assumes the source and target datasets are in the same domain or differ in marginal distributions. The method aims to bridge the gap in data distributions between domains in cross-domain transfer (Guo et al., 2024).

In their study, Kirielle et al. (2022) suggested incorporating structured attributes from personal data into entity resolution tasks. The researchers used a label generator for target instances and a classifier that assigned pseudo labels to improve precision by 13% and recall by 50% across seven datasets. In another study, Du et al. (2019) focused on online transfer learning methods that effectively handle situations where target data arrives online using a Hedge strategy that leveraged knowledge from both domains and aimed at reducing distribution and conditional distribution disparities through feature representations. In a further study, an asymmetric homogeneous feature-based transfer learning technique was used in convolutional neural networks to address the need for the large amounts of data required in training neural networks for functional Near Infrared Spectroscopy (fNIRS) – for recognizing brain activities and their translation (Chen et al., 2020). This approach achieved a higher accuracy of over 25% compared to the traditional CNNs.

#### **2.3.2.2 Heterogeneous**

This feature space classification assumes that the source and target domains are different. A good task example is the text-image classification. They can be symmetric

or asymmetric. For symmetric, feature transformations are learnt from the source and target spaces onto a joint subspace for adaptation in the target domain. The common representation of the two in the common subspace can then be utilized by a traditional algorithm such as the SVM. For the asymmetric, the source space features are aligned with the target space features, bridging the space gap and reducing the transfer problem, but it works well when both spaces have the same class label space.

Heterogeneous feature augmentation refers to using features and dimensions and applying transformations to both the source and limited target features (Bao et al., 2023). In a study by Mozafari and Jamzad (2016), the Heterogeneous Max margin Classifier Adaptation (HMCA) technique was introduced. The HMCA applies domain adaptation methods to the target domain while utilizing the parameters of a classifier trained on the source samples, enabling the learning of a margin classifier for the target domain by adapting it from the source classifier.

Wu et al. (2019) introduced the Heterogeneous Online Transfer Learning (HetOTLMS) algorithm for online transfer learning with sources tested on a fashion dataset. This algorithm divides instances into two groups: one that shares feature space with a source domain and another for training new learners in the target space. Weights are adjusted in both domains to adapt to the target domain effectively. More HTL studies include Chui et al. (2023), which introduced multiple incremental transfer learning. The method gave an accuracy improvement of 4.35% compared to the previous studies; Moon and Carbonell (2017) introduced the Attention heterogeneous transfer learning that learns from both domains by selecting an optimized subset of datasets. The objective is to minimize the differences between datasets and determine which elements should be prioritized for transfer.

In the transfer learning process, four main approaches are introduced (Zhuang et al., 2021):

**a) Instance-based transfer learning**

This approach supposes some parts of the source model's data are reusable in an instance weighting strategy. This process is done due to reduced marginal distribution. The samples that are likely to lower the adaptation are removed. The model is then fine-

tuned, or a new model uses the remaining data points (Zhang et al., 2023). This process can be symmetric or asymmetric. This transfer-learning approach has been applied in various studies, including to improve Soil Organic Carbon (SOC) value estimations – from soil and geochemical factors (Bursac et al., 2022). The SOC values in areas used for croplands are derived from source domains, including cropland and grassland areas.

Lam et al. (2022) developed a speech synthesis system using 45 target datasets that differed from the source voice using the text-to-speech (DC TTS) model. In automating vision-based quality identification of Longjing tea, Zhang et al. (2023) used a MobileNetV2 to extract features from both the source and target datasets using the TrAdaBoost algorithm.

#### **b) Feature-based**

This approach involves transferring knowledge from the feature level to create new ones. A "good" feature represents the target domain. Unseen network attacks can be detected using feature-based transfer learning since the machine learning approaches used in these detections rely on the features identified (Jung et al., 2021) with network intrusion detection. The feature-based approach has further been used by Sevani et al. (2023) through the weight-based feature transfer learning (WbFTL) technique to address the class imbalance between transfer learning domains. The approach involves creating a representation of features using class feature distance and selecting relevant features through Analysis of Variance (ANOVA) and Support Vector Machines (SVM) as the features classifier. In a study by Karim and Van Zyl (2021), a method for establishing a feature space for transfer learning was introduced, aiming at reducing the disparity between the source and target domains while preserving data features.

#### **c) Parameter-based**

This transfer learning approach involves transferring knowledge from shareable parameters in both target and source tasks. The learning approach transfers knowledge across the tasks via the shareable parameters. In a study by Pinto et al. (2022), parameter-based transfer learning was used to predict a building's behaviour, and they proposed a data-driven model that captures physical processes in high dimensions. The

researchers used over 250 data points, showing similar climate patterns across different schedules even with limited data availability and efficiency.

Zheng et al. (2023) proposed deep parameter-based transfer learning to tackle disruptions in tokamaks. They utilized a J-TEXT tokamak model and 20 discharges to predict disruptions in tokamaks by leveraging knowledge gained from existing ones. This approach enabled the prevention of plasma confinement termination and device damage. The approach has further been used to enhance the parameter selection in extreme learning machines (ELM), as noted by Wang et al. (2022), where they introduced a parameter projection framework.

#### **d) Relational-based**

This transfer learning approach focuses on the problems in the relational domain, defining the relationship between the two domains. In a study by Li et al. (2020), a proposed relational feature transfer algorithm directs the transfer procedure that uses relations and causality knowledge to tackle the challenge of practical transfer learning in relational-based transfer learning. Strömfelt et al. (2020) introduced a further study addressing domain relationships, which addresses relations systems to maintain coherence across properties and domains when addressing relational transfer learning.

Other categorizations of transfer learning methods are quickly gaining attention, including instance, network, adversarial and mapping-based. Furthermore, these categorizations can be viewed by combining the data, model approaches, and properties. With the new methods, four have been shown to deliver great potential in transfer learning (Liang et al., 2019):

#### **(a) Transitive transfer learning**

This type of transfer learning uses domain datasets with an intermediary dataset when there is a weak similarity between domains. An et al. (2021) introduced a transitive transfer learning model that assists in detecting SAR (aperture radar) targets when there is a scarcity of labelled data in SAR images. This scarcity poses challenges for improving deep convolutional neural network (DCNN) detection methods. The model allows for generalizing features and specific variations in the target domain. The results

show that transitive transfer learning is an approach that helps achieve state-of-the-art results in SAR.

In a further study by Sineglazov et al. (2022), transitive transfer learning was proposed for tuberculosis (TB) diagnosis using ImageNet and LIDC IDRI as the source domain and tuberculoma TB as the target dataset.

### **(b) Lifelong transfer learning**

This type of transfer learning involves new learning methods that use intelligent agents, thereby learning new knowledge in a new environment. In the field of agents, lifelong transfer has been employed to enable teams of agents to adapt to new tasks by utilising autonomous instructions. This adaptation involves controlling and coordinating policies. It has been successfully applied in both simulated (using sequential decision-making (SDM)) and physical robot settings (Zhu et al., 2024). Furthermore, lifelong transfer has been used in Autonomous systems operating in environments that need to learn and enhance their existing knowledge. For example, Irfan et al. (2021) developed an autoencoder that leveraged convolution filters for extracting image features.

### **(c) Transfer reinforcement learning**

This type of transfer learning involves using reinforcement agents capable of transferring knowledge and behaviours across many tasks and domains. The approach has successfully addressed Markov decision problems (MDPs), enhancing the learning process for reinforcement learning agents by leveraging the knowledge acquired by MDPs. One strategy of using transfer reinforcement learning involves using auto-pruned decision trees to extract knowledge - the trees are retrained using data samples generated by the policies of trained models. The knowledge-based transfer learning algorithm works with discrete and continuous action spaces (Lan et al., 2022).

This approach has been used by Li et al. (2022) through kernel discrepancies (kMMDs) to construct multiple deep transfer learning networks that facilitate single-source target transfers in addressing fault diagnosis in scenarios with multiple source domains. The differences in these kernel functions helped identify features between the domain spaces, aiding the DLTNs in their identification tasks. It also has promising capabilities



in dealing with differential power grid rescheduling, especially considering security constraints. The process occurs in two situations; one deals with security-constrained scheduling for power grid configurations, and the other involves domain transfer learning, which is compared to the pre-trained model (Wang & Tang, 2022).

#### **(d) Adversarial transfer learning**

This transfer learning type involves generative adversarial networks (GANs) to exploit its networks. Examples include image-to-image translation (Wang et al., 2021) and focusing on the input data by Zhao et al. (2022) to develop the MFAT NER (Multi-head attention Feature fusion Adversarial Transfer NER) model. The model addresses the problem of insufficient labelled data in Named Entity Recognition (NER) tasks. The model uses the head attention to capture relevant information from various sentence subspaces for semantic interaction. Furthermore, the model effectively represented both characters and words.

Another application of adversarial transfer learning is seen in COVID-19 diagnosis, as Alhares et al. (2023) noted. In their research, a model named AMTLDC successfully leveraged similar features from different sources despite the inherent and natural differences in datasets when predicting COVID-19 from images using a CNN event with limited data available.

According to Zhuang et al. (2021), transfer learning can use semi-supervised learning (instances), multiview learning (features) and multitask learning (tasks). Further categorization is based on data adaptation: 1) Feature-based focusing on methods that reshape the data feature spaces. 2) sample-based, focusing on correcting the bias in the data sample procedures, and 3) inference-based, focusing on incorporating data adaptation into the parameter estimation procedures (Kouw & Loog, 2021). However, despite the many methods that have been developed and categorized in transfer learning, classifier adaptation (Shi et al., 2023), instance re-weighting (Molina-Cabanillas et al., 2022), deep adaptation (Xu et al., 2020; Xu et al., 2023), universal representation (Zhang et al., 2020) and adversarial adaptation methods are the recent transfer learning methods frontiers.

In the transfer learning process, the source model (pre-trained) undergoes fine-tuning.

Fine-tuning is adjusting the source model's parameters to allow knowledge transfer to the target model (Han et al., 2024). The method is a primary deep-learning transfer technique that can utilize various strategies: optimization of all the pre-trained model parameters, fine-tuning the last few layers while freezing some retrained models (usually initial layers), and optimizing the last layers. Two main approaches have recently been adopted for standard fine-tuning (Li et al., 2022): good data points or regularization with source-task-related parameters.

## **2.4 Feature-Based Transfer learning**

Feature-based transfer learning optimizes feature representation in both domains (Iman et al., 2023). The initial step involves extracting features from the data, after which the new feature representation occurs from both domains before being fed into a classifier. This approach to transfer learning ensures that the distance between the source and the target feature mapping reduces the maximum transferability of knowledge (Priyatikanto et al., 2023). For example, in convolutional neural networks, low-level features like edges and curves are extracted to provide better abstraction and better represent new data sets (Zhao et al., 2024). The transfer learning in this method is conditional on two events: selecting data with similar features in the source and the target domains and using data from the source domain, which would improve learning when the acquired knowledge is used in the target task. In this study, the samples with similar textural features in both domains were selected to reduce the knowledge divergence, which improved new domain adaptability. Furthermore, this learning method reduces the probability distribution differences between the two domains.

### **2.4.1 Label Efficient Learning of Transferable Representations across Domains and Tasks**

Luo et al. (2017) used a Generative Adversarial Network (GAN) to extract the Kullback-Leibler divergence comparable features from the target and source domain images utilising two CNNs, one for each domain. The researchers also used a softmax function with a temperature variable that regulated the semantic loss and the usable source domain samples: high temperature meant samples were closely related to multiple classes and one class when it was low. The researchers only used a ResNet

architecture. Their research differed from this work in that 1) They used only one architecture while ResNet50, InceptionV3, DenseNet169, DenseNet121, MobileNet, MobileNetV2 and VGG16 were used. The one architecture used utilized the residual blocks, while this work considered architectures that did not use residual blocks - and used convolutional layers. 2) The researchers used images and extended the work in video action recognition in per-frame prediction. In contrast, this work used images only. 3) The feature extraction used convolutional layers, unlike the researchers who used GAN capabilities.

#### **2.4.2 Training SVMs Using Deep Features**

In this study, Nanni et al. (2021) aimed to identify and use high-quality textural features from pre-trained models' middle and upper layers of the models. Principal component analysis (PCA) and discrete cosine transform (DCT) were used for dimensionality reduction, but poor PCA performance restricted the researchers from using DCT. Feature analysis used the local binary pattern (LBP) with ranking utilising the Chi-square. The DCT helped select features for each image channel. Due to its good results and short training time, the support vector machine algorithm was added as the classifier on the pre-trained models: ResNet50, GoogleNet and DenseNet201.

In the researchers' study, the pre-trained models utilized layers in the middle, one layer in every ten and the last four layers. Their research differed from this study: For example, 1) The selection of layers utilized the model's middle and last layers compared to this work's utilization of dynamically selected layers. 2) The analysis of the textural features relied on the linear binary pattern (LBP) algorithm, while this work utilized both the LBP and the grey-level co-occurrence matrix (GLCM) algorithms. 3) The ranking of the features in the researchers' work involved using Chi-square. In contrast, this work utilized the Kullback-Leibler divergence measure to identify the images with the below-average DKL value of the extracted features in an image and low DKL values when comparing the weights in the layers. 4) The researchers used three pre-trained models, while this work had additional models and even evaluated the algorithm's performance on MobileNets. However, the researchers' work gave a consistent performance on thirteen virus test datasets.

### **2.4.3 Borrowing Treasures from the Wealthy (Selective Fine-Tuning)**

Ge and Yu (2017) utilized images in the target domain whose low-level characteristics with high probability match the source domain images to reduce data insufficiency in the target domain. The study used Gabor filters that returned response filters and CNN layer kernels through histograms. The features in the domains were compared using the k-nearest neighbour approach. Moreover, the domain data was used simultaneously to identify the required subset to bridge the data insufficiency. Their study used three pre-trained models: GoogleNet, VGG-19 and AlexNet. It used four publicly available datasets: Stanford Dogs120, Oxford Flowers 102, MIT Indoor 67 and Caltech 256.

This study differed from the researchers' work: 1) This work used pre-trained CNN layers to extract the features, unlike the ones that used the Gabor Filters. Although these filters have been known to behave like CNN layers, this work used the exact layers used in the final fine-tuning, giving better assurance and confidence - the filter behaved similarly to the layers but is not the same. 2) The researchers used only four datasets and three pre-trained models, while this work used six pre-trained models and nine datasets. 3) The researchers only focused on the features, but this work further explored the use of layers in improving the accuracy performance when performing transfer learning. These works also had similarities: they had three standard datasets - MIT Indoor 67, Caltech 256, and Stanford Dogs 120. Both explored extracting features from the CNN layers, although utilising different techniques.

### **2.4.4 CNNs Texture Classification Using Filter Banks**

This study used filter banks to explore textural features in a simple Texture CNN architecture (Andrearczyk & Whelan, 2016). The AlexNet layer output provided the textural descriptors: two CNN layers with an energy-pooling layer were used in this process and connected to a softmax function for the classification process. The researchers used one pre-trained model - AlexNet but tested the technique on three datasets: ImageNet-S1, ImageNet-S2 and ImageNet-T. These three datasets were subsets of the larger ImageNet dataset.

The researchers' approach differed from this work in the following ways. 1) It used the filter banks to extract the needed textural features with descriptors provided by the

output layer of the pre-trained model. This extraction and description of the features varied from this work, which involved the CNN layer extracting the features with description using the GLCM and LBP descriptors. 2) The researchers employed three datasets and one pre-trained model, which could be argued as insufficient, unlike this work that utilized nine datasets and six pre-trained models. 3) The datasets used are subsets of ImageNet with no new datasets, while this work introduced other datasets to identify the similarities of the ImageNet pre-trained models. This work also combined dynamically selected layers to the pipeline, which improved the transfer learning process. However, the two also had similarities: they identified similarities between the target and source domain datasets (which comprised ImageNet samples). The researchers also used the softmax function, like in this work, for classification during fine-tuning. The researchers' use of the energy-pooling layer was essential in improving the descriptors, and the use of middle layers allowed weight sharing, allowing efficient yet straightforward integration.

#### **2.4.5 Maximum-Likelihood Principle and DCNNs in Approximate Nearest Neighbour Search**

Savchenko (2017) introduced the Gaussian Mixture Model (GMM) to extract intensity features from images with efforts to identify the dissimilarities between them, normalizing intensity values across the voxels of the image. The nearest neighbour rules and probability dissimilarity measures were used in the ranking. This study used two pre-trained models - VGG13 and ResNet50 with the random forest as a classifier. This approach differed from this work in the following ways: 1) It relied on the Gaussian function to reduce image noise and extract the features. In contrast, this work depended on the convolutional layers to extract the images' features. 2) The researcher used two pre-trained models, while this work used nine datasets and six pre-trained models to validate the introduced approach. However, similarities exist:

- a) Both tested the concepts on ResNet and VGG pre-trained models.
- b) Both utilized dissimilarity measures - the researcher used the probability dissimilarity measure, while this work used the Kullback-Leibler divergence measure based on probability distributions.

#### **2.4.6 EEG-Based Human Intention Recognition Using Spatio-Temporal Pre-Serving Representations**

During the transfer learning process, knowledge loss can happen, and Zhang et al. (2020) used a deep neural network approach that used two recurrent neural networks (RNNs) for time and space dimensions in the transformation of a 1-D vector to a 2-D vector for Mesh Electroencephalography (EEG). The approach extracted the spatial features from the data meshes using the RNNs, where zero padding was applied on the convolutional layers to prevent information loss. This new approach addressed the complex pre-processing issues in a standard neural network as the meshes classified the segments as k-categories. Their work used an EEG dataset (PhysioNet) and data from an actual case study.

The researchers' work differed from this study in the following ways: 1) The feature extraction was done using RNNs, while this study used convolutional layers of the pre-trained models. 2) The researchers aimed to reduce the information loss between samples in both domains, while this study focused on the divergence of the samples between the source and target domains. 3) The researchers cited the inapplicability of their approach to mobile pre-trained models due to the large number of parameters introduced in their approach. In contrast, this study introduced new parameters and worked well with MobileNets. However, the two approaches utilized artificial neural networks to detect and extract patterns from the data points.

#### **2.4.7 Video Semantic Recognition Using Adaptive Semi-Supervised Feature Analysis**

Luo et al. (2017) introduced an optimal similarity graph with semi-supervised learning in joint feature selection to address the curse of dimensionality in selecting low-level visual features. This approach factored in the lack of labels in video semantic recognition, making the process expensive and time-consuming. Semi-supervised learning facilitates the discriminative knowledge of the original features in the labelled and non-labelled subsets. The researchers assumed that the close labels had a higher similarity that made the joint feature selection easier to learn the optimal similarity simultaneously in the local structure.

The use of optimal similarity graphs helped to reduce the curse of dimensionality while still capturing the features in the local structures of the video. The graph was updated when two video pairs were compared based on their labels. The researchers used their approach on five datasets. This work introduced an excellent approach that compared the source and target domain data points based on labels while evaluating the intrinsic local features, unlike this study's method looked at the textural elements and compared them based on informational divergence. The researchers also introduced semi-supervised learning, unlike the study approach that relied on convolutional neural network learning and still extracted features using the convolutions.

#### **2.4.8 Human Activity Recognition Using Semi-Supervised Recurrent Convolutional Attention Model**

Chen et al. (2020) introduced a pattern-balanced RNN attention model to address human activity recognition (HAR) labelling in semi-supervised learning. This method extracted the salient features from the data while addressing the inter-class variability and similarities, which are significant challenges in HAR processes. The attention mechanism in this approach was used to balance the less-balanced dataset from the multi-modal sensory data. It ensured that only salient features were used while limiting the irrelevant signals from the data. The approach learned the patterns through reinforcement learning and used the partially observable Markov Decision Process (POMDP) in the training and optimisation phases. The researchers' approach used two datasets to evaluate the concept.

This study's approach differed from the researchers' in the following views: 1) The researchers addressed the imbalance in the domain datasets. In contrast, this study looked at the divergence in information between the data points in the domains to determine the best-suited data points based on the textural features. 2) The researchers introduced attention and reinforcement learning into their model in balancing the dataset, while this study considered the use of CNNs and their default learning process. However, the two approaches looked into using features in the domains to identify the required features: the researchers balanced the datasets, and this study found similar low-level features for use in creating quality target datasets for fine-tuning.

## **2.5 Instance-based Transfer learning**

Instance-based transfer learning focuses on adjusting the weight values for the source domain data with similar samples in the target domain (Iman et al., 2023). The main aim of changing the source value weights is to improve the target model's chances of using relevant information from the source in the transfer learning process. Typical approaches used in this learning involve mean weight assignment to the source samples of the source and the target domains and the use of a binary classifier in estimating the source sample weights. The label-attribute relationship ensures that the deterministic list of attributes and the label space's joint probabilities come from the class and the attributes. In this work, the weights were not adjusted but used to determine the layers used in the transfer learning process. The various methods of selecting layers for fine-tuning are documented in the following research findings:

### **2.5.1 Adaptive Fine-Tuning**

Guo et al. (2019) presented the SpotTune technique to improve dynamic global fine-tuning strategies by selecting dynamic layers' in a network. The researchers used a light network with a discrete probability distribution that evaluated suitable layers: It used a Gumbel softmax sampling. The method used ten datasets and was evaluated on the ResNet50 and ResNet26 architectures with last k-layers, stochastic fine-tuning, feature extractor, and standard fine-tuning as the baselines. This study differed from the researchers' work in the following ways: 1) The selection of layers utilized weight divergence. In contrast, the researchers used a decision policy based on the Gumbel softmax sampling. 2) The selection of layers was evaluated on six pre-trained models to the two used by the researchers. 3) It used nine to ten datasets in the researchers' approach, and 4) The researchers used stochastic fine-tuning as a baseline, while this study utilized the other three baselines. However, the researchers noted that introducing the method reduced the parameters required for fine-tuning.

### **2.5.2 Adaptive Filter Fine-Tuning for Deep Transfer Learning (AdaFilter)**

In this approach, Guo et al. (2020) proposed the AdaFilter, which wisely decided the filters to use in pre-trained model layers: it addressed the reusability of filters in a pre-trained model, reducing the number of parameters and reducing overfitting. The



recurrent neural network was responsible for selecting the filters, which determined the fine-tuneable layers using the dataset's low-level features, as Tagnamas et al. (2024) explained. In this approach, the researchers utilized seven datasets: Aircraft (Chen et al., 2024), MIT Indoors 67, Omniglot (Lulu et al., 2024), UCF-101 (Ji & Tekalp, 2024), Caltech 256-60, Stanford Dogs, and Caltech 256-30. The approach used a ResNet50 model with fine-tuning half, standard fine-tuning, L2-SP (Feng et al., 2024), and random policy as the baselines. The researchers' work was different from this study in the following ways: 1) Only one type of pre-trained model was used: ResNet50, which used residual blocks; this work evaluated other pre-trained models, including MobileNets to evaluate the performance of the introduced approach in low-resource settings. 2) The approach was based on an RNN that selected the filters, while this study's layers were selected based on the weight filters but from the Convolutional layers. The two approaches were evaluated on similar datasets: Stanford Dogs, MIT Indoor 67 and Caltech 256.

### **2.5.3 Flex-Tuning**

Royer and Lampert (2020) introduced the Flex-tuning method. In the technique, each layer was analyzed while ignoring the rest. Early stopping was used to avoid overfitting and monitoring the validation dataset. The layer units were tested iteratively until the best layers were selected to participate in the transfer learning process. The Flex-tuning approach was evaluated on three architectures: five-layered CNN, three-layered CNN and one large Inception2 pre-trained model that utilises four datasets: PACS, MNIST, ILSVRC2012 and CIFAR. The results were baselined on last-k layers, scaling and shifting operations (Lian et al., 2024). The researchers' approach differed from this study in the following ways: 1) The layers were selected by evaluating each model's layer at a time. In contrast, this study's approach considered all the layers simultaneously. 2) The approach used one pre-trained and two custom models, while this study tested its approach on six pre-trained models with varying architectures. 3) The approach used four datasets, while the introduced method used nine. The two approaches also shared datasets and one architecture - The inception model.

#### **2.5.4 Using Genetic Algorithm in Transfer Learning Layer Selection**

The researchers' study introduced the PathNet algorithm using genetic algorithms (Nagae et al., 2020). The genotypes represented the layers' weights, with the layers of the highest validation accuracies being used in the transfer learning process. In selecting the layers, the layers were assigned binary numbers, with one assigned to a candidate layer and a zero to a non-selected layer frozen during the transfer learning process. This approach evaluated the method on the CIFAR-100 dataset and the InceptionV3 pre-trained model. Furthermore, two baselines were used: training from scratch and the standard fine-tuning process.

The study's approach differed from the introduced method in various ways. For example, 1) Divergence of signed weights was used in selecting weights, unlike the researchers' work that focused on the validation accuracies. 2) The introduced method focused on the divergence of the weights, while the researchers' work focused on the genetic concepts applying binary values to the model's layers. 3) The study also used only one pre-trained model and dataset. In contrast, the introduced approach used nine datasets and six pre-trained models. 4) The introduced method used the last  $k$  layers and feature extractor to the researchers' baselines. The researchers extended these layers' selection methods using the Stepwise PathNet that utilized the tournament genetic selection algorithm (Imai et al., 2020). The new approach was tested on the SVHN21, Food-101 and CIFAR-100 datasets.

#### **2.5.5 Adaptive Fine-Tuning in Transfer Learning**

Vrbančić and Podgorelec (2020) presented the DEFT algorithm for dynamically choosing fine-tunable layers. This algorithm decides the suitable layers using a differential evolution algorithm. The layers were assigned binary values, and the algorithm gave their predictive performance - the process of selecting, binary assignment and evaluation iterates until an optimal desired performance is achieved. The process only considered layers with the least cross-entropy. The researchers tested the approach using the Osteosarcoma dataset on a VGG architecture.

The researchers' approach differs from this study's approach in several ways. 1) The approach utilized a differential algorithm in the selection and evaluation of the pre-

trained model layers, while this study selected and evaluated the layers based on weights in the pre-trained model layers, and 2) The approach was evaluated on one dataset and one pre-trained model while this study considered nine datasets and six pre-trained models. The researchers' approach was similar to this study's approach by evaluating the difference between probability distributions (cross-entropy), and the study used probability distribution to measure divergence between the layers and the domain samples.

## **2.6 Distance Measures Literature**

This section examines the literature on distance and divergence measures used in fine-tuning the new approach methods. The measures are further reviewed at the new approach's two stages: layer selection, where similarity distance is used, and conflation of features, which uses divergence measures.

### **2.6.1 Similarity Distance Measures**

A similarity measure refers to the likeness of two objects or variables based on their features (Wang, 2020). Similarity distance measure expresses the likeness using the dimensions of the features. Using cosine similarity, the study evaluated weight features in the layers filters for their likeness. Other measures that could have been used included Euclidean distance, Jaccard similarity, Manhattan distance and Minkowski distance.

#### **2.6.1.1 Euclidean Distance**

This metric calculates the distance between two points by measuring their length. It has been used in network applications like clustering (Wang et al., 2022), image classification and object detection. The metric calculates the distance in space by taking the root of the sum of the squared differences between corresponding elements of the two points. Several recent studies have explored the application of distance in networks, including improving learning about structural information in fine-grained classification tasks by Lu et al. (2022) and in content-based image retrieval by Majhi et al. (2021).

### **2.6.1.2 Jaccard Similarity**

The metric returns a quotient of the cardinality of the intersection and the union of the sets. As Yuan et al. (2022) noted, it could also measure sets' dissimilarity. Jaccard similarity has gained popularity in clustering, recommendation systems, and natural language processing (NLP) applications. Several recent studies have explored using Jaccard similarity in neural networks, including the attention U-Net model for multi-label segmentation and detecting COVID-19 abnormalities (Arora et al., 2021) and NodeSim for network embedding (Saxena et al., 2022).

### **2.6.1.3 Manhattan distance**

This distance metric gives the sum of absolute differences between points in a set. It is also known as the L1 norm. It measures the absolute differences between the coordinates of two points, calculated as the sum of the fundamental differences of their corresponding coordinates. Various studies have explored using Manhattan distance in various AI applications, including image retrieval (Tian et al., 2024), memory recommender system (Uyanik & Orman, 2023), and embedding dimensions identification for financial time series (Zhu & Huang, 2022).

### **2.6.1.4 Minkowski Distance**

This distance metric measures the similarity between two points in a vector space. Minkowski distance is a hybrid of Euclidean distance and Manhattan distance. It can be defined as the  $p^{\text{th}}$  root of the sum of the  $p^{\text{th}}$  power of the absolute differences between the coordinates of two points. Minkowski distance is an adequate distance measure in various applications due to its flexibility and adaptability. These studies include improving EEG feature selection in EEG clustering by Al-Shammary et al. (2024) and content-based image retrieval systems (Hameed et al., 2021).

### **2.6.1.5 Cosine Similarity**

This work also looked at the similarity distance between weights using cosine similarity and then evaluated the layers using Kullback-Leibler divergence. Cosine similarity evaluates the cosine angle between any two vectors: a higher cosine similarity is given

by a lower angle. Given two vectors,  $u$  and  $v$ , the cosine similarity ( $Cos(u, v)$ ) can be expressed as shown in equation 2.2:

$$Cos(u, v) = \frac{|u||v|}{\|u\|\|v\|} \quad (2.2)$$

The cosine similarity has been documented in the literature, specifically in artificial neural networks. In a study investigating footprint image retrieval, cosine similarity has been used on closely related images given their picture vectors (Chen et al., 2021).

$$Cos(u, v) = \|u - v_i\|, i \in [1, N] \quad (2.3)$$

where  $u$  is the searched image, and  $v_i$  is the footprint in the picture library for the  $N$  features. Further work has been done by Jin et al. (2020) to evaluate the effects of weight correlation in neural networks and their impact on the generalisation ability of the probably approximately correct (PAC) Bayesian framework. In a CNN model, the correlation can be expressed as:

$$Cos_c(w) = \frac{1}{N_l - 1(N_l^2 - N_l)} \sum_{i,j=1}^{N_l} \sum_{z=1}^{N_l} \frac{|w'_{li_z}| |w'_{lj_z}|}{\|w'_{li_z}\| \|w'_{lj_z}\|} \quad (2.4)$$

where  $Cos(wl)$  is the cosine similarity value between the model layers filters, and  $w$  is the weight filter. The  $w'_{li_z}$  and  $w'_{lj_z}$  are the reshaped  $z^{th}$  columns of the  $i$  and  $j$  filters.

Karat et al. (2023) have done more work using cosine similarity to evaluate neurite density and dispersion in hippocampus microstructural distributions. Pieterse and Mocanu (2019) used the distance measure to assess the similarity between any two neurons in a model's layer with a cosine similarity value close to zero. This value indicates a closer similarity between the neurons and, therefore, a meaningful relationship. This method predicted the next neuron's importance in the model learning pattern.

## **2.6.2 Divergence Distance Measures**

Divergence refers to a function that measures the difference between two probability distributions. This study utilized the probability distributions of the weights, precisely the Kullback-Leibler divergence. However, other divergence measures were also considered in this work comparison. These include Hellinger distance, Wasserstein (Earth Mover) distance, Bhattacharyya distance and Jensen-Shannon distance.

### **2.6.2.1 Hellinger Distance**

This divergence measure evaluates the distance between two probability distributions that share a probability space. The distance value ranges are 0 and 1, with 0 giving the least distance (Lee et al., 2024). The measure has been used in various studies: in modelling certification frameworks to ensure distribution robustness for bounded loss functions and black box models (Weber et al., 2022) and in word embeddings for sentiment analysis (Agrawal & Moparthy, 2023).

### **2.6.2.2 Wasserstein Distance**

This divergence measure is based on the optimal transport theory, aiming to identify the optimal resource allocation and transportation (Okano & Imaizumi, 2022). It also measures the divergence between two probability distributions. Compared to the Kullback-Leibler divergence, this measure performs similarly but has a higher computational cost due to its iterative optimisation process (Bazán et al., 2019). Wasserstein distance has gained much attention recently, especially in neural networks and transfer learning. Unlike other distance measures, Wasserstein distance provides a more meaningful and intuitive measure of the difference between two distributions.

One of the benefits of Wasserstein distance is its ability to ensure a gradient flow throughout the training process. This gradient flow contributes to the stability and convergence of the network, which proves vital for tasks like image recognition and natural language processing. Furthermore, Wasserstein distance facilitates learning the transport map between two distributions, which proves valuable in transfer learning applications. Some of the applications include training of GANS (Chen et al., 2021), analysis of unsupervised domain adaptation (Si et al., 2024), domain image retrieval

(Zhou et al., 2022) and domain sentiment classification (Du et al., 2020).

### **2.6.2.3 Bhattacharyya Distance**

This divergence measure is related to the Bhattacharyya coefficient. It measures the overlapping degree between two given probability distributions (Begum et al., 2022). This asymmetrical divergence has a faster pace to saturation and sticks to the maximum value. It performs worse than the Kullback-Leibler divergence with higher source and target mean values. (Bazán et al., 2019). One of the advantages of the Bhattacharyya distance is its symmetry - it yields the result regardless of the order in which input distributions are considered. Researchers have been exploring the integration of Bhattacharyya distance into networks and transfer learning methods for years. For example, Van Molle et al. (2021) used it for image classification and uncertainty quantification, while Liang and Zhang (2023) used it in L1-norm discriminant analysis.

### **2.6.2.4 Jensen-Shannon divergence**

This divergence measure gives a total bounded symmetrized Kullback-Leibler divergence to the average mixture distribution (Zunino, 2024). Jensen-Shannon divergence (JSD) is a popular similarity measure between two probability distributions. It is widely used in neural networks and transfer learning applications due to its ability to measure the distance between distributions symmetrically. JSD is a modification of Kullback-Leibler divergence, which is asymmetric and does not satisfy the triangle inequality. JSD overcomes these issues and provides a reliable similarity measure between probability distributions.

The measure has been used in various studies, including automatic fabric inspection tasks (Asha, 2022), speech recognition, and sentiment analysis in transfer learning tasks (Su & Kabala, 2023). These divergences have suitable measures, but their shortcomings were the basis of using Kullback-Leibler divergence in this study: The Hellinger gave almost similar results to the Kullback-Leibler divergence with a higher time computation complexity; Jensen-Shannon had a higher computational cost (Toledo et al., 2022); Bhattacharyya had a higher saturation than Kullback-Leibler divergence, and Wasserstein had the iterative optimisation processing. This lower

saturation improved the Kullback-Leibler divergence in balancing the computational complexity and its performance.

### 2.6.2.5 Kullback-Leibler Divergence

This divergence calculates the dissimilarity between two probability distributions (Theodoridis, 2020). Given two probability distributions,  $Q$  and  $R$ , the Kullback-Leibler divergence (DKL) can be expressed as shown in equation 2.5:

$$DKL(Q \parallel R) = \int_{-\infty}^{+\infty} Q(x) \log \frac{Q(x)}{R(x)} dx \quad (2.5)$$

where  $DKL(Q \parallel R) \geq 0$ , with the identity if and only if  $Q = R$  almost everywhere for  $R$ ; since  $Q$  and  $R$  measures are similar.

The Kullback-Leibler divergence has been used in measuring divergence and is well-documented in the literature. Liu et al. (2023) used DKL to evaluate the divergence between the training and target probability distributions in a medical image segmentation model that used diverse image training data. The DKL assessed the differences between training and target images using the weights. However, the comparison was based on a target's probability density functions (PDFs) for joint image weights. Pimentel and Oranfai (2021) proposed using DKL to address procedural data assimilation constraints, and they believe that they work better than Kalman filter approaches.

DKL has also evaluated probabilistic output signals in separating multi-channel speech sources in deep neural networks (Togami et al., 2019). The probabilistic signs came from deep neural network signals (supervised) in a teleconferencing system. More work has been done by Sharvan and Farshad, where DKL compared extracted features in an impulse-noise-resistant version of LBP. The researchers aimed to address combining LBP features with impulse noise-sensitivity and colour combinations (Fekri-Ershad & Tajeripour, 2017). Researchers used the DKL to identify the difference in feature distributions between source and target domains in the



Bidirectional Encoder Representations from Transformers (BERT) model. The BERT mapped the source domain features to the shared feature space in the target domain (Cao et al., 2021). Using the DKL, researchers calculated the divergence between probability distributions between Pix2pix-generated and authentic images. The Pix2pix images were generated by a GAN (deep convolutional) and a cycle-GAN (Lee et al., 2021).

## **2.7 Summary**

This chapter has highlighted the different types of transfer learning and then narrowed it down into feature-based and instance-based transfer learning. The main focus has been addressing the literature that uses selecting the most suitable features and the proper layers for effective transfer. The review in this chapter shows that choosing the most convenient features and suitable layers is essential to many researchers and, most importantly, to the transfer learning process. The researchers have used various methods, including binary values, freezing layers, GANs, and genetic algorithms. However, most methods differed by using one to four models and datasets. They did not evaluate the effects of signed weights and feature divergence in the domains using conflation for better domain adaptation.

This study differs from the existing literature by developing a new model that utilises the following elements:

- a) The most suitable transferrable features in the source and target domains.
- b) The suitable layers for use in the fine-tuning process and
- c) Using Kullback-Leibler divergence in textural features and fine-tunable layer selection improves the transfer learning process performance.

## CHAPTER THREE

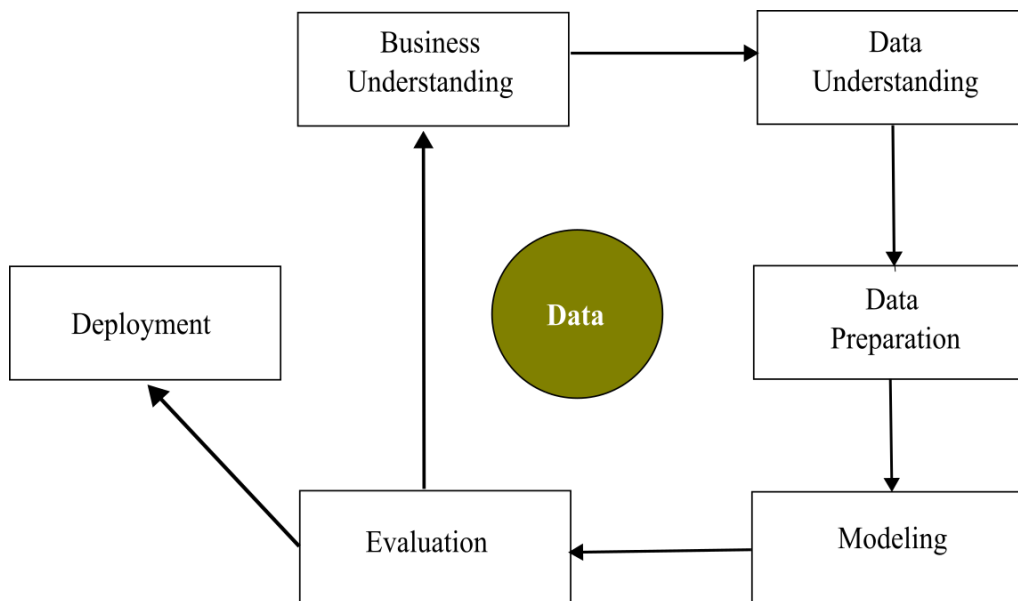
### METHODOLOGY

#### 3.1 Introduction

This chapter explains the methodology used in this study. Section one discusses the methodology overview, section two discusses the datasets used in the experiments, section three addresses the experimental setups, and section four looks at the proposed and baseline methods.

#### 3.2 Methodology

This study follows the Cross Industry Process Model for Data Mining (CRISP-DM) methodology, adopted as the de facto approach in data mining, machine learning, and artificial intelligence projects (Solano et al., 2022). The approach is illustrated in Figure 3.1, and it has six stages: business understanding, data understanding, data preparation, modelling, evaluation, and deployment.



**Figure 3.1: CRISP-DM Illustration**

In the business understanding stage, the project's objectives are listed to underscore the value of the project. In this research, the following objectives are key:

- a) To analyze the existing feature and instance-based transfer learning approaches using various pre-trained models, models' parameters, and datasets. The multiple datasets, pre-trained models and setups have been explained in sections 3.3, section 3.4 and section 3.8
- b) To determine a suitable distance metric for comparison of suitable textural features and fine-tuning layers. The study's metrics have been addressed in section 3.6.
- c) To develop a fine-tuneable model that utilises textural features, data points, and source model weights: a feature-instance-based transfer learning model. The new model has been conceptualized in section 3.5 and addressed in section 3.6.
- d) To evaluate the effectiveness of using the developed model with various datasets and pre-trained CNN models, comparing it with several standard transfer learning baselines. The various baselines for evaluation have been discussed in section 3.7.3.

The second stage examines data understanding through the following tasks: data collection, data description, data exploration, and data quality verification. In this study, we adopted nine secondary datasets discussed in section 3.3.

The third stage addressed data preparation. The stage has the following activities: data selection, cleaning, transformation, and integration. In this work, the conflation approach allowed the selection of quality data points needed in the modelling, and data transformation was done by converting pixels to greyscale for easier feature extraction and scaling to 224×224 pixels. These steps were necessary to accommodate the inputs of the study's pre-trained models.

The fourth stage involves modelling – selecting suitable techniques that use the ready dataset to meet the study's objectives, generating test designs, building models, and assessing. This study evaluated various pre-trained models to fit the developed algorithm. A new model was proposed and developed, as discussed in sections 3.5 and 3.6.

The fifth stage involves model evaluation, which looks at results evaluation to check if the new model meets the business value criteria, process review to address any

overlooked procedures and findings and determine the next steps. In this research, experiments were carried out to address the objectives, as documented in Chapter 4.

In the last stage, the approach addresses the deployment of the new model. It contains the following tasks: deployment planning, monitoring and maintenance, final report development and other reviews. This study's coded new model is deployed in a cloud environment (PaperSpace), addressed in section 3.8. Its effectiveness is compared using the baseline methods in section 3.7 with unseen data to conclude if the model meets the study's objectives.

### **3.3 Data Collection**

The study's data was categorized into source and target datasets. The source dataset was used in the conflation process to reduce the poor adaptation to the target domain using the closely-related target data points. The target dataset was used in the conflation process, training and evaluation of the model.

#### **3.3.1 Source Dataset**

The dataset utilized for training the trained models was ImageNet. For this purpose, a smaller subset called TinyImageNet (Mnmoustafa, 2017), consisting of 100,000 images, was employed. ImageNet is a widely-used dataset in computer vision that offers a collection of labelled images spanning 200 categories, each containing 500 images. The dataset has over one million labelled images categorized into one thousand classes and is considered one of the most comprehensive public datasets.

The ImageNet dataset offers various advantages, including diversity and size, ensuring trained models can effectively identify and classify objects and scenes. This dataset, therefore, becomes vital in transfer learning, where models are initially trained on datasets and adjusted for more specific tasks. The dataset has been used in many studies, including improving the performance of learning models trained by Yun et al. (2021) and addressing label-related issues. Fukuyama et al. (2024) used GANs leveraging the ImageNet dataset. Furthermore, Raffel et al. (2020) used it in natural language processing in specific tasks.

### 3.3.2 Target datasets

Nine datasets, described in Table 3.1, were used in the conflation process to select the quality images for transfer learning.

**Table 3.1: Distribution of Samples for the Experimental Datasets**

Dataset	Training	Validation	Classes
CIFAR-10	50000	10000	10
CIFAR-100	50000	10000	100
MNIST	60000	10000	10
Fashion-MNIST	60000	10000	10
Caltech 256	21425	9182	257
Stanford Dogs 120	12000	8580	120
MIT Indoor Scenes	5360	1340	67
ISIC 2016	900	379	2
ChestX-ray8	3200	800	4

#### 3.3.2.1 CIFAR-10

The CIFAR-10 dataset is an image dataset widely used in computer vision research. The dataset has 60,000 images categorized into ten classes. Each class contains 6,000 images. The dataset has aero planes, deer, dogs, frogs, horses, cars, birds, ships, cats, and trucks, with images of  $32 \times 32$  pixels. The CIFAR-10 dataset has been used in many studies, such as evaluating image classification on different architectures (Antonio et al., 2023), where they found that combining convolutional and dense layers produced the best results. In another study, Dosovitskiy et al. (2020) used the dataset in image recognition tasks, demonstrating that reliance on CNNs is no longer necessary, and a pure transformer applied to sequences of image patches could be used for classification tasks.

#### 3.3.2.2 CIFAR-100

The CIFAR-100 dataset is more complex compared to CIFAR-10. It has 60000 images with up to 100 classes with image dimensions of  $32 \times 32$  pixels. The dataset is an essential benchmarking dataset owing to its range of classes. Several researchers have used the dataset, including Su et al. (2022), in developing the tensor train network architecture Meir et al. (2024) in applying depth networks, where they achieved better performance compared to traditional transfer learning approaches.

### **3.3.2.3 MNIST**

This widely-known public image dataset is used in computer vision and machine learning tasks. The dataset has 70,000 hand-drawn digits images: 60,000 training images and 10,000 testing images. The dataset has been used in various character recognition and object detection applications and is often used to evaluate and compare machine learning algorithms. Abo-Zahhad et al. (2023) introduced the Fast Embedded Capsule Network and Deep Fast Embedded Capsule Network using the MNIST dataset, leading to a 58% reduction in trainable parameters. Additionally, Kadam et al. (2020) examined the benefits of transfer learning on this dataset and showcased how it improves the performance of learning models.

### **3.3.2.4 Fashion-MNIST**

The Fashion MNIST dataset is a public computer vision created in 2017 to replace the MNIST dataset and, just like the MNIST, contains 70,000 fashion images: 60,000 training images and 10,000 testing images. The Fashion-MNIST dataset has been used in various applications, including image classification, object detection, and generative modelling. Meng et al. (2021) used the dataset in their Parallel selective Kernel Attention-based Network. Vijayara et al. (2022) used the Fashion-MNIST dataset for apparel manufacturers to improve clothing recognition, suggestions, and searches on their e-commerce platforms. Additionally, the Fashion MNIST dataset is beneficial in applications, including the development of generative models (Wang et al., 2024), where their model produced high-quality fashion images using the Fashion-MNIST dataset.

### **3.3.2.5 Caltech 256**

The Caltech 256 dataset is another public benchmarking dataset. It contains 30,607 images in 256 object categories encompassing animals, vehicles and various household items. The dataset has found applications in object recognition, image classification and retrieval. Over the years, this dataset has increased interest, with multiple studies showcasing its ongoing significance and value in machine learning research. For example, Wang and Sun (2022) introduced a deep neural network structure that showed superior performance in image classification compared to existing models on the

Caltech 256 dataset – achieving an accuracy of 95.4%, surpassing other study’s benchmarks. In a further study, Luo and Hu (2023) explored the attention mechanisms with feature extraction in image classification using the Caltech 256 dataset.

### **3.3.2.6 Stanford Dogs 120**

The Stanford Dogs 120 dataset consists of 20,580 dog images representing 120 breeds and is used in image recognition, classification, and retrieval systems. Sun et al. (2023) proposed a novel graph-based discriminative features learning network to mine more discriminative features in fine-grained image retrieval. Borwarnginn et al. (2021) used it in a study addressing identifying dog breeds from images. They introduced a technique that employs neural networks with transfer learning to improve the accuracy of recognizing dog breeds using the Stanford Dogs 120 and Columbia Dogs datasets. In addition, Minami et al. (2021) introduced an approach to image retrieval that combines neural networks with graph-based representations. Their method effectively matched dog images with content.

### **3.3.2.7 MIT Indoor Scenes**

This dataset contains images depicting scenes and is used in object detection, image classification, and scene recognition. It contains 67 categories and uses images of  $3504 \times 2336$  pixels. The dataset comprises images captured in various indoor scenes, such as bedrooms, offices, kitchens, and living rooms. MIT Indoor Scenes dataset has been used in several studies: Afif et al. (2020) introduced an approach to scene recognition by leveraging various learning techniques using the MIT Indoor Scenes dataset and achieved results emphasizing its importance in scene recognition research; Heikel and Espinosa-Leal (2022) introduced a method for identifying objects in environments using this dataset and achieved state of the art results.

### **3.3.2.8 ISIC 2016**

The dataset known as ISIC 2016 is a collection of skin lesion images frequently used in skin lesion classification tasks and as a reference point for developing cutting-edge machine learning algorithms and models. Xu et al. (2024) presented a method for segmenting skin lesions with the help of the ISIC 2016 dataset, while Debelee (2023)

used the dataset in skin lesion classification using DNN techniques, further cementing its importance in skin cancer detection and classification.

### **3.3.2.9 ChestX-Ray8**

ChestX ray8 is a collection of chest X-ray images that has become increasingly popular among medical image analysis researchers and practitioners. It comprises 108,948 frontal view chest X-ray images from 32,717 patients. It has been labelled for up to 14 thoracic diseases. These diseases include pneumonia, tuberculosis, lung nodules and pleural effusions. For this study, we focused on four classes with a subset of 4,000 images to mimic scenarios where data is available for transfer learning. The dataset has been used in Pneumonia diagnosis (Kundu et al., 2021), while Āadawi and Elgazzar (2021) used transfer learning to detect COVID-19 from ChestX-ray8 images, achieving an accuracy rate of 99.62%. Furthermore, the dataset has helped identify lung nodules and diagnose tuberculosis (Rea et al., 2021).

## **3.4 Modelling Architectures**

The following pre-trained convolutional neural network models were used in this research to develop the new models that apply the proposed algorithm. Pre-trained models in this study are already developed CNN models whose weights are used to develop the new models through fine-tuning of the models' parameters:

### **3.4.1 VGG16**

The VGG16 pre-trained model is one of the VGG (Visual Geometry Group 16) models used in various computer vision tasks (Sikha & Bharath, 2022). The VGG16 pre-trained model architecture has 16 layers - 13 convolutional layers and three fully connected layers. The network has 138 million parameters, making it one of the largest neural networks. The layers comprise  $3 \times 3$  filters stacked one after the other. These stacked filters allow the network to learn more complex features while reducing the number of parameters. The model accepts input sizes of  $224 \times 224$  pixels.

The VGG16 pre-trained model can adapt effectively to various public datasets, enabling it to grasp high-level features from images. Moreover, using filters helps tackle the



issue of overfitting, which is quite common in deep neural networks. Numerous research studies have investigated the application of VGG16 in transfer learning scenarios. For instance, Wu (2021) in expression recognition, Sitaula and Hossain (2021) classify chest X-ray images and Lim et al. (2023) in recognizing traffic signs in a transfer learning experiment.

### **3.4.2 DenseNet169**

The DenseNet169 pre-trained model, commonly used as a DNN architecture, is highly connected, with each layer having connections to all other layers (Dalvi et al., 2023). This unique characteristic enables DenseNet169 to learn features from scales and makes it more parameter-efficient than other architectures. The model accepts  $224 \times 224$  pixels input sizes and was trained on over 1000 ImageNet classes. The model consists of four dense blocks, each containing several convolutional layers with batch normalization and rectified linear unit (ReLU) activation. The blocks are connected by transition layers, reducing the feature maps' spatial dimensions. The final layer of the network comprises a global average pooling layer with a fully connected layer for classification. The model has over 14 million parameters and uses dense connections between its layers. The extracted features usually are concatenated to improve performance before passing through the layers.

The pre-trained model can address the vanishing gradient problem through the dense connections, enabling gradients to propagate efficiently throughout the network. Several studies have used DenseNet169 in transfer learning scenarios, including Anand et al. (2024) in grading diabetic retinopathy; Wakili et al. (2022) in classifying breast cancer; Kundu et al. (2021) in pneumonia detection in chest X-ray images and Kong and Cheng (2022) to classify COVID-19 in chest X-ray images reporting an improved performance to the ResNet and InceptionV3 pre-trained models.

### **3.4.3 MobileNetV2**

The MobileNetV2 architecture is known for its efficiency, design and fast performance, making it a preferred option for small devices and embedded systems. The model uses depthwise convolutions, effectively decreasing the number of parameters and computations needed compared to regular convolutions of CNNs (Zhu et al., 2024).

The MobileNetV2 consists of several blocks containing depthwise separable and pointwise convolutions with batch normalization and ReLU activation. The blocks are connected by bottleneck layers, which reduce the feature maps' spatial dimensions. The final layer of the network comprises a global average pooling layer with a fully connected layer for classification.

The architecture has over 3.5 million parameters and convolutional filters of  $3 \times 3, 224 \times 224$  pixels input sizes. Using depthwise separable convolutions significantly reduces the required computations, resulting in a smaller model size and faster inference times.

The pre-trained model has been used in several studies, including plant disease classification (Alirezazadeh et al., 2023); Ahsan et al. (2021) to classify COVID-19 in chest X-ray images and Mutawa et al. (2023) in detecting retinopathy.

#### **3.4.4 InceptionV3**

InceptionV3 is a convolutional neural network (CNN) architecture that can improve the accuracy of image recognition tasks while maintaining a reasonable model size (Bhardwaj et al., 2021). InceptionV3's architecture comprises inception modules consisting of multiple convolutional layers with varying filter sizes and max pooling. These modules allow the network to capture features at different scales and reduce the required parameters. This pre-trained model uses label smoothing along with  $7 \times 7$  filters. It requires an input size of  $79 \times 79$  pixels. The model was also trained on the ImageNet dataset. Several studies have applied InceptionV3 in transfer learning scenarios. The pre-trained model has been used in different applications by various researchers, including Jain et al. (2021) to classify skin lesions; Manokaran et al. (2021) to detect COVID-19 in chest X-ray images; Mutawa et al. (2023) to identify retinopathy and Shankar et al. (2022) to classify breast cancer.

#### **3.4.5 ResNet50**

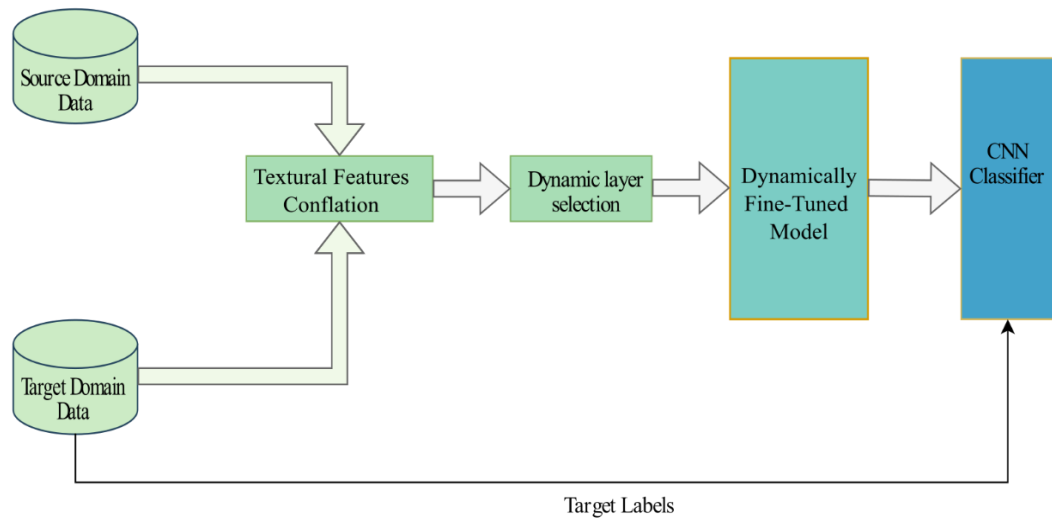
ResNet50 is a popular convolutional neural network (CNN) architecture widely used in transfer learning applications (Wightman et al., 2021). The architecture is based on residual blocks, which makes the network understand more complex features by adding shortcut connections that bypass one or more layers. This approach has

been shown to improve the accuracy of image recognition tasks while reducing the required number of parameters.

This architecture was trained on the ImageNet dataset. The model can handle input sizes of  $224 \times 224$  pixels and can train deeper networks without experiencing the problem of vanishing gradients, just like the DenseNet169. By incorporating connections, the gradients can flow smoothly through the network, making it easier to train and allowing the network to understand intricate features. Consequently, ResNet50 has achieved benchmark performance, including the widely recognized ImageNet dataset. The pre-trained model has been used in several studies, including Alwakid et al. (2023) to classify cases of retinopathy; El Le1 (2023) to classify COVID-19 in chest X-ray images and Al-Haija and Adebajo (2020) to classify breast cancer as either malignant or benign on the BreakHis dataset.

### **3.5 Conceptual New Model**

A well-defined conceptual model of the dynamically fine-tuned transfer learning process is crucial for gaining a high-level understanding of the process (Swaen & George, 2024). The conceptual model effectively illustrates the interaction of various elements and defines critical components, such as the conflation of features and layer selection, which are essential for achieving the objectives of this study. Figure 3.2 illustrates the study's conceptual model. In this process, the target domain images search for images with similar textural features (low-level characteristics) in the source domain, and the returned images subset is used in the source model's training. The new model dynamically selected layers are then used in the fine-tuning process. Kullback-Leibler divergence is used in both model steps to decide the data points and the layers specified in the transfer learning process.



**Figure 3.2: Transfer Learning Conceptual Model**

### 3.6 Proposed Methods

This research used several experimental methods based on the two approaches introduced in the transfer learning approach. These methods utilized the textural features analysis or the filter weights.

#### 3.6.1 Features conflation Approach

This approach was used in the first of the two stages of the new model: feature extraction and comparison to select the best data points with the best features for use in the pre-trained models. The features conflation approach was based on the merging of probability distributions, which minimizes the information loss during the consolidation of distributions and provides optimal consolidation regarding the likelihood-ratio criteria (Hernán-Caballero et al., 2024).

The approach design comprised the feature extraction and the selected dataset. Within the features extraction was a CNN with only one convolutional layer for extracting the features. This process was extended with a pooling layer that reduced the dimensionality of the extracted features, effectively addressing the computational complexities of the many extracted features. This first convolutional layer behaves like the Gabor filters used in edge extraction in the filter banks or the spatial filters commonly used in textural analysis (Varghese et al., 2023). CNN was used in this work

due to its discriminative feature extraction ability. Once the first layer of the CNN had extracted the features, their dimensionality was reshaped into a one-dimensional vector to create a probability distribution for each feature map from the pooling layer. A softmax function with a temperature was used in this conversion. The following assumptions were considered when using the softmax function:

- a) Each one-dimensional vector element was between 0 and 1,  $w_i \in [0, 1]$ .
- b) The sum value of the elements in the one-dimensional vector  $w_1, \dots, w_n$  was 1.

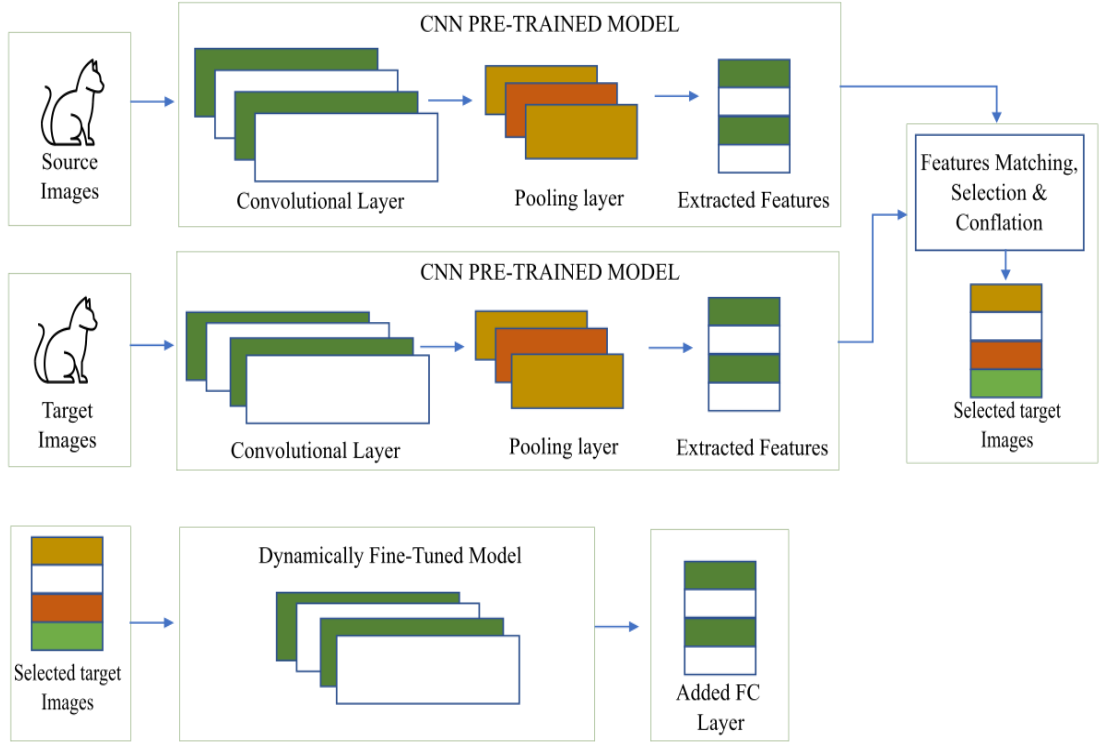
For the element  $w_i$ , its softmax value  $p(w_i)$  was then represented as shown below:

$$p(w_i) = \frac{\exp(w_i)}{\sum_n \exp(w_n)} \quad (3.1)$$

We then expressed the above equation using the temperature in the softmax function, which controlled the probability distribution's entropy levels. The equation becomes:

$$p(w_i) = \frac{\exp\left(\frac{w_i}{T}\right)}{\sum_n \exp\left(\frac{w_n}{T}\right)} \quad (3.2)$$

The result was a set of probability distributions that could be merged into one probability distribution for a given image sample in either the source or target domain dataset. The merged probability distributions between the target and source domains could then be compared. In the comparison phase, the filter banks of merged probability distributions were used to determine which images could appear in the final dataset for use in the transfer learning process. This proposed approach ensured that only quality dataset samples were used in the transfer learning process, filtering low-quality samples. Figure 3.3 below illustrates this process where the features are extracted from both the target and source datasets, and they are compared using Kullback-Leibler divergence with the samples below the average Kullback-Leibler divergence selected as the final source target images by the dynamically fine-tuned model.



**Figure 3.3: Conflation Approach Conceptual Diagram**

The probability distribution conflation was defined as follows:

**Definition 1 (Discrete probability distribution Conflation).** Given a set of probability distributions  $p_1, \dots, p_n$ , a conflated probability distribution  $Q$  could be expressed as shown in the equation below:

$$Q = (P_1, \dots, P_n) \quad (3.3)$$

The equation can be rewritten using probability density functions (probability mass function as referred to discrete probability distribution)  $f_1, \dots, f_n$ , as shown below:

$$(P_1, \dots, P_n) = \frac{f_1(k), \dots, f_n(k)}{\sum_l f_1(l), \dots, f_n(l)} \quad (3.4)$$

where  $k$  and  $l$  are dataset samples

**Definition 2 (Continuous probability distribution Conflation).** Given a set of probability distributions  $p_1, \dots, p_n$ , a conflated probability distribution  $Q$  could be expressed as shown in the equation below:

$$Q = (P_1, \dots, P_n) \quad (3.5)$$

The equation can be rewritten using probability density functions  $f_1, \dots, f_n$  as shown below:

**Definition 3 (Feature vectors' continuous probability distributions Conflation).**

Given an image with probability density functions  $p_i^f, \dots, p_n^f$  and continuous probability distributions  $p_i, \dots, p_j$ , we could express equation 3.5 as follows;

$$(P_i, \dots, P_j) = \frac{\prod_f P_{ij}^f}{\int \prod_f P_{ij}^f} \quad (3.6)$$

When using a specific label  $l_i$ , a set of images  $Xsl = (x_{i1}, \dots, x_{in})$  each with its own conflated probability distribution, a further merged probability distribution can be expressed as follows;

$$l_i(P_{li}, \dots, P_{lj}) = \frac{\prod_x P_{lij}^{fx}}{\int \prod_x P_{lij}^{fx}} \quad (3.7)$$

the  $l_i$  represents a specific image label, while the  $l_s$  represent the feature classes for the source images. The target domain image conflations can, therefore, be expressed as follows:

$$(P_{ti}, \dots, P_{tj}) = \frac{\prod_{tf} P_{titj}^{tf}}{\int \prod_{tf} P_{titj}^{tf}} \quad (3.8)$$

The  $t$  in the above expression indicates the target domain. The  $\&$  for the target image sample can then be compared with the source domain using the DKL. These DKL values are banked in a set of DKLs as expressed below;

$$k = \{DKL_{li}, \dots, DKL_{lj}\} \quad (3.9)$$

where  $k$  is the list of DKLs. The mean of  $k$  is expressed as;

$$\bar{X} = \text{mean}(k) \quad (3.10)$$

Algorithmic steps for features conflation Approach

The conflation approach was described stepwise as follows:

- i) Input the source image and extract the feature vectors.
- ii) Reshape the feature vectors into one-dimensional vectors.
- iii) Using the softmax function, create the set of probability distributions,  $P(x_n)$ .  $P(x_n)$  is from the features extracted using the linear-binary patterns (LBP) properties and the grey-level co-occurrence matrix (GLCM). The probability distributions and their densities have been expressed in Equations 3.3 to 3.8.
- iv) Create a conflated probability distribution of the set of probability distributions created in step iii.
- v) Repeat steps i-iv for all the source domain images.
- vi) Using an image from the target domain dataset, repeat steps i-iv. This step is expressed in Equations 3.7 and 3.8.
- vii) Using DKL, compare a conflated probability distribution of an image in step vi with the set of conflated probability distributions in step v as expressed in Equation 3.9.
- viii) From step vii, calculate the mean DKL for the images in the target domain dataset as expressed in Equation 3.10.
- ix) Select the target images lower than the mean DKL value in Step vii and add them to the final target domain dataset.

In figure 3.4, the extraction of features and their comparison after conflation is done to illustrate the algorithmic steps presented in section 3.6.1.1. Step 1 extracts the source and target image low-level features, creating their respective feature probability distributions in step 2. The distributions are conflated in step 3 and compared using Kullback-Leibler divergence in step 4. The average Kullback-Leibler divergence is calculated in step 5, with step 6 identifying all the images below the average Kullback-Leibler divergence selected as part of the final target dataset.

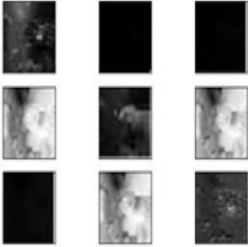


Source Extracted Features – Tiny ImageNet  
 Image: n01443537\_355.JPEG  
 Image Class: n01443537  
 GLCM property: Energy

**Step 1: Extract source features**

```
[[0.99928902 0.99926981 0.99928902 1.          ]
 [0.94807993 0.94742714 0.94809913 0.94740687 ]
 ...
 [0.62436125 0.60740609 0.62467229 0.59870604 ]
 [0.70272142 0.6895107  0.72230437 0.69530947 ]
```

Sample n01443537\_355.JPEG Features



**Step 2: Create Source features probability distributions**

```
[[0.24991349 0.24990389 0.24991349 0.25026912 ]
 [0.25016333 0.24983693 0.25017294 0.2498268  ]
 ...
 [0.2552802  0.2467687  0.25543906 0.24251204 ]
 [0.25005295 0.24353271 0.26004081 0.24637353 ]
```

**Step 3: Conflate Source features probability distributions**

N01443537\_355.JPEG(n01443537) - P<sub>S1</sub>

```
[[0.156252296, 0.156127904, 0.156098285,
 0.156112535, 0.156252296, 0.15252296]]
```

...

N02814860\_174.JPEG(n02814860) - P<sub>Sn</sub>

```
[[0.156220753, 0.156173264, 0.156142416,
 0.156184275, 0.156220753, 0.1562200753]]
```

a)

Target Extracted Features – MIT\_INDOOR  
 Image: airport\_inside\_0604.jpg  
 Image Class: MIT indoor - airportinside  
 GLCM property: Energy

**Step 1: Extract target features**

```
[[0.69709045 0.68926535 0.71380901 0.6865024  ]
 [0.94807993 0.94742714 0.94809913 0.94740687 ]
 ...
 [0.91984273 0.91374497 0.91408388 0.91215293 ]
 [0.6505744  0.62843736 0.66161686 0.62091909 ]
 ...
 [0.25015506 0.24627056 0.25866094 0.24491345 ]
 [0.25016333 0.24983693 0.25017294 0.2498268  ]
 ...
 [0.25015506 0.24627056 0.25866094 0.24491345 ]
 [0.25016333 0.24983693 0.25017294 0.2498268  ]
```

**Step 3: Conflate Source features probability distributions**

airport\_inside\_0604.jpg - P<sub>T1</sub>

```
[[0.156283285, 0.156274773, 0.156014109,
 0.156033265, 0.1516283285, 0.156235666]]
```

**Step 4: Compare target to source conflated distributions**

$$DKL(P_{T1}||P_{S1}) = \int_{-\infty}^{+\infty} P_{T1} \log \frac{P_{T1}}{P_{S1}}$$

$$DKL(P_{T1}||P_{Sn}) = \int_{-\infty}^{+\infty} P_{T1} \log \frac{P_{T1}}{P_{Sn}}$$

**Step 5: Calculate mean divergence**

$$\bar{x} = \{DKL(P_{T1}||P_{S1}), \dots, DKL(P_{T1}||P_{Sn})\}$$

**Step 6: Select samples above mean divergence**

Selected Samples = {airport\_inside\_0604.jpg(airportinside), ..., claustro\_si  
 los.jpg(cloister)}

b)

**Figure 3.4: Example of the Conflation Process. (a) Source Images Conflation Steps and (b) Target Images Conflation with Selected Samples**

Merging the probability distributions using the conflation approach minimizes the loss

of Shannon information (Sayantan et al., 2020). The above algorithm (in section 3.6.1.1) can be further presented using pseudocodes 1 to 3.

Pseudocode 1 illustrates how the source image features are conflated with the function `sourceDomainConflation`, returning an image features conflated distribution.

```
#source conflation
def sourceDomainConflation(image):
    domain_image_conflation_distribution = []
    #outer loop
    for GLCM/LBP(feature) in image:
        features_distributions = []
        #inner loop
        for feature_map in feature:
            dimensioned_feature_map = feature_map.reshape(1 Dimension)
            feature_map_distribution = softmax(dimensioned_feature_map)
            features_distributions.append(feature_map_distribution)

        conflated_feature_distribution = &(amp;features_distributions)
        domain_image_conflation_distribution.append(conflated_feature_distribution)

    return domain_image_conflation_distribution
```

### **Pseudocode 1: Source Conflation process function**

Pseudocode 2 shows how the target image features are defined by the `targetDomainConflation` function that returns a target's image features conflated distribution.

```

#target conflation
def targetDomainConflation(image):
    domain_image_conflation_distribution = []
    #outer loop
    for GLCM/LBP(feature) in image:
        features_distributions = []
        #inner loop
        for feature_map in feature:
            dimensioned_feature_map = feature_map.reshape(1 Dimension)
            feature_map_distribution = softmax(dimensioned_feature_map)
            features_distributions.append(feature_map_distribution)

        conflated_feature_distribution = &(ampersand(features_distributions)
        domain_image_conflation_distribution.append(conflated_feature_distribution
        )

    return domain_image_conflation_distribution

```

### **Pseudocode 2: Target conflation process function**

In pseudocode 3, the distributions returned by the two functions are compared using the DKL, with the images with lower than average DKL being added to the dataset to be used in the transfer learning process.

#adding images to dataset

for image in source\_domain or target\_domain:

```

DKL          =          DKL(sourceDomainConflation(image),
targetDomainConflation(image))
mean_DKL = DKL/DKL_distribution_items
below_DKL = DKL_distribution_items
dataset = []
dataset.append(below_DKL)

```

**Pseudocode 3: Selection and addition of selected samples to the final dataset.**

### 3.6.2 Dynamic Layer Selection Approach

This approach was based on the correlation and divergence of weights in the convolutional neural network layers, specifically the convolutional layers.

**Definition 1 (Convolutional layers weight correlation).** Given the weight filter tensor  $w^{fl}$  in  $\mathbb{R}^{s \times s}$  of the  $l^{th}$  layer, where  $s \times s$  is the size of the kernel. The  $w^{fln} \in \mathbb{R}^{s \times s}$  and  $w^{fln-1} \in \mathbb{R}^{s \times s}$  are the  $n$  and  $n - 1$  filters in the tensor. Reshaping the  $w^{fln} \in \mathbb{R}^{s \times s}$  and  $w^{fln-1} \in \mathbb{R}^{s \times s}$  as  $w^{fln}$  and  $w^{fln-1}$  into single-dimensional tensor filters respectively; the weight cosine correlation of the layer filters is expressed as;

$$\text{Cos}(w_l^f) = \sum_{n=1}^{N_l} \left( \frac{|w_{ln,z}^{f'}| |w_{ln-1,z}^{f'}|}{\|w_{ln,z}^{f'}\| \|w_{ln-1,z}^{f'}\|} \right) \quad (3.11)$$

where  $z$  is the  $z^{th}$  column of the filters and  $N_l$  is the number of filters in the  $l^{th}$  layer.  $w$  refers to a weight,  $fl$  represents the filter in the layer,  $w^{fl}$  refers to a weighting filter in the layer while  $w^{fln}$  represents a weighting filter in a layer at position  $n$ .

**Definition 2 (Signed convolutional layers' weight correlation).** Given the  $l^{th}$  layer with reshaped filters  $w_{l1}^{f'} \in \mathbb{R}^{s \times s}$ ,  $w_{l2}^{f'} \in \mathbb{R}^{s \times s}$ ,  $w_{l3}^{f'} \in \mathbb{R}^{s \times s}$ , ...,  $w_{ln}^{f'} \in \mathbb{R}^{s \times s}$ . If  $w_{ln}^{f'+ve} \in \mathbb{R}^{s \times s}$  and  $w_{ln}^{f'-ve} \in \mathbb{R}^{s \times s}$  are single-dimensional tensor filters with positive and negative weights, respectively; the signed weight cosine correlation of the layer filters is defined as follows:

$$\text{Cos}(w_l^{f^\pm}) = \sum_{n=1}^{N_l} \left( \frac{|w_{ln,z}^{f'+ve}| |w_{ln-1,z}^{f'-ve}|}{\|w_{ln,z}^{f'+ve}\| \|w_{ln-1,z}^{f'-ve}\|} \right) \quad (3.12)$$

where  $+ve$ ,  $-ve$ , and  $\pm$  refer to the positive, negative and positive-negative filter weights. The  $N_l$  is the number of filters in the  $l^{\text{th}}$  layer,  $w$  refers to a weight,  $w_l^{f'}$  refers to a single-dimensional weight vector in the layer while  $w_{ln}^{f'}$  represents a single-dimensional vector of a weight filter in a layer at position  $n$ .  $z$  is the  $z^{\text{th}}$  column of the filters.

**Definition 3 (Cosine similarity DKL).** Given the  $l^{\text{th}}$  layer with reshaped filters  $w_{l1}^{f'} \in \mathbb{R}^{s \times s}$ ,  $w_{l2}^{f'} \in \mathbb{R}^{s \times s}$ ,  $w_{l3}^{f'} \in \mathbb{R}^{s \times s}$ , ...,  $w_{ln}^{f'} \in \mathbb{R}^{s \times s}$ , whose weight cosine similarities are represented as  $\text{Cos}(w_{l1}^{f'})$ ,  $\text{Cos}(w_{l2}^{f'})$ ,  $\text{Cos}(w_{l3}^{f'})$ , ...,  $\text{Cos}(w_{ln}^{f'})$ , the *DKL* of any two filters in the layer can be defined in terms of filters' cosine similarities as expressed below.

$$\text{DKL}(w_{ln}^f \parallel w_{ln-1}^f) = \sum P(\text{Cos } w_{ln}^{f'}) \log \frac{P(\text{Cos}(w_{ln,z}^{f'}))}{P(\text{Cos}(w_{ln-1,z}^{f'}))} \quad (3.13)$$

where  $P$  refers to the probability distribution,  $w$  refers to the weight,  $w_l^{f'}$  refers to a single-dimensional weight vector in the layer while  $w_{ln}^{f'}$  represents a single-dimensional vector of a weight filter in a layer at position  $n$ .  $z$  is the  $z^{\text{th}}$  column of the filters.

A softmax function was then used to convert the cosine similarity vectors of filters into probability distributions. The softmax function relied on two assumptions:

- a) A weight  $p(w_{n-1})$  is in the range of 0 and 1 such that  $p(w_{n-1}) \in [0,1]$ .
- b) The sum of probabilities  $p(w_1), \dots, p(w_n)$  is 1, such that  $\sum_1^n p(w) = 1$ .

where  $p$  refers to the probability of weight value between 1 and  $n$ . Given

$r \in [0,1]$ , the softmax  $p(r)$  for the item  $r$  is expressed as;

$$p(r) = \frac{\exp(r)}{\sum_n \exp(r_n)} \quad (3.14)$$

where the value of  $r$  belongs to the probability distribution  $P$  where  $p(r)$  is the probability of  $r$ . The equation 3.14 can then be re-expressed as;

$$\text{DKL}(w_{ln}^f \parallel w_{ln-1}^f) = \sum P(w_{ln}^{f'}) P(w_{ln}^{f'} - w_{ln-1}^{f'}) \quad (3.15)$$

where  $w$  refers to a weight,  $w_l^{f'}$  refers to a single-dimensional weight filter in the layer,  $w_{ln}^f$  represents a weighting filter in a layer at position  $n$ , while  $w_{ln}^{f'}$  represents a single-dimensional vector of a weight filter in a layer at position  $n$ .

The DKL gave differences between the signed weight filter distributions and weight filters of the convolutional layers. The layers with lower DKL values were then used in the transfer learning process.

#### 3.4.1.1 Algorithmic steps for dynamic layer selection Approach

The following algorithmic steps were used in the dynamic layer selection step in the convolutional neural networks:

- (i) Identify a convolutional layer  $l^c$  in a pre-trained model  $M$ . Any convolutional layer has a prefix *conv*. The  $l^c$  can, therefore, be expressed as;

$$l^c = \begin{cases} M^l, & \text{if } l^n = \text{"conv"} \\ \text{otherwise} & \end{cases} \quad (3.16)$$

where  $n$  refers to the number of layers in the pre-trained model.

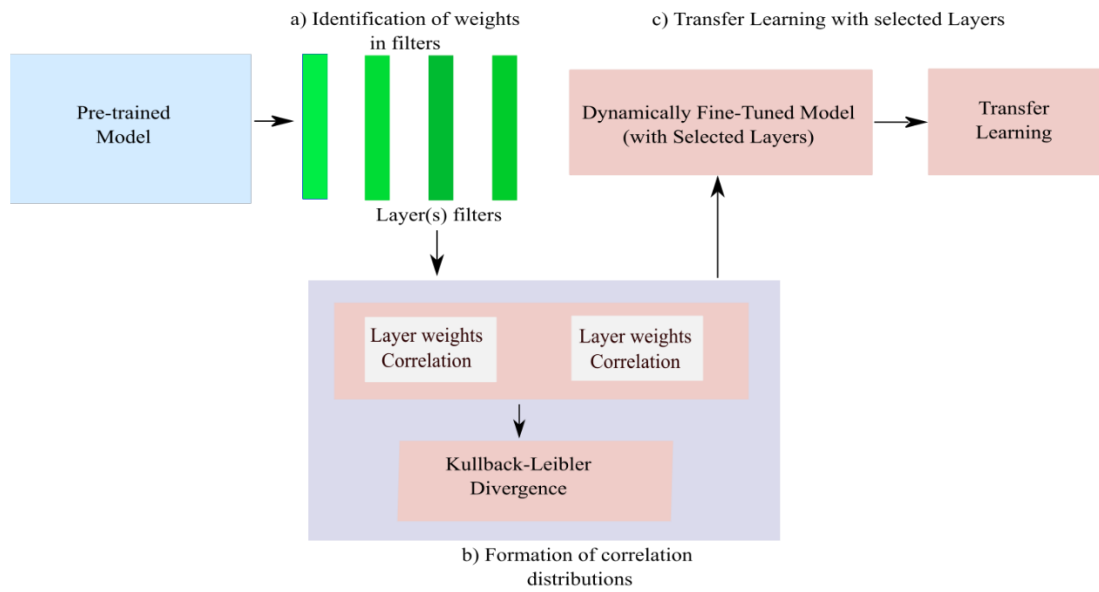
- (ii) Identify the ImageNet filter weight(s)  $w$  in  $l_c$ .
- (iii) Identify the positive and negative weights in equation 3.12.
- (iv) Create a list  $ll^c$  of  $l^c$ s identified in (i).

$$ll^c = (l_1^c, l_2^c, l_3^c, \dots, l_n^c) \quad (3.17)$$

- (v) Calculate the filter(s) weights correlations as expressed in equations 3.11 to 3.15.

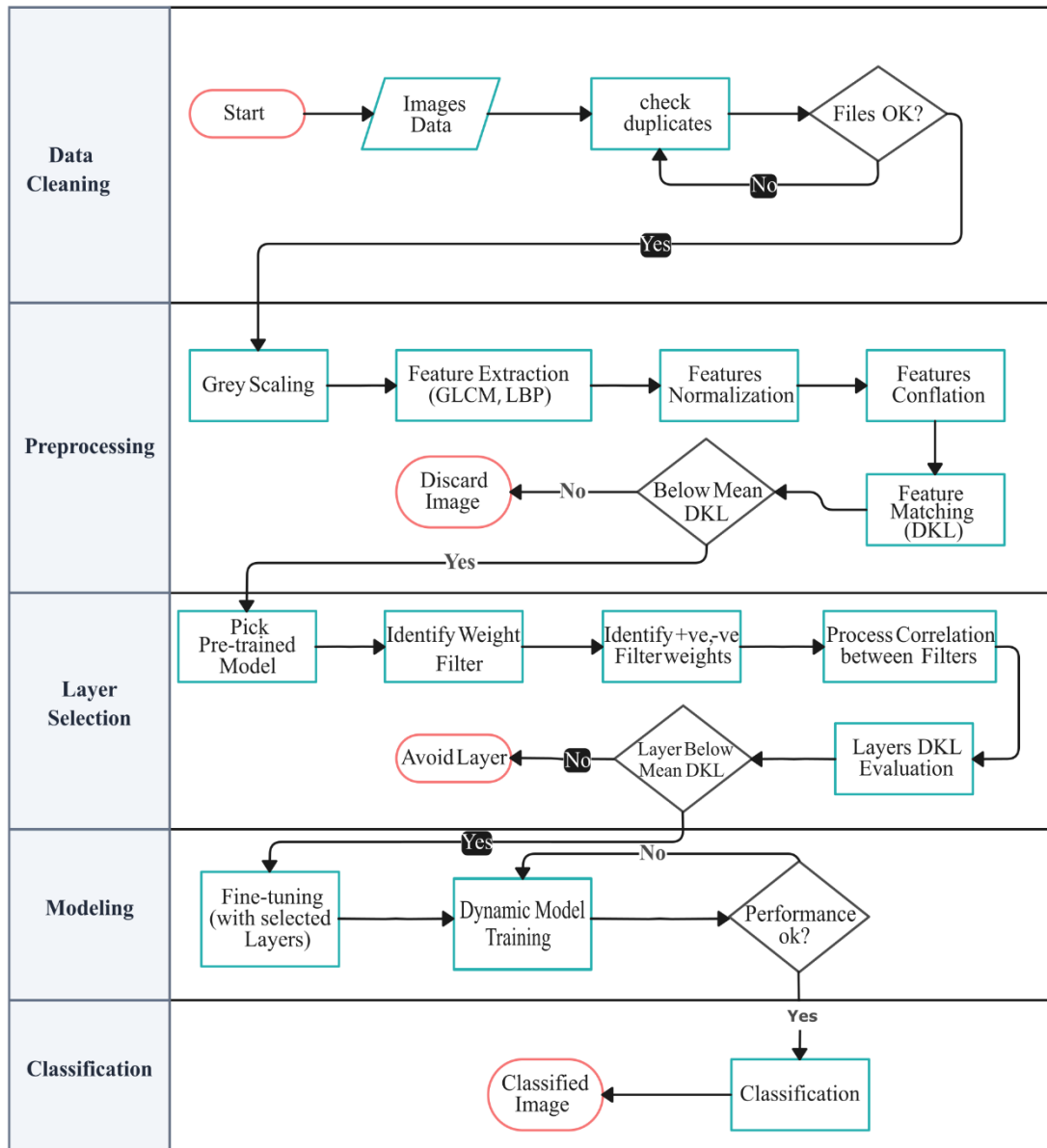
(vi) Select the  $l^c$ s with the lowest DKL values as the selected fine-tunable layers for use in the transfer learning process.

This process is illustrated in figure 3.5, showing the three main stages of the process. The first stage (a) identifies the weights in the filters and their sign, and the second stage (b) checks the correlation and divergence of the layer weights to select suitable layers. Finally, the dynamically fine-tuned model transfers learning using the selected layers in stage (c).



**Figure 3.5: Pipeline of the Dynamic Layer Selection Approach: (a) Layer(s) and Its Filter Weights Identification, (b) Correlation and Ranking of the Layers Using DKL and (c) Use of the Selected Layers in the Fine-Tuning Process**

The two approaches can be illustrated in Figure 3.6, and where they appear in the methodology steps. The conflation approach takes place in the preprocessing phase of the methodology. In contrast, dynamic layer selection occurs before the modelling, with the fine-tuned model used in the modelling phase.



**Figure 3.6: A Flowchart of the New Approach**

### 3.7 Experimental Methods

This section addresses the methods used in the algorithm stages: phase 1: feature extraction and analysis in the conflation process and phase 2 – weights comparison methods in the dynamic layer selection. The section also discusses the various baseline methods used in results comparisons.

#### 3.7.1 Phase 1: Textural Features Extraction and Analysis Methods

The textural feature analysis methods used in this work are GLCM and LBP. In feature



extraction, precisely texture, many methods use grayscale and colour levels, exploiting their first and second-order properties. The first-order properties include variance and mean, and they are derived from individual image pixels. In the second-order properties, the properties, which include correlation and homogeneity, are derived through a comparison between two pixels of an image. The GLCM and LBP utilize the second-order properties in their analysis.

The GLCM has been shown as a more straightforward algorithm to implement while giving outstanding results in applying large systems such as satellite imagery systems that apply textural analysis (Anusha et al., 2024). It has also been recorded to give outstanding results in non-complicated textural analysis tasks (Ramola et al., 2020). Its performance (high accuracy and low complexity) has also been recorded while processing real-time scenarios in object recognition processes (Kurniati et al., 2024). In the case of LBP, it has been recorded as an easy algorithm to implement, with low computational complexity and does not change (invariance) to the monotonic illumination changes.

#### **3.7.1.1 Grey-Level Co-Occurrence Matrix (GLCM)**

This matrix analyses texture in an image utilising the grayscale levels. It represents different combinations of grey levels or brightness of pixels in an image and shows a spatial window of interest of two parameters (Alibabaei et al., 2023). The algorithm utilises second-order statistics (Barburiceanu et al., 2021) and can analyse features showing the spatial shape attributes on interval and amplitude changes and the grey level directions. It has been used in various applications, including behavioral analysis, image retrieval, image segmentation, biometrics applications and image motif recognition. The initial algorithm groups the 14 features into four categories: correlation, entropy, statistical and textural visual measures. However, orientation and distance are the most critical features when determining an image's GLCM (Varghese et al., 2023; Kurniati et al., 2024). The two determine relationships between pixels that form the textural description. In this work, four features: energy, correlation, dissimilarity, and homogeneity, are explored.

#### **3.7.1.1.1 Energy**

Energy is a widely used statistical feature in GLCM (Gray-Level Co-occurrence Matrix) textural analysis, characterizing an image's uniformity and homogeneity. This attribute measures the grey levels intensity of an image, returning the GLCM's sum of the squared elements. Energy has proven to be an essential feature in GLCM textural analysis because it captures the overall image intensity and texture information. The property has been used in various studies, including the examination of textural characteristics that can help distinguish between malignant thyroid nodules (Hristu et al., 2020), a study by Al-Hasani et al. (2022) on the use of GLCM texture analysis to help detect liver fibrosis at a stage and Anand et al. (2023) in the diagnosis of patients with prostate cancer. In these studies, the higher the energy values, the more the disease is likely.

#### **3.7.1.1.2 Correlation**

Correlation plays a role in GLCM (Gray Level Co-occurrence Matrix) textural analysis. The property describes the relationship between pairs of pixels in an image. It measures the connection between the grey level values of two pixels at a specific distance and direction. The connection between the two pixels allows essential insights into the texture directionality and regularity within the image. It is, therefore, an important property when differentiating textures through GLCM textural analysis.

Several studies have revealed that correlation is vital in several medical imaging applications. Hristu et al. (2020) discovered that correlation is a factor in distinguishing between malignant thyroid nodules; Ramadan (2020) emphasized the importance of correlation in identifying benign lesions in mammogram images for detecting breast cancer, and Mujeeb et al. (2022) found correlation to be an indispensable feature in identifying different stages of diabetic retinopathy.

#### **3.7.1.1.3 Dissimilarity**

In GLCM textural analysis, dissimilarity plays a role by describing the connection between pairs of pixels in an image. It quantifies the variances in patterns in texture analysis. The property has been used in several applications, including the medical

imaging application by Nguyen et al. (2021), which was used to distinguish benign breast tumors in ultrasound images. Similarly, Al-Hasani et al. (2022) used the property as a factor in detecting early-stage liver fibrosis through ultrasound images.

#### **3.7.1.1.4 Homogeneity**

Homogeneity measures the similarity in grey-level values between two pixels at a specific distance and direction, providing essential information about an image's texture homogeneity. Homogeneity is preferred in GLCM textural analysis as it captures the consistency in texture patterns, making it a necessary feature for texture analysis. The property has been used by Park et al. (2021) in ultrasound images in diagnosing liver fibrosis, while Arun et al. (2023) discovered that detecting breast cancer through mammography images heavily relies on homogeneity.

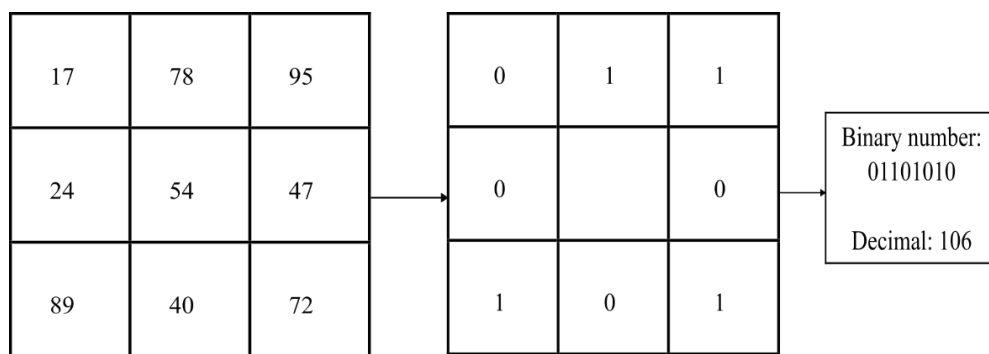
#### **3.7.1.2 Linear Binary Pattern (LBP)**

This textural-visual descriptor is an alternative to the GLCM and uses second-order statistics. However, it compares the pixels' neighbours by assigning binary values 0 and 1 (Simon & V, 2020). In its textural analysis, block sizes of  $3 \times 3$  are used, although more block sizes have been added since its discovery (Zhuang et al., 2021).

The center pixel in an image is used as the thresholding, and the LBP can show the correlation between the pixels with the selected blocks, detecting the edges in the image based on the pixel intensities (Dharma et al., 2022). Once the LBPs have been identified in the image, the description is done using histograms, illustrating the distribution of the feature signals in the image. With its encoding capabilities of the image features, it is possible to detect edges, corners and other properties in an image. This algorithm is an excellent unsupervised learning method widely used in textural and edge detection (Zeebaree et al., 2021).

LBP has been used in different studies, including emotion recognition (Niu et al., 2021), brain tumor classification (Kaplan et al., 2020) and fabric defect detection (Pourkaramdel et al., 2022). Due to the algorithm's low complexity, easier local structure description, faster computational speed, robustness in illumination variations and simplicity (Barburiceanu et al., 2021), it has been applied in computer vision

applications. Other studies have noted that it can withstand monotonic illumination changes (Wu & Liu, 2021) and has computational efficiency. An example of LBP analysis is shown in Figure 3.7 below. The pixel values on the left matrix are compared to the center pixel (54). Any pixel whose value is below the center pixel is assigned 0 and a 1 if above it, resulting in binary values matrix on the right. The final descriptive LBP value is then read clockwise from the top-left binary number, giving a decimal value 106.



**Figure 3.7: LBP Decimal Value Extraction on an Image**

This research used the uniform LBP method to acquire the probability density functions (PDFs) regardless of the orientation. These consistent patterns detect edges, spots and corners (Kaplan et al., 2020) since LBP has the structural information for the corners and edges (Niu et al., 2021), similar to the Gabor filters. In the LBP usage of this work, the images were first converted into grayscale to reduce the dimensionalities, focusing on luminance and binarisation (Qiao et al., 2022). The following algorithmic steps are applied to LBP:

Step 1: Choose an image pixel  $i(x,y)$  and choose the neighbouring pixels ( $p$ ) within the radius  $R$ .

Step 2: Calculate the difference between  $i(x,y)$  intensity and of its neighbours ( $p$ ).

Step 3: Assign a value of 1 or 0 to neighbours whose value is higher than  $i(x,y)$  or lower, respectively.

Step 4: Create a vector of binary values assigned in step 1.

Step 5: Convert all vector elements into decimal values (0-255), creating a new vector of decimal values.

Step 6: Read the values clockwise, as seen in the figure above.

The following equation expresses the LBP calculation:

$$LBP(P, R) = \sum_{p=0}^{p-1} f(g_p - g_c) 2^p \quad (3.18)$$

where  $g_p$  and  $g_c$  refer to the center and neighbouring pixel's intensities, respectively. Six LBP classes can be used on the extracted features: thresholding and quantization, combining complementary features, traditional LBP encoding and regrouping, sampling neighbouring and topology and other inspired methods (Chen et al., 2021). However, different variants still exist (Nanni et al., 2021).

### 3.7.2 Phase 2: Weights Comparison methods in Dynamic layer selection

The methods for identifying the layers were based on weight correlation and divergence in selecting suitable dynamic layers for transfer learning.

There were six methods used in this research:

- a) Positive weights cosine similarity: This method compared the cosine similarity between two positive filter weights.
- b) Negative weights cosine similarity: This method compared two cosine similarity values of negative weight elements of a filter.
- c) Positive–negative weights cosine similarity: This method compared the cosine similarity between positive and negative weights elements in a convolutional neural network layer filter.
- d) Positive weights cosine similarity DKL: This method compared the divergence between two positive weight elements in a filter.
- e) Negative weights cosine similarity DKL: This method compared two negatively signed weights cosine similarity divergence in a CNN layer's filter.
- f) Positive–negative weights cosine similarity DKL: This method evaluated the

divergence between a positive and negatively signed weight's cosine similarity divergence in a CNN layer.

### **3.7.3 Baseline (Conventional) Methods**

Domain adaptability performance in this study was evaluated using three baselines. The baselines were adopted based on the previous research on conflation, dynamic layer selection and transfer learning. The three were:

#### **3.7.3.1 Standard Fine-Tuning**

Standard fine-tuning is a technique in transfer learning that enables the model to adjust for a task. This process involves replacing the model's layer with a classification layer tailored to fit the categories of the new task. To accomplish this, we choose a trained model, like a convolutional neural network (CNN), and fine-tune it precisely for the new task. One of the benefits of standard fine-tuning is its ability to use pre-trained models effectively. Pre-training models demand large amounts of data in their initial training, which can be expensive. However, we can still obtain excellent outcomes by fine-tuning these models for a task using a smaller dataset without investing extensive resources in training entirely new models. The method has been used in several studies, including vision transformers (ViT) model performance improvements by Chen et al. (2021) and tuning the BERT model for medical image protocol classification tasks (Talebi et al., 2024).

#### **3.7.3.2 Fine-Tuning Last-K Layers**

This baseline involves the removal of the few last layers in the pre-trained model. Typically, the last layer,  $k - 1$ , the second previous layer,  $k - 2$  and the third last layer,  $k - 3$ , are used in this process. This method utilises the  $k$  layers of a trained model to extract features and then trains a new classification layer on these extracted features. The value of  $k$  can be adjusted depending on the problem's complexity and the dataset's size. The last  $k$  layers serve as an approach for transfer learning across domains.

#### **3.7.3.3 Feature Extraction**

Feature extraction is a transfer learning method in deep neural networks, and it should

always be seen as a starting point. This approach is simple and effective, using pre-trained models from CNNs or transformers. We generate a feature representation of the input data by removing the classification layer and utilising the output from the preceding layer. Subsequently, this representation is passed through a trained classification layer using the desired dataset.

The method is simple and efficient when working with limited computational resources, including datasets. The technique has been used by several researchers, including Ugail et al. (2023) in visual analysis and attribution of paintings by fine-tuning a trained ResNet50 model using feature extraction and Wu et al. (2022) in their Feature Transferring Autonomous machine learning pipeline (FTAP), which outperforms the state-of-the-art autonomous Tree-based pipeline optimisation tools (TPOT) in the transfer learning tasks.

### **3.8 Experimental Setups**

The experiments performed in this research were implemented on the Paperspace Cloud platform (Quadro P4000, 30GB RAM with GPU) using TensorFlow 2.4.1. The following experimental settings were observed:

#### **3.8.1 Epochs**

The training used 50-150 epochs with samples batched in 8 images. An epoch refers to the number of times the model goes through the training data. The number of epochs is a significant factor in training an AI model, and it can affect how well the model performs since the learnt features come from the data points. One approach to determining the number of epochs is early stopping. The process involves monitoring the model's performance on a validation set during training and stopping when the performance levels off or starts to decline. The number of epochs helps prevent overfitting to the training data, enabling generalisation of the entire dataset. Several studies have explored how the number of epochs affects model performance. For instance, in a study focused on diagnosing breast cancer using images, Sadeghi et al. (2024) discovered that a few epochs resulted in good performance when using the six different transfer learning models for slide analysis. Finding the right number of epochs is a hyper parameter that requires careful consideration when training AI

models.

### **3.8.2 Feature Extraction Settings**

In the feature extraction phase of the research, one convolutional layer (first convolutional layer) in the CNN was used to extract the input features. The features were then analyzed using two methods: GLCM and LBP. The images were scaled to  $224 \times 224$  pixels in the input phase. This size ensured uniformity in selecting the quality dataset and a size match for the final environment, reducing any differences that could have resulted from input sizes. The probability distribution softmax temperature was set to 0.5.

### **3.8.3 Optimizer**

The stochastic gradient descent was used as the optimizer with a learning rate 0.0001. The optimizer plays a role in training AI models by updating the model weights during training. Consequently, choosing the optimizer can significantly affect how well the model performs for a task and dataset, ultimately aiming for optimal performance. Optimizers in training AI models include stochastic gradient descent (SGD) Adam and RMSprop. Stochastic gradient descent is the commonly preferred optimizer because of its simplicity and task effectiveness.

The SGD iteratively updates the model weights by calculating the loss function's gradient to the weights and then adjusting the weights in the opposite direction of the gradient until convergence. The SGD can improve the training performance by adding momentum, which smoothens the updates and allows faster convergence.

The choice of optimizer relies on the task and dataset. It's crucial to experiment with optimizers to determine the most effective one. Kandel et al. (2020) examined how various optimizers impact the performance of a network in image classification. Their study reported that the SGD with momentum yielded higher accuracy and faster convergence than the other optimizers.

### **3.8.4 Loss Function**

The model training used categorical cross-entropy as the loss function. Choosing the loss



function is of significance in AI model training. This parameter determines how well the predicted output matches the output after each training iteration. Each loss function plays an essential role in evaluating the model's performance, and it is crucial to identify the appropriate loss function for a given task and dataset.

Various loss functions can be used for training AI models, such as mean squared error (MSE) binary cross entropy and categorical cross-entropy. Out of these options, categorical cross entropy is the utilized loss function in this study. The loss function was preferred because of its effectiveness in a wide range of tasks, and it yields gradients, making convergence faster and enhancing overall generalisation performance. For class classification tasks, we utilized categorical cross-entropy. This method helps to determine the variation between the actual probability distribution and the predicted probability distribution. The choice of which loss function to use in a model relies on the task and dataset. It's important to experiment with loss functions to identify the effective ones. Xie et al. (2021) explored the influence of loss functions on bioacoustics signal classification. Their findings revealed that focal loss outperformed tested loss functions, exhibiting accuracy and faster convergence.

### **3.8.5 Other settings**

In the fine-tuning process, other settings were considered to improve the model's performance to faster convergence. These settings included the flattening layer for dimensionality reduction of the neurons, dropout layer for model independence, batch normalization and softmax classifier for categorical classification.

#### **3.8.5.1 Flattening Layer**

This particular layer takes the feature map values from the pooled layer, converting them into vectors in a one-dimensional format. Flattening layers are critical in the architecture of networks. They transform the output from the layer into one array output, enabling it to be quickly passed on to the next layer for further processing. Flattening layers play a role in training AI models as they help reduce parameter count, enhance model efficiency and accelerate training speed. Furthermore, flattening mitigates the possibility of overfitting a challenge encountered in learning models. Flattening the layers also has the added benefit of enhancing the model's efficiency. By

obtaining one input from the flattened layers, it becomes much more straightforward to feed it into layers, ultimately reducing the computational complexity of the model. Consequently, this speeds up the training process and minimizes the cost associated with the model. Flattening layers also facilitate the implementation of the model on hardware platforms with limited resources.

Various studies have documented flattening. Garg et al. (2023) investigated the impact of deep learning using VGG16 in real-world applications in Yoga pose classification, where flattening was used. Flattening layers significantly improved the model's accuracy and reduced the computational cost. The study highlighted the importance of flattening layers in deep neural network architecture for a real-world application.

### **3.8.5.2 Dropout Layer**

The layers' neuron drop had a probability index of 0.5. Dropout is a DNN optimisation technique used in AI model training (Salehin et al., 2023). The method helps to improve the model's generalisation ability by reducing overfitting and has become one of the preferred model regularization techniques. The dropout technique randomly deactivates a percentage of neurons during each training iteration. This technique helps to ensure that the model doesn't overly rely on any neuron or group of neurons, making it more resistant to noise and overfitting. This approach creates an ensemble of models that collaborate to make predictions, ultimately enhancing the accuracy and generalisation capabilities of the model. To understand the importance of dropout when training AI models, Salehin et al. (2023) used its impact on network performance. The findings demonstrated that incorporating dropout can address overfitting problems and improve model accuracy. This study emphasized the value of utilising dropout as a regularization technique during a model's training phase.

### **3.8.5.3 Batch Normalization Layer**

Batch normalization is a DNN optimisation method that normalizes the input data within each network layer to improve model performance and address overfitting problems. It has since become an accepted approach for training AI models (Peng et al., 2024). The method adjusts the values of the dataset's mini-batch by subtracting its mean and dividing it by its deviation - ensuring the input data has one deviation and is

centered on zero. The approach can counteract any fluctuations in input data distribution during model training, significantly impacting the model's performance. By implementing batch normalization, the model's training stabilizes and improves accuracy while addressing overfitting problems.

During the normalization process, each layer's dependence on the layer's output enables the model to converge quickly. Furthermore, this technique reduces sensitivity to the parameter values, leading to the generalisation and robustness of the model when faced with unseen data. Huang et al. (2023) investigated how normalization techniques affect network performance. The results showed that batch normalization improves model accuracy and helps mitigate overfitting, further cementing the theoretical concepts of batch normalization.

#### **3.8.5.4 Softmax Classifier**

This hyperparameter was used in the fine-tuning process for all the pre-trained models. The softmax function transforms the model's output into a probability distribution over multiple classes. The softmax function is preferred over other activation functions because it provides better gradients, allowing faster convergence and better generalisation performance. The softmax function is commonly used in tasks involving class classification. It helps to determine the likelihood of each class and selects the one with the probability as the predicted outcome. Compared to activation functions, like tanh and sigmoid, the softmax function is favored due to its reliable gradients. This process leads to convergence and an improved overall performance in terms of generalisation.

Recent studies have emphasized the significance of the softmax function when training AI models. In a study conducted by Nanni et al. (2021), the effect of activation functions on the performance of a learning model for image classification was examined. The results indicated that the softmax function performed better than tested activation functions, exhibiting higher accuracy and faster convergence. This performance underscores its effectiveness as an activation function in AI models. Among the activation functions, the softmax function is the most commonly used and preferred one due to its effectiveness in handling multiple tasks. However, it is essential

to note that selecting the activation function should be based on experimenting with alternatives specific to the task and dataset.

We can significantly enhance our AI model's performance and convergence speed by employing the optimal activation function with the following:

- a) 4096 neurons- the VGG16 architecture uses 4096 neurons in the last dense layer.
- b) 64 neurons - the MobileNetv2 uses 64 neurons in the last dense layer of the pre-trained model.

### **3.9 Summary**

This chapter addresses the methodology used to realise the objectives of this research: the CRISP-DM approach. It also discusses the various datasets and pre-trained models. The study's approaches are also discussed in detail, outlining the feature analysis methods, the weight comparison methods, and even the conventional methods used to evaluate our results.

## CHAPTER FOUR

### RESEARCH RESULTS AND DISCUSSION

#### 4.1 Introduction

This chapter presents the results and research findings of the methods used to develop the new model. The results provide the behaviour of the two approaches used in the new model pipeline and the combined performance of the two sections.

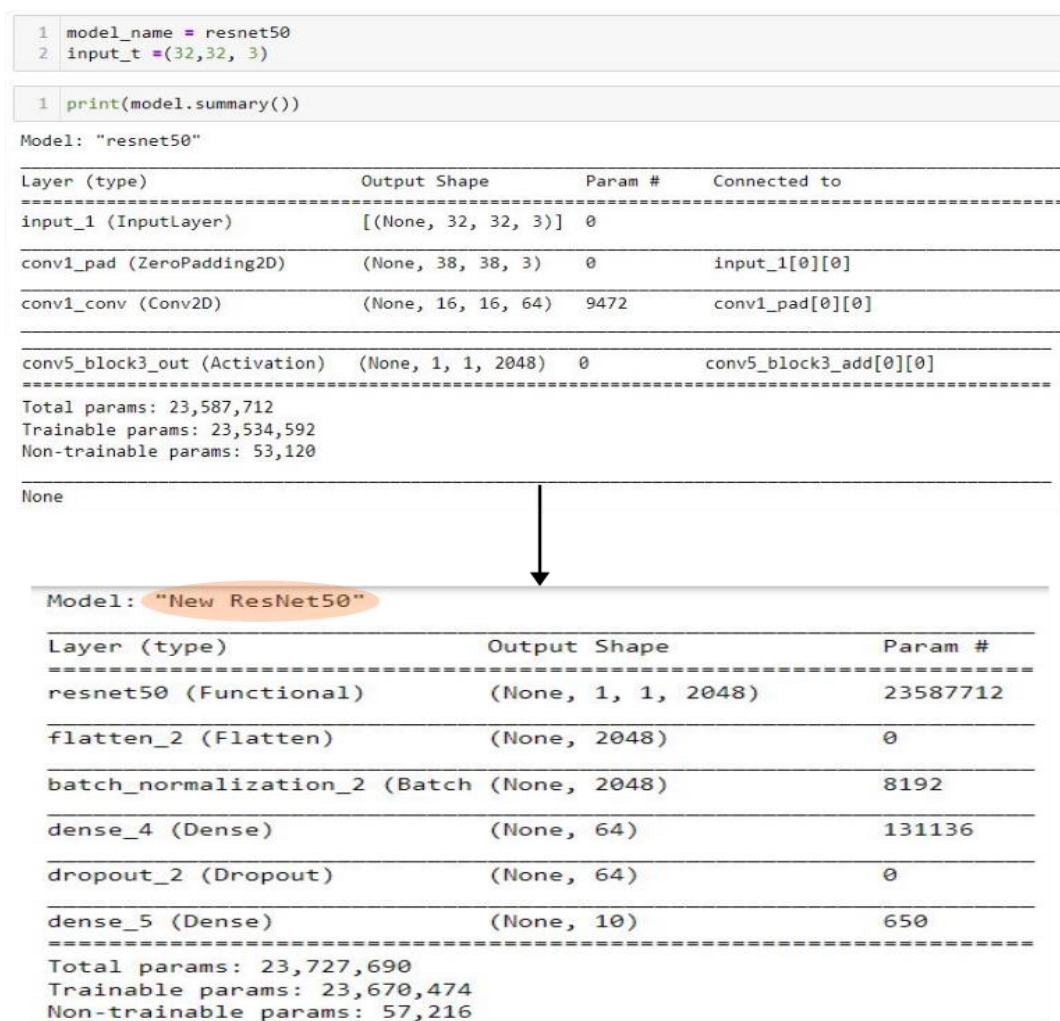
The results are presented in an order that addresses the study's objectives in tables and figures. Section 4.2 addresses the new model and how a pre-trained model changes to the new model, which addresses objective 3. Sections 4.3 to 4.5 assess the effectiveness of the new model on various datasets addressing objective 4, while section 4.6 addresses the new model complexities and objective 2 of the study.

#### 4.2 Investigation of Parameter Changes in the Dynamically Fine-Tuned Model

Figure 3.2 introduces the conceptual model, illustrating the selection of closely matching source - target images to reduce the marginal distribution between the spaces. The concept further shows the selection of fine-tunable layers in the new model to perform the target task. These two stages of the introduced approach are further expressed by equations 3.3 to 3.10 and 3.12 to 3.15, respectively. In the first step (whose algorithmic steps are presented in section 3.6.1.1), pre-trained models are used to identify the textural features without altering them. However, assessing the performance of the selected samples involves fine-tuning with the new models, as labelled in Figure 3.3. The fine-tuning process uses standard fine-tuning, which adds a softmax classification layer. The results of this step are presented in tables 4.1 to 4.3. The selected samples were fed into the new model input layer shown in figures 4.1 to 4.3.

For the second step, which involves a dynamic selection of layers, the various pre-trained models introduced in section 3.4 are used in the experiments (from section 4.4) to implement the algorithm discussed in 3.6.2.1. The resultant model after these changes is the new model, which inherits the weights of the pre-trained models and

the fine-tuned parameters. The names of the new models bear the prefix “New” to indicate our new models, as labelled in figures 3.2, 3.8 and 3.9. Figures 4.1 and 4.2 show how the parameters change after fine-tuning. The two pre-trained models used in figures 4.1 and 4.2 can transfer the learned knowledge using the layers “resnet 50 (functional)” and “vgg16 (functional)” layers to the New ResNet50 and New VGG16 models, respectively. This process further informs Figure 1.1 in section 1.1 to show where the knowledge is and how it is transferred to the new model.



**Figure 4.1: Illustration of Parameter Changes of the ResNet50 Pre-Trained Model and the New ResNet50 Model**

```

1 model_name = vgg16
2 input_t =(32,32, 3)

1 model = model_name(include_top=False,
2                   weights="imagenet",
3                   input_shape=input_t)

1 print(model.summary())

```

Model: "vgg16"

Layer (type)	Output Shape	Param #
input_11 (InputLayer)	[(None, 32, 32, 3)]	0
block1_conv1 (Conv2D)	(None, 32, 32, 64)	1792
block1_conv2 (Conv2D)	(None, 32, 32, 64)	36928
block1_pool (MaxPooling2D)	(None, 16, 16, 64)	0
block2_conv1 (Conv2D)	(None, 16, 16, 128)	73856
block2_conv2 (Conv2D)	(None, 16, 16, 128)	147584
block5_conv2 (Conv2D)	(None, 2, 2, 512)	2359808
block5_conv3 (Conv2D)	(None, 2, 2, 512)	2359808
block5_pool (MaxPooling2D)	(None, 1, 1, 512)	0

Total params: 14,714,688  
Trainable params: 14,714,688  
Non-trainable params: 0

None

↓

```

1 print(t_model.summary())

```

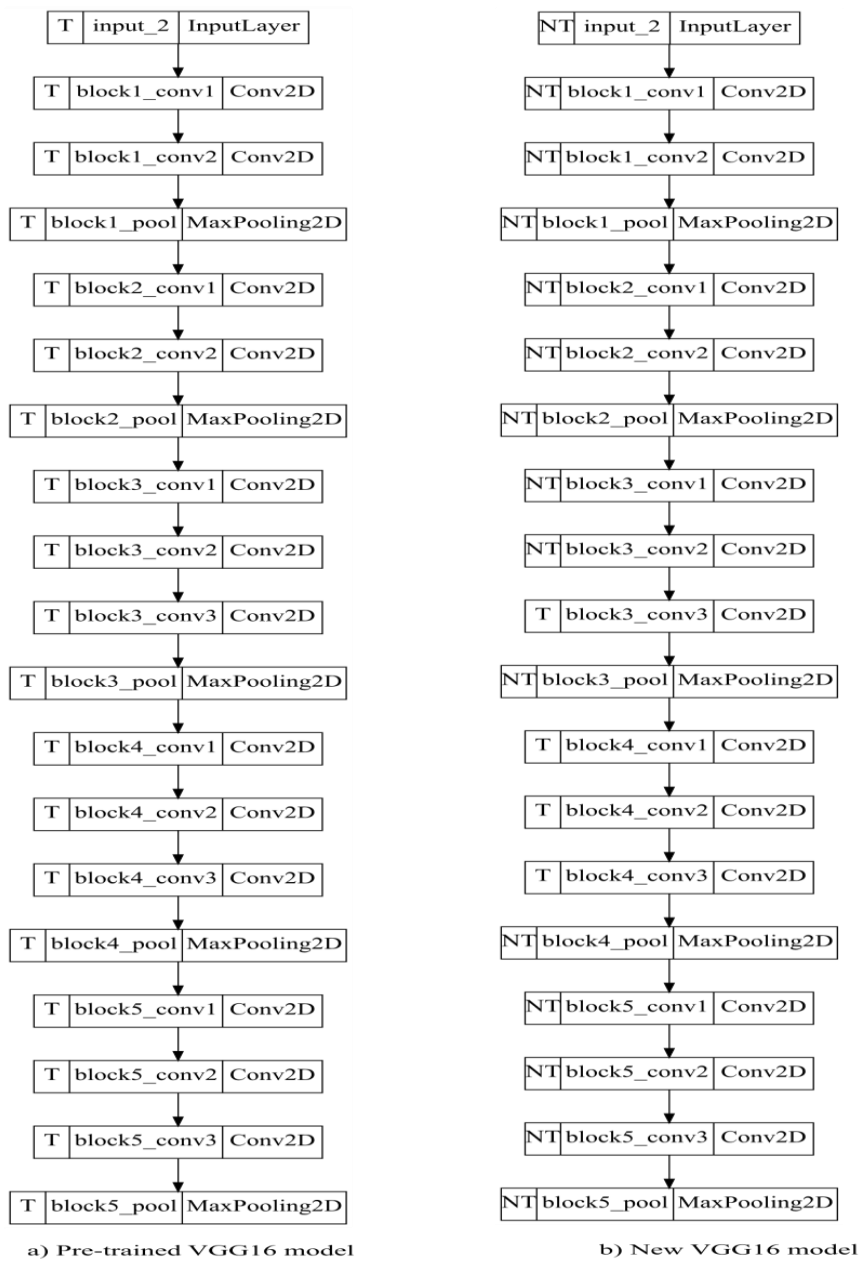
Model: "New VGG16"

Layer (type)	Output Shape	Param #
vgg16 (Functional)	(None, 1, 1, 512)	14714688
flatten_3 (Flatten)	(None, 512)	0
batch_normalization_3 (Batch Normalization)	(None, 512)	2048
dense_6 (Dense)	(None, 64)	32832
dropout_3 (Dropout)	(None, 64)	0
dense_7 (Dense)	(None, 10)	650

Total params: 14,750,218  
Trainable params: 14,749,194  
Non-trainable params: 1,024

**Figure 4.2: Illustration of Parameter Changes of VGG16 Pre-Trained Model and the New VGG16 Model**

In Figure 4.3, the pre-trained VGG16 model has all its layers in the training mode and would use all the layers' weights in a transfer learning task. The layers in the training mode are represented by the "T" and the non-trainable ones by "NT".



**Figure 4.3: An Illustration of the Pre-Trained VGG16 Model and the New VGG16 Model Showing the ‘T’ (Trainable) and ‘NT’ (Non-Trainable) Layers**

However, the trainable layers change after selecting and freezing the non-trainable ones. This new architecture is the new model that performs the classification tasks in the study. The same process happens for all the other selected pre-trained models.



### **4.3 Investigation of Textural Features Conflation on the Dynamically Fine-Tuned Model**

This section presents the results of the conflation approach using the GLCM and LBP properties performance (accuracy, recall, and precision), comparison to baselines, comparison between samples, probability distance measures, and computational complexities.

#### **4.3.1 Performance on GLCM and LBP Properties**

The GLCM and LBP properties were evaluated based on their accuracy, precision, and recall to determine the best or quality samples to use as target dataset samples.

##### **4.3.1.1 Accuracy**

Accuracy refers to the ratio of correct predictions to total predictions (Machine Learning Glossary, 2024). Accuracy can be expressed using the equation below:

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \quad (4.1)$$

where  $TN$  refers to true negative predictions,  $TP$  refers to true positive predictions,  $FP$  refers to false positive predictions, and  $FN$  refers to false negative predictions.

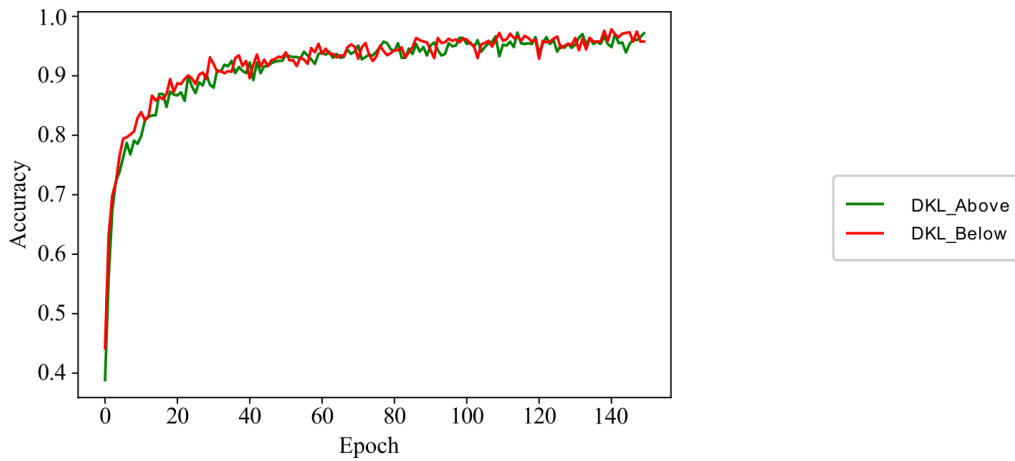
In Table 4.1, the accuracy results of the new VGG16 model have been presented when using the various datasets for the conflation process. The results consider both the GLCM and LBP methods.

**Table 4.1: Comparison of Accuracy Performance (%) on Selected Datasets on New VGG16**

Dataset	Methods									
	GLCM								LBP	
	Correlation		Homogeneity		Energy		Dissimilarity		BLW	ABV
	BLW	ABV	BLW	ABV	BLW	ABV	BLW	ABV		
Caltech 256	98.69	96.93	97.73	96.01	98.45	95.65	96.32	96.03	97.77	96.84
MIT Indoor	97.25	95.37	90.81	88.90	90.76	89.11	91.99	88.04	90.62	90.40
Stanford Dogs	98.64	98.11	94.26	91.43	94.29	89.74	92.07	90.77	93.27	92.22

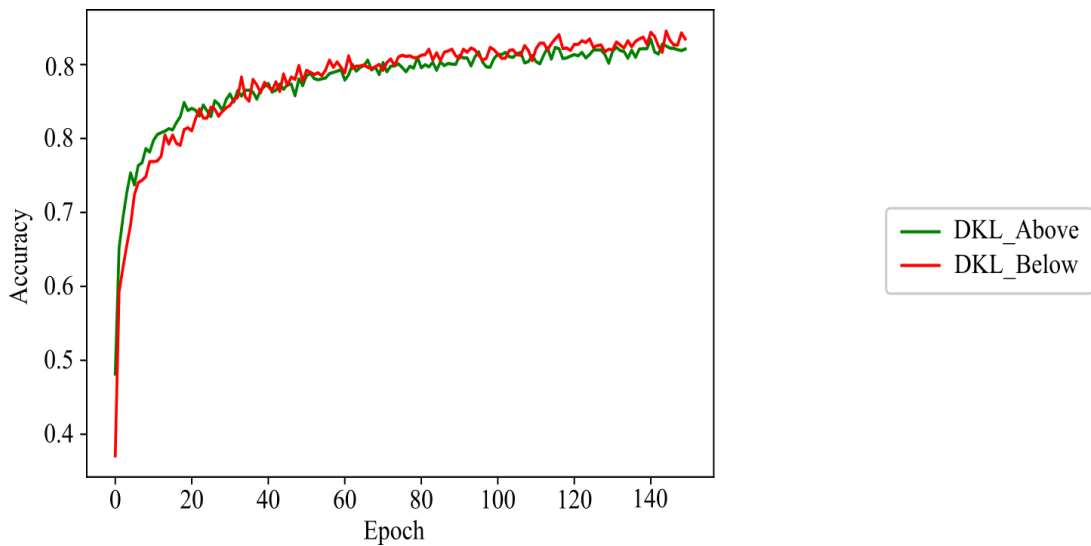
\*BLW; ABV represent Below-average and Above-average DKL

From the table, the Caltech256 dataset gave a good accuracy performance on the GLCM’s correlation property and performed the least with the dissimilarity property. BLW stands for below-average DKL, while ABV stands for above-average DKL. Similar results were achieved with the MIT indoor and Stanford dogs’ datasets. The accuracy of the LBP properties was lower in Caltech 256 GLCM properties and better in the Stanford dogs’ dataset. In the new MobileNetV2, the highest performance was noted in Correlation, Energy and Homogeneity. Like in the new VGG16 accuracy performance, correlation performed well in the new MobileNetV2; although the LBP performed well for the Caltech256, the GLCM properties dominated the other datasets. The selected samples in the Caltech256 dataset showed that images with overall conflation above the mean DKL (green) gave lower accuracy performance than those with lower DKL (red), as seen in Figure 4.4. The “DKL\_Below” samples gave an accuracy performance of 98.69% compared to the 96.93% for the “DKL\_Above” samples after the 140 epochs.



**Figure 4.4: GLCM Correlation Using Caltech256 on the New VGG Model**

In Figure 4.5, the LBP conflated features with higher than mean DKL also gave a lower performance in the new MobileNetV2 model, further illustrating the importance of the lower informational difference noted in, the lower than mean DKL samples. The “DKL\_Below” samples gave an accuracy performance of 95.41% compared to the 93.05% for the “DKL\_Above” samples.



**Figure 4.5: LBP Using MIT Indoor on the New MobileNetV2 Model**

#### 4.3.1.2 Precision

Precision is the identification of the correct samples in data (Machine Learning Glossary, 2024) as expressed below;

$$\text{Precision} = \frac{TP}{TP + FP} \quad (4.2)$$

where  $TP$  refers to true positive predictions, and  $FP$  refers to false positive predictions. Precision is typically used alongside accuracy performance metrics. The new VGG16 model precision performance is shown in the table below using the selected datasets.

Table 4.2 presents the precision results of the new VGG16 model when using the various datasets for the conflation process considering the GLCM and LBP methods.

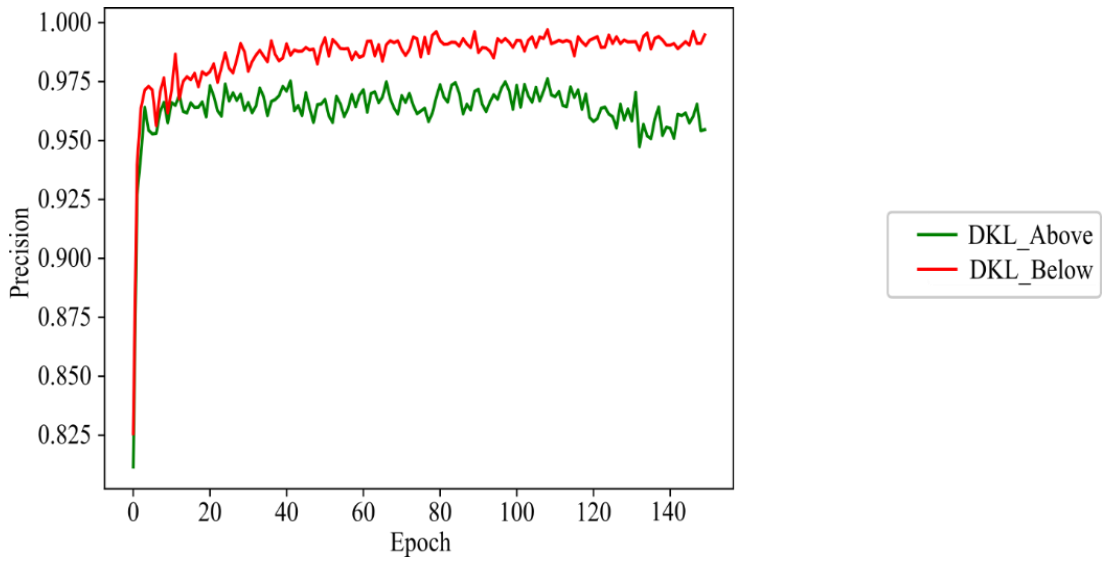
**Table 4.2: New VGG16 Precision Performance (%) on Selected Datasets**

Dataset	Methods									
	GLCM								LBP	
	Correlation		Homogeneity		Energy		Dissimilarity		BLW	ABV
	BLW	ABV	BLW	ABV	BLW	ABV	BLW	ABV	BLW	ABV
Caltech 256	98.83	97.87	98.38	97.31	98.81	97.07	98.00	97.46	98.69	97.92
MIT Indoor	97.36	96.04	93.23	91.19	92.85	91.40	93.81	90.68	93.03	91.05
Stanford Dogs	99.69	97.56	95.37	93.31	96.94	91.74	93.19	92.78	96.19	93.07

\*BLW; ABV represent Below-average and Above-average DKL

The conflated lower DKL indicated that BLW gave the highest precision. In the case of GLCM, the correlation and energy properties had higher precision than the other properties. The Stanford Dogs and Caltech256 datasets gave the least precision for the dissimilarity property. Furthermore, the GLCM properties gave higher Caltech256 and MIT Indoor datasets scores.

Similarly to Table 4.2, the GLCM’s energy continued to give good precision performance compared to the other GLCM properties. Still, the LBP performed better in the MIT Indoor dataset in the new MobileNetV2 model. However, the GLCM dominated the other two datasets. Figure 4.6 shows the performance of the Caltech256 LBP precision on the new MobileNetV2. The “DKL\_Below” samples gave a precision of 99.54% compared to the “DKL\_Above” of 97.26%. The “DKL\_Below” also showed a better training curve, as noted in the figure, which could result from using better-quality data points.



**Figure 4.6: LBP Precision Plot of the New MobileNetV2 on the Caltech256 Dataset**

As illustrated in the Figures above, the samples with lower DKL values had higher precision performance.

#### 4.3.1.3 Recall

The recall is a function of the correctly identified samples of actual positives (Machine Learning Glossary, 2024) to the total predictions expressed in Equation 4.3:

$$\text{Recall} = \frac{TP}{TP + FN} \quad (4.3)$$

where  $TP$  refers to true positive predictions, and  $FN$  refers to false negative predictions.

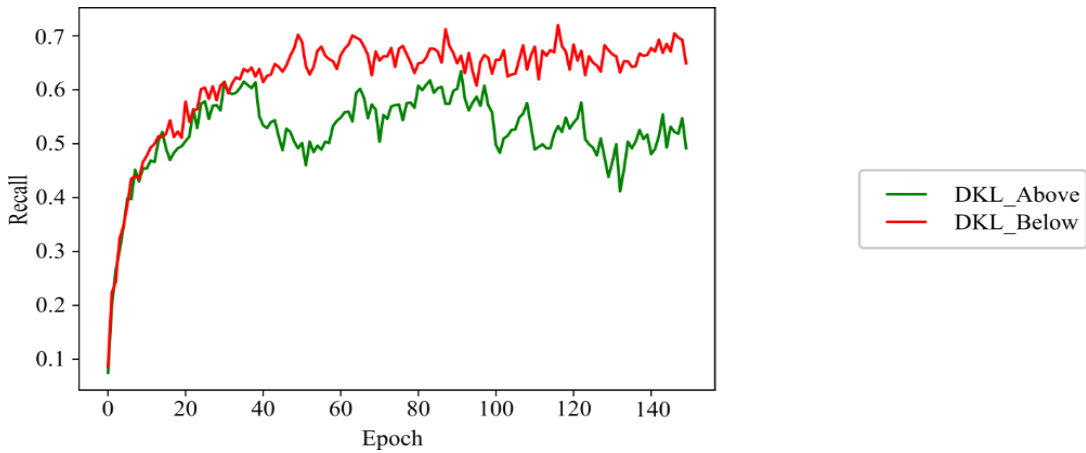
Table 4.3 presents the new MobileNetV2 model's accuracy results for the various datasets and methods used in the conflation process.

**Table 4.3: New MobileNetV2 Recall Performance (%) on Selected Datasets**

Dataset	Methods									
	GLCM								LBP	
	Correlation		Homogeneity		Energy		Dissimilarity		BLW	ABV
	BLW	ABV	BLW	ABV	BLW	ABV	BLW	ABV		
Caltech 256	87.34	71.00	73.64	73.07	90.04	85.62	76.87	76.00	95.68	88.72
MIT Indoor	92.10	90.38	90.35	86.58	92.06	90.00	89.46	89.22	92.54	89.38
Stanford Dogs	96.47	96.26	97.11	93.68	96.78	95.09	97.51	95.08	95.80	94.99

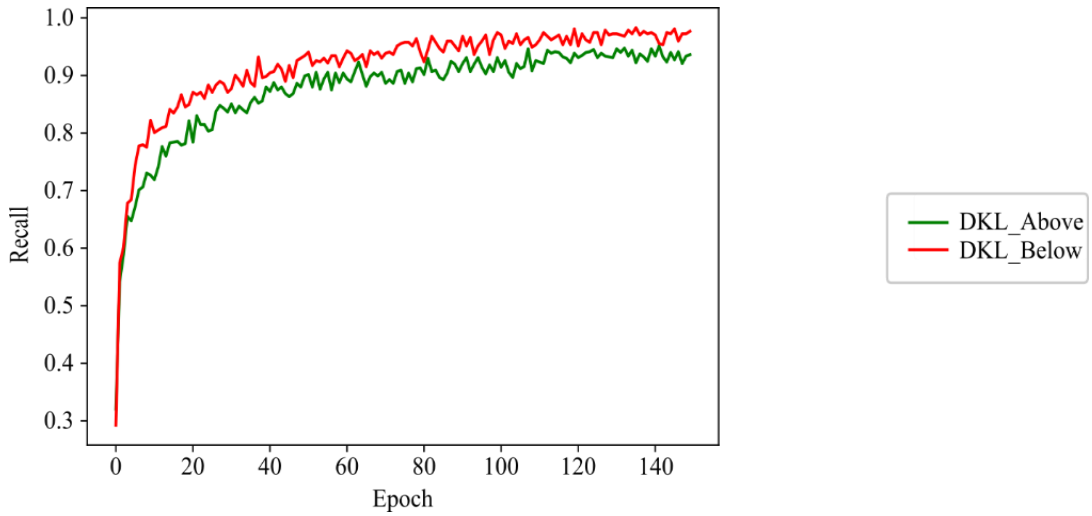
\*BLW; ABV represent Below-average and Above-average DKL

As recorded in Table 4.3, the correlation and energy properties performed well with the other GLCM properties. The LBP still outperformed the GLCMs in the Caltech256 and MIT Indoor datasets. The same trend of energy and correlation GLCM properties to give the best recall values among the GLCM properties was observed in the new VGG16. In contrast, dissimilarity gave the least recall performance. Figure 4.7 illustrates the new VGG16 correlation recall performance on the Caltech256 dataset. In the figure, the training curve when using the “DKL\_Below” looks more stable and gives the highest recall of 97.39% compared to the use of “DKL\_Above” samples that gave 96.38%.



**Figure 4.7: Correlation Recall Plot on Caltech256 Dataset Using New VGG16**

In Figure 4.8, a similar trend of recall performance is noted on the new MobileNet when using the Stanford dogs’ dataset. The samples with lower DKL outperform those with DKL above the mean. The “DKL\_Above” samples gave a recall rate of 93.68% compared to the 97.11% for the “DKL\_Below” samples.



**Figure 4.8: Homogeneity Recall Plot on New MobileNetV2 on Stanford Dogs' Dataset**

#### 4.4 Investigation of Dynamic Layer selection Approach on the New Model

This section presents the results of the dynamic selection of fine-tunable layers. The results are shown in the following subsections: accuracy performance comparison based on probability distance and divergence metrics, accuracy performance comparison using the weights methods (cosine similarity-positive weights, cosine similarity-negative weights, cosine similarity-positive-negative weights and their respective Kullback-Leibler divergence methods (on signed weights - positive, negative, and positive-negative)). The other comparisons are on the standard baselines and the computational complexities.

##### 4.4.1 Comparison between Distance Measures

This section looks at the accuracy performance of the five new models used in the dynamic selection of layers approach. The positive and negative weight methods are presented and compared. Table 4.4 presents the results of the Fashion-MNIST dataset accuracy performance on the various new models.

**Table 4.4: Comparison of Fashion-MNIST Dataset Accuracy Performance (%) of Selected New Models**

Model	Fashion-MNIST											
	Cosine Similarity						Kullback-Leibler Divergence					
	Positive		Negative		Positive-Negative		Positive		Negative		Positive-Negative	
	H	L	H	L	H	L	H	L	H	L	H	L
ResNet50	90.15	90.30	89.87	89.92	90.02	90.21	90.07	90.08	89.97	89.91	91.89	91.35
VGG16	90.88	90.21	90.75	90.77	90.20	90.46	90.62	90.19	90.34	90.55	91.58	90.78
Inception V3	91.46	90.78	90.59	90.65	90.45	90.31	90.60	90.29	90.42	90.16	91.03	90.40
DenseNet169	90.56	90.28	90.13	90.63	90.55	90.70	90.52	90.46	90.25	90.03	92.19	91.36

\*H; L represent Higher than Average and Lower than Average

From the presented results, the positive and the Positive-Negative methods in Cosine Similarity and Kullback-Leibler divergence perform better than the Negative methods. Table 4.5 further presents the MNIST accuracy performance for the various models.

**Table 4.5: Comparison of MNIST Dataset Accuracy Performance (%) of Selected Models**

Model	MNIST											
	Cosine Similarity						Kullback-Leibler Divergence					
	Positive		Negative		Positive-Negative		Positive		Negative		Positive-Negative	
	H	L	H	L	H	L	H	L	H	L	H	L
ResNet50	98.98	99.01	98.76	98.90	98.85	98.83	98.72	99.03	98.96	98.99	99.45	99.25
VGG16	99.34	98.29	99.28	99.27	99.25	99.24	99.26	99.28	99.20	99.28	99.31	99.18
Inception V3	98.80	98.78	99.28	99.27	99.63	99.35	98.10	98.06	98.04	98.03	93.68	91.70
DenseNet169	88.45	89.20	88.56	89.69	99.14	99.04	91.09	90.29	90.24	90.15	90.40	89.70

\*H; L represent Higher than Average and Lower than Average

The accuracy performance in Table 4.5 was lower than the accuracy values in Table 4.4 for the negative weights methods. However, the performance of the Positive and Positive-Negative methods in the Cosine Similarity and Kullback-Leibler divergence continued over the Negative methods. This performance is further noted in Table 4.6, which presents the results of the CIFAR-10 dataset.



**Table 4.6: Comparison of CIFAR-10 Dataset Accuracy Performance (%) of Selected New Models**

Model	CIFAR-10											
	Cosine Similarity						Kullback-Leibler Divergence					
	Positive		Negative		Positive-Negative		Positive		Negative		Positive-Negative	
	H	L	H	L	H	L	H	L	H	L	H	L
ResNet50	68.08	67.25	67.68	66.94	67.98	67.30	67.72	67.65	67.50	67.04	68.58	68.21
VGG16	80.13	80.01	79.73	80.00	80.35	80.15	80.46	80.18	80.50	79.88	82.45	80.12
InceptionV3	86.31	84.23	85.76	85.70	85.43	86.01	81.98	80.19	82.27	81.65	73.56	69.55
DenseNet 169	77.23	75.36	76.78	76.44	76.90	76.84	77.46	77.32	76.88	77.37	78.13	77.10

\*H; L represent Higher than Average and Lower than Average

From the table, the performance of the CIFAR-10 dataset in the various models for the Cosine Similarity and Kullback-Leibler divergence is lower compared to the Fashion-MNIST and MNIST results presented in Tables 4.4 and 4.5. The results of the CIFARs are further presented in Table 4.7 for the CIFAR-100.

**Table 4.7: Comparison of CIFAR-100 Dataset Accuracy Performance (%) of Selected New Models**

Model	CIFAR-100											
	Cosine Similarity						Kullback-Leibler Divergence					
	Positive		Negative		Positive-Negative		Positive		Negative		Positive-Negative	
	H	L	H	L	H	L	H	L	H	L	H	L
ResNet50	18.92	16.89	18.82	17.91	18.08	18.00	19.03	18.61	18.68	18.73	19.52	19.98
VGG 16	38.49	36.24	36.56	38.11	38.64	36.28	37.21	37.13	37.49	36.58	20.77	13.53
Inception V3	57.16	54.56	54.01	54.16	58.23	57.25	30.24	29.45	29.36	26.47	35.22	30.55
DenseNet 169	34.31	33.08	35.00	34.15	34.25	34.17	34.68	34.32	35.25	35.02	57.16	54.56

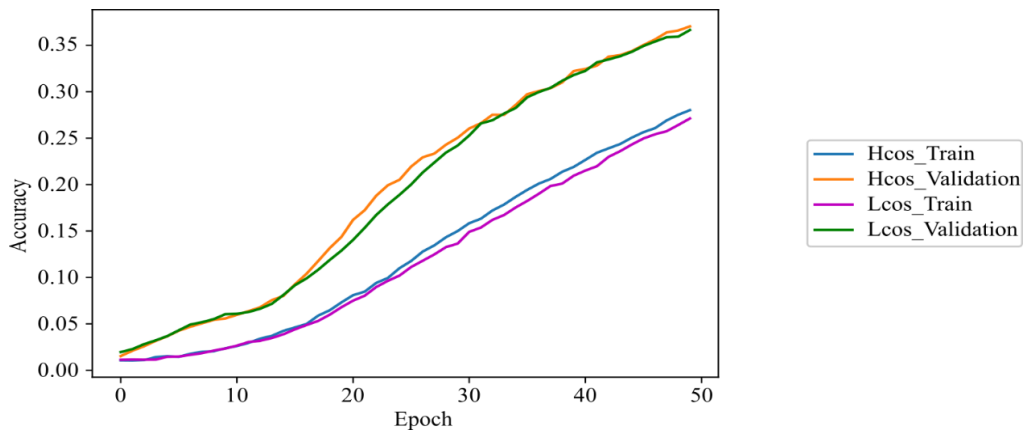
\*H; L represent Higher than Average and Lower than Average

In Table 4.7, the performance of the new models on various datasets still showed that the positive-negative weights and the positive weights outweighed the negatives. This pattern was very consistent with the previous performances. In Tables 4.4-4.7, the positives and the positive-negatives gave the best accuracy performance compared to the negatives.

From the result presented:

- a) Positive cosine methods gave 14 out of 16 best accuracies compared to negative ones.
- b) The method accuracy margins were between 1.01% to 3.15%
- c) Positive-negative cosine methods gave 10 out of 16 best accuracies compared to the negative cosine methods. The methods' accuracy margins were 9.69% to 10.58% over the negative cosine methods.
- d) Negative cosine methods gave only two best accuracies compared to the positive-negative or positive ones.
- e) The negative-weighted methods did not perform better in DKL comparisons than the positive-negative and positive-weighted DKL methods.
- f) The positive-negative DKL methods performed better than the positive and the negative DKL methods. Their accuracy margins were between 16.44% to 16.72%.

The cosine similarity methods targeted the higher layers, while the DKL methods utilized the lower and middle layers in the networks. This layer selection affected their performance, with the DKL methods improving performance. Figure 4.9 shows that high Cosine similarity among the layers influences the training and validation datasets without overfitting the new VGG16 model. The "Hcos\_Train" gave 26.58%, while the "Hcos\_Validation" gave a 38.64%. The low Cosine similarity validation (Lcos\_Validation) dataset also gave a higher value of 36.28% to the 25.21% for the low Cosine similarity training set.



**Figure 4.9: New VGG16 Accuracy Plot on CIFAR-100**

"Hcos" refers to higher positive-negative cosine similarity, and "Lcos" refers to lower positive-negative cosine similarity

When comparing the performance of the new models based on the datasets, the following was noted:

- a) The MNIST performed better than the CIFAR datasets. This performance was evident, with the CIFAR-100 performing poorly in all four new models. The poor performance in the models has also been reported by i) Bastidas Rodriguez et al. (2020), who noted that CIFAR-10 performed better than CIFAR-100 in new ResNet50 and new InceptionV3 models with standard deviations of 0.306 and 9.95, respectively. ii) In a study by Nagae et al. (2020), the CIFAR-100 still gave lower values than the other datasets. However, in this case, their models did not overfit.
- b) Similar to the previous studies reported in (i), CIFAR-10 performed better than CIFAR-100, which gave very low performance, as shown in Figure 4.9. However, a higher performance was noted when the epochs were increased.

#### **4.4.2 Comparison between the Selected Methods and Conventional Baseline Methods**

The baseline (commonly used) methods refer to those techniques that have or are being used in transfer learning tasks to evaluate the performance of the new models when used in new target tasks. The methods are essential in assisting researchers in assessing

the performance of their newly designed methods. The DKL methods were used in this section due to their superiority, reported in the previous section, over the cosine similarity methods. In Table 4.8, a comparison of the various baselines and the presented methods' accuracy (Acc) and loss is made on the new InceptionV3 model.

**Table 4.8: Comparison of DKL Accuracy (%) and Loss Using Selected Fine-Tuning Methods on the New InceptionV3 Model**

Method	Datasets							
	CIFAR-10		CIFAR-100		MNIST		Fashion-MNIST	
	Acc	Loss	Acc	Loss	Acc	Loss	Acc	Loss
DKL	77.28	0.65	22.68	3.48	98.18	0.06	87.90	0.33
Positive								
DKL	78.67	0.62	23.17	3.52	98.16	0.06	87.95	0.33
Negative								
DKL	77.73	0.64	23.16	3.47	98.1	0.06	87.81	0.33
Positive-Negative								
1 <sup>st</sup> Layer	66.26	0.99	12.06	4.01	91.66	0.29	80.71	0.54
Fine-tuning								
2 <sup>nd</sup> Layer	67.22	0.99	11.95	4.07	91.78	0.28	80.87	0.55
Fine-tuning								
3 <sup>rd</sup> Layer	66.62	0.99	10.85	4.14	92.12	0.27	79.94	0.57
Fine-tuning								
Feature	66.93	0.98	13.12	3.98	91.97	0.28	80.67	0.55
Extraction								

\* Acc represent Accuracy

From the presented results, the new InceptionV3 model gave a higher accuracy margin of at least 5% for the selected datasets with the proposed methods. The MNIST performed better than the Fashion-MNIST. Table 4.9 further shows the accuracy (Acc) and loss performance of the new techniques to the baselines in the new VGG16 model on the CIFARs and MNISTs datasets.

**Table 4.9: Comparison of DKL Accuracy (%) and Loss Using Selected Fine-Tuning Methods on the New VGG16 Model**

Method	Datasets							
	CIFAR-10		CIFAR-100		MNIST		Fashion-MNIST	
	Acc	Loss	Acc	Loss	Acc	Loss	Acc	Loss
DKL Positive	73.65	1.52	74.66	0.76	99.55	0.02	88.25	0.31
DKL Negative	73.94	1.55	74.96	0.77	99.30	0.02	88.10	0.32
DKL Positive-Negative	72.45	1.65	75.42	0.76	99.70	0.01	87.65	0.31
1 <sup>st</sup> Layer Fine-tuning	53.31	2.58	57.45	1.27	82.70	0.64	75.45	0.69
2 <sup>nd</sup> Layer Fine-tuning	58.04	2.34	61.35	1.16	89.75	0.36	77.60	0.58
3 <sup>rd</sup> Layer Fine-tuning	61.44	2.18	64.63	1.06	95.65	0.15	81.40	0.48
Feature Extraction	24.76	3.84	66.97	0.99	68.70	1.32	69.55	1.07

\* Acc represent Accuracy

In Table 4.9, the MNISTs performed better than the CIFARs for the new VGG16 pre-trained model. Furthermore, the CIFAR-100 was noted to adapt to the new VGG16 and even performed better than the CIFAR-10, which had been dominant in the other new models. Table 4.10 further shows the accuracy (Acc) and loss performance of the new DenseNet169 model.

**Table 4.10: Comparison of DKL Accuracy (%) and Loss Using Selected Fine-Tuning Methods on the DenseNet169 Model**

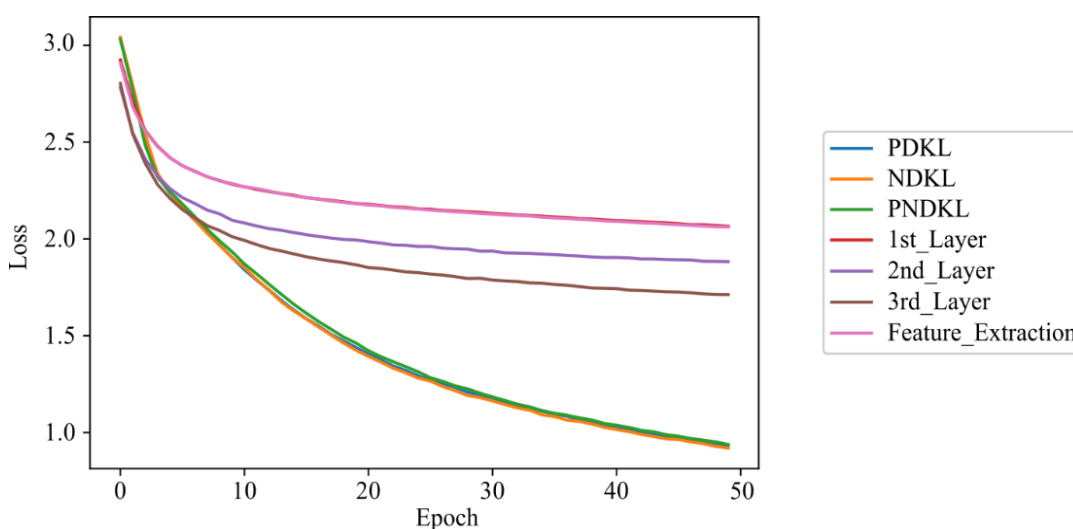
Method	Datasets							
	CIFAR-10		CIFAR-100		MNIST		Fashion-MNIST	
	Acc	Loss	Acc	Loss	Acc	Loss	Acc	Loss
DKL Positive	69.19	1.81	21.60	3.82	99.05	0.02	87.75	0.32
DKL Negative	69.02	1.84	22.17	4.02	99.01	0.02	88.20	0.32
DKL Positive-Negative	69.20	1.87	5.86	4.61	98.78	0.04	87.50	0.33
1 <sup>st</sup> Layer Fine-tuning	39.61	3.36	5.82	4.62	98.65	0.05	79.30	0.58
2 <sup>nd</sup> Layer Fine-tuning	40.74	3.17	6.60	4.59	98.60	0.05	79.20	0.58
3 <sup>rd</sup> Layer Fine-tuning	45.20	3.07	3.95	4.70	98.65	0.05	79.45	0.56
Feature Extraction	35.55	3.41	5.96	4.35	91.85	0.30	75.30	0.71

\* Acc represent Accuracy

In Table 4.10, the CIFAR-100 did not adapt well to this sizeable new model (new DenseNet169) and required an adjustment of the hyperparameters, for example,

additional epochs. However, the MNISTs (presented in Tables 4.8 – 4.10) adapted well to the new model, giving a 99.05% on the MNIST dataset. Again, the introduced methods provided better performance than the conventional methods. Among the traditional baseline methods, the  $k$ -3 method performed better, while feature extraction gave the least accuracy performance.

The conventional baselines and the introduced methods' accuracy performance can further be illustrated using Figures 4.10-4.12 below, with the positive-negative DKL that gave an excellent accuracy performance and a substantial loss of up to 0.985, as noted in Figure 4.10. Although the negative DKL (NDKL) showed the most negligible loss, the positive-negative gave the best accuracy of 56.34% compared to the NDKL's 53.89%. Among the baseline methods, the feature extraction gave the least loss of 2.345, while the best-performing 3<sup>rd</sup>\_Layer gave 1.789.

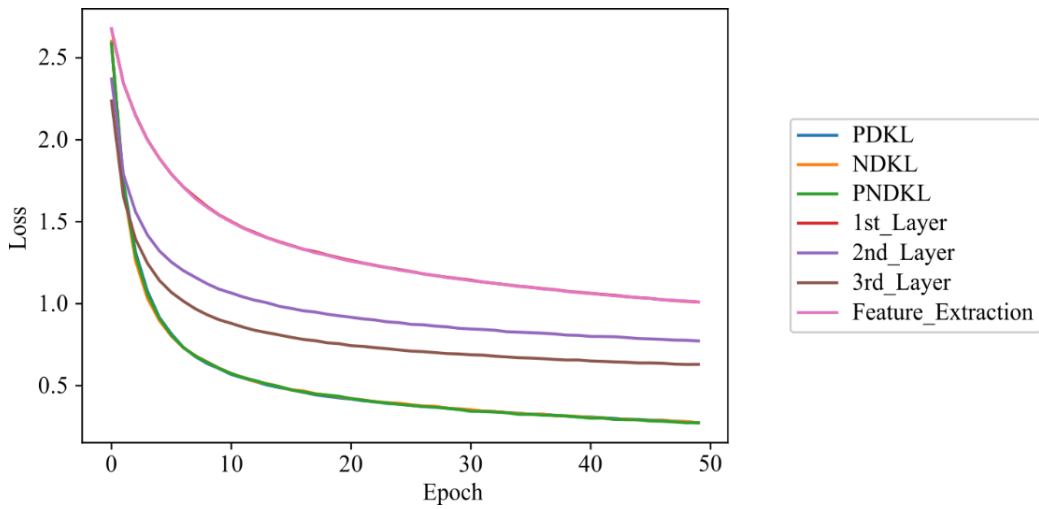


**Figure 4.10: Loss on New ResNet50 on CIFAR-10 Dataset**

The word “PDKL” refers to positive DKL, “NDKL” refers to negative DKL and “PNDKL” refers to positive- negative DKL

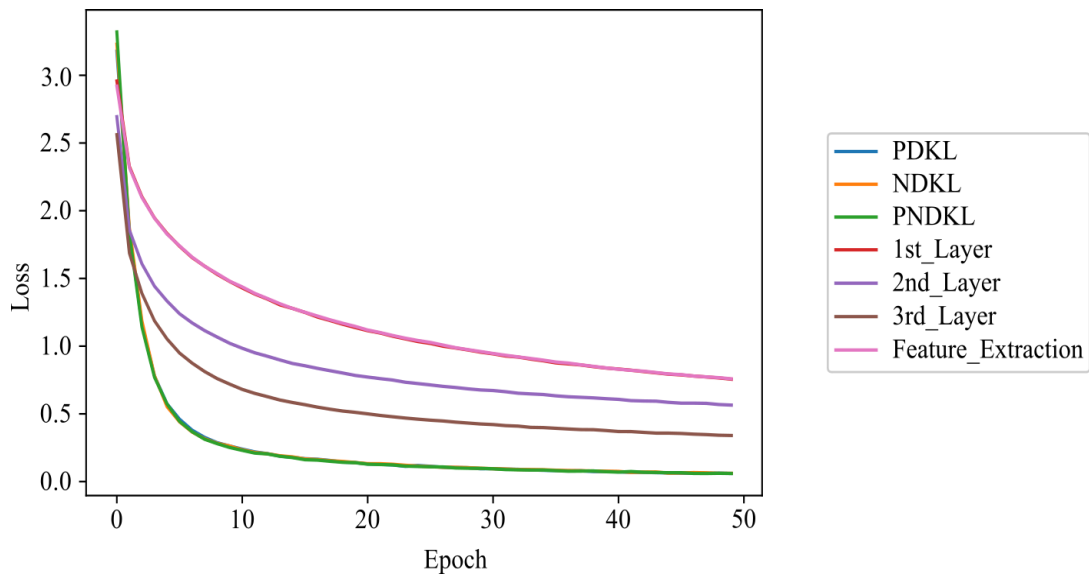
In figure 4.11, the three introduced methods show good model learning as the number of epochs increases. The baseline methods, especially the 2<sup>nd</sup>\_Layer and 3<sup>rd</sup>\_Layer, show a fast decline in the learning rate from the 20<sup>th</sup> epoch, resulting in a loss value of 0.869 and 0.729, respectively. However, the introduced methods still gave lower loss values, with the PNDKL giving the most negligible loss, 0.358, for the Fashion-

MNIST dataset.



**Figure 4.11: Loss of the New ResNet50 on the Fashion-MNIST Dataset**

In comparing the introduced methods and the conventional baselines, the approach selected the best-suited layers for fine-tuning. It is also evident that the traditional baselines gave the lowest results – as illustrated by Figure 4.12, where the PNDKL gave the best loss at 0.248 while the best baseline gave a loss of 0.943. This observation was presented in previous studies: Royer and Lampert (2020) used the Flex-tuning technique, which fine-tuned a network’s layers and its fully connected units; Guo et al. (2019) used the standard transfer learning method that only replaced the classification layer of the model, the L2-SP and random policy methods. In this study, the three methods gained accuracy margins of 15.52%-16.89% in the new ResNet50, 10.8%-11.74% for the new InceptionV3, 23.82%-24% for the new DenseNet169 and 11.01%-12.5% for the new VGG16 models.



**Figure 4.12: Loss of the New ResNet50 on the MNIST Dataset**

When comparing the conventional baselines, the feature extractor performed less accurately for the study’s new models. This performance supports the study by Guo et al. (2019). In further efforts to understand the impacts of weights in selecting the layers, regularization was added to the new models, and similar margins were observed. However, the regularization improved the new models’ performance, reduced the chances of overfitting, improved the model’s domain adaptation and gave stable model training (Vrbančič & Podgorelec, 2020).

#### 4.4.3 Evaluation of Dynamic Layer Selection Approach on New MobileNets

Understanding the dynamic layer selection method in the transfer learning process was vital to observe its effects on various architectures, including new mobile nets. MobileNets are relatively small compared to pre-trained models such as DenseNet169. In this study, two mobile nets, MobileNet and MobileNetV2, were considered. Table 4.11 shows the accuracy (Acc) and loss performance of the new MobileNet model on the various datasets using various methods.



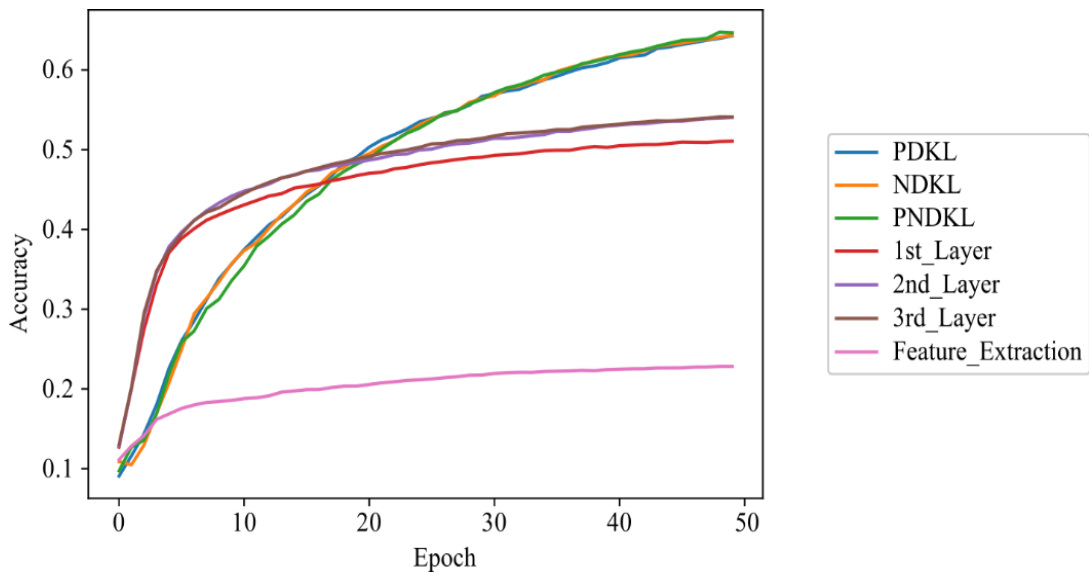
**Table 4.11: Comparison of DKL Accuracy (%) and Loss Using Selected Fine-Tuning Methods on the New MobileNet Model**

Method	Datasets							
	CIFAR-10		CIFAR-100		MNIST		Fashion-MNIST	
	Acc	Loss	Acc	Loss	Acc	Loss	Acc	Loss
DKL	70.26	1.72	29.67	3.95	99.24	0.03	90.24	0.45
Positive								
DKL	70.01	1.81	30.58	3.87	99.03	0.03	89.47	0.44
Negative								
DKL	71.21	1.79	19.27	4.01	99.70	0.06	89.36	0.39
Positive-								
Negative								
1 <sup>st</sup> Layer	45.29	3.21	18.74	4.52	97.63	0.07	81.29	0.55
Fine-tuning								
2 <sup>nd</sup> Layer	47.89	3.06	20.49	4.43	97.34	0.07	81.04	0.54
Fine-tuning								
3 <sup>rd</sup> Layer	50.45	3.23	14.96	4.97	97.56	0.08	80.98	0.52
Fine-tuning								
Feature	39.18	3.26	15.10	4.18	93.45	0.25	78.36	0.89
Extraction								

\* Acc represent Accuracy

In Table 4.11, the baselines gave lower values compared to the introduced methods, indicating more suitability and effectiveness in choosing suitable fine-tuneable layers.

The comparison of these approaches is also illustrated in Figure 4.13, where the accuracy of the introduced methods continues to increase during the learning process. The PNDKL gave the highest accuracy of 64.67%, while the best-performing baseline (3<sup>rd</sup>\_Layer) gave an accuracy of 54.12%.

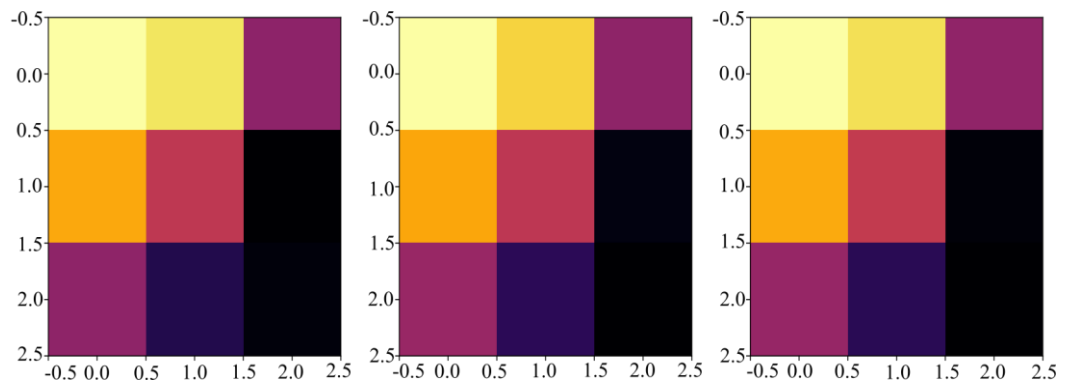


**Figure 4.13: CIFAR-10 on the New MobileNetV2**

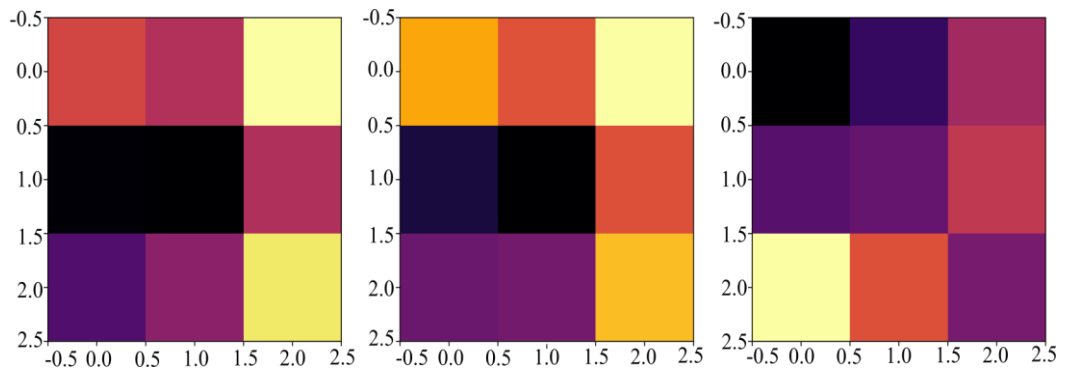
In the two illustrated Mobilenets instances, the new MobileNet gave an improved accuracy range between 7.2% for the MNIST dataset and 33.65% for the CIFAR dataset. The new MobileNetV2 also maintained high accuracy while utilising fewer parameters, facilitating its use in low-resource settings like mobile devices. These mobile nets are reliable, just like the new ResNets and the new VGGs, despite fewer parameters, as Praveen et al. (2021) noted.

#### **4.5 Investigation of New Model Pipeline (Textural Features Conflation and Dynamic Layer Selection)**

This section presents the pre-trained models' performance results using the selected samples and the dynamically selected layers. Additionally, non-selected samples were compared with the selected samples to validate the results of these models. Figure 4.14 below compares two feature maps to show a layer's weight behaviour when using samples with lower and higher DKL values. As seen in the illustration, layer 1 of the new model had lighter colours representing excitatory weights in the first channel of the layer. In comparison, darker colours representing inhibitory weights are noted in layer 9 of the new model. The learning difference is a clear indicator of improved learning in the first layer of the new model compared to the later layers, which are primarily concerned with classification purposes.



a) VGG16 - Layer 1, Channel 1 Weights



b) VGG16 - Layer 9, Channel 1 Weights

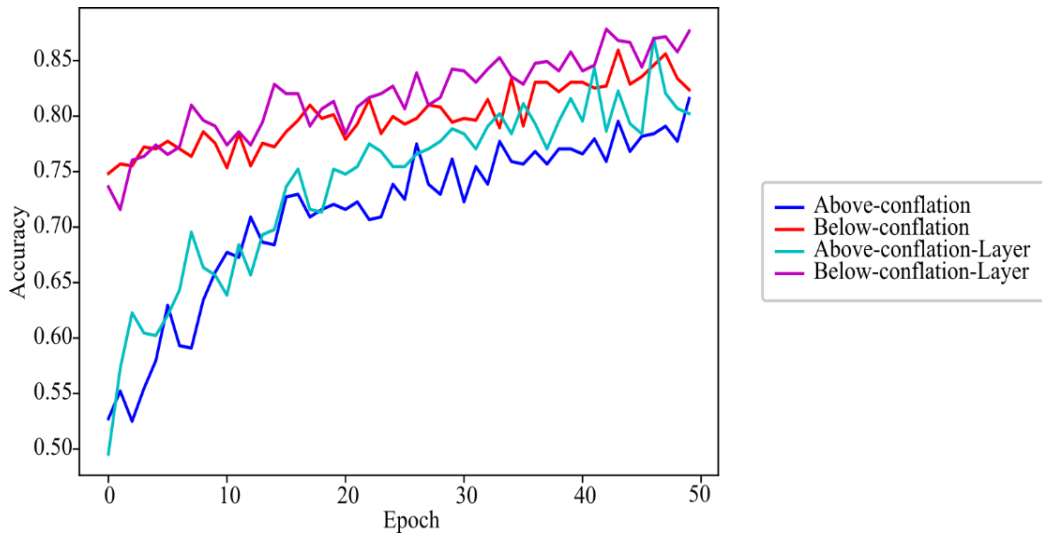
**Figure 4.14: New VGG16's Layers 1 and 9 Weights Visualization**

The tables (Tables 4.12 – 4.16) onwards show the performance of the various new models with and without the selected samples and the DKL methods. Table 4.12 shows the performance of the selected DKL methods on samples before and after conflation for the various datasets on the new ResNet50 model.

**Table 4.12: New ResNet50 Accuracy Performance (%) Using Selected DKL Methods**

Dataset	ResNet50			
	Positive		Negative	
	Before Samples	After Samples	Before Samples	After Samples
Caltech 256	97.04	98.15	96.23	96.25
MIT Indoor	96.64	97.24	95.98	96.12
Stanford Dogs	92.16	93.01	91.06	91.77
CIFAR10	54.18	55.29	53.58	53.89
CIFAR100	19.10	31.24	19.01	32.05
MNIST	98.20	99.27	98.25	98.69
Fashion MNIST	88.12	89.06	88.14	88.42
CRX8	82.96	83.46	80.20	81.11
Melanoma	78.68	79.36	77.23	77.85

From Table 4.12, selected conflated samples gave better results regardless of using both the positive and the negative-weighted Kullback-Leibler divergence methods. It is also worth noting that the positive-weighted Kullback-Leibler divergence methods performed better than the negative ones. The MNIST datasets gave the best results with the dynamic layers, while the CIFAR-100 gave the least performance. When the conflated data points were used, there was an improvement in the CIFAR-100. The new VGG’s performance in Figure 4.15 further illustrates the performance of the ISIC2016 dataset, with and without the conflated images (selected) and the dynamically chosen layers. Using conflated data points (with below) and dynamically selected layers gave an accuracy rate of 87.34% compared to just “Below\_conflation” without the selected layers, which gave the highest accuracy of 82.69%. A similar trend applies to “Above\_conflation,” only giving an accuracy of 78.34% for the combined method with 85.12%.



**Figure 4.15: New VGG16 Accuracy Comparison Plots between Conflated Samples Methods and Conflated Samples with Dynamically Selected Layers Methods**

As seen in Figure 4.15, the new VGG16 pre-trained model improved drastically for samples with below-average DKL and more when the dynamically selected layers were used in the fine-tuning. The results of the new InceptionV3 pre-trained model are presented in Table 4.13.

**Table 4.13: New InceptionV3 Accuracy Performance (%) Using Selected DKL Methods**

Dataset	InceptionV3			
	Positive		Negative	
	Before Samples	After Samples	Before Samples	After Samples
Caltech 256	93.24	94.02	92.14	92.34
MIT Indoor	89.42	90.23	87.24	88.34
Stanford Dogs	88.36	91.05	88.09	89.02
CIFAR10	77.58	78.29	78.47	79.11
CIFAR100	30.59	34.12	28.98	32.21
MNIST	98.09	99.08	98.01	98.78
Fashion MNIST	87.53	88.24	87.45	88.03
CRX8	80.48	81.45	78.32	79.38
Melanoma	85.36	85.98	81.39	82.35

From Table 4.13, the fine-tuned dynamically selected layers improved the new model's performance when the below-average samples were used, with the MNIST performing

better than the CIFARs. It is also noted that the smaller datasets - ISIC2016 and ChestX-ray8 performed very well. Table 4.14 further presents the performance of these methods and samples on the new DenseNet169 model.

**Table 4.14: New DenseNet169 Accuracy Performance (%) Using Selected DKL Methods**

Dataset	Kullback-Leibler Divergence Methods			
	Positive		Negative	
	Before Samples	After Samples	Before Samples	After Samples
Caltech 256	91.54	92.69	90.47	90.65
MIT Indoor	87.28	88.90	86.88	86.82
Stanford Dogs	86.65	87.35	84.07	84.98
CIFAR10	69.18	72.49	69.05	71.56
CIFAR100	34.58	42.15	32.15	41.49
MNIST	99.02	99.52	98.89	98.94
Fashion MNIST	87.69	88.14	87.12	88.01
CRX8	78.24	79.61	77.36	77.58
Melanoma	81.36	82.35	79.35	81.44

From Table 4.14, the new DenseNet169 pre-trained model showed a similar trend, with CIFAR-100 giving the least accuracy among the datasets. The MNISTs still had the best adaptation, as seen in their previously recorded accuracy performance in Table 4.13. Table 4.15 further presents the performance of the selected methods and samples for the new MobileNetV2.

**Table 4.15: New MobileNetV2 Accuracy Performance (%) Using Selected DKL Methods**

Dataset	Kullback-Leibler Divergence Methods			
	Positive		Negative	
	Before Samples	After Samples	Before Samples	After Samples
Caltech 256	91.45	92.34	93.65	94.04
MIT Indoor	86.24	87.95	86.02	86.39
Stanford Dogs	88.56	88.96	87.34	87.58
CIFAR10	64.21	66.20	64.11	65.14
CIFAR100	29.99	32.41	29.38	30.04
MNIST	97.98	98.56	97.90	97.87
Fashion MNIST	87.35	88.69	87.04	88.96
CRX8	77.53	79.34	67.08	70.12
Melanoma	76.40	78.15	73.24	73.69

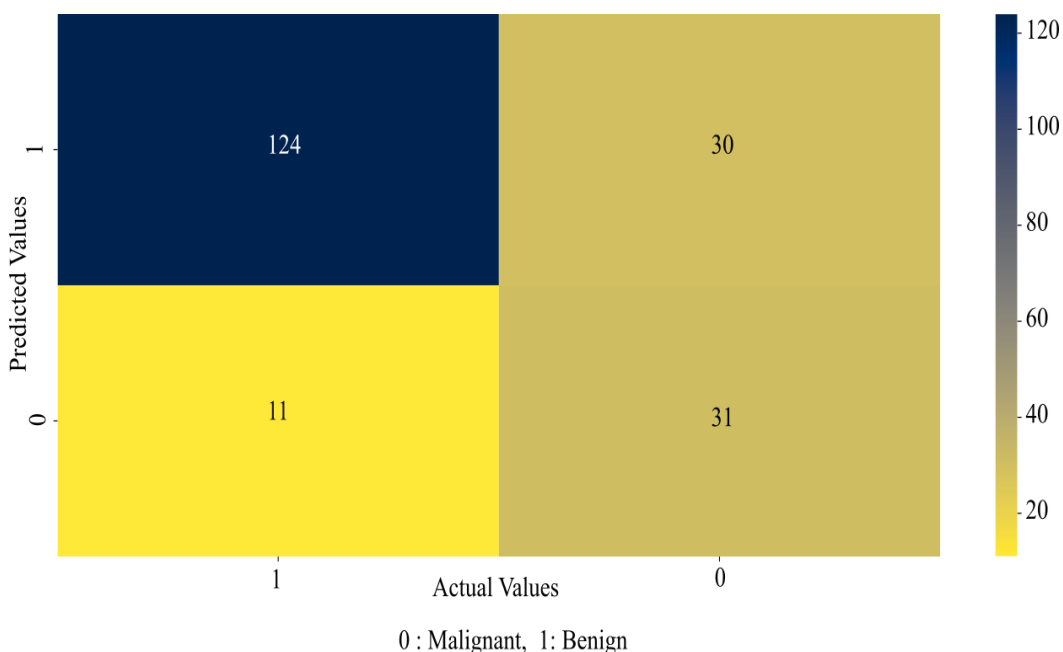
In using the new MobileNetV2 pre-trained model, the evaluated datasets behaviour was similar to the already reported pre-trained models for the CIFARs and the MNISTs - this was possible as a result of using the below-average DKL samples and the dynamically selected layers of the new MobileNetV2. The precision performance of the new MobileNetV2 was used to validate the accuracy performance, as shown in Table 4.16.

**Table 4.16: New MobileNetV2 Precision Performance (%) Using Selected DKL Methods**

Dataset	Kullback-Leibler Divergence Methods			
	Positive		Negative	
	Before Samples	After Samples	Before Samples	After Samples
Caltech 256	90.34	91.85	93.15	93.88
MIT Indoor	86.02	87.36	85.85	85.74
Stanford Dogs	89.01	90.37	86.48	86.95
CIFAR10	65.23	67.92	64.06	66.24
CIFAR100	30.08	33.74	29.04	29.51
MNIST	98.36	98.87	96.47	96.94
Fashion MNIST	88.41	90.21	88.14	89.38
CRX8	78.1	80.39	66.47	69.54
Melanoma	77.63	79.54	73.12	74.85

The precision performance of the models was used to indicate the repeatability of obtaining good performance for pre-trained models. In Table 4.16, the precision values were not very far off the accuracy values, validating the accuracy performance

from the samples on the pre-trained model. In Figure 4.16, a confusion matrix further illustrates the validity of the results on the ISIC2016 testing dataset. The model can correctly classify the positive class data points (124) and the negative ones (31).

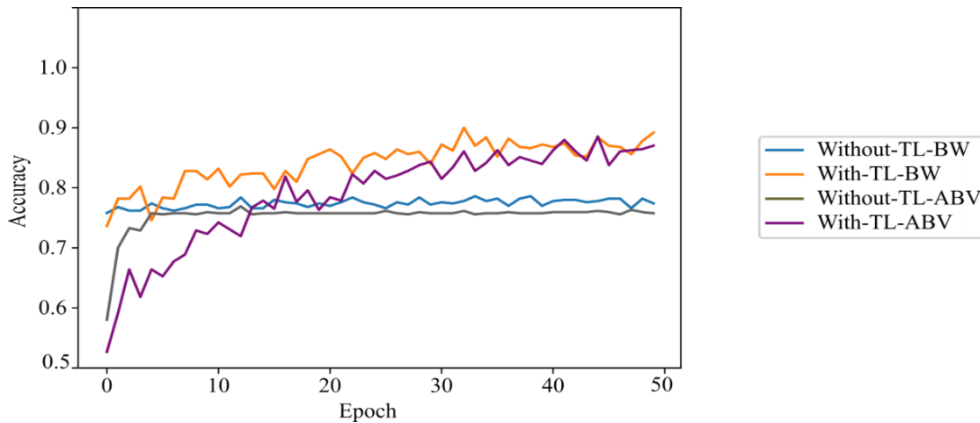


**Figure 4.16: New MobileNetV2 Confusion Matrix on ISIC 2016**

#### 4.5.1 Evaluation of the Proposed Methods against Conventional Baseline Methods

Combining the two methods was compared to the conventional methods: standard fine-tuning,  $k-1$ ,  $k-2$ , and  $k-3$ . The use of conflated data points and dynamically selected layers in a new model allowed the average accuracy performance to improve by 0.87%, with the new DenseNet169 giving the most improved accuracy of 1.57% and the new VGG16 giving a slight improvement of 0.06% compared to the standard fine-tuning for the ChestX-ray8 dataset. Figure 4.17 illustrates the accuracy performance of samples without transfer learning and with transfer learning. The model gives an accuracy value of 85.27% when the transfer learning is done with the selected layers and when using the chosen samples, compared to 84.63% when using all the samples in the dataset. The performance is even lower when using the standard transfer learning in either case, but the selected samples still perform better at 77.78% to 75.56% when using all the samples.





**Figure 4.17: New VGG16 Accuracy Comparison between Methods with and without Transfer Learning (TL) on ISIC 2016**

BW Refers to Below-Average DKL, and ABV Refers to Above-Average DKL

From the results illustrated in Figure 4.17, the below-average DKL not only performed better than the above-average method but also took less time (50 seconds) to train on ISIC2016 samples than a typical convolutional neural network that was used to ascertain the improvements from the use of pre-trained models. The typical convolutional neural network was a 12-layer CNN. Table 4.17 presents the performance of the ChestX-ray8 samples on the various methods and new models.

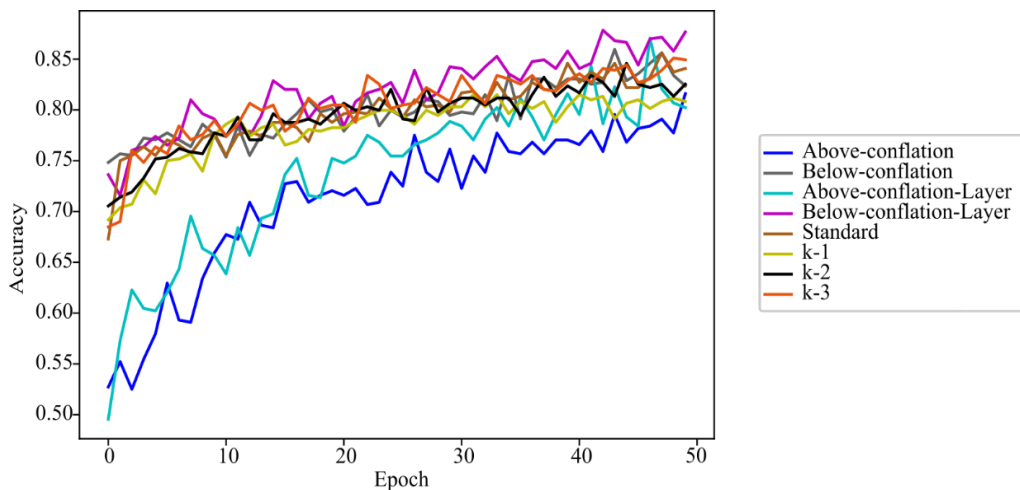
**Table 4.17: Proposed Methods and Commonly Used Methods Accuracy Performance (%) on ChestX-Ray8 (Homogeneity)**

Method	New Models				
	ResNet50	VGG16	InceptionV3	DenseNet169	MobileNetV2
Below DKL	84.34	72.14	84.15	81.26	77.25
Above DKL	83.97	68.90	81.20	78.50	72.45
Positive DKL	82.96	85.38	80.48	78.24	77.53
Negative DKL	80.20	82.30	78.32	77.36	67.08
Positive Above DKL	+83.46	86.44	81.45	79.61	79.34
Standard fine-tuning	83.08	86.38	80.05	78.04	78.40
Last k-1	80.14	82.54	78.98	77.39	78.18
Last k-2	82.26	84.03	79.59	77.65	79.04
Last k-3	82.98	84.84	79.56	77.84	79.12

From Table 4.17, the combined methods of the dynamically selected layers positive DKL methods and the use of conflated below-average DKL gave an average improvement of 0.87%, with the new VGG16 pre-trained model giving the slightest

improvement of 0.06%.

The  $k$ -3 method was the most improved among the conventional techniques, with its starting training accuracy of 68.46%, as shown in Figure 4.18. Combining these two methods seemed to work well across all the baseline methods. It is also important to note that the below-average methods using the conflated samples only were the second best methods after the combined methods, highlighting the importance of selecting the quality samples in the datasets. The model gives an accuracy value of 85.27% with the selected layers (Below-conflation-Layer) and when using the chosen samples compared to 84.63% when using all the samples in the dataset. This performance is lower for the baseline methods with the standard fine-tuning, which gives the best accuracy of 84.75%.



**Figure 4.18: New VGG16 Accuracy Comparison between Proposed Methods and Commonly Used Methods on ISIC 2016**

#### 4.6 Investigations on Computational Complexities of the New Model

The computational complexity in the proposed approach was evaluated from two views: the selection of samples and the dynamic selection of the layers. The complexity looked at the time taken to complete a computational process and the memory taken.

##### 4.6.1 Evaluation of Conflation Approach Computational Complexity

In the computation of the conflation of textural features, the time taken in the process

and memory used in the process were measured in milliseconds and bits, respectively. Table 4.18 shows the computation complexity of the conflation process on a new VGG16 pre-trained model using the LBP method.

**Table 4.18: Comparison of Complexity Distances on New VGG16 Using LBP**

Dataset	Divergence Measures									
	DKL		Wasserstein		Hellinger		Jensen-Shannon		Bhattacharya	
	T	Mem	T	Mem	T	Mem	T	Mem	T	Mem
Caltech256	6.02	12262	6.18	12262	6.16	12262	6.18	12262	6.20	12262
MIT Indoor67	6.08	12262	6.15	12262	6.22	12262	6.31	12262	6.36	12262
Stanford Dogs	6.24	12262	6.29	12262	6.29	12262	6.28	12262	6.34	12262

\*T; Mem represents Time (s) and Memory (bits)

As noted in the table, the Bhattacharya divergence metric gave the highest complexity compared to the other divergences. The complexity was based on the conflation of features of one target sample and its comparison to the source samples. The Jensen-Shannon came second, while the DKL was the least computationally complex divergence measure. From the table, it can also be noted that the memory complexity was similar for all the divergence measures. The computational complexity was also evaluated using a single target data sample using the new MobileNet pre-trained model, as shown in Table 4.19.

**Table 4.19: Comparison of Complexity Distances on New MobileNet Using GLCM Correlation**

Dataset	Divergence Measures									
	DKL		Wasserstein		Hellinger		Jensen-Shannon		Bhattacharya	
	T	Mem	T	Mem	T	Mem	T	Mem	T	Mem
Caltech 256	6.81	12262	6.97	12262	6.93	12262	7.01	12262	7.10	12262
MIT Indoor 67	7.02	12262	7.04	12262	7.22	12262	7.22	12262	7.16	12262
Stanford Dogs	6.91	12262	7.09	12262	7.11	12262	7.20	12262	7.21	12262

\*T; Mem represents Time (s) and Memory (bits)

In the new MobileNet pre-trained model, the same trend of Bhattacharya gave the highest computational complexity. It is worth noting that DKL did not provide a very significant time complexity margin over the other divergences per sample, but this scale increases as the samples increase.

#### 4.6.2 Evaluation of Dynamic selection Approach Computational Complexity

In the dynamic selection of the pre-trained model’s layers, the computational complexity between DKL and other divergence measures was compared to understand the reason behind the choice of DKL in this study. Table 4.20 shows the various divergences when using a CIFAR-10 dataset sample.

**Table 4.20: DKL Complexity Comparison to Other Divergence Measures**

Model	Divergence Measures									
	DKL		Hellinger		Jensen-Shannon		Bhattacharya		Wasserstein	
	T	Mem	T	Mem	T	Mem	T	Mem	T	Mem
MobileNet	3.33	40102	4.29	141018	1.20	76520	11.94	49537828	2.05	618209
MobileNetV2	1.92	40323	2.68	193214	0.51	129121	6.20	716486	0.84	916694
ResNet50	32.89	85444	48.47	370086	5.25	232424	120.36	1450276	15.86	1811638
VGG16	0.11	13106	0.18	51806	0.05	32183	0.22	176199	0.08	255810
Inception V3	33.84	90332	66.91	656161	5.35	406236	194.11	1627010	17.89	3224079
DenseNet169	56.25	125025	96.24	4212014	110.69	797576	219.56	3501420	508.47	6202006

\*T; Mem represents Time (s) and Memory (bits)

As shown in Table 4.20, the DKL could balance the time and memory computational complexities over the other divergences. It was also noted that the new DenseNet169 took 143 seconds, while the new VGG16 took 34 seconds to use CIFAR-10 when selecting the trainable layers. This time difference signified a higher computational complexity in the large pre-trained models.

#### 4.6.3 Evaluation of New Model’s Pipeline (Merged Proposed Methods) Computational Complexities

This section looks into the combined computational complexity of the conflation process and the dynamic selection. In Tables 4.21 and 4.22, a sample was taken through the conflation and dynamic processes in a classification task for the Caltech256 dataset. Table 4.21 shows the computational complexities of a Caltech256 data point on various divergences and new models during conflation.

**Table 4.21: Divergence Measures Computational Complexity Using Conflation Approach on a Sample Caltech256 Data Point**

Model	Divergence Measures									
	DKL		Wasserstein		Hellinger		Jensen-Shannon		Bhattacharya	
	T	Mem	T	Mem	T	Mem	T	Mem	T	Mem
ResNet50	37.48	104137	13.96	30297	51.16	376325	6.51	238501	118.78	1159044
InceptionV3	36.83	28151	18.35	44451	67.74	672025	7.10	414769	164.84	1639686
MobileNet V2	14.57	40900	3.65	39695	4.67	40527	4.12	40475	3.66	37805
VGG16	5.34	25315	3.75	269011	3.98	384880	3.66	187836	3.88	187836
DenseNet169	908.15	320543	1174.06	632111	1768.06	1262959	78.93	787334	1836.14	1862471

\*T; Mem represents Time (s) and Memory (bits)

Table 4.21 shows the DKL performed lower than the Wasserstein but better than the other methods. However, it also gave a good balance when working with large models such as the DenseNet169. Table 4.22 further shows the complexity results of a sample Caltech256 data point in the dynamic layer selection method.

**Table 4.22: Divergence Measures Computational Complexity Using Dynamically Selected Layers Approach on a Sample Caltech 256 Data Point**

Model	Divergence Measures									
	DKL		Wasserstein		Hellinger		Jensen-Shannon		Bhattacharya	
	T	Mem	T	Mem	T	Mem	T	Mem	T	Mem
ResNet50	32.89	85444	15.86	1811638	48.47	370086	5.25	232424	120.36	1450276
Inception V3	33.84	90332	17.89	3224079	66.91	656161	5.35	406236	194.11	1627010
MobileNetV2	1.92	40323	0.84	916694	2.68	193214	0.51	129121	6.20	716486
VGG16	0.11	13106	0.08	255810	0.18	51806	0.05	32183	0.22	176199
DenseNet 169	56.25	125025	508.47	6202006	96.24	4212014	110.69	797576	219.56	3501420

\*T; Mem represents Time (s) and Memory (bits)

In the dynamic selection of layers, the DKL maintained a better balance than Wasserstein in all the new models with lower memory complexities. From Tables 4.21 and 4.22, the DKL complexities were much lower than Bhattacharyya and Hellinger, which took 239.14ms and 99.63ms on the new ResNet50 model, respectively. The Jensen-Shannon and Wasserstein took less computational time when DKL took 70.37ms. However, they took more computational resources, making DKL ideal among the divergence measures; it had a better balance of time and memory.

#### **4.7 Discussion on the Selection of Target Samples using Textural Features Conflation**

The first phase of the model introduced the use of conflation of features. The method involved merging features probability distributions and comparing them using the Kullback-Leibler divergence, described and illustrated in section 3.3. From the two features analysis methods, LBP and GLCM, it was clear that samples with lower Kullback-Leibler divergence performed better than samples with high Kullback-Leibler divergence. The homogeneity and correlation of GLCM's properties gave the best performance in Table 4.1. This performance was due to similar grey levels in the neighbouring pixels and the GLCM elements in the pixel's diagonals. From the results, these varied between 1.96% and 6.18%. The homogeneity values were noted to be better in the edges where the density and the distances between the textural patches were lower. As Chaves (2022) noted, this trend of elements in the main diagonal changes smoothly. The elements away from the main diagonal are less critical, so their homogeneity is lower. Tables 4.1 and 4.2 present correlation performance that aids in determining better sensitivity and validates the classifier's accuracy.

As noted in Table 4.2, the GLCMs performed better than the LBP property of the sample's textural features. The homogeneity was reported to match the energy property values (Kurniati et al., 2024) in matching the simplicity and uniformity of the textural features. The performance of GLCM properties is also noted in Table 4.3, where better values for energy and correlations were achieved. The recall values show the identification of relevant data points in the dataset by the model (Al-Abboodi et al., 2024). Other works have also reported such performance (Changwei et al., 2020; Kabir et al., 2024). The GLCM properties were also shown to perform better than the CNN-based descriptors since they are less discriminating in selecting features in the grey levels (Iqbal et al., 2021). In the reported instance, the GLCM performed very well, improving a CNN's model performance when using a limited dataset, a common occurrence when performing transfer learning.

The GLCM's properties helped the model to learn the data patterns much better (Iqbal et al., 2021) by selecting the samples with suitable features - those with lower Kullback-Leibler divergence. The chosen samples had lower informational differences

than those with higher Kullback-Leibler divergence. The GLCM properties were better in the selection process than the LBP features descriptor. Among the GLCM properties, the dissimilarity provided the least values due to the sharp changes of the grey levels in the pixels' diagonals; when there were abrupt grey level changes, the contrast (dissimilarity) was considered to be high, and the features change was higher compared to the slant diagonals.

Apart from the comparison in performance between the GLCM and LBP described features used in the pre-trained model, the divergence measures used in the conflation approach showed that the Bhattacharya and the Wasserstein performed better than the Kullback-Leibler divergence. However, they also gave very high computational complexities, presented in tables 4.21 and 4.22, with the Bhattacharya taking 0.244 milliseconds for new MobileNetV2 with the GLCM properties forming the basis of using the DKL in selecting the target samples. When the conflation approach was used with the baseline transfer learning methods for the selected new models, it was noted that an improvement ranging from 2.51% and 9.15% was achieved for the standard method and the other methods, respectively. The feature extraction method improved the least among the baseline methods due to freezing parameters, which hindered the new model from learning new patterns in the data and could not suit the features' domain shift.

#### **4.8 Discussion on the Dynamic Selection of New Model Layers for Fine-Tuning**

The dynamic layer selection process aimed to select the most suitable layers for participation in fine-tuning and, consequently, good new model performance. This study looked at the layers based on the similarity and the divergence measures and picked the divergence measures as the best methods to use in the selection process. The divergence was based on the weights in the layers, which was compared using the Kullback-Leibler divergence, and the process is described in section 3.4. The layers whose weights presented low Kullback-Leibler divergence were selected and were mainly at the higher end of the models since the bottom layers are primarily used in the feature extraction of the given samples and involve freezing of parameters, which often leads to poor adaptation. The performance of these selections is presented in Tables 4.8 to 4.11, and feature extraction gave the least performance.

Additionally, the improved adaptation with the dynamic selection of fine-tunable layers gave better model stability, as evident in figures 4.13 to 4.15, with gradual loss without overfitting. The positive weights in the layers were considered excitatory, a property that made the selection of features much better (with improved stimuli) (Montesinos López et al., 2022). The positive weights in the filters converged the gradient descent in the training process much faster, ensuring better learning of the sample features (Wang et al., 2023).

Furthermore, apart from other factors such as the learning rate, momentum, and error surface, the weights have been shown to significantly affect the gradient descent process (Hassan et al., 2023). The gradient descent was updated through a reduction or increase of the direction of the weights when the previous and the current derivatives had opposite weight signs - typically, the positive weight indicates a similar direction of the learning parameters (learning rate, momentum, error surface) for convergence. The weight signs change due to the learning process's temporal behaviours to the minimum. The negative weights in the learning process are inhibitory and aim to diverge and correct the gradient descent and control the effects of the model learning rate (Hassan et al., 2023). The gradient can affect the magnitude and the weight sign. The negative weights are predominant or consecutive when a layer's nodes do not contribute much to the learning process. This negligible contribution happens as the minor weight adjustments during the learning process result in small gradient changes and error curves, necessitating minor modifications to the learning rate (Lamjiak et al., 2024).

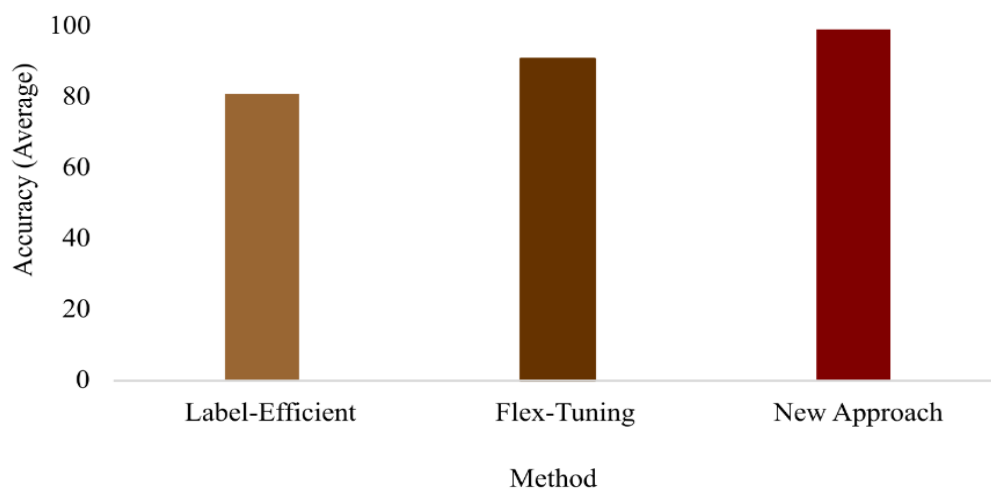
During a model's learning process, the positive weights are used when the current derivative and the previous derivatives are in the same direction; otherwise, the weights are decreased (with a negative sign) to ensure the model's stability and to prevent further divergence from the global minima (convergence) (Zhang et al., 2023). This behaviour explains the influence of the negatively signed weights in this study and the slow or lower performance compared to the positively signed weights. Combining the positive and the negative weights ensures a balance in the performance, creating stability. However, the performance is not the best due to taking too long to reach the model's convergence (minima), but with better results than the inhibitory weights.



Even for the new MobileNets, the models' dynamic choice of suitable layers gave good performance irrespective of their small number of parameters. The new MobileNetV2 facilitates transfer learning in low-resource environments like mobile devices while maintaining good pre-trained model performance. These mobile nets comprise convolutional layers broken into depthwise separable and pointwise convolutional layers. The mobile nets also have linear bottlenecks in inverted residuals. The depthwise part of the convolutional layer can perform light filtering using single convolutional filters in the channels. In contrast, the pointwise part creates features for the input channels using linear combinations. This part helps reduce the input channels required (Zhu et al., 2024).

#### 4.9 Comparison to Existing Methods

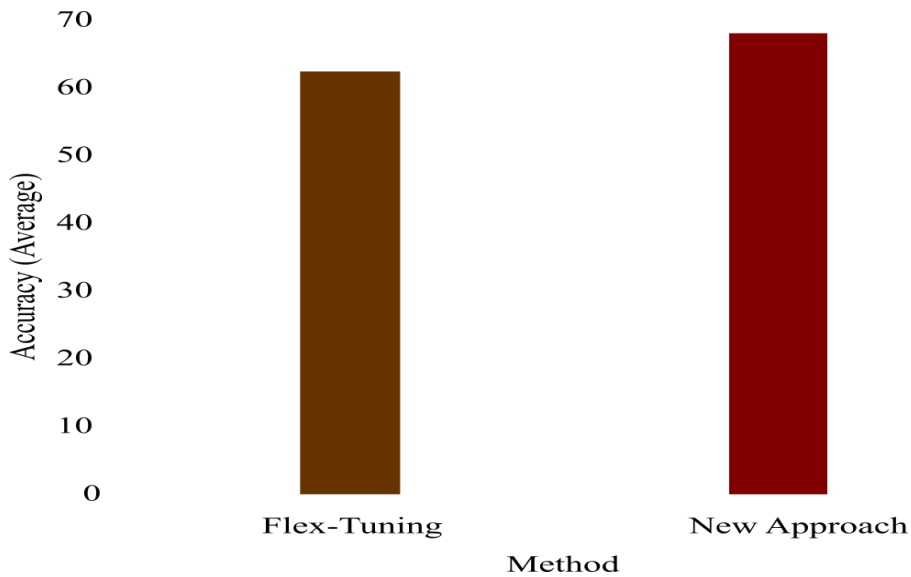
The new model's performance was compared against state-of-the-art methods on the MNIST, MITIndoor 67, Stanford Dogs 120, Caltech256, CIFAR10 and CIFAR100 datasets. The performances are shown in figures 4.19 to 4.23 below. In Figure 4.19, the new approach has an average accuracy of 98.1%, while the Flex-tuning and Label-Efficient methods have an average accuracy of 90.8% and 81%, respectively. The introduced method outperformed both methods by up to 7.3% improvement over the Flex-Tuning, which could result from the positive learning process introduced by the positive weights in the selected layers.



**Figure 4.19: MNIST Dataset Accuracy Performance in Various Methods**

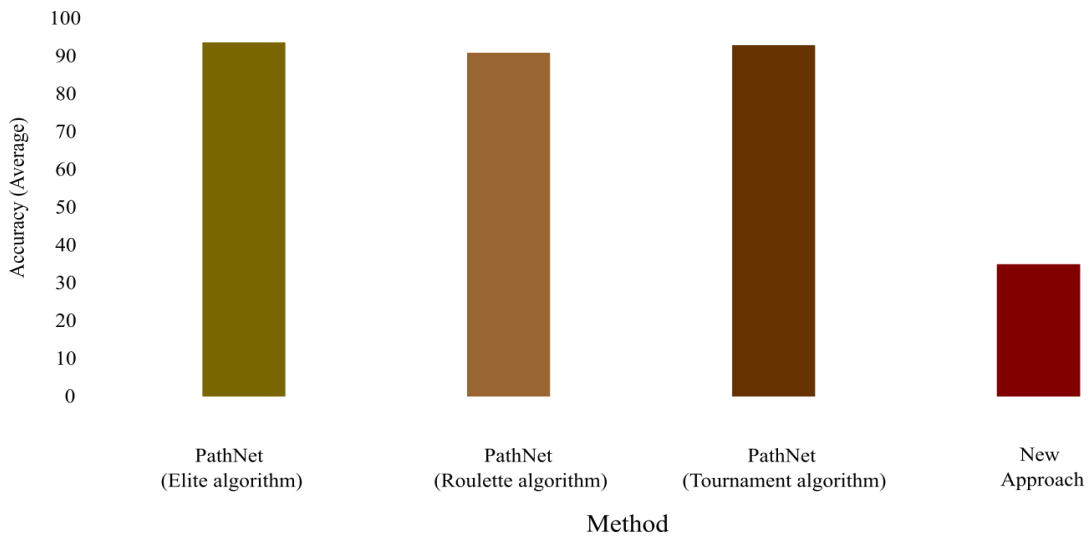
Compared to the methods in the CIFAR10 dataset, the introduced method gave an

average accuracy of 68.07% compared to 62.4% for the Flex-Tuning, as illustrated in Figure 4.20 for the various models. The 5.67% average difference can be attributed to positive weights that lead to faster model convergence.



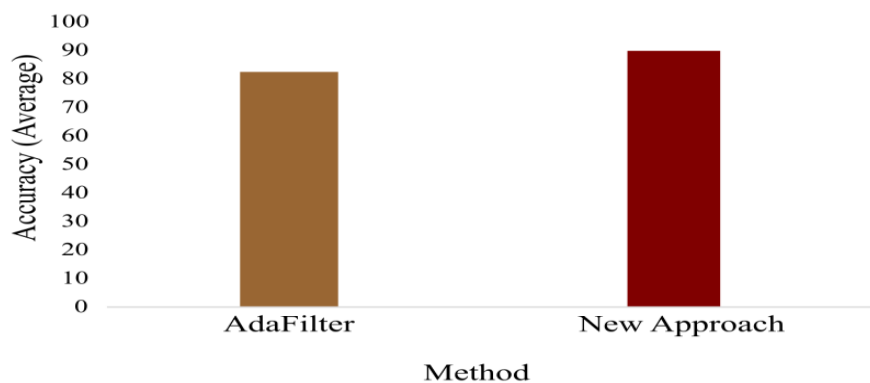
**Figure 4.20: CIFAR10 Dataset Accuracy Performance in Various Methods**

For the CIFAR100, the new approach gave an average accuracy performance of 34.98% compared to 93.7%, 90.9% and 92.9% for the PathNet Elite, Roulette and Tournament algorithms, respectively. The introduced method was performed dismally compared to the three PathNet genetic algorithm variants, as noted in Figure 4.21. However, this was due to using small data samples and only five epochs in three genetic algorithms compared to the introduced method, which needed more epochs for the data points to fit the new models.



**Figure 4.21: CIFAR100 Dataset Accuracy Performance in Various Methods**

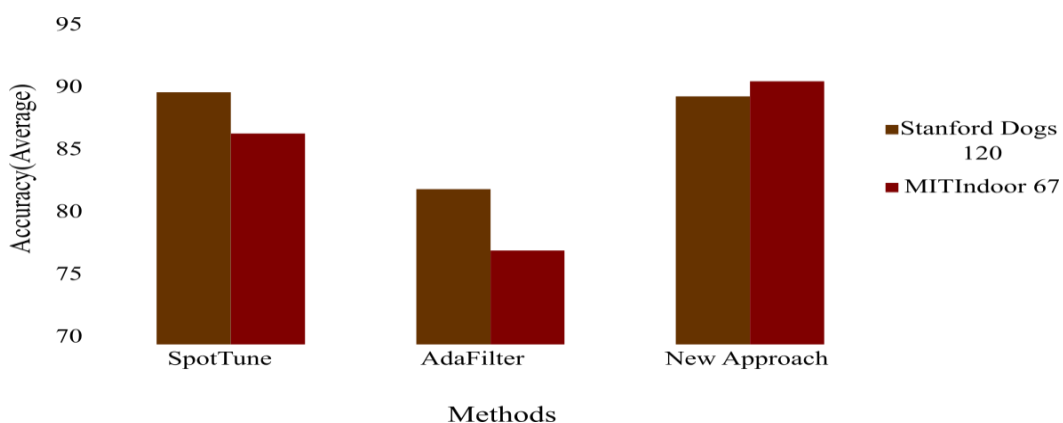
In Figure 4.22, the new approach is compared with the AdaFilter method, which shows a similar trend in good performance. The new approach gave an average accuracy performance of 89.88% compared to an average accuracy of 82.44% for the AdaFilter method in the Caltech256 dataset. This average accuracy represents an average improvement of 7.44%. The difference could result from using RNNs in the AdaFilter, which could not be adopted very well with the CNN, and the added advantage of selecting the filters with the most positive weights for the introduced approach.



**Figure 4.22: Caltech256 Dataset Accuracy Performance in Various Methods**

In comparing the new approach with the other methods in the MITIndoor 67 and Stanford Dogs 120, the new approach was better by up to 4.18% and 7.44%,

respectively, for the AdaFilter. The new approach gave an average accuracy of 89.88%, while the SpotTune and AdaFilter gave 90.2% and 82.44%, respectively, in the Stanford Dogs 120. For the MIT dataset, the new approach gave 91.08%, while SpotTune and AdaFilter gave 86.9% and 77.53% respectively. However, SpotTune performed better than the new approach for the Stanford Dogs 120, as noted in Figure 4.23, due to its image-specific fine-tuning method (Guo et al., 2020). The excellent performance was the use of the positive weights compared to the use of softmax sampling (for SpotTune) and the RNN (for AdaFilter), which does not address the low-level features targeted in the new approach



**Figure 4.23: MITIndoor 67 and Stanford Dogs 120 dataset Accuracy Performance in Various Methods**

From figures 4.19 to 4.23, the new approach performs better than the other approaches apart from the CIFAR10 dataset since it had broken down into smaller datasets and therefore required more epochs in our study to get closer to their results and the SpotTune method.

#### 4.10 Discussion on the Computational Complexity of the New Model

Apart from the comparison in performance between the GLCM and LBP described features used in the new models, the divergence measures used in the conflation approach showed that the Bhattacharya and the Wasserstein performed better than the Kullback-Leibler divergence. However, they also gave very high computational complexities, presented in tables 4.21 and 4.22, with the Bhattacharya taking 0.244 milliseconds for new MobileNetV2 with the GLCM properties and 0.192 milliseconds

when using the LBP described samples in the new VGG16 model. The complexity may not seem high for one sample, but it is complex when the number increases, which makes DKL suitable for selecting the samples used in the fine-tuning process. The conflation algorithm has  $O(n \log(n))$  complexity, which is higher when the number of samples is high. For example, for a Stanford Dogs 120 images, the conflation process takes 4.906 milliseconds on top of 78780 milliseconds for one image, which can quickly compound for a large dataset. However, the complexity is also subject to other factors, such as hardware specifications (memory and processing speed) (Sarker, 2021) When the conflation approach is used with the baseline transfer learning methods for the selected new models.

The heavy models in this study for the dynamic selection of new model layers were noted to give lower accuracy performance than the other models due to their complexities (large number of parameters and layers). The DenseNet169 is an example of a pre-trained model with 169 layers compared to the VGG16, which has just 16 layers. Larger datasets such as the CIFAR-100 also recorded lower performance compared to smaller datasets; CIFAR-100 gave lower accuracies when trained for 50 epochs, but an increase in this number gave better results, but this came with higher computational complexity for the model to identify the right features in the data. The DEFT researchers also noted these issues (Vrbančič & Podgorelec, 2020).

#### **4.11 Summary**

This chapter has presented the results of the new approach: results on the conflation of textural features and dynamic selection of fine-tunable layers. In the new approach, it is clear that informational divergence is a crucial element that could result in lower domain adaptation in the transfer learning process. The new approach was tested on various datasets and pre-trained models as source domain models, as evident in tables 4.12 to 4.16, and was effective in improving domain adaptation. It effectively improved domain adaptation and performed well against baselines and state-of-the-art methods.

## CHAPTER FIVE

### SUMMARY, CONCLUSIONS AND FUTURE WORK

#### 5.1 Dynamically Fine-Tuned Model Summary

This study aimed to develop a dynamically fine-tuned model that addresses the conflation of data samples used in the transfer learning tasks and the fine-tuning layers. The model was developed based on the conflation of feature distribution and comparison of signed weights in fine-tunable layers. The concept of conflation (a method primarily used in GIS mapping) is introduced in section 3.6, while the idea of signed weights is derived from Equation 2.4, with more discussion in section 3.6. When feature distributions are conflated, they give a uniform representation, making comparisons with other features easier. The weight signs in a model's learning process significantly affect how fast it can learn patterns in data and consequent use in the neural network tasks.

The model was informed by experiments conducted in chapter four that validated the mathematical model defined and expressed in sections 3.6 and 3.7. From the experimental results in Tables 4.1 to 4.11, it is clear that conflated feature distributions with lower Kullback-Leibler divergence give closely related source-target domain images and dynamically selected layers whose Kullback-Leibler divergence is lower for the positively signed weights.

Furthermore, the model was tested on various datasets, as presented in Chapter Four, indicating the effects of the conflated distributions and signed weights on the transfer learning process at the feature adaptation and fine-tuning stages. The testing was done on the nine datasets presented in chapter three, section 3.3, on pre-trained models described in section 3.4.

#### 5.2 Objectives Review

This study's main objective was to investigate using similar textural features from the target and source models on selectable data subsets in CNN models with a dynamic layer selection mechanism on suitable fine-tunable new models to enhance

performance accuracy. Combining the two elements - closely related textural features (feature-based transfer learning) and weight instances (instance-based transfer learning) - was expected to improve the transfer learning process in classification tasks.

The objectives are outlined in Chapter One, section 1.6.2. The first objective was to review existing studies that have addressed the use of features and transfer learning approaches with parameters in optimizing the transfer learning process. These have been discussed in section 2.4, addressing the aspect of labels, low-textural features, and the various methods used in extracting these features. The section also highlights some real-life applications of extracting these features, as noted in sections 2.4.6 to 2.4.8.

Furthermore, the review of various studies in section 2.4 shows a clear context of the applicability of low-level textural features in different architectures. Like in this study, the features are extracted and compared using various methods. These methods have used different parameters, including GANs, dimensionality reduction techniques such as DCT, Gabor filters with histograms, approximation methods with Gaussian Mixture models, and optimal similarity graphs. Some of these methods are similar to those used in extracting features (convolutional neural network) and comparing them (dissimilarity of the features). The uniqueness of this objective is found in the introduction of the conflation that merges the distribution of textural features, creating better uniformity of features in a given sample for improved comparison between the domains.

The second objective was to determine the most suitable distance metric for comparing the samples and layers in the new model. This determination has been addressed in sections 2.4, 4.4.1, and 4.6, where the best measures were selected based on the least complexity.

The third objective was to develop a new model using the selected conflated samples and dynamically selected layers participating in fine-tuning. The choice of dynamic layers in the new models has been discussed in section 2.5. The literature outlines various methods for pre-trained models' layers based on filters, weights, binary numbers, and evolution algorithms. These methods differ from those that look at the weights in the pre-trained model filters based on the signs. The second objective

follows a methodology addressed in section 3.6. The section clearly outlines how the features are conflated and how the layers are selected, creating a pipeline for fine-tuning. The methods used in each subprocess, the datasets and the architectures are also discussed in sections 3.3 and 3.4, respectively.

Furthermore, the third objective also includes integrating the algorithmic steps in the various new models. The algorithm was implemented in new models discussed in section 3.4. The necessary setups required to achieve this objective are discussed in section 3.8, which outlines the various parameters and even optimisation techniques such as dropout and batch normalization. Within this implementation phase, all the architectures and techniques are justified in their selection, with architectures such as MobileNets influenced by evaluating the models in low-resource architectures. Others, such as DenseNet169, have been used to identify the algorithm's behaviour in large new models.

The fourth objective evaluates the effectiveness of the developed algorithm in fine-tuning by comparing the various datasets and commonly used transfer learning methods. First, this objective is addressed by selecting the standard techniques used in the transfer learning process described in section 3.7.3. These methods' performance (accuracy, recall and precision) are compared with those introduced by the conflation-dynamic layer selection method. The results of the new techniques are extensively presented in sections 4.3 and 4.4, while their comparisons to the baselines are presented in section 4.5. The validation of the model is also done using a confusion matrix on the validation dataset samples presented in Table 3.1. This validation is further illustrated in Figure 4.16 on the new MobileNetV2 model using the ISIC 2016 dataset.

### **5.3 Knowledge Contributions**

This study's main contribution was developing a new model that facilitates the selection of source-closely related samples from the target dataset and fine-tunable layers for practical transfer learning tasks. The model uses the conflation of features and weights probability distributions using Kullback-Leibler divergence. The model presents a new approach in pipelining transfer learning to the conventional methods, allowing the users to select quality data points in the target datasets and suitable fine-tuning layers,



an advancement in transfer learning methods that highlights the importance of features and weights in the transfer learning process with applications to different types of data and models.

The study's contributions come from the two parts of the model: the target samples and suitable fine-tunable layer selection. In the first part, the target samples are selected based on closeness in textural features with the source domain dataset. In the second part, fine-tunable layers are chosen based on the signed weights within a layer, selecting the layers with the most positive weights. The selection of the closely source-related featured samples in the target domain and the chosen dynamic layers gave a good performance compared to the samples with higher average Kullback-Leibler divergence. A comparison was made of the features of the samples extracted using the first layer of the pre-trained model, which was then converted into textural features probability distributions. The total probability distributions were then conflated (merged) into one probability distribution and compared to the merged probability distributions of each source image feature. Divergence measures were used to compare the extracted LBP or matrix features. The introduced evaluation of features using conflation presents a simple way of representing the overall look of an image's features, making the comparison much more effortless. This method allows users to select quality data points with surety of positive transfer.

In the second part of the model, the fine-tuning layers are selected dynamically based on the weights. The layers whose Kullback-Leibler divergence measures are below average are chosen in the fine-tuning process. For a layer to be selected, the weights for the layer's features are selected (positive or negative), creating a probability distribution of positive and negative weights. The created distribution is then compared to distributions of other layers using Kullback-Leibler divergence. The layers are then compared based on the average divergence, and the layers with the lower values to the average divergence become good candidates for fine-tuning. This new dynamic approach in the selection of fine-tuneable layers presents a quantified approach to the traditional manual methods that rely on the layer's position, providing further research into the role of weights in effective transfer learning.

For the conflation of features in the dataset samples, the new model has shown that

selecting samples closely related to the dataset used in training the pre-trained model allows the consolidation of textural features in an image sample, allowing easier comparison between images. This process allows the selection of better-quality images or samples when performing fine-tuning, leading to a reliable method for selecting target images. This process also increases confidence when working with sensitive data such as medical images whose correctness is key in diagnosis. It reduces trial and error chances in the current methods.

Additionally, the second part of the new model proved that hyperparameters such as weight play a critical factor in the adaptability of pre-trained models in transfer learning tasks: transfer learning is affected by signed weights. The layers with the highest number of positive weights performed better than the negative weights due to their influence towards faster convergence when training a model.

Finally, by merging the two parts of the new model, it is clear that selecting closely domain-related data samples is an essential factor that determines the target domain data adaptation in classification tasks. Furthermore, selecting least-weight divergent layers is another important factor influencing fine-tuning: choosing the suitable layers ensures the model converges much faster and aids in the pre-trained model adaptation process. When these two are in the new model's pipeline, there is confidence in the data and the fine-tuning process.

## **5.4 Conclusion**

This research aimed to develop a new classification model (dynamically fine-tuned model) that uses Kullback-Leibler divergence to improve the target data selection process and fine-tunable layers for an effective transfer learning process. The new model uses the conflation of features and dynamic layer selection utilising the Kullback-Leibler divergence measure. The conflation process uses the textural features of the images or data samples to create probability distributions for each image, merging them to form one feature probability distribution. The conflated probability distribution is then compared to other probability distributions with the below-average Kullback-Leibler divergence of the selected samples for the target task.

The dynamic selection of layers uses Kullback-Leibler divergence to select layers

whose signed weights comparison divergence is below the average of the total compared layers. Both parts of the new model use Kullback-Leibler divergence methods in their selection process, creating the model's pipeline. The new model was evaluated on five model architectures, including the new MobileNetV2, to cater to the resource-constrained environment and nine datasets. The standard transfer learning baselines were used to validate the model's performance, with the introduced methods aiding the new models in performing better when using the standard transfer learning method. The results demonstrate that divergence measures such as the Kullback-Leibler between the target data points and the new model's layers (weights) can influence the target domain's adaptation. However, these two additional steps in using the new CNN models introduce a higher computational complexity than the commonly used transfer learning methods but significantly reduce the trial and error instances and better model selection confidence for the users.

## **5.5 Future Work**

This study has introduced an enhanced transfer learning method involving conflated target data features in selecting the best-fit target data points for transfer learning. It has also introduced a dynamic process of layer selection using signed weights divergence in new model layers. The two pipeline items used a divergence measure (Kullback-Leibler divergence) and improved the transfer learning process. The conflation of features can be used in other datasets since they can be conflated, allowing uniform comparison of these target data points with the source data points' features. The dynamic selection of layers could also be extended to other models apart from the convolutional neural networks by determining which layers participate in the learning process. This enhanced model's pipeline creates an avenue for strengthening many pre-trained models irrespective of their domain and application since all data has features and all models have weights.

The use of divergence measures has shown a significant method in addressing these two issues, and it would be worth exploring other dissimilarities (hamming distance, total variation and Mahalanobis distance) and similarity measures (Kendall rank, Canberra distance and centered kernel alignment) to understand their effects on the feature and layer differences and similarities. Applying this study's method will help developers

gain confidence when selecting pre-trained models and solidify the research insights of transfer learning, consequently reducing the time and cost of developing pre-trained models.

Finally, the introduced methods can be used in small and large tasks depending on the available resources - in areas with low resources (data or computing power), the new MobileNets have been shown to work well, and in a resourced environment, large tasks can still be performed well as evident with the new DenseNet169.

## REFERENCES

- Abdulazeez, F. A., Ahmed, I. T., & Hammad, B. T. (2024). Examining the Performance of Various Pretrained Convolutional Neural Network Models in Malware Detection. *Applied Sciences*, *14*(6). <https://doi.org/10.3390/app14062614>
- Abo-Zahhad, M., Eldifrawi, I., Abdelwahab, M., & El-Malek, A. H. A. (2023). Deep and shallow fast embedded capsule networks: going faster with capsules. *Analog Integrated Circuits and Signal Processing*, *114*(3), 315–324. <https://doi.org/10.1007/s10470-022-02108-w>
- Afif, M., Ayachi, R., Said, Y.F., & Atri, M. (2020). Deep Learning Based Application for Indoor Scene Recognition. *Neural Processing Letters*, *51*, 2827 - 2837.
- Agrawal, M., & Moparthy, N. R. (2023). An efficient multiple-word embedding-based cross-domain feature extraction and aspect sentiment classification. *Measurement: Sensors*, *28*, 100851. <https://doi.org/10.1016/j.measen.2023.100851>
- Ahsan, M. M., Ahad, M. T., Soma, F. A., Paul, S., Chowdhury, A., Luna, S. A., ... Huebner, P. (2021). Detecting SARS-CoV-2 From Chest X-Ray Using Artificial Intelligence. *IEEE access: practical innovations, open solutions*, *9*, 35501–35513. <https://doi.org/10.1109/ACCESS.2021.3061621>
- Asha, V. (2022). Automatic Fabric Inspection using GLCM-based Jensen-Shannon Divergence. *Informatica (Slovenia)*, *46*(1). <https://doi.org/10.31449/INF.V46I1.3015>
- Al-Abboodi, R. H., & Al-Ani, A. A. (2024). A Novel Technique for Facial Recognition Based on the GSO-CNN Deep Learning Algorithm. *Journal of Electrical and Computer Engineering*, *2024*(1), 3443028. <https://doi.org/10.1155/2024/3443028>
- Al-Haija, Q. A., & Adebajo, A. (2020). Breast Cancer Diagnosis in Histopathological Images Using ResNet-50 Convolutional Neural Network. *2020 IEEE*

*International IOT, Electronics and Mechatronics Conference (IEMTRONICS)*, 1–7. <https://doi.org/10.1109/IEMTRONICS51293.2020.9216455>

Al-Hasani, M., Sultan, L. R., Sagreiya, H., Cary, T. W., Karmacharya, M. B., & Sehgal, C. M. (2022). Ultrasound Radiomics for the Detection of Early-Stage Liver Fibrosis. *Diagnostics (Basel, Switzerland)*, 12(11), 2737. <https://doi.org/10.3390/diagnostics12112737>

Alhares, H., Tanha, J., & Balafar, M. A. (2023). AMTLDC: a new adversarial multi-source transfer learning framework to diagnosis of COVID-19. *Evolving Systems*, 14(6), 1101–1115. <https://doi.org/10.1007/s12530-023-09484-2>

Alibabaei, S., Rahmani, M., Tahmasbi, M., Tahmasebi Birgani, M. J., & Razmjoo, S. (2023). Evaluating the Gray Level Co-Occurrence Matrix-Based Texture Features of Magnetic Resonance Images for Glioblastoma Multiform Patients' Treatment Response Assessment. *Journal of Medical Signals & Sensors*, 13(4).

Alirezazadeh, P., Schirrmann, M., & Stolzenburg, F. (2023). Improving Deep Learning-based Plant Disease Classification with Attention Mechanism. *Gesunde Pflanzen*, 75(1), 49–59. <https://doi.org/10.1007/s10343-022-00796-y>

Al-Shammary, D., Hakem, E., Mahdi, A. M., Ibaida, A., & Ahmed, K. (2024). A novel brain EEG clustering based on Minkowski distance to improve intelligent epilepsy diagnosis. *Informatics in Medicine Unlocked*, 47, 101492. <https://doi.org/10.1016/j.imu.2024.101492>

Alwakid, G., Gouda, W., Humayun, M., & Jhanjhi, N. Z. (2023). Enhancing diabetic retinopathy classification using deep learning. *Digital health*, 9, 20552076231203676. <https://doi.org/10.1177/20552076231203676>

Andrearczyk, V., & Whelan, P. F. (2016). Using filter banks in convolutional neural networks for texture classification. *Pattern Recognition Letters*, 84, 63–69. <https://doi.org/10.1016/j.patrec.2016.08.016>

Anand, L., Mewada, S., Shamsi, W., Ritonga, M., Aflisia, N., KumarSarangi, P., &

- NdoleArthur, M. (2023). Diagnosis of Prostate Cancer Using GLCM Enabled KNN Technique by Analyzing MRI Images. *BioMed research international*, 2023, 3913351. <https://doi.org/10.1155/2023/3913351>
- Anand, V., Koundal, D., Alghamdi, W. Y., & Alsharbi, B. M. (2024). Smart grading of diabetic retinopathy: an intelligent recommendation-based fine-tuned EfficientNetB0 framework. *Frontiers in Artificial Intelligence*, 7.
- Antonio, B., Moroni, D., & Martinelli, M. (2023). Efficient adaptive ensembling for image classification. *Expert Systems*, n/a(n/a). <https://doi.org/10.1111/exsy.13424>
- Anusha, N., Vasanth, K., & Masurkar, S. P. (2024). Automated Extraction of Textural Features From Segmented Sentinel-1A Synthetic Aperture Radar Satellite Image Using Grey Level Co-Occurrence Matrix. *Procedia Computer Science*, 235, 2124–2134. <https://doi.org/10.1016/j.procs.2024.04.201>
- Archana, R., & Jeevaraj, P. S. E. (2024). Deep learning models for digital image processing: a review. *Artificial Intelligence Review*, 57(1), 11. <https://doi.org/10.1007/s10462-023-10631-z>
- Arora, R., Saini, I., & Sood, N. (2021). Multi-label segmentation and detection of COVID-19 abnormalities from chest radiographs using deep learning. *Optik*, 246, 167780. <https://doi.org/10.1016/j.ijleo.2021.167780>
- Arun Kumar, S., & Sasikala, S. (2023). Review on Deep Learning-Based CAD Systems for Breast Cancer Diagnosis. *Technology in cancer research & treatment*, 22, 15330338231177977. <https://doi.org/10.1177/15330338231177977>
- Badawi, A., & Elgazzar, K. (2021). Detecting Coronavirus from Chest X-rays Using Transfer Learning. *COVID*, 1(1), 403–415. <https://doi.org/10.3390/covid1010034>
- Bao, R., Sun, Y., Gao, Y., Wang, J., Yang, Q., Chen, H., Mao, Z., & Ye, Y. (2023). A Survey of Heterogeneous Transfer Learning. *ArXiv*, *abs/2310.08459*.

- Barburiceanu, S., Terebes, R., & Meza, S. (2021). 3d texture feature extraction and classification using glcm and lbp-based descriptors. *Applied Sciences*, *11*(5). <https://doi.org/10.3390/app1105233220>
- Bastidas Rodriguez, M. X., Gruson, A., Polanía, L. F., Fujieda, S., Ortiz, F. P., Takayama, K., & Hachisuka, T. (2020). Deep Adaptive Wavelet Network. *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 3100–3108. <https://doi.org/10.1109/WACV45572.2020.9093580>
- Bazán, E., Dokládál, P., & Dokládálová, E. (2019). Quantitative analysis of similarity measures of distributions. *British Machine Vision Conference*
- Begum, N., Badshah, N., Rada, L., Ademaj, A., Ashfaq, M., & Atta, H. (2022). An improved multi-modal joint segmentation and registration model based on Bhattacharyya distance measure. *Alexandria Engineering Journal*, *61*(12), 12353–12365. <https://doi.org/10.1016/j.aej.2022.06.018>
- Bhardwaj, P., & Kaur, A. (2021). A novel and efficient deep learning approach for COVID-19 detection using X-ray imaging modality. *International journal of imaging systems and technology*, *31*(4), 1775–1791. <https://doi.org/10.1002/ima.22627>
- Bhatia, P., Arumae, K., & Busra Celikkaya, E. (2020). Dynamic transfer learning for named entity recognition. In A. Shaban-Nejad & M. Michalowski (Eds.), *Precision health and medicine: A digital revolution in healthcare* (pp. 69– 81). Springer International Publishing. [https://doi.org/10.1007/978-3-030-24409-5\\_7](https://doi.org/10.1007/978-3-030-24409-5_7)
- Borwarnginn, P., Kusakunniran, W., Karnjanapreechakorn, S., & Thongkanchorn, K. (2021). Knowing Your Dog Breed: Identifying a Dog Breed with Deep Learning. *Machine Intelligence Research*, *18*(1), 45–54. doi:10.1007/s11633-020-1261-0
- Bu, H. H., Kim, N. C., & Kim, S. H. (2023). Content-based image retrieval using a fusion of global and local features. *ETRI Journal*, *45*(3), 505–518. <https://doi.org/10.4218/etrij.2022-0071>



- Bursać, P., Kovačević, M., & Bajat, B. (2022). Instance-based transfer learning for soil organic carbon estimation. *Frontiers in Environmental Science*, 10. <https://doi.org/10.3389/fenvs.2022.1003918>
- Cao, Z., Zhou, Y., Yang, A., & Peng, S. (2021). Deep transfer learning mechanism for fine-grained cross-domain sentiment classification. *Connection Science*, 33(4), 911–928. <https://doi.org/10.1080/09540091.2021.1912711>
- Changwei, Z., Lili, Z., Xiaojun, Z., Yuanbo, W., Di, W., & Zhi, T. (2020). Classification of normal and pathological voices using convolutional neural network. *2020 International Conference on Sensing, Measurement Data Analytics in the era of Artificial Intelligence (ICSMD)*, 325–329. <https://doi.org/10.1109/ICSMD50554.2020.9261730>
- Chaves, M. (2022, January 31). GLCMs — a great tool for your ML arsenal - towards data science. *Medium*. <https://towardsdatascience.com/glcms-a-great-tool-for-your-ml-arsenal-7a59f1e45b65>
- Chen, D., Chen, Y., Ma, J., Cheng, C., Xi, X., Zhu, R., & Cui, Z. (2021). An Ensemble Deep Neural Network for Footprint Image Retrieval Based on Transfer Learning. *Journal of Sensors*, 2021(1), 6631029. <https://doi.org/10.1155/2021/6631029>
- Chen, F., Wang, N., Tang, J., Yan, P., & Yu, J. (2023). Unsupervised person re-identification via multi-domain joint learning. *Pattern Recognition*, 138, 109369. <https://doi.org/10.1016/j.patcog.2023.109369>
- Chen, H., Yu, Y., Jia, Y., & Gu, B. (2023). Incremental learning for transductive support vector machine. *Pattern Recognition*, 133, 108982. <https://doi.org/10.1016/j.patcog.2022.108982>
- Chen, K., Yao, L., Zhang, D., Wang, X., Chang, X., & Nie, F. (2020). A Semisupervised Recurrent Convolutional Attention Model for Human Activity Recognition. *IEEE Transactions on Neural Networks and Learning Systems*, 31, 1747-1756.

- Chen, T., Gao, T., Li, S., Zhang, X., Cao, J., Yao, D., & Li, Y. (2021). A novel face recognition method based on fusion of LBP and HOG. *IET Image Processing*, *15*(14), 3559–3572. <https://doi.org/10.1049/ipr2.12192>
- Chen, W. L., Wagner, J., Heugel, N., Sugar, J., Lee, Y. W., Conant, L., ... Whelan, H. T. (2020). Functional Near-Infrared Spectroscopy and Its Clinical Application in the Field of Neuroscience: Advances and Future Directions. *Frontiers in Neuroscience*, *14*. <https://doi.org/10.3389/fnins.2020.00724>
- Chen, Z., Wu, J., Wang, W., Su, W., Chen, G., Xing, S., ... Dai, J. (2024). InternVL: Scaling up Vision Foundation Models and Aligning for Generic Visual-Linguistic Tasks. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 24185–24198.
- Cherkassky, V., & Lee, E. H. (2024). To understand double descent, we need to understand VC theory. *Neural Networks*, *169*, 242–256. <https://doi.org/10.1016/j.neunet.2023.10.014>
- Chen, Y., Gao, Q., & Wang, X. (2021). Inferential Wasserstein Generative Adversarial Networks. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, *84*(1), 83–113. <https://doi.org/10.1111/rssb.12476>
- Chui, K. T., Arya, V., Band, S. S., Alhalabi, M., Liu, R. W., & Chi, H. R. (2023). Facilitating innovation and knowledge transfer between homogeneous and heterogeneous datasets: Generic incremental transfer learning approach and multidisciplinary studies. *Journal of Innovation & Knowledge*, *8*(2), 100313. <https://doi.org/10.1016/j.jik.2023.100313>
- Cohn, R., & Holm, E. (2021). Unsupervised Machine Learning Via Transfer Learning and k-Means Clustering to Classify Materials Image Data. *Integrating Materials and Manufacturing Innovation*, *10*(2), 231–244. <https://doi.org/10.1007/s40192-021-00205-8>
- Cote, M., Dash, A., & Branzan Albu, A. (2023). Semantic segmentation of textured mosaics. *EURASIP Journal on Image and Video Processing*, *2023*(1), 13.

<https://doi.org/10.1186/s13640-023-00613-0>

Dalvi, P. P., Edla, D. R., & Purushothama, B. R. (2023). Diagnosis of Coronavirus Disease From Chest X-Ray Images Using DenseNet-169 Architecture. *SN Computer Science*, 4(3), 214. <https://doi.org/10.1007/s42979-022-01627-7>

Debelee, T. G. (2023). Skin Lesion Classification and Detection Using Machine Learning Techniques: A Systematic Review. *Diagnostics*, 13(19). <https://doi.org/10.3390/diagnostics13193147>

DeepAI. (2020, June 25). WEIGHT (Artificial neural Network). DeepAI. <https://deepai.org/machine-learning-glossary-and-terms/weight-artificial-neural-network>

Dewan, J. H., & Thepade, S. D. (2020). Image Retrieval Using Low Level and Local Features Contents: A Comprehensive Review. *Applied Computational Intelligence and Soft Computing*, 2020(1), 8851931. <https://doi.org/10.1155/2020/8851931>

Dharma, A. S., Tambunan, N., & Sinaga, L. E. (2022). Face Recognition with Edge Detection and LBP Feature Extraction. *2022 IEEE International Conference of Computer Science and Information Technology (ICOSNIKOM)*, 1–7. <https://doi.org/10.1109/ICOSNIKOM56551.2022.10034925>

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, ... Houlsby, N. (2020). An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *ArXiv, abs/2010.11929*.

Du, C., Sun, H., Wang, J., Qi, Q., & Liao, J. (2020). Adversarial and Domain-Aware BERT for Cross-Domain Sentiment Analysis. In D. Jurafsky, J. Chai, N. Schluter, & J. Tetreault (Eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (pp. 4019–4028). <https://doi.org/10.18653/v1/2020.acl-main.370>

Du, Y., Tan, Z., Chen, Q., Zhang, Y., & Wang, C. (2019). Homogeneous Online Transfer Learning with Online Distribution Discrepancy

Minimization. *European Conference on Artificial Intelligence*.

- El Lel, T., Ahsan, M., & Haider, J. (2023). Detecting COVID-19 from Chest X-rays Using Convolutional Neural Network Ensembles. *Computers*, 12(5). <https://doi.org/10.3390/computers12050105>
- Fekri-Ershad, S., & Tajeripour, F. (2017). Impulse-Noise Resistant Color-Texture Classification Approach Using Hybrid Color Local Binary Patterns and Kullback–Leibler Divergence. *The Computer Journal*, 60(11), 1633–1648. <https://doi.org/10.1093/comjnl/bxx033>
- Feng, L., Yang, Y., Tan, M., Zeng, T., Tang, H., Li, Z., ... Feng, F. (2024). Adaptive multi-source domain collaborative fine-tuning for transfer learning. *PeerJ. Computer Science*, 10(e2107), e2107. <https://doi.org/10.7717/peerj-cs.2107>
- Fukuyama, S., Asakawa, T., Shimizu, K., Nomura, K., & Aono, M. (2024). KDE Lab at ImageCLEFmedical GANs 2024. *Conference and Labs of the Evaluation Forum*.
- Garg, S., Saxena, A., & Gupta, R. (2023). Yoga pose classification: a CNN and MediaPipe inspired deep learning approach for real-world application. *Journal of Ambient Intelligence and Humanized Computing*, 14(12), 16551–16562. <https://doi.org/10.1007/s12652-022-03910-0>
- Ge, W., & Yu, Y. (2017). Borrowing treasures from the wealthy: Deep transfer learning through selective joint fine-tuning. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 10–19. <https://doi.org/10.1109/CVPR.2017.9>
- Guo, W., Dong, Y., & Hao, G.-F. (2024). Transfer learning empowers accurate pharmacokinetics prediction of small samples. *Drug Discovery Today*, 29(4), 103946. <https://doi.org/10.1016/j.drudis.2024.103946>
- Guo, Y., Li, Y., Wang, L., & Rosing, T. (2020). AdaFilter: Adaptive Filter Fine-Tuning for Deep Transfer Learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04), 4060–4066. <https://doi.org/10.1609/aaai>.

- Guo, Y., Shi, H., Kumar, A., Grauman, K., Rosing, T., & Feris, R. (2019). SpotTune: Transfer Learning Through Adaptive Fine-Tuning. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 4800–4809. <https://doi.org/10.1109/CVPR.2019.00494>
- Hameed, I. M., Abdulhussain, S. H., Mahmmud, B. M., & Pham, D. T. (2021). Content-based image retrieval: A review of recent trends. *Cogent Engineering*, 8(1). <https://doi.org/10.1080/23311916.2021.1927469>
- Han, X., Zhang, Z., Ding, N., Gu, Y., Liu, X., Huo, Y., ... Zhu, J. (2021). Pre-trained models: Past, present and future. *AI Open*, 2, 225–250. <https://doi.org/10.1016/j.aiopen.2021.08.002>
- Han, Z., Gao, C., Liu, J., Zhang, J., & Zhang, S. Q. (2024). Parameter-Efficient Fine-Tuning for Large Models: A Comprehensive Survey. *arXiv [Cs.LG]*. Retrieved from <http://arxiv.org/abs/2403.14608>
- Haq, I., Mazhar, T., Nasir, Q., Razzaq, S., Mohsan, S. A. H., Alsharif, M. H., ... Mostafa, S. M. (2022). Machine Vision Approach for Diagnosing Tuberculosis (TB) Based on Computerized Tomography (CT) Scan Images. *Symmetry*, 14(10). <https://doi.org/10.3390/sym14101997>
- Hassan, E., Shams, M. Y., Hikal, N. A., & Elmougy, S. (2023). The effect of choosing optimizer algorithms to improve computer vision tasks: a comparative study. *Multimedia Tools and Applications*, 82(11), 16591–16633. <https://doi.org/10.1007/s11042-022-13820-0>
- Heikel, E., & Espinosa-Leal, L. (2022). Indoor Scene Recognition via Object Detection and TF-IDF. *Journal of imaging*, 8(8), 209. <https://doi.org/10.3390/jimaging8080209>
- Hernán-Caballero, A., Akhlaghi, M., López-Sanjuán, C., Vázquez Ramió, H., Laur, J., Varela, J., ... Taylor, K. (2024). The miniJPAS survey: Maximising the photo-z accuracy from multi-survey datasets with probability conflation. *A&A*, 684.

doi:10.1051/0004-6361/202348513

- Hong, W., Ren, W., Lao, J., Xie, L., Zhong, L., Wang, J., ... Chu, W. (2024). Training Object Detectors from Scratch: An Empirical Study in the Era of Vision Transformer. *International Journal of Computer Vision*, 132(8), 2929–2942. <https://doi.org/10.1007/s11263-024-01988-x>
- Hristu, R., Eftimie, L. G., Paun, B., Stanciu, S. G., & Stanciu, G. A. (2020). Pixel-level angular quantification of capsular collagen in second harmonic generation microscopy images of encapsulated thyroid nodules. *Journal of Biophotonics*, 13(12), e202000262. <https://doi.org/10.1002/jbio.202000262>
- Huang, L., Qin, J., Zhou, Y., Zhu, F., Liu, L., & Shao, L. (2023). Normalization Techniques in Training DNNs: Methodology, Analysis and Application. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 45(08), 10173–10196. <https://doi.org/10.1109/TPAMI.2023.3250241>
- Hung, J. C., & Chang, J.-W. (2021). Multi-level transfer learning for improving the performance of deep neural networks: Theory and practice from the tasks of facial emotion recognition and named entity recognition. *Applied Soft Computing*, 109, 107491. <https://doi.org/10.1016/j.asoc.2021.107491>
- Imai, S., Kawai, S., & Nobuhara, H. (2020). Stepwise pathnet: A layer-by-layer knowledge-selection-based transfer learning algorithm. *Scientific Reports*, 10
- Iman, M., Arabnia, H. R., & Rasheed, K. (2023). A Review of Deep Transfer Learning and Recent Advancements. *Technologies*, 11(2). <https://doi.org/10.3390/technologies11020040>
- Iqbal, N., Mumtaz, R., Shafi, U., & Zaidi, S. M. H. (2021). Gray level co-occurrence matrix (GLCM) texture based crop classification using low altitude remote sensing platforms. *PeerJ Comput. Sci.*, 7(e536), e536
- Irfan, M., Jiangbin, Z., Iqbal, M., & Arif, M. H. (2021). A novel lifelong learning model based on cross domain knowledge extraction and transfer to classify underwater images. *Information Sciences*, 552, 80–101. <https://doi.org/10.1016/>

- Ismail, W. N., Alsalamah, H. A., & Mohamed, E. A. (2024). Genetic-efficient fine-tuning with layer pruning on multimodal Covid-19 medical imaging. *Neural Computing and Applications*, 36(6), 3215–3237. <https://doi.org/10.1007/s00521-023-09194-5>
- Jahanian, M., Karimi, A., Eraghi, N. O., & Zarafshan, F. (2024). MedTransCluster: Transfer learning for deep medical image clustering. *Intelligence-Based Medicine*, 9, 100139. <https://doi.org/10.1016/j.ibmed.2024.100139>
- Jain, S., Singhanian, U., Tripathy, B., Nasr, E. A., Aboudaif, M. K., & Kamrani, A. K. (2021). Deep Learning-Based Transfer Learning for Classification of Skin Cancer. *Sensors*, 21(23). <https://doi.org/10.3390/s21238142>
- Ji, R., & Tekalp, A. M. (2024). A new multi-picture architecture for learned video deinterlacing and demosaicing with parallel deformable convolution and self-attention blocks. *Image and Vision Computing*, 146, 105023. <https://doi.org/10.1016/j.imavis.2024.105023>
- Jiang, S., Li, Y., Firouzi, F., & Chakrabarty, K. (2024). Federated clustered multi-domain learning for health monitoring. *Scientific Reports*, 14(1), 903. <https://doi.org/10.1038/s41598-024-51344-9>
- Jin, G., Yi, X., Zhang, L., Zhang, L., Schewe, S., & Huang, X. (2020). How does Weight Correlation Affect the Generalisation Ability of Deep Neural Networks. *ArXiv, abs/2010.05983*.
- Jung, I., Lim, J., & Kim, H. K. (2021). PF-TL: Payload Feature-Based Transfer Learning for Dealing with the Lack of Training Data. *Electronics*, 10(10). <https://doi.org/10.3390/electronics10101148>
- Kabir, M., Unal, F., Akinci, T. C., Martinez-Morales, A. A., & Ekici, S. (2024). Revealing GLCM Metric Variations across a Plant Disease Dataset: A Comprehensive Examination and Future Prospects for Enhanced Deep Learning Applications. *Electronics*, 13(12). <https://doi.org/10.3390/electronics13122299>

- Kadam, S.S., Adamuthe, A., & Patil, A. (2020). CNN Model for Image Classification on MNIST and Fashion-MNIST Dataset. *Journal of scientific research*.
- Kandel, I., & Castelli, M. (2020). The effect of batch size on the generalizability of the convolutional neural networks on a histopathology dataset. *ICT Express*, 6(4), 312–315. <https://doi.org/10.1016/j.icte.2020.04.010>
- Kandhro, I. A., Manickam, S., Fatima, K., Uddin, M., Malik, U., Naz, A., & Dandoush, A. (2024). Performance evaluation of E-VGG19 model: Enhancing real-time skin cancer detection and classification. *Heliyon*, 10(10), e31488. <https://doi:10.1016/j.heliyon.2024.e31488>
- Kaplan, K., Kaya, Y., Kuncan, M., & Ertunç, H. M. (2020). Brain tumor classification using modified local binary patterns (lbp) feature extraction methods. *Medical Hypotheses*, 139, 109696. <https://doi.org/https://doi.org/10.1016/j.mehy.2020.109696>
- Karat, B. G., DeKraker, J., Hussain, U., Köhler, S., & Khan, A. R. (2023). Mapping the macrostructure and microstructure of the in vivo human hippocampus using diffusion MRI. *Human Brain Mapping*, 44(16), 5485–5503. <https://doi.org/10.1002/hbm.26461>
- Karim, Z., & van Zyl, T. L. (2021). Deep/Transfer Learning with Feature Space Ensemble Networks (FeatSpaceEnsNets) and Average Ensemble Networks (AvgEnsNets) for Change Detection Using DInSAR Sentinel-1 and Optical Sentinel-2 Satellite Data Fusion. *Remote Sensing*, 13(21). <https://doi.org/10.3390/rs13214394>
- Kim, J., & Lee, O. (2024). Correlation analysis between texture features and elasticity of skin hyperspectral images in the near-infrared band. *Skin Research and Technology*, 30(3), e13654. <https://doi.org/10.1111/srt.13654>
- Kirielle, N., Christen, P., & Ranbaduge, T. (2022). TransER: Homogeneous Transfer Learning for Entity Resolution. *International Conference on Extending Database Technology*.



- Kong, L., & Cheng, J. (2022). Classification and detection of COVID-19 X-Ray images based on DenseNet and VGG16 feature fusion. *Biomedical signal processing and control*, 77, 103772. <https://doi.org/10.1016/j.bspc.2022.103772>
- Kouw, W. M., & Loog, M. (2021). A review of domain adaptation without target labels. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(3), 766–785. <https://doi.org/10.1109/TPAMI.2019.2945942>
- Kundu, R., Das, R., Geem, Z. W., Han, G. T., & Sarkar, R. (2021). Pneumonia detection in chest X-ray images using an ensemble of deep learning models. *PloS one*, 16(9), e0256630. <https://doi.org/10.1371/journal.pone.0256630>
- Kurniati, F., Sembiring, I., Setiawan, A., Setyawan, I., & Huizen, R. (2024). GLCM-Based Feature Combination for Extraction Model Optimization in Object Detection Using Machine Learning. *Jurnal Ilmiah Teknik Elektro Komputer Dan Informatika*, 9(4), 1196-1205.
- Kushibar, K., Salem, M., Valverde, S., Rovira, À., Salvi, J., Oliver, A., & Lladó, X. (2021). Transductive transfer learning for domain adaptation in brain magnetic resonance image segmentation. *Frontiers in Neuroscience*, 15. <https://doi.org/10.3389/fnins.2021.608808>
- Lam, P., Zhang, H., Chen, N.F., & Sisman, B. (2022). EPIC TTS Models: Empirical Pruning Investigations Characterizing Text-To-Speech Models. *ArXiv, abs/2209.10890*.
- Lamjiak, T., Sirinaovakul, B., Kornthongnimit, S., Polvichai, J., & Sohail, A. (2024). Optimizing Artificial Neural Network Learning Using Improved Reinforcement Learning in Artificial Bee Colony Algorithm. *Applied Computational Intelligence and Soft Computing*, 2024(1), 6357270. <https://doi.org/10.1155/2024/6357270>
- Lan, Y., Xu, X., Fang, Q., Zeng, Y., Liu, X., & Zhang, X. (2022). Transfer reinforcement learning via meta-knowledge extraction using auto-pruned decision trees. *Knowledge-Based Systems*, 242, 108221. <https://doi.org/10.1016>

/j.knosys.2022.108221

- Lavine, B. K., Booksh, K. S., & Neal, S. L. (2024). Transductive and Transfer Learning. *Journal of Experimental and Theoretical Analyses*, 2(2), 56–57. <https://doi.org/10.3390/jeta2020005>
- Lee, G. Y., Dam, T., Poenar, D. P., Duong, V. N., & Ferdous, M. M. (2024, January). HELA-VFA: A Hellinger Distance-Attention-Based Feature Aggregation Network for Few-Shot Classification. *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2173–2183.
- Lee, J.-W., Goo, N. H., Park, W. B., Pyo, M., & Sohn, K.-S. (2021). Virtual microstructure design for steels using generative adversarial networks. *Engineering Reports*, 3(1), e12274. <https://doi.org/10.1002/eng2.12274>
- Li, J., Horiguchi, Y., & Sawaragi, T. (2022). Counterfactual inference to predict causal knowledge graph for relational transfer learning by assimilating expert knowledge --Relational feature transfer learning algorithm. *Advanced Engineering Informatics*, 51, 101516. <https://doi.org/10.1016/j.aei.2021.101516>
- Li, L., Yang, J., Kong, X., Zhang, J., & Ma, Y. (2022). Unsupervised domain adaptation via discriminative feature learning and classifier adaptation from center-based distances. *Knowledge-Based Systems*, 250, 109022. <https://doi.org/10.1016/j.knosys.2022.109022>
- Lian, D., Zhou, D., Feng, J., & Wang, X. (2024). Scaling & shifting your features: a new baseline for efficient model tuning. *Proceedings of the 36th International Conference on Neural Information Processing Systems*. Presented at the New Orleans, LA, USA. Red Hook, NY, USA: Curran Associates Inc.
- Liang, Q., Wu, P., & Huang, C. (2019). An efficient method for text classification task, 92–97. <https://doi.org/10.1145/3341620.3341631>
- Liang, Z., & Zhang, L. (2023). L1-norm discriminant analysis via Bhattacharyya error bounds under Laplace distributions. *Pattern Recognition*, 141, 109609. <https://doi.org/10.1016/j.patcog.2023.109609>

- Lim, X. R., Lee, C. P., Lim, K. M., Ong, T. S., Alqahtani, A., & Ali, M. (2023). Recent Advances in Traffic Sign Recognition: Approaches and Datasets. *Sensors (Basel, Switzerland)*, *23*(10), 4674. <https://doi.org/10.3390/s23104674>
- Liu, B., Desrosiers, C., Ben Ayed, I., & Dolz, J. (2023). Segmentation with mixed supervision: Confidence maximization helps knowledge distillation. *Medical Image Analysis*, *83*, 102670. <https://doi.org/10.1016/j.media.2022.102670>
- Liu, Q., Zhao, Y., Gao, R., Bu, X., & Hanajima, N. (2024). SpanEffiDet: Span-Scale and Span-Path Feature Fusion for Object Detection. *Neural Processing Letters*, *56*(3), 193. <https://doi.org/10.1007/s11063-024-11653-6>
- Lu, J., Zhang, W., Zhao, Y., & Sun, C. (2022). Image local structure information learning for fine-grained visual classification. *Scientific Reports*, *12*(1), 19205. <https://doi.org/10.1038/s41598-022-23835-0>
- Lulu, Q., Ranhui, X., Shaojie, Z., Mingming, Z., & Weiqin, Y. (2024). TMNIO: Triplet merged network with involution operators for improved few-shot image classification. *IET Image Processing*, *18*(6), 1629–1641. <https://doi.org/10.1049/ipr2.13055>
- Luo, J., & Hu, D. (2023). An Image Classification Method Based on Adaptive Attention Mechanism and Feature Extraction Network. *Computational Intelligence and Neuroscience*, *2023*(1), 4305594. <https://doi.org/10.1155/2023/4305594>
- Luo, Z., Zou, Y., Hoffman, J., & Fei-Fei, L. (2017). Label efficient learning of transferable representations across domains and tasks. *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 164–176
- Machine Learning Glossary. (2024, August 24). Retrieved from <https://developers.google.com/machine-learning/glossary#accuracy>
- Machine Learning Glossary. (2024, August 24). Retrieved from <https://developers.google.com/machine-learning/glossary#precision>
- Majhi, M., Pal, A. K., Islam, S. K. H., & Khurram Khan, M. (2021). Secure content-

based image retrieval using modified Euclidean distance for encrypted features. *Transactions on Emerging Telecommunications Technologies*, 32(2), e4013. <https://doi.org/10.1002/ett.4013>

Manokaran, J., Zabihollahy, F., Hamilton-Wright, A., & Ukwatta, E. (2021). Detection of COVID-19 from chest x-ray images using transfer learning. *Journal of medical imaging (Bellingham, Wash.)*, 8(Suppl 1), 017503. <https://doi.org/10.1117/1.JMI.8.S1.017503>

Mecheter, I., Abbod, M., Zaidi, H., & Amira, A. (2024). Transfer learning from T1-weighted to T2-weighted Magnetic resonance sequences for brain image segmentation. *CAAI Transactions on Intelligence Technology*, 9(1), 26–39. <https://doi.org/10.1049/cit2.12270>

Meir, Y., Tzach, Y., Hodassman, S., Tevet, O., & Kanter, I. (2024). Towards a universal mechanism for successful deep learning. *Scientific Reports*, 14(1), 5881. <https://doi.org/10.1038/s41598-024-56609-x>

Meli, D., Sridharan, M., & Fiorini, P. (2021). Inductive learning of answer set programs for autonomous surgical task planning. *Machine Learning*, 110(7), 1739–1763. <https://doi.org/10.1007/s10994-021-06013-7>

Meng, Z. H. U., Weidong, M. I. N., Yu, Z. H. A. N. G., & Jingwen, D. U. A. N. (2021). Parallel Selective Kernel Attention Based on HardSoftmax. *Journal of Computer Engineering & Applications*, 57(21).

Michau, G., & Fink, O. (2021). Unsupervised transfer learning for anomaly detection: Application to complementary operating condition transfer. *Knowledge-Based Systems*, 216, 106816. <https://doi.org/10.1016/j.knosys.2021.106816>

Minami, S., Hirakawa, T., Yamashita, T., & Fujiyoshi, H. (2021). Knowledge Transfer Graph for Deep Collaborative Learning. In H. Ishikawa, C.-L. Liu, T. Pajdla, & J. Shi (Eds.), *Computer Vision -- ACCV 2020* (pp. 203–217). Cham: Springer International Publishing.

Mnmoustafa, M. A. (2017). *Tiny ImageNet*. Retrieved from <https://kaggle.com/>

competitions/tiny-imagenet

- Molina-Cabanillas, M. A., Jiménez-Navarro, M. J., Arjona, R., Martínez-Álvarez, F., & Asencio-Cortés, G. (2022). DIAFAN-TL: An instance weighting-based transfer learning algorithm with application to phenology forecasting. *Knowledge-Based Systems*, 254, 109644. <https://doi.org/10.1016/j.knosys.2022.109644>
- Montesinos López, O. A., Montesinos López, A., & Crossa, J. (2022). Fundamentals of Artificial Neural Networks and Deep Learning. In O. A. Montesinos López, A. Montesinos López, & J. Crossa (Eds.), *Multivariate Statistical Machine Learning Methods for Genomic Prediction* (pp. 379–425). [https://doi.org/10.1007/978-3-030-89010-0\\_10](https://doi.org/10.1007/978-3-030-89010-0_10)
- Moon, S., & Carbonell, J. (2017). Completely Heterogeneous Transfer Learning with Attention - What And What Not To Transfer. *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, 2508–2514. <https://doi.org/10.24963/ijcai.2017/349>
- Mozafari, A. S., & Jamzad, M. (2016). A SVM-based model-transferring method for heterogeneous domain adaptation. *Pattern Recogn.*, 56(C), 142–158. <https://doi.org/10.1016/j.patcog.2016.03.009>
- Mujeeb Rahman, K. K., Nasor, M., & Imran, A. (2022). Automatic Screening of Diabetic Retinopathy Using Fundus Images and Machine Learning Algorithms. *Diagnostics (Basel, Switzerland)*, 12(9), 2262. <https://doi.org/10.3390/diagnostics12092262>
- Mutawa, A. M., Alnajdi, S., & Sruthi, S. (2023). Transfer Learning for Diabetic Retinopathy Detection: A Study of Dataset Combination and Model Performance. *Applied Sciences*, 13(9). <https://doi.org/10.3390/app13095685>
- Myung, S., Huh, I., Jang, W., Choe, J.M., Ryu, J., Kim, D., Kim, K. & Jeong, C.. (2022). PAC-Net: A Model Pruning Approach to Inductive Transfer Learning. *Proceedings of the 39th International Conference on Machine Learning*, in

Proceedings of Machine Learning Research 162:16240-16252 Available from <https://proceedings.mlr.press/v162/myung22a.html>.

Nagae, S., Kawai, S., & Nobuhara, H. (2020). Transfer learning layer selection using genetic algorithm. *2020 IEEE Congress on Evolutionary Computation (CEC)*, 1–6. <https://doi.org/10.1109/CEC48606.2020.9185501>

Nanni, L., Ghidoni, S., & Brahmam, S. (2021). Deep features for training support vector machines. *Journal of Imaging*, 7(9). <https://doi.org/10.3390/jimaging7090177>

Nayman, N., Golbert, A., Noy, A., & Zelnik-Manor, L. (2024, January). Diverse Imagenet Models Transfer Better. *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 1914–1925.

Nguyen, T.-T., Nguyen, T.-H., & Ngo, B.-V. (2021). A GLCM Algorithm for Optimal Features of Mammographic Images for Detection of Breast Cancer. *2021 International Conference on System Science and Engineering (ICSSE)*, 295–299. <https://doi.org/10.1109/ICSSE52999.2021.9538426>

Niu, B., Gao, Z., & Guo, B. (2021). Facial expression recognition with lbp and orb features. *Computational Intelligence and Neuroscience*, 2021

Okada, K., & Motoyoshi, I. (2021). Human Texture Vision as Multi-Order Spectral Analysis. *Frontiers in Computational Neuroscience*, 15. <https://doi.org/10.3389/fncom.2021.692334>

Okano, R., & Imaizumi, M. (2022). Inference for Projection-Based Wasserstein Distances on Finite Spaces. *Statistica Sinica*.

Park, E. J., Kim, S. H., Park, S. J., & Baek, T. W. (2021). Texture Analysis of Gray-Scale Ultrasound Images for Staging of Hepatic Fibrosis. *Taehan Yongsang Uihakhoe chi*, 82(1), 116–127. <https://doi.org/10.3348/jksr.2019.0185>

Peng, H., Yu, Y., & Yu, S. (2024). Re-Thinking the Effectiveness of Batch Normalization and Beyond. *IEEE Transactions on Pattern Analysis & Machine*

*Intelligence*, 46(01), 465–478. <https://doi.org/10.1109/TPAMI.2023.3319005>

- Peng, M., Li, Z., & Juan, X. (2022). Similarity-based domain adaptation network. *Neurocomputing*, 493, 462–473. <https://doi.org/10.1016/j.neucom.2021.12.089>
- Pieterse, J., & Mocanu, D.C. (2019). Evolving and Understanding Sparse Deep Neural Networks using Cosine Similarity. *ArXiv*, *abs/1903.07138*.
- Pimentel, S., & Qranfal, Y. (2021). A data assimilation framework that uses the Kullback-Leibler divergence. *PloS one*, 16(8), e0256584. <https://doi.org/10.1371/journal.pone.0256584>
- Pinto, G., Messina, R., Li, H., Hong, T., Piscitelli, M. S., & Capozzoli, A. (2022). Sharing is caring: An extensive analysis of parameter-based transfer learning for the prediction of building thermal dynamics. *Energy and Buildings*, 276, 112530. <https://doi.org/10.1016/j.enbuild.2022.112530>
- Pourkaramdel, Z., Fekri-Ershad, S., & Nanni, L. (2022). Fabric defect detection based on completed local quartet patterns and majority decision algorithm. *Expert Systems with Applications*, 198, 116827. <https://doi.org/10.1016/j.eswa.2022.116827>
- Praveen Gujjar, J., Prasanna Kumar, H. R., & Chiplunkar, N. N. (2021). Image classification and prediction using transfer learning in colab notebook. *Global Transitions Proceedings*, 2(2), 382–385. <https://doi.org/10.1016/j.gltp.2021.08.068>
- Priyatikanto, R., Lu, Y., Dash, J., & Sheffield, J. (2023). Improving generalisability and transferability of machine-learning-based maize yield prediction model through domain adaptation. *Agricultural and Forest Meteorology*, 341, 109652. <https://doi.org/10.1016/j.agrformet.2023.109652>
- Qiao, S., Yu, Q., Zhao, Z., Song, L., Tao, H., Zhang, T., & Zhao, C. (2022). Edge extraction method for medical images based on improved local binary pattern combined with edge-aware filtering. *Biomedical Signal Processing and*

*Control*, 74, 103490. <https://doi.org/10.1016/j.bspc.2022.103490>

- Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., ... Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(1).
- Ramadan, S. Z. (2020). Methods Used in Computer-Aided Diagnosis for Breast Cancer Detection Using Mammograms: A Review. *Journal of Healthcare Engineering*, 2020(1), 9162464. <https://doi.org/10.1155/2020/9162464>
- Ramola, A., Shakya, A. K., & Van Pham, D. (2020). Study of statistical methods for texture analysis and their modern evolutions. *Engineering Reports*, 2(4), e12149. <https://doi.org/10.1002/eng2.12149>
- Rea, G., Sperandeo, M., Lieto, R., Bocchino, M., Quarato, C. M. I., Feragalli, B., Valente, T., Scioscia, G., Giuffreda, E., Foschino Barbaro, M. P., & Lacedonia, D. (2021). Chest Imaging in the Diagnosis and Management of Pulmonary Tuberculosis: The Complementary Role of Thoracic Ultrasound. *Frontiers in medicine*, 8, 753821. <https://doi.org/10.3389/fmed.2021.753821>
- Rezaei, M., Yang, H., & Meinel, C. (2018). voxel-GAN: Adversarial Framework for Learning Imbalanced Brain Tumor Segmentation. *BrainLes@MICCAI*.
- Royer, A., & Lampert, C.H. (2020). A Flexible Selection Scheme for Minimum-Effort Transfer Learning. *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2180-2189.
- Sadeghi, A., Sadeghi, M., Sharifpour, A., Fakhar, M., Zakariaei, Z., Sadeghi, M., ... Hajati, F. (2024). Potential diagnostic application of a novel deep learning-based approach for COVID-19. *Scientific Reports*, 14(1), 280. <https://doi.org/10.1038/s41598-023-50742-9>
- Saito, K., Kim, D., Teterwak, P., Sclaroff, S., Darrell, T., & Saenko, K. (2021). Tune it the Right Way: Unsupervised Validation of Domain Adaptation via Soft Neighborhood Density. *arXiv [Cs.CV]*. Retrieved from <http://arxiv.org/abs/2108.10860>



- Salehin, I., & Kang, D.-K. (2023). A Review on Dropout Regularization Approaches for Deep Neural Networks within the Scholarly Domain. *Electronics*, 12(14). <https://doi.org/10.3390/electronics12143106>
- Sarker, I. H. (2021). Machine Learning: Algorithms, Real-World Applications and Research Directions. *SN Computer Science*, 2(3), 160. <https://doi.org/10.1007/s42979-021-00592-x>
- Savchenko, A. V. (2017). Maximum-likelihood approximate nearest neighbor method in real-time image recognition. *Pattern Recognition*, 61, 459–469. <https://doi.org/10.1016/j.patcog.2016.08.015>
- Saxena, A., Fletcher, G., & Pechenizkiy, M. (2022). NodeSim: node similarity based network embedding for diverse link prediction. *EPJ Data Science*, 11(1), 24. <https://doi.org/10.1140/epjds/s13688-022-00336-8>
- Sayantana, M., Sriparna, S., & Mohammed, H. (2020). Multi-view clustering for multi-omics data using unified embedding. *Scientific Reports*, 13654
- Sevani, N., Azizah, K., & Jatmiko, W. (2023). A Feature-based Transfer Learning to Improve the Image Classification with Support Vector Machine. *International Journal of Advanced Computer Science and Applications*, 14(6). <https://doi.org/10.14569/IJACSA.2023.0140632>
- Sferrazza, P. (2023). Grey level co-occurrence matrix and learning algorithms to quantify and classify use-wear on experimental flint tools. *Journal of Archaeological Science: Reports*, 48, 103869. <https://doi.org/10.1016/j.jasrep.2023.103869>
- Sharifi-Noghabi, H., Peng, S., Zolotareva, O., Collins, C. C., & Ester, M. (07 2020). AITL: Adversarial Inductive Transfer Learning with input and output space adaptation for pharmacogenomics. *Bioinformatics*, 36(Supplement\_1), i380–i388. <https://doi.org/10.1093/bioinformatics/btaa442>
- Shankar, K., Dutta, A. K., Kumar, S., Joshi, G. P., & Doo, I. C. (2022). Chaotic Sparrow Search Algorithm with Deep Transfer Learning Enabled Breast Cancer

- Classification on Histopathological Images. *Cancers*, 14(11), 2770. <https://doi.org/10.3390/cancers14112770>
- Shi, Y., Li, L., Yang, J., Wang, Y., & Hao, S. (2023). Center-based Transfer Feature Learning With Classifier Adaptation for surface defect recognition. *Mechanical Systems and Signal Processing*, 188, 110001. <https://doi.org/10.1016/j.ymssp.2022.110001>
- Si, L., Dong, H., Qiang, W., Zheng, C., Yu, J., & Sun, F. (2024). Regularized Hypothesis-Induced Wasserstein Divergence for unsupervised domain adaptation. *Knowledge-Based Systems*, 283, 111162. <https://doi.org/10.1016/j.knosys.2023.111162>
- Sikha, O. K., & Bharath, B. (2022). VGG16-random fourier hybrid model for masked face recognition. *Soft Computing*, 26(22), 12795–12810. <https://doi.org/10.1007/s00500-022-07289-0>
- Siddhant, A., Goyal, A., & Metallinou, A. (2018). Unsupervised Transfer Learning for Spoken Language Understanding in Intelligent Agents. *ArXiv*, *abs/1811.05370*.
- Simon, P., & V, U. (2020). Deep learning based feature extraction for texture classification [Third International Conference on Computing and Network Communications (CoCoNet'19)]. *Procedia Computer Science*, 171, 1680–1687. <https://doi.org/10.1016/j.procs.2020.04.180>
- Sineglazov, V., Klanovets, O., & Riazanovskiy, K. (2022). Transitive transfer learning for lungs ct segmentation. *2022 IEEE 3rd International Conference on System Analysis Intelligent Computing (SAIC)*, 1–5. <https://doi.org/10.1109/SAIC57818.2022.9923023>
- Sitaula, C., & Hossain, M. B. (2021). Attention-based VGG-16 model for COVID-19 chest X-ray image classification. *Applied intelligence (Dordrecht, Netherlands)*, 51(5), 2850–2863. <https://doi.org/10.1007/s10489-020-02055-x>
- Solano, J. A., Lancheros, D. J., Umaña, S. F., & Coronado-Hernández, J. R. (2022). Predictive models assessment based on CRISP-DM methodology for students

performance in Colombia - Saber 11 Test. *Procedia Computer Science*, 198, 512–517. <https://doi.org/10.1016/j.procs.2021.12.278>

Srinivas, K., Gagana Sri, R., Pravallika, K., Nishitha, K., & Polamuri, S. R. (2024). COVID-19 prediction based on hybrid Inception V3 with VGG16 using chest X-ray images. *Multimedia Tools and Applications*, 83(12), 36665–36682. <https://doi.org/10.1007/s11042-023-15903-y>

Strömfelt, H., Dickens, L., Garcez, A.S., & Russo, A. (2020). On the Transferability of VAE Embeddings using Relational Knowledge with Semi-Supervision. *ArXiv*, *abs/2011.07137*.

Su, J., Li, J., Liu, X., Ranadive, T., Coley, C., Tuan, T.C., & Huang, F. (2022). Compact Neural Architecture Designs by Tensor Representations. *Front Artif Intell*. 2022 Mar 8;5:728761. <https://doi.org/10.3389/frai.2022.728761> . PMID: 35355829; PMCID: PMC8959219.

Su, Y., & Kabala, Z. J. (2023). Public Perception of ChatGPT and Transfer Learning for Tweets Sentiment Analysis Using Wolfram Mathematica. *Data*, 8(12). <https://doi.org/10.3390/data8120180>

Sun, H., Lang, W., Xu, C., Liu, N., & Zhou, H. (2023). Graph-based discriminative features learning for fine-grained image retrieval. *Signal Processing: Image Communication*, 110, 116885. <https://doi.org/10.1016/j.image.2022.116885>

Swaen, B. & George, T. (2024, March 18). *What Is a Conceptual Framework? | Tips & Examples*. Scribbr. Retrieved from <https://www.scribbr.com/methodology/conceptual-framework/>

Tagnamas, J., Ramadan, H., Yahyaouy, A., & Tairi, H. (2024). Multi-task approach based on combined CNN-transformer for efficient segmentation and classification of breast tumors in ultrasound images. *Visual Computing for Industry, Biomedicine, and Art*, 7(1), 2. <https://doi.org/10.1186/s42492-024-00155-w>

Talebi, S., Tong, E., Li, A., Yamin, G., Zaharchuk, G., & Mofrad, M. R. K. (2024).

- Exploring the performance and explainability of fine-tuned BERT models for neuroradiology protocol assignment. *BMC Medical Informatics and Decision Making*, 24(1), 40. <https://doi.org/10.1186/s12911-024-02444-z>
- Tang, H., & Jia, K. (2020). Discriminative Adversarial Domain Adaptation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04), 5940-5947. <https://doi.org/10.1609/aaai.v34i04.6054>
- Theodoridis, S. (2020). Probability and stochastic processes. *Machine Learning*
- Tian, M., Zhang, Y., Zhang, Y., Xiao, X., & Wen, W. (2024). A privacy-preserving image retrieval scheme with access control based on searchable encryption in media cloud. *Cybersecurity*, 7(1), 22. <https://doi.org/10.1186/s42400-024-00213-z>
- Togami, M., Masuyama, Y., Komatsu, T., & Nakagome, Y. (2019). Unsupervised training for deep speech source separation with kullback-leibler divergence based probabilistic loss function. *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 56–60.
- Toledo, A. S., Silini, R., Carpi, L. C., & Masoller, C. (2022). Outlier mining in high-dimensional data using the jensen–shannon divergence and graph structure analysis. *Journal of Physics: Complexity*, 3(4), 045011.
- Uyanik, B., & Orman, G.K. (2023). A Manhattan distance based hybrid recommendation system. *International Journal of Applied Mathematics Electronics and Computers*.
- Ugail, H., Stork, D. G., Edwards, H., Seward, S. C., & Brooke, C. (2023). Deep transfer learning for visual analysis and attribution of paintings by Raphael. *Heritage Science*, 11(1), 268. <https://doi.org/10.1186/s40494-023-01094-0>
- Varghese, B. A., Fields, B. K. K., Hwang, D. H., Duddalwar, V. A., Matcuk, G. R., Jr, & Cen, S. Y. (2023). Spatial assessments in texture analysis: what the radiologist needs to know. *Frontiers in radiology*, 3, 1240544. <https://doi.org/10.3389/>

[fradi.2023.1240544](https://doi.org/10.1007/s00521-021-05789-y)

- Van Molle, P., Verbelen, T., Vankeirsbilck, B., De Vylder, J., Diricx, B., Kimpe, T., ... Dhoedt, B. (2021). Leveraging the Bhattacharyya coefficient for uncertainty quantification in deep neural networks. *Neural Computing and Applications*, 33(16), 10259–10275. <https://doi.org/10.1007/s00521-021-05789-y>
- Vandikas, K., Moradi, F., Larsson, H., & Johnsson, A. (2024). Transfer Learning and Domain Adaptation in Telecommunications. IntechOpen. <https://doi.org/10.5772/intechopen.114932>
- Vijayaraj, A., Vasanth Raj, P. T., Jebakumar, R., Gururama Senthilvel, P., Kumar, N., Suresh Kumar, R., & Dhanagopal, R. (2022). Deep Learning Image Classification for Fashion Design. *Wireless Communications and Mobile Computing*, 2022(1), 7549397. <https://doi.org/10.1155/2022/7549397>
- Vrbančič, G., & Podgorelec, V. (2020). Transfer learning with adaptive fine-tuning. *IEEE Access*, 8, 196197–196211. <https://doi.org/10.1109/ACCESS.2020.3034343>.
- Wakili, M. A., Shehu, H. A., Sharif, M. H., Sharif, M. H. U., Umar, A., Kusetogullari, H., ... Uyaver, S. (2022). Classification of Breast Cancer Histopathological Images Using DenseNet and Transfer Learning. *Computational Intelligence and Neuroscience*, 2022(1), 8904768. <https://doi.org/10.1155/2022/8904768>
- Wang, J., Chen, Y., Feng, W., Yu, H., Huang, M., & Yang, Q. (2020). Transfer Learning with Dynamic Distribution Adaptation. *ACM Trans. Intell. Syst. Technol.*, 11(1). <https://doi.org/10.1145/3360309>
- Wang, J., & Dong, Y. (2020). Measurement of Text Similarity: A Survey. *Information*, 11(9). <https://doi.org/10.3390/info11090421>
- Wang, J., Lu, S., Wang, S.-H., & Zhang, Y.-D. (2022). A review on extreme learning machine. *Multimedia Tools and Applications*, 81(29), 41611–41660. <https://doi.org/10.1007/s11042-021-11007-7>

- Wang, L., & Sun, Y. (2022). Image classification using convolutional neural network with wavelet domain inputs. *IET Image Processing*, *16*(8), 2037–2048. <https://doi.org/10.1049/ipr2.12466>
- Wang, Q., Powell, M. A., Geisa, A., Bridgeford, E., Priebe, C. E., & Vogelstein, J. T. (2023). Why do networks have inhibitory/negative connections? *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, 22494–22502. <https://doi.org/10.1109/ICCV51070.2023.02061>
- Wang, T., & Tang, Y. (2022). Transfer-Reinforcement-Learning-Based rescheduling of differential power grids considering security constraints. *Applied Energy*, *306*, 118121. <https://doi.org/10.1016/j.apenergy.2021.118121>
- Wang, Y., Mantecón, H.L., Weijer, J.V., Lopez-Fuentes, L., & Raducanu, B. (2021). TransferI2I: Transfer Learning for Image-to-Image Translation from Small Datasets. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 13990-13999.
- Wang, Y., Zhang, Q., Wang, G.-G., & Cheng, H. (2024). The application of evolutionary computation in generative adversarial networks (GANs): a systematic literature survey. *Artificial Intelligence Review*, *57*(7), 182. <https://doi.org/10.1007/s10462-024-10818-y>
- Weber, M., Li, L., Wang, B., Zhao, Z., Li, B., & Zhang, C. (2022). Certifying Out-of-Domain Generalization for Blackbox Functions. *International Conference on Machine Learning*.
- Wei, H., Jia, K., Wang, Q., Ji, F., Cao, B., Qi, J., ... Yan, X. (2024). A texture feature extraction method considering spatial continuity and gray diversity. *International Journal of Applied Earth Observation and Geoinformation*, *130*, 103896. <https://doi.org/10.1016/j.jag.2024.103896>
- Wightman, R., Touvron, H., & Jégou, H. (2021). ResNet strikes back: An improved training procedure in timm. *ArXiv, abs/2110.00476*.
- Wu, H., Yan, Y., Ye, Y., Min, H., Ng, M. K., & Wu, Q. (2019). Online Heterogeneous

- Transfer Learning by Knowledge Transition. *ACM Trans. Intell. Syst. Technol.*, 10(3). <https://doi.org/10.1145/3309537>
- Wu, S. (2021). Expression Recognition Method Using Improved VGG16 Network Model in Robot Interaction. *Journal of Robotics*, 2021(1), 9326695. <https://doi.org/10.1155/2021/9326695>
- Wu, X., Chen, C., Li, P., Zhong, M., Wang, J., Qian, Q., ... Guo, Y. (2022). FTAP: Feature transferring autonomous machine learning pipeline. *Information Sciences*, 593, 385–397. <https://doi.org/10.1016/j.ins.2022.02.006>
- Xie, J., Hu, K., Guo, Y., Zhu, Q., & Yu, J. (2021). On loss functions and CNNs for improved bioacoustic signal classification. *Ecological Informatics*, 64, 101331. <https://doi.org/10.1016/j.ecoinf.2021.101331>
- Xu, Q., Wei, X., Bai, R., Li, S., & Meng, Z. (2023). Integration of deep adaptation transfer learning and online sequential extreme learning machine for cross-person and cross-position activity recognition. *Expert Systems with Applications*, 212, 118807. <https://doi.org/10.1016/j.eswa.2022.118807>
- Xu, H., Ebner, S., Yarmohammadi, M.A., White, A.S., Durme, B.V., & Murray, K. (2021). Gradual Fine-Tuning for Low-Resource Domain Adaptation. *ArXiv, abs/2103.02205*.
- Xu, J., Pan, Y., Pan, X., Hoi, S., Yi, Z., & Xu, Z. (2023). RegNet: Self-Regulated Network for Image Classification. *IEEE Transactions on Neural Networks and Learning Systems*, 34(11), 9562–9567. <https://doi.org/10.1109/TNNLS.2022.3158966>
- Xu, W., He, J., & Shu, Y. (2020). Transfer Learning and Deep Domain Adaptation. IntechOpen. <https://doi.org/10.5772/intechopen.94072>
- Xu, Z., Da, Yin., Qingyang, Z., Hongxia, Y., Zhilin, Y., & Jie, T. (2021). Controllable generation from pre-trained language models via inverse prompting. ArXiv preprint *ArXiv, abs/2103.10685*.

- Xu, Z., Guo, X., & Wang, J. (2024). Enhancing skin lesion segmentation with a fusion of convolutional neural networks and transformer models. *Heliyon*, *10*(10), e31395. <https://doi.org/10.1016/j.heliyon.2024.e31395>
- Yang, Q., Zhang, Y., Dai, W., & Pan, S. J. (2020). Feature-Based Transfer Learning. In *Transfer Learning* (pp. 34–44). chapter, Cambridge: Cambridge University Press.
- Yin, M. J., Wang, B., Dong, Y., & Ling, C. (2024). Source-Free Domain Adaptation for Question Answering with Masked Self-training. *Transactions of the Association for Computational Linguistics*, *12*, 721–737. <https://doi.org/10.1162/tacl.a.00669>
- Yones, C., Stegmayer, G., & Milone, D. H. (09 2017). Genome-wide pre-miRNA discovery from few labeled examples. *Bioinformatics*, *34*(4), 541–549. <https://doi.org/10.1093/bioinformatics/btx612>
- Yuan, B., Heiser, W., & de Rooij, M. (2022). A comparison of two dissimilarity functions for mixed-type predictor variables in the  $\delta$ -machine. *Advances in Data Analysis and Classification*, *16*(4), 875–907. <https://doi.org/10.1007/s11634-021-00463-6>
- Yun, S., Oh, S., Heo, B., Han, D., Choe, J., & Chun, S. (2021). Re-labeling ImageNet: from Single to Multi-Labels, from Global to Localized Labels. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2340-2350.
- Zeebaree, D.Q., Abdulazeez, A.M., Zebari, D.A., Haron, H., & Hamed, H.N. (2021). Multi-Level Fusion in Ultrasound for Cancer Detection based on Uniform LBP Features. *Cmc-computers Materials & Continua*, *66*, 3363-3382.
- Zhao, D., Zhang, P., Meng, J., & Wu, Y. (2022). Adversarial Transfer Learning for Named Entity Recognition Based on Multi-Head Attention Mechanism and Feature Fusion. *Natural Language Processing and Chinese Computing: 11th CCF International Conference, NLPCC 2022, Guilin, China, September 24–25, 2022, Proceedings, Part I*, 272–284. Presented at the Guilin, China.



[https://doi.org/10.1007/978-3-031-17120-8\\_22](https://doi.org/10.1007/978-3-031-17120-8_22)

- Zhao, X., Wang, L., Zhang, Y., Han, X., Deveci, M., & Parmar, M. (2024). A review of convolutional neural networks in computer vision. *Artificial Intelligence Review*, 57(4), 99. <https://doi.org/10.1007/s10462-024-10721-6>
- Zhang, C., Wang, J., Yan, T., Lu, X., Lu, G., Tang, X., & Huang, B. (2023). An instance-based deep transfer learning method for quality identification of Longjing tea from multiple geographical origins. *Complex & Intelligent Systems*, 9(3), 3409–3428. <https://doi.org/10.1007/s40747-023-01024-4>
- Zhang, S., Zhou, Y., Geng, P., & Lu, Q. (2024). Functional Neural Networks for High-Dimensional Genetic Data Analysis. *IEEE/ACM transactions on computational biology and bioinformatics*, 21(3), 383–393. <https://doi.org/10.1109/TCBB.2024.3364614>
- Zhang, Y., Huang, M., Chen, Y., Xiao, X., & Li, H. (2024). Land cover classification in high-resolution remote sensing: using Swin Transformer deep learning with texture features. *Journal of Spatial Science*, 1–25. <https://doi.org/10.1080/14498596.2024.2386317>
- Zhang, Z., Cheng, H., & Yang, T. (2020). A recurrent neural network framework for flexible and adaptive decision making based on sequence learning. *PLOS Computational Biology*, 16, 1–26. <https://doi.org/10.1371/journal.pcbi.1008342>.
- Zhang, Y., Zhang, Y., Zhou, G., Zhang, W., Li, K., Mu, Q., ... Tang, K. (2023). A New Ensemble Learning Method for Multiple Fusion Weighted Evidential Reasoning Rule. *Journal of Electrical and Computer Engineering*, 2023(1), 8987461. <https://doi.org/10.1155/2023/8987461>
- Zheng, W., Xue, F., Chen, Z., Chen, D., Guo, B., Shen, C., ... Pan, Y. (2023). Disruption prediction for future tokamaks using parameter-based transfer learning. *Communications Physics*, 6(1), 181. <https://doi.org/10.1038/s42005-023-01296-9>

- Zhou, X., Han, X., Li, H., Wang, J., & Liang, X. (2022). Cross-domain image retrieval: methods and applications. *International Journal of Multimedia Information Retrieval*, 11(3), 199–218. <https://doi.org/10.1007/s13735-022-00244-7>
- Zhu, D., Bu, Q., Zhu, Z., Zhang, Y., & Wang, Z. (2024). Advancing autonomy through lifelong learning: a survey of autonomous intelligent systems. *Frontiers in neurorobotics*, 18, 1385778. <https://doi.org/10.3389/fnbot.2024.1385778>
- Zhu, H., & Huang, J. (2022). A New Method for Determining the Embedding Dimension of Financial Time Series Based on Manhattan Distance and Recurrence Quantification Analysis. *Entropy*, 24(9). <https://doi.org/10.3390/e24091298>
- Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H., & He, Q. (2021). A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1), 43–76. <https://doi.org/10.1109/JPROC.2020.3004555>.
- Zhu, Q., Zhuang, H., Zhao, M., Xu, S., & Meng, R. (2024). A study on expression recognition based on improved mobilenetV2 network. *Scientific Reports*, 14(1), 8121. <https://doi.org/10.1038/s41598-024-58736-x>
- Zunino, L. (2024). Revisiting the Characterization of Resting Brain Dynamics with the Permutation Jensen–Shannon Distance. *Entropy*, 26(5). <https://doi.org/10.3390/e26050432>

## APPENDICES

### Appendix I: Author's Journal Publications during PhD Study

Wanjiku, R. N., Nderu, L., & Kimwele, M. (2023). Dynamic fine-tuning layer selection using Kullback–Leibler divergence. *Engineering Reports*, 5(5), e12595. doi:10.1002/eng2.12595

Wanjiku, R. N., Nderu, L., & Kimwele, M. (2023). Transfer learning data adaptation using conflation of low-level textural features. *Engineering Reports*, 5(5), e12603. doi:10.1002/eng2.12603

Wanjiku R. N., Nderu, L., & Kimwele, M. (2023). Improved transfer learning using textural features conflation and dynamically fine-tuned layers. *PeerJ Computer Science* 9:e1601 <https://doi.org/10.7717/peerj-cs.1601>

## **Appendix II: Author's Conference Publications during PhD Study**

Wanjiku, R., Nderu, L., & Kimwele, M. (2023). Dynamic Pre-trained Models Layer Selection Using Filter-Weights Cosine Similarity. In T. M. Ngatched Nkouatchah, I. Woungang, J.-R. Tapamo, & S. Viriri (Eds.), *Pan-African Artificial Intelligence and Smart Systems* (pp. 95–108). Cham: Springer Nature Switzerland.

Wanjiku, R., Nderu, L., & Kimwele, M. (2024). Improved Medical Imaging Transfer Learning through the Conflation of Domain Features. In J. Marx Gómez, A. Elikana Sam, & D. Godfrey Nyambo (Eds.), *Artificial Intelligence Tools and Applications in Embedded and Mobile Systems* (pp. 113–124). Cham: Springer Nature Switzerland.