PROGRESSIVE HIERARCHICAL MODELS FOR MULTI-CATEGORY IMAGE CLASSIFICATION

A THESIS SUBMITTED TO THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILMENT OF THE REQUIREMENTS
FOR THE DEGREE

TE-CHUAN (VICKY) KUO
SUPERVISOR: PROF. STEPHEN CHEN
SUPERVISORY COMMITTEE: PROF. MANAR JAMMAL

MASTER OF ARTS
GRADUATE PROGRAM IN
INFORMATION SYSTEMS AND TECHNOLOGY
YORK UNIVERSITY
TORONTO, ONTARIO
AUGUST 2024

Abstract

This thesis evaluates a hierarchical classification model applied to the CIFAR-10 dataset, focusing on addressing the limitations of existing methods, which often struggle with (i) overlapping features and (ii) poor interpretability of classification decisions. Designed to improve interpretability and targeted accuracy, the model demonstrates strengths in specifically targeted categorization through an incremental, multi-stage approach. While the model demonstrates strengths in targeted categorization, it introduces trade-offs in overall performance, particularly in categories with subtle differences. These insights provide a foundation for further research into optimizing such models for balanced accuracy across diverse categories.

Table of Contents

List of Tables **vii**

List of Figures **xi**

1 Introduction **1**

2 Background **6**

2.1 Supervised Learning for Image Classification 6

2.2 Evaluation of Image Classification Models 9

2.2.1 Accuracy 10

2.2.2 Confusion Matrix 10

2.3 Convolutional Neural Networks 11

3 Literature Review **14**

3.1 Convolutional Neural Networks 14

3.2 Deep Convolutional Neural Networks 16

3.3 Challenges of Convolutional Neural Networks 17

3.4 Hierarchical Models in Image Classification 21

4 Methodology **27**

4.1 Computational Tools and Framework 28

4.1.1 Deep Learning Framework 28

4.1.2 GPU Acceleration 28

4.2 Data 29

4.2.1 Dataset 29

4.2.2 Data Preprocessing 29

4.3 Model Architecture 33

4.3.1 Convolutional Layers 33

4.3.2 Convolutional Layer Design Considerations 36

4.3.3 Batch Normalization 37

4.3.4 Activations Functions 39

4.3.5 Residual Connections 41

4.3.6 Fully Connected Layers 42

4.4 Training Parameters 45

4.4.1 Reproducibility through Random Seed Setting 45
4.4.2 Batch Size 46
4.4.3 Number of Epochs 47
4.4.4 Optimizer 47
4.5 Evaluation Protocol 49
4.5.1 Accuracy in Baseline Model 49
4.5.2 Normalized Confusion Matrices 49
4.5.3 Accuracy in Hierarchical Models 51

5 Experiments **56**

5.1 Baseline Model 56
5.1.1 Analysis of Vehicle Classification 56
5.1.2 Analysis of Animal Classification 58
5.2 Hierarchical Classification Models 62
5.2.1 Two-Stage Hierarchical Classification for Broad Visual Distinction . . 62
5.2.2 Three-Stage Hierarchical Classification for Detailed Subgroup Discrimination 76
5.2.3 Four-Stage Hierarchical Classification for Animals 86

6 Summary **91**

7 Future Work **95**

8 Contributions **98**

Bibliography **100**

List of Tables

2.1 Confusion Matrix 11

5.1 Classification percentages of each class into vehicle and animal categories by the baseline model. These percentages are calculated by summing the relevant rates allocated to vehicle and animal types, as illustrated in Figure 5.1. . . . 64

5.2 Classification percentages of each class into vehicle and animal categories by the two-stage hierarchical classification model. These percentages are calculated by summing the relevant rates allocated to vehicle and animal types, as illustrated in Figure 5.5. 65

5.3 Difference in classification accuracy between the baseline model and the two-stage hierarchical model for vehicle and animal categories. The values represent the change in classification percentages, calculated by subtracting the baseline model values Table 5.1 from the two-stage model values for each category 5.2. 66

5.4 Classification percentages of each class into vehicle and animal categories by the two-stage hierarchical classification model. These percentages are calculated by summing the relevant rates allocated to vehicle and animal types, as illustrated in Figure 5.5. 70

5.5 Classification percentages of each class into vehicle and animal categories by the two-stage hierarchical classification model 70

5.6 Difference in classification accuracy between the baseline model and the two-stage hierarchical model for vehicle and animal categories. The values represent the change in classification percentages, calculated by subtracting the baseline model values Table 5.4 from the two-stage model values for each category 5.5. 71

5.7 Confusion matrices (Formula 4.12) for 'Vehicle' and 'Animal' superclasses comparison between the baseline model and the hierarchical classification model. 76

5.8 Confusion matrices for the second stage of the vehicles' three-stage model. . 78

5.9 Confusion matrices for airplanes and ships in the third stage of the vehicles' three-stage model. 79

5.10 Baseline and expected vs. achieved binary classification accuracies in the hierarchical model 80

5.11 Confusion matrices for automobiles and trucks in the third stage of the vehicles' three-stage model. 81

5.12 Confusion matrices for the second stage of the animals' three-stage model. . 82

5.13 Confusion matrices for deers and horses in the third stage of the animals' three-stage model. 82

5.14 Confusion matrices for birds and frogs in the third stage of the animals' three-stage model. 83

5.15 Confusion matrices for cats and dogs in the third stage of the animals' three-stage model. 84

5.16 Baseline and expected vs. achieved binary classification accuracies in the hierarchical model 85

5.17 Confusion matrices for the second stage of the animals' four-stage model. . . 87

5.18 Confusion matrices for ('Cat', 'Dog') and ('Deer', 'Horse') pairs in the third stage of the animals' four-stage model. 87

5.19 Confusion matrices for birds and frogs in the third stage of the animals' four-stage model. 89

5.20 Confusion matrices for cats and dogs in the fourth stage of the animals' four-stage model. 90

5.21 Confusion matrices for deers and horses in the fourth stage of the animals' four-stage model. 90

6.1 The overall accuracy of each class calculated by Formula 4.13. 93

List of Figures

2.1 Architecture of a neural network with an input layer, a hidden layer, and an output layer. 12

3.1 Illustration of the hierarchical classification process, where the dataset is progressively divided into subsets [57]. 22

3.2 Hierarchical Deep Convolutional Neural Network (HD-CNN) architecture [58]. The coarse category component and each branching fine category component can be implemented as standard deep CNN models. Branching components share shallow layers while having independent deep layers. 24

4.1 The example of CIFAR-10 dataset 30

4.2 Illustration of the operations within a neural network node, showing how inputs (X_i) are weighted (weights W_i), summed, and then transformed by an activation function to produce an output. This process underpins the functionality of convolutional layers in feature extraction and transformation. 34

4.3 Graphical representation of the activation functions, illustrating ReLu function's effect of zeroing out negative inputs and maintaining positive inputs unchanged. 40

4.4 Diagram illustrating the data flow through convolutional layers enhanced with residual connections. This setup highlights the feedforward path, batch normalization, and ReLU activations. The process starts with an input x that undergoes a 1st Convolution, followed by Batch Normalization and ReLU Activation. It then goes through a 2nd Convolution, Batch Normalization, and ReLU Activation to produce Output a. Simultaneously, a 3rd Convolution, Batch Normalization, and ReLU Activation occur to produce Output b. The residual connection adds Output a and Output b, resulting in the final output of the convolutional block, which is fed into a Fully Connected Layer to produce the classification result y. 44

5.1 Confusion matrix showing the accuracy of each class for the baseline model. The vertical axis represents the true classes, while the horizontal axis represents the predicted labels. The values in each row sum to 100%, indicating the distribution of predictions for each true class. Values in each column may not sum to 100%, reflecting the nature of misclassification errors. 61

5.2 Diagram of the Two-Stage Hierarchical Classification framework 63

5.3 Confusion matrix for the second stage within 'Vehicle' superclass in the two-stage hierarchical classification model. The vertical axis represents the true classes, while the horizontal axis represents the predicted labels. 67

5.4 Confusion matrix for the second stage within the 'Animal' superclass in the two-stage hierarchical classification model. The vertical axis represents the true classes, while the horizontal axis represents the predicted labels. 72

5.5 Confusion matrix for each class' accuracy of the two-stage hierarchical model. The vertical axis represents the true classes, while the horizontal axis represents the predicted labels. 75

5.6 Diagram of the Three-Stage Hierarchical Classification framework 77

5.7 If a confused pair in the second stage consists of Class A and Class B, we assume the original confusion rate splits equally between the two classes because of the added stage, leading to the expected binary classification accuracies. . . 79

5.8 Confusion matrix for each class' accuracy of the three-stage model. The vertical axis represents the true classes, while the horizontal axis represents the predicted labels. 86

5.9 Diagram of the four-stage hierarchical classification framework 88

5.10 Confusion matrix for each class' accuracy of the two-stage hierarchical model. The vertical axis represents the true classes, while the horizontal axis represents the predicted labels. 91

1. Introduction

Image classification remains a cornerstone of computer vision, vital for applications from automated tagging systems to autonomous vehicle navigation [1]. Traditional classification methods often struggle with overlapping features and subtle distinctions between categories [2].

Recent advancements in deep learning have led to the development of very deep convolutional neural networks (CNNs) [3, 4, 5, 6, 7, 8, 9] that have set new benchmarks in image classification accuracy. Models such as ResNet, which features residual learning frameworks to enable the training of networks that are substantially deeper than those previously used [10], and Google's EfficientNet, which scales up CNNs in a more structured manner, have demonstrated remarkable performance on standard datasets like ImageNet [10]. However, the increased depth and complexity of these models have also made their decision-making processes less interpretable. The layers of transformations in very deep CNNs (more than 100 layers) [11] tend to obscure the logic behind their classifications, rendering them opaque and challenging to audit, especially in applications where understanding the model's reasoning is

crucial.

Despite ongoing advancements, improvements in both accuracy and interpretability have been modest. This situation arises primarily from the limitations inherent in traditional classification architectures, particularly when applied to state-of-the-art models. While these advanced models, such as very deep convolutional neural networks, achieve high levels of accuracy, they often do so by becoming increasingly complex. This complexity tends to obscure the internal mechanisms of the models, reducing their interpretability. The need for transparency is particularly critical in safety-sensitive areas such as healthcare [12] and finance [13], where explaining and justifying model decisions is often mandated by regulations. In these domains, opaque models can hinder trust and compliance, making it essential to develop methods that enhance the explainability of deep learning systems.

To address the aforementioned concerns, this thesis introduces a hierarchical model to enhance transparency and targeted error correction. The hierarchical classification approach organizes the decision-making process into multiple layers, focusing on specific categories with high confusion rates. By structuring the decision process hierarchically, each stage can be examined independently, facilitating a clearer understanding of how classifications are made and allowing for incremental refinements. This layered approach also allows for targeted analysis of specific categories or pairs of categories, making it easier to understand and address the reasons behind certain misclassifications in subsequent layers.

The hierarchical model adopts a 'divide-and-conquer' strategy [14], where instead of tackling the entire classification problem at once, it breaks it down into smaller, more manageable sub-problems. For instance, rather than classifying all n elements simultaneously, the model first divides them into $n/2$ subcategories. Each subcategory is then further classified, greatly reducing the complexity at each stage. This method leverages the principle that smaller sub-problems should, theoretically, lead to fewer errors [15]. However, the initial classification into these subcategories introduces a new source of error. Thus, the thesis explores whether this hierarchical approach leads to overall improvements in classification accuracy, when and why it leads to improvements, and conversely, when and why it might result in worse performance.

The key research objectives (ROs) in this thesis to guide our exploration of hierarchical classification models are as follows:

1. **RO1:** Examine how using a consistent classifier ensures comparative fairness in hierarchical classification by isolating the hierarchical structure as the primary variable, rather than differing model architectures.

2. **RO2:** Identify the most confused class pairs in the baseline model and design a hierarchical model structure that specifically addresses these pairs, analyzing the impact on classification accuracy.

3. **RO3:** Evaluate the overall performance trade-offs of implementing a hierarchical model, focusing on changes in accuracy and training time, to understand the implications of hierarchical segmentation.

Using the CIFAR-10 dataset as a benchmark [16, 17, 18], the standalone model establishes a reference point for evaluating the improvements offered by the hierarchical models. Our experimental design progresses from a two-stage to a four-stage hierarchical model that relies on the divide-and-conquer principle, allowing us to examine the impact of each additional stage on performance compared to the baseline model. Within this framework, each model layer focuses on sets of categories frequently mistaken for one another. This approach enables targeted analysis and precise error correction at every stage[19].

To ensure the reproducibility of the experiments, we set the same random seed at the beginning of each experiment. This standardization controls all stochastic components of the model, making the training process deterministic and ensuring that identical results are produced when the experiments are repeated, even if conducted on different computers [20, 21, 22].

The novelty of this research lies in its iterative hierarchical model, which enhances both targeted accuracy and interpretability in image classification. This approach lays a robust foundation for further exploration and practical application of these strategies across diverse

and complex datasets.

The remainder of the thesis is organized as follows: **Chapter 2** introduces the background information on image classification and supervised learning, including an overview of convolutional neural networks (CNNs) and deep CNNs. **Chapter 3** reviews existing research on hierarchical models in image classification, discussing their advantages and challenges and identifying gaps that this thesis seeks to address. **Chapter 4** details the research methodology, including the dataset, preprocessing techniques, model architecture, and training parameters used in this study. **Chapter 5** presents the results of the experiments conducted with the baseline and hierarchical classification models. **Chapter 6** summarizes the key findings, and outlines the study's limitations. **Chapter 7** suggests directions for future research. **Chapter 8** concludes the research contributions.

2. Background

Image classification represents a critical application within the broader domain of artificial intelligence, where the goal is to categorize images into predefined classes based on their visual content. This task is primarily accomplished through supervised learning, a method wherein a model is trained on a dataset comprised of input-output pairs. The model learns to map inputs (in this case, images) to their correct outputs (class labels) by recognizing patterns and features that distinguish each class.

2.1 Supervised Learning for Image Classification

Supervised learning in the context of image classification is an iterative process where a computational model is trained on a dataset comprised of images, each paired with an optimizing label. The primary goal of such a model is to accurately predict the class of new, unseen images by leveraging the patterns and features it has discerned during the training

phase. This involves optimizing the model's internal parameters to minimize the discrepancy, often quantified by a loss function, between the predicted labels and the actual labels of the training images.

Algorithm 1 Supervised Learning Algorithm for Image Classification

Data: Training set $\mathcal{D} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$
Result: Optimized model parameters $\boldsymbol{\theta}$
Initialize model parameters $\boldsymbol{\theta}$
  Define learning rate α
  while *not converged* **do**
    Shuffle $\mathcal{D}$ at the start of each epoch
    Divide $\mathcal{D}$ into mini-batches of size M
    for *mini-batch* $\mathcal{B} = \{(x_i, y_i), \ldots\}$ **do**
      Compute the predicted outputs for $\mathcal{B}$ using $\hat{y}_i = f(x_i; \boldsymbol{\theta})$
      Compute the loss for $\mathcal{B}$ using $\mathcal{L}_{\mathcal{B}} = \frac{1}{M} \sum_{(x_i, y_i) \in \mathcal{B}} \text{LossFunction}(y_i, \hat{y}_i)$
      Compute the gradient of $\mathcal{L}_{\mathcal{B}}$ w.r.t. $\boldsymbol{\theta}$ as $\nabla_{\boldsymbol{\theta}} \mathcal{L}_{\mathcal{B}}$
      Update $\boldsymbol{\theta}$ by $\boldsymbol{\theta} = \boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\mathcal{B}}$
    end
  end
Return the learned parameters $\boldsymbol{\theta}$

The Algorithm 1 presents a step-by-step process designed explicitly for image classification tasks in supervised learning. This process involves adjusting the model's internal settings, or parameters, to improve its ability to identify the categories of images correctly.

The algorithm begins by initializing the model's parameters, denoted as θ. In machine learning, parameters like weights and biases are crucial. Weights control the impact of each input feature on the model's predictions, and biases allow adjustments to the output that are not dependent on the input values. Starting with random values for these parameters allows the model to explore possible solutions.

The learning rate (α) is a key value that determines how much the parameters change with each update. It's a delicate balance: too high, and the model might skip over the best solution; too low, and the model might take too long to find it or get stuck in a suboptimal solution.

The algorithm's core is an iterative process that continues until a predefined performance ('convergence') criterion is met, such as a specific number of loops ('epochs') or minimal improvement in loss.

Each time the loop starts (an 'epoch'), the training data ($\mathcal{D}$) is shuffled. This means changing the order of the data to ensure the model doesn't just memorize the sequence but learns the distinguishing features of each category.

After shuffling, the data is divided into 'mini-batches'. This makes computing more manageable and faster and helps the model to generalize better, meaning it can perform well on data it hasn't seen before.

Within each mini-batch, the algorithm employs the current set of model parameters θ to predict the class of each image. It then assesses the accuracy of these predictions by comparing them to the true labels of the images, a process that yields a numerical value known as the 'loss'. This loss quantifies the discrepancy between the predicted and actual labels, serving as a critical indicator of the model's performance on that particular mini-batch.

To enhance the model's predictive accuracy, the algorithm leverages a backpropagation

technique. Backpropagation calculates the gradient of the loss function concerning each parameter θ, effectively determining how parameter changes will impact the loss. This gradient information guides the model in adjusting its parameters, aiming to reduce the loss in subsequent iterations.

Specifically, the parameters are updated by moving them in the opposite direction of the loss gradient, a step conceptually akin to descending a hill to reach the lowest point. This updating process, governed by the learning rate α, incrementally improves the model's ability to make accurate predictions. By continuously iterating over mini-batches and updating the parameters based on backpropagation, the model systematically refines its understanding of the complex relationships within the data, ultimately enhancing its capability to classify images correctly.

2.2 Evaluation of Image Classification Models

Evaluating the performance of supervised machine learning models is crucial for understanding their effectiveness and identifying areas for improvement. Various evaluation techniques offer insights into different aspects of a model's performance.

2.2.1 Accuracy

Accuracy is the initial metric often used to evaluate an image classification model. It represents the proportion of correct predictions made by the model out of all predictions. Accuracy is calculated as:

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}}$$

While accuracy provides a straightforward measure of effectiveness, it can be misleading in imbalanced datasets where some classes are more prevalent than others.

2.2.2 Confusion Matrix

A confusion matrix offers a detailed breakdown of the model's performance by showing the counts of true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN). The confusion matrix is structured as follows:

		Predicted Class	
		Positive	**Negative**
Actual Class	**Positive**	TP	FN
	Negative	FP	TN

Table 2.1: Confusion Matrix

- **True Positives (TP)**: The number of instances correctly predicted as positive.
- **False Positives (FP)**: The number of instances incorrectly predicted as positive.
- **True Negatives (TN)**: The number of instances correctly predicted as negative.
- **False Negatives (FN)**: The number of instances incorrectly predicted as negative.

The confusion matrix provides a granular view of the model's strengths and weaknesses, highlighting specific areas for improvement.

2.3 Convolutional Neural Networks

Among supervised learning algorithms, Convolutional Neural Networks (CNNs) have become a cornerstone, particularly in image and video recognition [23, 24]. CNNs are designed to automatically and adaptively learn spatial hierarchies of features from input images. They

consist of an input layer, multiple hidden layers, and an output layer. The hidden layers of a CNN typically include convolutional layers and fully connected layers [25].

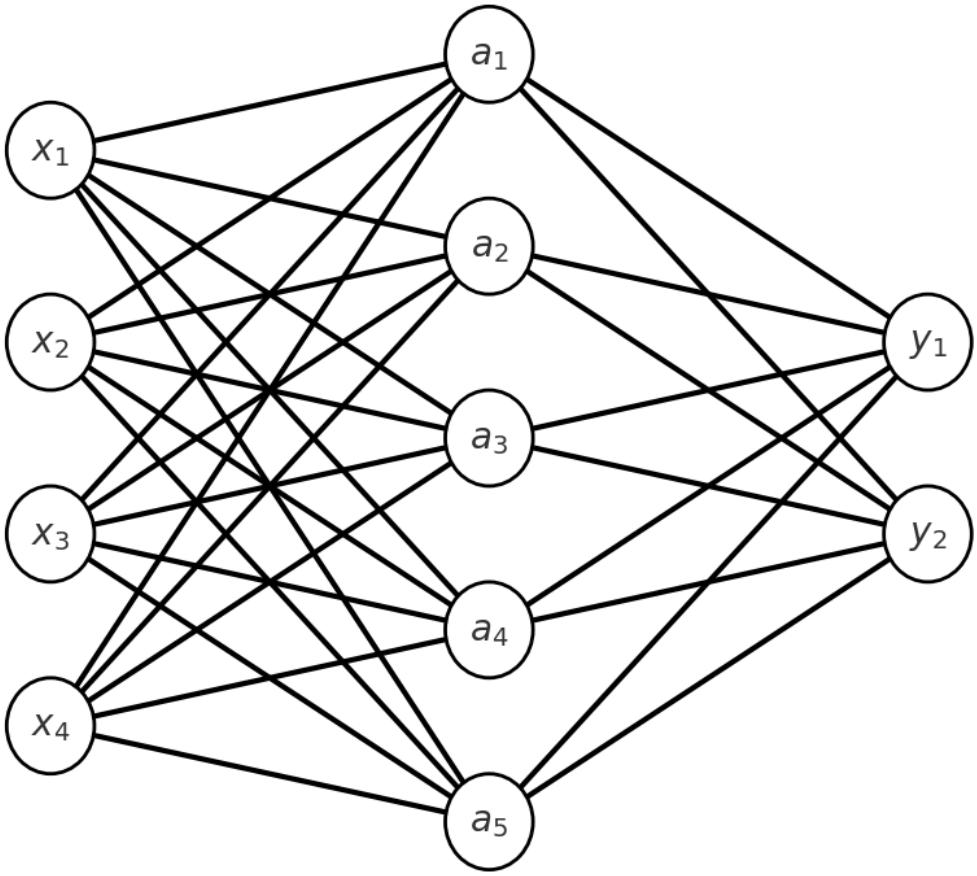

Figure 2.1: Architecture of a neural network with an input layer, a hidden layer, and an output layer.

Figure 2.1 illustrates the architecture of a neural network with an input layer, a hidden layer, and an output layer. The input layer consists of neurons (x_1, x_2, x_3, x_4) that receive the raw data. These neurons are connected to the hidden layer neurons $(a_1, a_2, a_3, a_4, a_5)$ through weighted connections, represented by different colored lines. The hidden layer processes the

inputs and passes the output to the final layer, which consists of output neurons (y_1, y_2) that produce the final prediction.

CNNs enhance this basic architecture by incorporating convolutional layers. These layers apply convolution operations to the input, extracting features like edges, textures, and patterns. These features are then passed through additional convolutional layers, further refining the feature maps. Finally, fully connected layers, similar to the hidden layers in traditional neural networks, are used to make the final prediction.

3. Literature Review

This chapter aims to provide a comprehensive understanding of the current state of research and identify gaps that this thesis seeks to address.

3.1 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) have emerged as a powerful class of deep learning models, particularly well-suited for tasks involving grid-like data structures, such as images. As described by Singh et al. [26], CNNs are designed with layers that progressively extract higher-level features from the input data, enabling robust and efficient processing of complex visual information.

The success of CNNs extends across various domains, as highlighted by Dhaka et al. [27]. Their survey explores applications in image recognition, object detection, and even medical image analysis. Notably, they showcase the flexibility and effectiveness of CNNs in

agriculture by demonstrating their ability to predict plant leaf diseases. Furthermore, Dhaka et al. [27] emphasize the importance of pre-processing techniques like resizing, normalization, and augmentation for improved CNN model accuracy and efficiency.

Optimizing the training process for CNNs is crucial, and one important factor is the training set batch size [28]. Radiuk [28]'s research explores the impact of batch size on CNN performance, emphasizing the importance of selecting an appropriate size to achieve optimal accuracy. Their study, which used datasets like MNIST and CIFAR-10, revealed that larger batch sizes generally lead to higher recognition accuracy, but they also come with increased computational demands.

Goyal et al. [29] further explored this concept by demonstrating that large mini-batch sizes can accelerate the training process without sacrificing model accuracy. They achieved this by utilizing advanced Stochastic Gradient Descent (SGD) variants, which help stabilize the training process and enhance the model's ability to generalize to unseen data [29]. This approach is particularly beneficial in scenarios where computational resources and training time are limited.

In practical implementations, frameworks like PyTorch have facilitated the development and training of CNNs [27]. PyTorch provides dynamic computation graphs, particularly useful for research and development as they allow for more flexible model design and debugging. This flexibility has made PyTorch a popular choice among researchers for implementing

state-of-the-art CNN architectures and experimenting with novel ideas.

3.2 Deep Convolutional Neural Networks

Deep Convolutional Neural Networks (DCNNs) have profoundly advanced the field of image classification [30, 31]. DCNNs differ from traditional CNNs in their depth and complexity. Traditional CNNs typically consist of a few convolutional layers, whereas DCNNs can have dozens or even hundreds of convolutional layers [31], which allow them to find more complex hidden patterns [32]. One notable advancement in DCNNs is the introduction of techniques like ReLU activation functions, which are piecewise linear functions that lead to faster convergence by introducing non-linearity [11]. In DCNNs, each network layer builds on the previous layers to create increasingly abstract and complex representations of the input data. This hierarchical structure allows the network to capture fine-grained details in the early layers and more abstract, high-level features in the deeper layers.

The success of DCNNs is also due to the availability of larger datasets [33, 34, 35], which, during training, impose a greater computational burden. Additionally, DCNN models have many parameters, which are the variables that the model learns during training. These parameters lead to increased storage and memory requirements [33, 36, 37]. For instance, the DCNN from Krizhevsky et al. [33] had 60 million parameters and took six days to train on

two Graphic Processor Units (GPUs), while the largest model presented by Simonyan and Zisserman [37] consisted of 144 million parameters, trained on four GPUs in two to three weeks. Consequently, subsequent research has focused on reducing the computational costs and storage space requirements of DCNNs [11].

To address these challenges, researchers have explored various solutions. Ensemble learning has demonstrated efficiency when integrated with DCNNs [38]. This approach involves training multiple instances of the same model with different random initializations and averaging their predictions in a parallel fashion. Similarly, hierarchical models in image classification share conceptual similarities with ensemble learning by utilizing multiple models or stages of processing. By decomposing the classification task into simpler, more manageable subtasks [39, 40, 41], hierarchical models can reduce overfitting and enhance generalization [39, 41]. Moreover, hierarchical classification can reduce the computational resources required for DCNNs, particularly in scenarios where broad category classifications suffice or where computational efficiency is a priority.

3.3 Challenges of Convolutional Neural Networks

Deep learning models, especially CNNs, are notoriously data-hungry, requiring vast amounts of labeled data to achieve optimal performance [42, 43, 44]. This issue is particularly

pronounced in domains with limited available data where obtaining labeled data is both time-consuming and expensive [45]. Win et al. [44] discuss this issue in their study on ensemble learning for COVID-19 detection, emphasizing the need for large datasets to avoid overfitting. Also, Boulent et al. [46] provide a comprehensive review of the use of CNNs for the automatic identification of plant diseases, emphasizing that robust model performance in real-world conditions requires datasets that capture the variability of field environments, including different lighting conditions, backgrounds, and disease symptoms. Boulent et al. [46] highlights the importance of data diversity, noting that models trained on images from controlled environments often fail to generalize to field conditions. They recommend using images captured under various conditions and incorporating different plant parts to improve model robustness. Additionally, Adadi [43] discusses the critical need for data-efficient algorithms in the big data era, highlighting that obtaining labeled data is both costly and labor-intensive, exacerbating the challenges faced by deep learning models in data-scarce environments.

In imbalanced datasets, CNNs tend to be biased towards the majority class, leading to poor performance on the minority class. Gomez et al. [9] propose data augmentation as a technique to mitigate these issues, which artificially increases the diversity of the training set by applying transformations to the existing data. Seth et al. [47] discuss the use of synthetic data generation, specifically using Generative Adversarial Networks (GANs), to create new

training examples and address the class imbalance. Ekim et al. [48] further emphasize the effectiveness of GANs for synthetic data generation, noting that these techniques can improve model robustness and performance on diverse datasets.

Furthermore, data quality is a pressing concern. Alzubaidi et al. [49] discuss how in-the-wild data often contains artifacts and noise, which affect deep learning models. They emphasize the importance of addressing these issues through various advanced augmentation techniques, such as Generative Adversarial Networks (GANs) and diffusion models, although their effectiveness can vary depending on the context and implementation [49].

An example that highlights these challenges is the classification of fruits and vegetables at supermarket self-checkouts [50]. This domain is characterized by massive variability in physical features such as color, texture, shape, and size, influenced by factors like ripeness and storage conditions. Constructing an exhaustive dataset to encompass all these variations is practically infeasible, complicating the classification task. A class distribution-aware adaptive margins approach with cluster embedding has been proposed to enhance the classification of highly variable classes by maintaining large inter-class separability and intra-class compactness [50].

The challenges of data quality and quantity are also evident in driver activity recognition systems, where CNNs are used to monitor driver behaviors. Roitberg et al. [42] emphasize that many classification errors in such systems stem from underrepresenting certain activities in the training set. Their study reveals that object- or movement-specific biases, which lead

to misclassifications, often arise due to the distribution and quality of the training data. They highlight the necessity for a more balanced and comprehensive dataset to improve model performance on rare behaviors.

In addition, training deep learning models is computationally intensive, often requiring hefty hardware resources [51, 52]. Banerjee and Chakraborty [51] highlight the challenges of using high-end graphic processor units (GPUs) and distributed computing infrastructures for applications with limited memory and computational resources, such as those running on mobile platforms. They propose a novel framework to address the problem of fine-tuning pre-trained deep learning models in resource-constrained applications by selecting an informative subset of training data, thereby reducing the computational load without significantly compromising performance [51]. Their method involves posing the subset selection as a constrained NP-hard integer quadratic programming problem and deriving an efficient linear relaxation to select a subset of exemplar instances. This requirement poses a barrier to entry for many researchers and practitioners, particularly those in resource-constrained environments [51]. Additionally, the energy consumption associated with training large models has raised concerns about the environmental impact of deep learning research [53, 52].

Barbierato et al. [52] discuss the energy consumption associated with training large models, raising concerns about the environmental impact of deep learning research. They emphasize the need for more energy-efficient algorithms and hardware to mitigate this

impact. Additionally, the high cost associated with cloud services such as Amazon Web Services, Microsoft Azure, and Google Cloud Platform can be prohibitive [51]. These services offer robust computational resources but come at a hefty price, and data privacy concerns further complicate their use. Consequently, local training becomes a viable alternative when considering privacy and cost issues.

Quetu et al. [53] address similar concerns, emphasizing the importance of developing simpler, less resource-intensive models to reduce the environmental footprint of deep learning. They advocate for green AI (Artificial Intelligence) principles, prioritizing energy efficiency and sustainability in AI research and practice.

3.4 Hierarchical Models in Image Classification

Hierarchical methods represent an advanced approach to classification. They combine multiple classifiers at various hierarchy levels to enhance accuracy and robustness [54, 55, 56]. Berno et al. [57] illustrates a hierarchical classification process where the classification problem is divided into subsets, each handled at different stages of the hierarchy, as shown in Figure 3.1. Initially, all data points (ALL) are considered. The process begins by classifying the first category (CAT 1) against the rest (REST 1). Once CAT 1 is classified, the remaining data (REST 1) is further divided to classify the next category (CAT 2) against the rest (REST

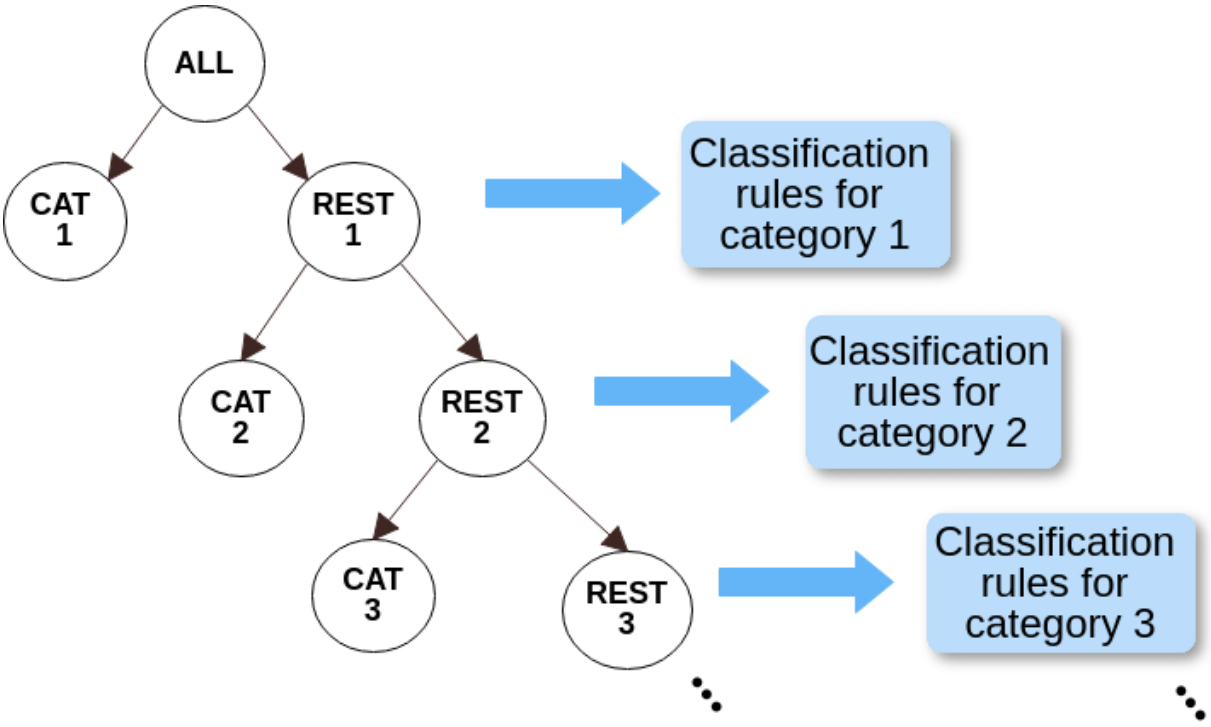

Figure 3.1: Illustration of the hierarchical classification process, where the dataset is progressively divided into subsets [57].

2). This continues until all categories are classified. This method allows for targeted error correction and enhances model interpretability by systematically narrowing down the decision space.

In their experiments, Berno et al. [57] used the Darknet framework for object detection on book covers, identifying relationships between visual attributes (such as predominant colors and objects) and book categories. They employed data mining techniques to analyze these relationships. Their findings revealed strong connections between co-purchased books and their visual features, demonstrating the potential of hierarchical classification in e-commerce

and recommendation systems. However, the methodology involves several computationally intensive steps, including object detection, color analysis, and data mining, making it challenging to scale this approach to larger datasets or real-time applications [57]. Additionally, validating the findings and the accuracy of the classification rules can be difficult due to the lack of a definitive ground truth. The subjective nature of book cover design and interpretation adds to this challenge.

Yan et al. [58] introduce the Hierarchical Deep Convolutional Neural Network (HD-CNN), which extends the hierarchical classification approach by incorporating a coarse-to-fine classification strategy within a deep learning framework [58]. HD-CNN first uses a 'coarse' classifier CNN to separate classes that are easy to distinguish. For example, it might easily differentiate between animals and vehicles. Then, for more difficult classifications, such as distinguishing between different types of animals, the model routes these to specialized 'fine' CNNs, each focusing on subsets of confusing classes. The final prediction is made by taking a weighted average of the predictions from all the branches, where the probabilities from the coarse category classifier determine the weights.

The architecture of HD-CNN is illustrated in Figure 3.2. Initially, an input image is processed by the coarse category CNN component, which generates broad (coarse) predictions. These predictions determine the routing of the image to one of several fine category branching components. Each fine branching component specializes in distinguishing between a subset

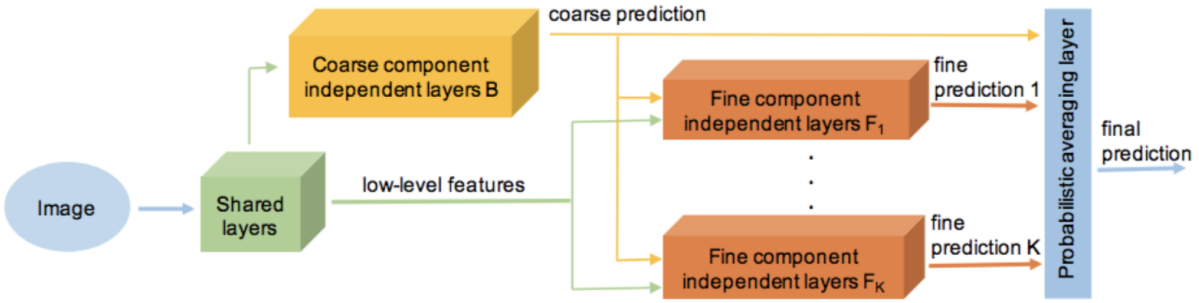

Figure 3.2: Hierarchical Deep Convolutional Neural Network (HD-CNN) architecture [58]. The coarse category component and each branching fine category component can be implemented as standard deep CNN models. Branching components share shallow layers while having independent deep layers.

of more similar and confusing categories. The predictions from these fine branches are then combined in a probabilistic averaging layer to produce the final, detailed (fine-grained) classification.

Yan et al. [58] demonstrated the effectiveness of HD-CNN on the CIFAR100 dataset. They compared HD-CNN's performance with standard CNN models and showed that HD-CNN outperformed these models. For instance, HD-CNN with CIFAR100 as the building block achieved a testing error of 32.62%, compared to 35.27% testing error for the standard CNN model, which improves the building block net by 2.65%. This method reduces interpretability because it becomes unclear which branch's prediction had the most influence on the final decision. It is also difficult to understand why certain branches were given more weight than others in the final prediction.

Another notable application of hierarchical classification methods is in the diagnosis

of Alzheimer's Disease (AD) and its early stage, mild cognitive impairment (MCI). Liu et al. [59] introduced a hierarchical classification method to analyze brain MR images, aiming to improve diagnostic accuracy by effectively handling high-dimensional imaging data. The model integrates various features extracted from both local brain regions and interactions between different regions. The process begins by partitioning the brain image into smaller local patches. For each patch, low-level classifiers are trained to transform local imaging features and inter-region correlations into higher-level features. Specifically, the local features include gray matter (GM) densities, while the inter-region features capture correlations between different brain regions. These transformed features are then used to train multiple high-level classifiers, each focusing on different larger brain regions. The outputs of these high-level classifiers are combined to make the final classification decision. This hierarchical approach decomposes the large-scale classification problem into smaller, more manageable tasks, improving accuracy and robustness [59]. The final decision is made by aggregating the results from the high-level classifiers using a method that assigns weights based on their performance (weighted voting strategy of ensemble learning) [59].

Liu et al. [59]'s method demonstrated high classification accuracies, achieving 92.0% accuracy for distinguishing AD from normal controls and 85.3% accuracy for distinguishing MCI from normal controls. This hierarchical classification approach proved effective in handling the complexity of medical imaging data and improved the interpretability and

diagnostic performance of traditional single-classifier methods.

However, similar to HD-CNN, this hierarchical method also reduces interpretability. This is because multiple intermediate steps and transformations influence each decision in this model. As a result, it becomes difficult to trace back and identify which specific features (e.g., gray matter densities or inter-region correlations) and which specific classifiers (e.g., low-level or high-level classifiers) were most responsible for a particular classification outcome. This complexity can obscure the decision-making process, making it difficult for clinicians to trust and validate the model's predictions. Additionally, Liu et al. [59] noted that while a three-level hierarchy provided the best performance, increasing the number of levels beyond three did not enhance performance and significantly increased computational costs. This highlights the balance needed between model complexity and interpretability, as well as the computational feasibility of deploying such models in practical settings.

4. Methodology

Building on the insights from the Literature Review, this research proposes a more straightforward yet interpretable approach to hierarchical methods for multi-category classification. The primary objective of the method design is to explore a more interpretable and scalable model with an easily available balanced dataset that can serve as a foundation for future enhancements in multi-category image classification.

The methodology involves starting with a baseline Convolutional Neural Network (CNN) and progressively adding hierarchical stages to refine classifications. While the training and testing principles remain consistent across all models, the hierarchical classification models incorporate a unique hierarchical data flow design.

4.1 Computational Tools and Framework

4.1.1 Deep Learning Framework

This research employs PyTorch, an open-source machine learning library developed by Facebook's AI Research Lab (FAIR), as the primary tool for model development, training, and evaluation. PyTorch offers a flexible and intuitive platform for building and training deep neural networks using Python. Its user-friendly interface is compatible with beginner and expert users, allowing for rapid prototyping and efficient experimentation with various model architectures [27].

Additionally, PyTorch provides a rich set of tools and utilities for tasks like data loading, preprocessing, and augmentation, as well as built-in support for GPU acceleration. These features streamline the development process and enable efficient utilization of hardware resources, making PyTorch well-suited for training deep neural networks on large-scale datasets.

4.1.2 GPU Acceleration

The research utilizes a CUDA-enabled GPU to accelerate processing and reduce training times. GPU acceleration is crucial for handling the extensive computations required in

training deep neural networks, especially when processing large datasets like CIFAR-10.

4.2 Data

4.2.1 Dataset

To bridge the research gap using an easily accessible and balanced benchmark dataset for image classification, this research utilizes the CIFAR-10 dataset [60] (Figure 4.1). CIFAR-10 consists of 60,000 32x32 color images evenly distributed among 10 classes, encompassing four types of vehicles and six types of animals: airplanes, automobiles, birds, cats, deer, dogs, frogs, horses, ships, and trucks. Each class contains 6,000 images, ensuring a balanced representation. The dataset is split into 50,000 training and 10,000 testing images, following a standard division widely used for evaluating different modeling approaches [60].

4.2.2 Data Preprocessing

To enhance the dataset's diversity and robustness, we employ data augmentation techniques during preprocessing [49], aiming to enhance the model's ability to generalize from the training data to unseen data. These techniques introduce variability and complexity into the training process, which is crucial for developing a robust neural network model. The employed techniques are as follows:

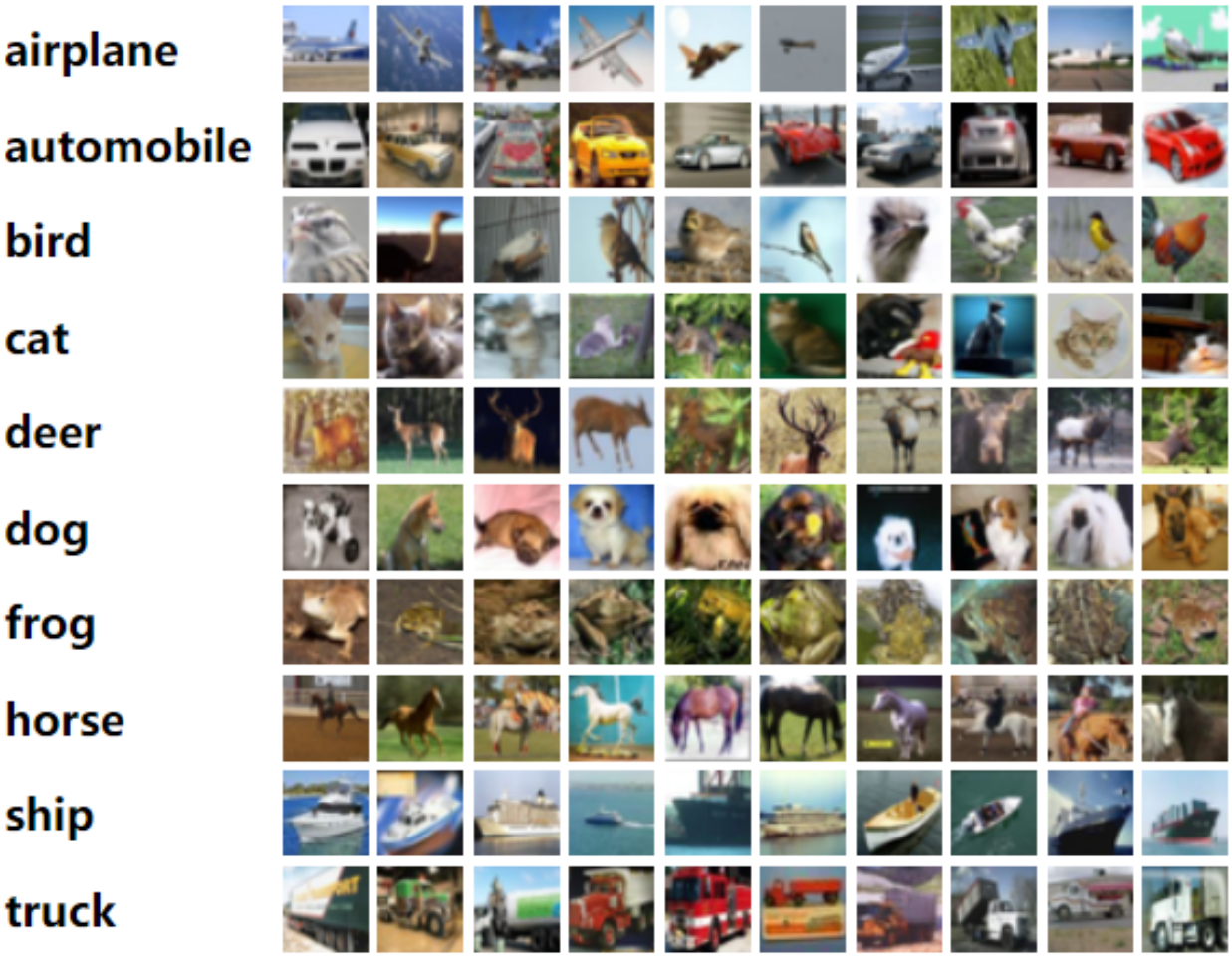

Figure 4.1: The example of CIFAR-10 dataset

- **Random Horizontal Flip:** Images are horizontally flipped with a certain probability, which introduces variations and helps the model learn invariant features. This strategy is particularly effective in teaching the model to identify features invariant to horizontal orientation. Such invariance is desirable in many real-world applications where the orientation of objects in images can vary.

- **Random Cropping:** A random crop is applied to the images, with padding of 4 pixels around the original 32x32 size based on common practice. This results in a 40x40 image. Then, a 32x32 crop is randomly selected from this padded image. This crop could be from the center, a corner, or any random position within the padded area. This technique introduces spatial variability by cropping images in different areas, which helps the model learn to recognize objects even when they are shifted, scaled, or partially occluded.

- **Normalization:** In our study, we implement a channel-wise normalization of the image data, a standard practice in deep learning for image processing tasks. This process involves adjusting the pixel values across the dataset to achieve a specific mean and standard deviation for each color channel in RGB images. The details of this normalization process are as follows:

  - **Mean Values:** The normalization process sets the mean of the pixel values to these

specific numbers for each of the three channels—Red, Green, and Blue—respectively. These mean values (0.4914 for Red, 0.4822 for Green, and 0.4465 for Blue) are computed based on the pixel intensity distributions across the entire dataset. Aligning the mean of each channel to these values ensures a consistent baseline for the input data, which facilitates the learning process of the neural network.

- **Standard Deviation Values:** Alongside mean normalization, we standardize the pixel values' spread or variance. The standard deviation for each channel is set to 0.247 for Red, 0.243 for Green, and 0.261 for Blue. This standardization ensures that the range of pixel values in each channel is consistent, mitigating issues related to scale and variance in the input data.

By applying data augmentation and tuning hyperparameters, we aim to:

- Increase dataset diversity and robustness.
- Mitigate overfitting by introducing variations in the training data.
- Ensure that the model converges efficiently during training [49].

4.3 Model Architecture

4.3.1 Convolutional Layers

The neural network model in this research starts with a single convolutional layer that begins the feature extraction process. Convolutional layers are a fundamental component of CNNs designed to process grid-like data such as images [61]. Each convolutional layer applies filters (kernels) to the input image to produce feature maps [61]. These filters slide over the image, computing dot products between the filter and patches of the input image [62]. This process enables the network to detect local features such as edges, textures, and shapes, which are crucial for image recognition tasks [62, 49].

The basic building block of neural networks is a neuron. A neuron in a neural network performs operations on the input data. This process is depicted in Figure 4.2 [63].

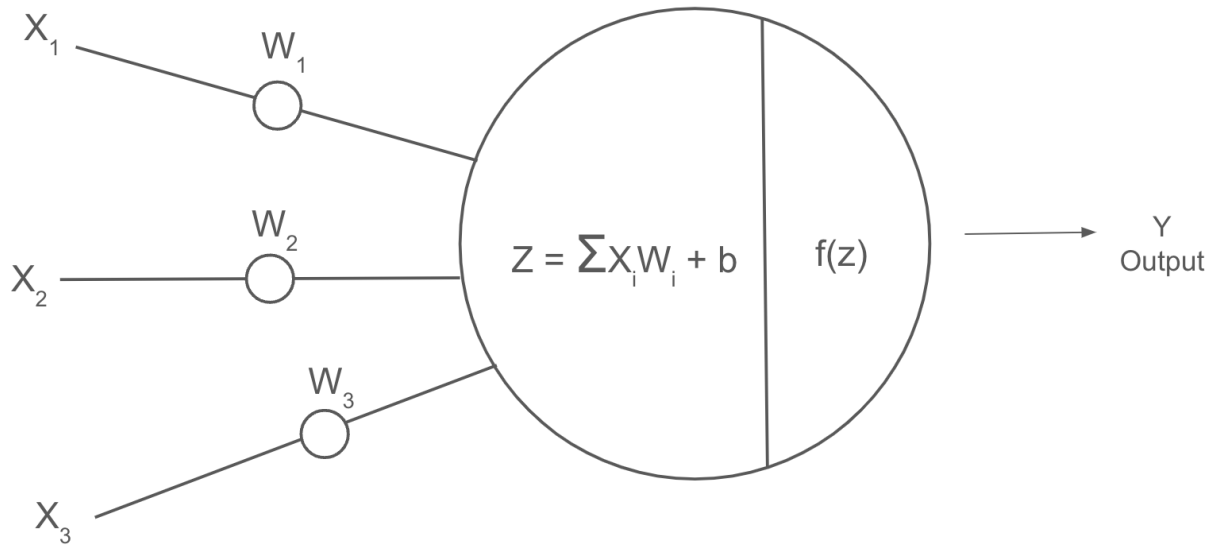

Figure 4.2: Illustration of the operations within a neural network node, showing how inputs (X_i) are weighted (weights W_i), summed, and then transformed by an activation function to produce an output. This process underpins the functionality of convolutional layers in feature extraction and transformation.

In a convolutional layer, the operations can be described as follows:

1. **Input (X_i)**: The input to a convolutional layer is typically a multidimensional array representing the image or feature map from the previous layer. For an RGB image, this would be a 3D array with dimensions corresponding to height, width, and color channels.

2. **Weights (W_i)**: Convolutional layers use small matrices of weights known as filters or kernels. Each filter is convolved with the input image to produce a feature map. The filter size is smaller than the input dimensions, typically 3x3 or 5x5.

3. **Summation (Z)**: For each filter position over the input image, a weighted sum of the input values and the filter weights is computed. This operation is called convolution. The result is a single value in the output feature map.

4. **Bias (b)**: An additional parameter, the bias, is added to the weighted sum to shift the output.

5. **Activation Function ($f(z)$)**: After the convolution operation, an activation function is applied to introduce non-linearity into the model. Common activation functions include the Rectified Linear Unit (ReLU), which replaces negative values with zero.

The entire process of convolution, summation, and activation is repeated for each filter across the entire input image, producing multiple feature maps that capture different aspects of the image's local patterns.

In this experiment, the neural network model includes three convolutional layers: one in the initial input layer and two in the hidden layers. This multi-layer setup allows the model to progressively extract more complex and abstract features from the input images, enhancing its ability to perform accurate image classification.

4.3.2 Convolutional Layer Design Considerations

The design of the convolutional layers in the model is tailored to optimize feature detection and facilitate effective learning:

- **Number of Filters:** Each layer uses 64 filters to ensure robust extraction of various features from the input images. This number allows the network to learn an extensive array of features without being overly computationally expensive [60].

- **Kernel Size:** A kernel size 3x3 is selected for all convolutional layers to capture sufficient spatial information without losing detail. This size balances capturing enough contextual data and maintaining reasonable computational efficiency [64].

- **Stride:** A stride of 1 is employed to process every pixel of the input image, preserving the spatial resolution and ensuring that valuable information is not skipped over during convolution. Using a stride of 1 allows the network to maintain the input dimensions and capture fine-grained image details [65].

- **Padding:** Padding of 1 is used to maintain the dimensionality of the feature maps, allowing edges and corners of the images to be processed adequately without distortion or loss of information. This technique, known as the same padding, ensures that the output feature maps have the same dimensions as the input, which is crucial for deep

networks where the spatial dimensions might otherwise shrink too quickly [66].

4.3.3 Batch Normalization

In this neural network model, a batch normalization layer follows each convolutional layer.

Batch normalization is a technique to improve the training of deep neural networks by normalizing the input of each mini-batch to have zero mean and unit variance [67]. The steps involved in batch normalization can be described as follows:

1. **Mini-Batch Mean (μ_B):** Calculate the mean of the mini-batch:

$$\mu_B = \frac{1}{m} \sum_{i=1}^{m} x_i \tag{4.1}$$

   where m is the number of examples in the mini-batch, and x_i represents the input values.

2. **Mini-Batch Variance (σ_B^2):** Calculate the variance of the mini-batch:

$$\sigma_B^2 = \frac{1}{m} \sum_{i=1}^{m} (x_i - \mu_B)^2 \tag{4.2}$$

3. **Normalization ($\hat{x}_i$):** Normalize the input values:

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \tag{4.3}$$

Here, ϵ is a small constant added to the variance to avoid division by zero.

4. **Scale and Shift:** Introduce two learnable parameters, γ and β, to scale and shift the normalized value:

$$y_i = \gamma \hat{x}_i + \beta \tag{4.4}$$

The parameters γ and β allow the model to undo the normalization if necessary and ensure that the batch normalization layer can represent the identity transformation if optimal for the learning process.

The batch normalization process can be summarized as follows:

$$\text{BN}(x_i) = \gamma \left(\frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \right) + \beta \tag{4.5}$$

By normalizing the inputs, batch normalization reduces the internal covariate shift, which refers to the change in the distribution of network activations due to the updating of parameters during training. This reduction helps stabilize the learning process and allows

higher learning rates, ultimately leading to faster convergence [67].

4.3.4 Activations Functions

Activation functions introduce non-linearity into the neural network models, allowing them to learn and represent complex patterns and relationships within the data [68, 69]. Without activation functions, neural networks would be limited to linear transformations, greatly reducing their modeling power and effectiveness. Activation functions are applied after each neuron computes its weighted sum and bias, transforming the result into an output that can be passed to the next layer in the network.

Several activation functions are commonly used in neural networks, including sigmoid, tanh, and Rectified Linear Unit (ReLU) [70, 68, 69]. In this research, the ReLU activation function is employed after the batch normalization layer normalizes and scales the data due to its simplicity and effectiveness.

Mathematically, the ReLU function is defined as:

$$f(x) = \max(0, x) \tag{4.6}$$

where z is the input to the activation function. The ReLU function outputs z if z is positive and 0 otherwise.

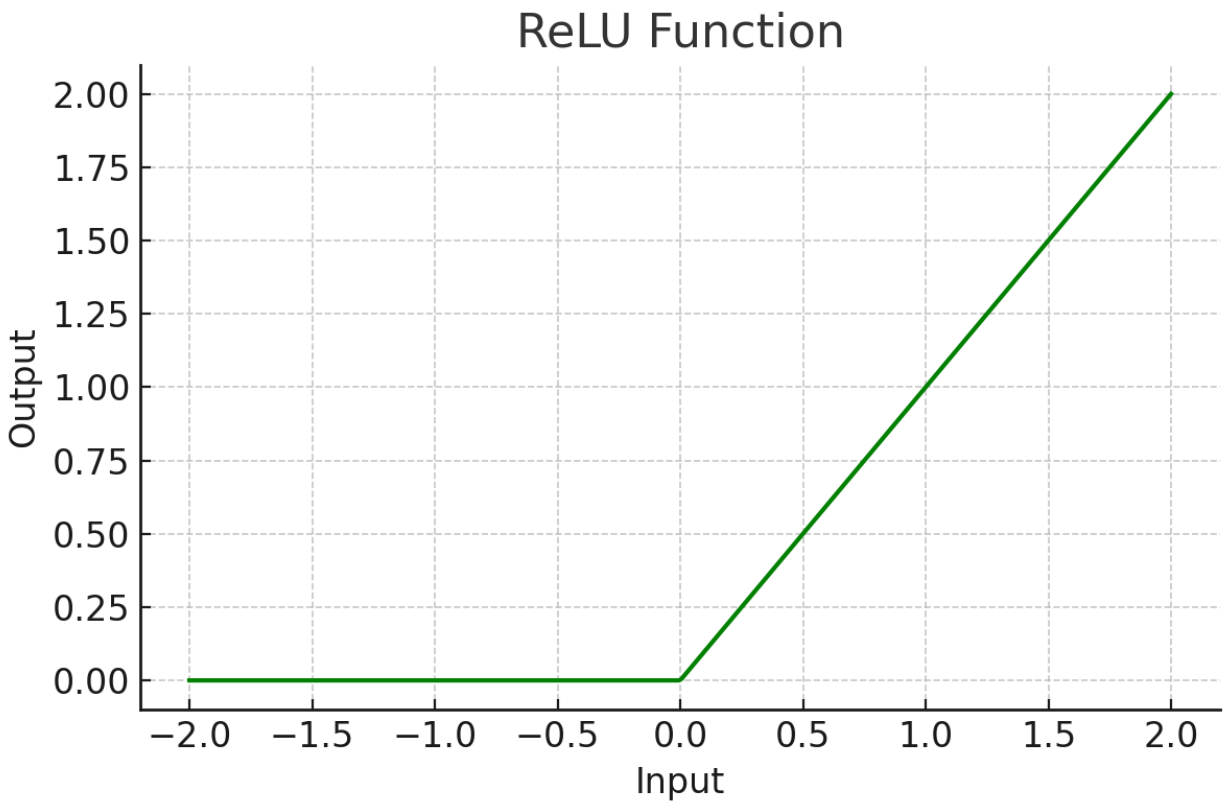

Figure 4.3: Graphical representation of the activation functions, illustrating ReLu function's effect of zeroing out negative inputs and maintaining positive inputs unchanged.

ReLU introduces non-linearity by outputting the input directly if it is positive; otherwise, it outputs zero, as Figure 4.3 shows. This mechanism allows ReLU to address some limitations of earlier activation functions, such as the vanishing gradient problem encountered with sigmoid and tanh functions [69]. The vanishing gradient problem occurs when gradients become extremely small during backpropagation, leading to slow or stalled learning in deeper layers. ReLU mitigates this issue by maintaining gradients for positive values, thereby ensuring that the network continues to learn effectively. Furthermore, ReLU is computationally

efficient because it involves simple thresholding at zero, which speeds up the training process compared to the more complex computations required for sigmoid and tanh functions. Its sparsity-inducing property, where neurons are deactivated (output zero) for negative inputs, can also lead to more efficient representations and improved generalization. Given these benefits, ReLU is chosen in this research to enhance the performance and efficiency of the hierarchical classification models, enabling them to learn more effectively and handle complex patterns within the CIFAR-10 dataset.

4.3.5 Residual Connections

In our neural network model, residual connections [71] are introduced after every second convolutional layer. Residual connections, or skip connections, are an essential component of modern deep neural networks. These connections help mitigate the vanishing gradient problem and enable the training of much deeper networks by allowing gradients to flow more easily through the network during backpropagation.

A residual connection essentially allows the input of a layer to bypass the layer and be added directly to the output. This approach helps preserve the information from earlier layers and stabilizes the training process. The main idea is to learn a residual mapping, denoted as $\mathcal{F}(x)$, which represents the difference between the desired mapping $\mathcal{H}(x)$ and the identity

mapping x. Mathematically, this can be expressed as:

$$\mathcal{H}(x) = \mathcal{F}(x) + x \tag{4.7}$$

where $\mathcal{H}(x)$ is the original mapping to be learned, and x is the input to the layer. The function $\mathcal{F}(x)$ represents the residual mapping that is easier to optimize.

4.3.6 Fully Connected Layers

Fully connected layers, also known as dense layers, are an integral part of neural networks, especially in the final stages where high-level reasoning is required [25]. Unlike convolutional layers, which focus on extracting spatial features from the input data, fully connected layers process these features to make the final classification decisions [25]. In a fully connected layer, each neuron is connected to every neuron in the previous layer, allowing the network to learn complex representations and relationships between features.

A fully connected layer performs a linear transformation followed by an activation function. Mathematically, the output y of a fully connected layer can be expressed as:

$$y = f(Wx + b) \tag{4.8}$$

where x is the input vector, typically a flattened feature map from the previous layer, W is the weight matrix, b is the bias vector, and f is the activation function for the final layer.

In our neural network model, the fully connected layer follows the convolutional and residual layers. After the final convolutional layer, the feature maps are flattened into a one-dimensional vector, which serves as the input to the fully connected layer. This layer processes the input to produce the final class scores for the CIFAR-10 dataset.

The overall structure and flow of these elements, including the convolutional layers, batch normalization, ReLU activations, residual connections, and fully connected layers, are visually represented in Figure 4.4.

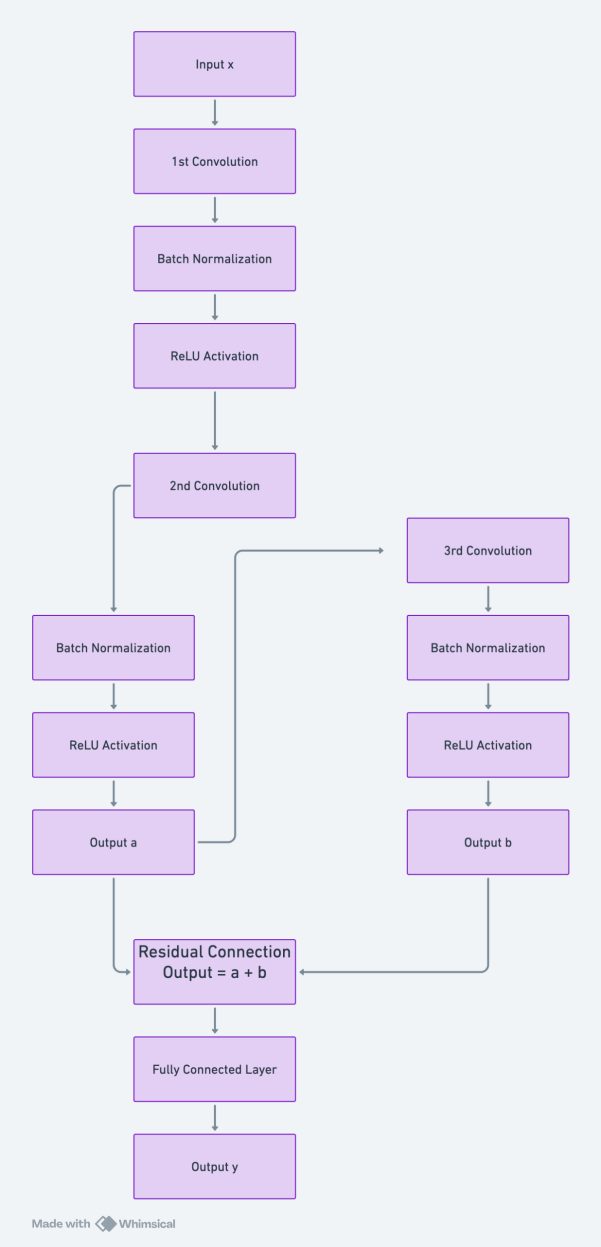

Figure 4.4: Diagram illustrating the data flow through convolutional layers enhanced with residual connections. This setup highlights the feedforward path, batch normalization, and ReLU activations. The process starts with an input x that undergoes a 1st Convolution, followed by Batch Normalization and ReLU Activation. It then goes through a 2nd Convolution, Batch Normalization, and ReLU Activation to produce Output a. Simultaneously, a 3rd Convolution, Batch Normalization, and ReLU Activation occur to produce Output b. The residual connection adds Output a and Output b, resulting in the final output of the convolutional block, which is fed into a Fully Connected Layer to produce the classification result y.

4.4 Training Parameters

The performance of a neural network heavily depends on the choice of training parameters. These parameters control various aspects of the training process, including how the model learns from the data, the speed of learning, and the stability of the training. The key training parameters in this research are batch size, learning rate, momentum, the number of epochs, and the choice of optimizer.

4.4.1 Reproducibility through Random Seed Setting

To ensure the reproducibility of the experiments, the same random seed is set at the beginning of each experiment. This standardization controls all stochastic components of the model, making the training process deterministic. As a result, identical results are produced when the experiments are repeated, even if conducted on different computers [20, 21, 22].

- **Role of the Seed:** The random seed functions as a fixed numerical starting point for generating sequences of random numbers. This guarantees that each run of the model when initiated with the same seed, produces identical sequences of random numbers that are used throughout the training process. Consequently, each model run is identical, allowing direct comparisons between different training configurations under controlled

conditions.

- **Implementation Details:** The seed is implemented in the PyTorch framework with the following command:

```
torch.manual_seed(42)
```

This command sets PyTorch's random number generator to '42', a commonly used example value that ensures repeatability. It is crucial to set this seed before initializing the model or processing any training data to maintain consistency across training sessions.

4.4.2 Batch Size

In this research, a batch size of 8 is used for both training and testing. Batch size is a parameter that defines the number of training examples used in one iteration [72]. A smaller batch size allows for more frequent updates to the model weights, leading to faster convergence but might introduce noise in the gradient estimates. Conversely, a larger batch size provides more stable gradient estimates but requires more memory and might lead to slower convergence. The choice of batch size in this research strikes a balance between efficient memory usage and the stability of gradient updates.

4.4.3 Number of Epochs

In this research, the number of epochs is set to 20. The number of epochs defines how often the entire training dataset is passed through the network during training. A higher number of epochs allows the model to learn more from the data, but it also increases the risk of overfitting [49] if the model starts to learn noise in the training data. The epoch value is selected to be a starting point for exploration, allowing us to assess the model's behavior in early training stages before committing to longer training sessions.

4.4.4 Optimizer

In this research, we use the Stochastic Gradient Descent (SGD) optimizer with a momentum of 0.9 and a learning rate of 0.001. These parameters are chosen based on standard practice and empirical experimentation to ensure a good balance between convergence speed and training stability [73]. An optimizer determines how the model's weights are updated based on the calculated gradients of the loss function, and the choice of optimizer can greatly impact the speed and efficiency of the training process.

SGD is a widely used optimization algorithm known for its simplicity and effectiveness. Unlike traditional gradient descent, which uses the entire dataset to compute the gradient of the loss function, SGD updates the model weights iteratively using small batches of data.

This approach allows for more frequent updates, leading to faster convergence.

SGD with momentum further enhances the optimization process by incorporating past gradients into the current update. This helps smooth the updates, accelerate convergence, and avoid oscillations. The momentum term essentially helps the optimizer navigate through the optimization landscape more effectively. The momentum parameter is typically set between 0 and 1, where a value closer to 1 can lead to faster convergence.

The learning rate determines the step size at which the model weights are updated during training. It controls how quickly or slowly a neural network learns. A higher learning rate can lead to faster convergence but might cause the training process to overshoot the optimal solution. Conversely, a lower learning rate ensures more stable convergence but can make the training process excessively slow.

Mathematically, the weight update rule for SGD with momentum is given by:

$$v_t = \mu v_{t-1} + \eta \nabla_w J(w) \tag{4.9}$$

$$w_{t+1} = w_t - v_t \tag{4.10}$$

where: - v_t is the velocity at time step t, - μ is the momentum coefficient, - η is the learning rate, - $\nabla_w J(w)$ is the gradient of the loss function for the weights w, - w_{t+1} are the updated

weights.

4.5 Evaluation Protocol

4.5.1 Accuracy in Baseline Model

Accuracy is a fundamental metric for evaluating classification models, representing the proportion of correctly predicted instances out of the total instances [74]. It provides a straightforward measure of the overall effectiveness of the baseline model. The accuracy is calculated as:

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}} \tag{4.11}$$

While accuracy is an important measure, accuracy alone does not provide a complete picture because it does not reveal where the misclassifications occur or which classes are causing the confusion.

4.5.2 Normalized Confusion Matrices

Confusion matrices provide a detailed breakdown of the model's performance by showing the counts of true positive, false positive, true negative, and false negative predictions for each class [75, 74]. Normalized confusion matrices are particularly useful as they represent

these counts as proportions, allowing for easier comparison across classes regardless of their representation in the dataset [74]. The entries of a normalized confusion matrix are calculated as follows:

$$\text{Normalized Entry}_{i,j} = \frac{\text{Count of Class } i \text{ predicted as Class } j}{\text{Total Count of Class } i} \tag{4.12}$$

Where:

- Normalized $\text{Entry}_{i,j}$ represents the normalized value for the number of instances of Class i that were predicted as Class j.
- Count of Class i predicted as Class j is the raw count of instances from Class i that were classified as Class j.
- Total Count of Class i is the total number of instances of Class i in the dataset.

An essential principle of normalized confusion matrices is that the entries in each row sum to 100%. This indicates the distribution of predictions for each actual class. However, the entries in each column do not necessarily sum to 100% as they represent the distribution of predicted classes across different actual classes.

4.5.3 Accuracy in Hierarchical Models

In a hierarchical classification model, the overall accuracy for each specific category is influenced by the accuracy of each stage in the hierarchy. The first stage usually classifies images into broad categories (e.g., animals vs. vehicles), and subsequent stages classify images within those broad categories into specific categories. The overall accuracy for a specific category is determined by tracking the correct classifications through each stage of the hierarchy. Let:

- N be the total number of images.
- C be the specific category (e.g., dogs).
- A_{broad} be the accuracy of the broad classification stage.
- A_{specific} be the accuracy of the specific classification stage within the broad category.

The overall accuracy for the specific category C can be calculated as:

$$OverallAccuracy(C) = \frac{\text{Number of images correctly classified as } C}{N} \tag{4.13}$$

Example

Suppose we have a dataset consisting of 40 images, equally distributed among 4 categories: dogs, cats, automobiles, and trucks. Each category has 10 images. We use a two-stage hierarchical model for classification. The first stage classifies images into broad categories: animals or vehicles. The second stage further classifies the images into specific categories within each superclass.

In the first stage, there are 40 images in total (10 dogs, 10 cats, 10 automobiles, 10 trucks). The classification results are as follows:

- 13 out of 20 animal images (dogs + cats) are correctly classified as animals.
- 7 out of 20 animal images are misclassified as vehicles.
- 15 out of 20 vehicle images (automobiles + trucks) are correctly classified as vehicles.
- 5 out of 20 vehicle images are misclassified as animals.

The confusion matrix for the first stage is:

	Animals	Vehicles
Animals	$\frac{13}{20}$	$\frac{7}{20}$
Vehicles	$\frac{5}{20}$	$\frac{15}{20}$

Within each superclass (animals and vehicles), images are further classified into specific categories in the second stage. For the animal superclass:

- Dogs: 5 out of 6 dog images (correctly classified as animals in the first stage) are correctly classified as dogs.
- Cats: 5 out of 7 cat images (correctly classified as animals in the first stage) are correctly classified as cats.

For the vehicle superclass:

- Automobiles: 6 out of 8 automobile images (correctly classified as vehicles in the first stage) are correctly classified as automobiles.
- Trucks: 5 out of 7 truck images (correctly classified as vehicles in the first stage) are correctly classified as trucks.

The confusion matrix for the second stage is:

	Cats	Dogs
Cats	$\frac{5}{7}$	$\frac{2}{7}$
Dogs	$\frac{1}{6}$	$\frac{5}{6}$

	Automobiles	Trucks
Automobiles	$\frac{6}{8}$	$\frac{2}{8}$
Trucks	$\frac{2}{7}$	$\frac{5}{7}$

The overall accuracy for each specific category is calculated by considering the correct classifications at each stage.

For dogs, 13 out of 20 images are correctly classified as animals in the first stage. In the second stage, 5 out of 6 dog images (correctly classified as animals) are correctly classified as dogs. Therefore, the overall accuracy for dogs is:

$$\text{Overall Accuracy (Dogs)} = \frac{\text{Number of correctly classified dogs}}{\text{Total number of dogs}} = \frac{5}{10} = 50\%$$

For cats, 13 out of 20 images are correctly classified as animals in the first stage. In the second stage, 5 out of 7 cat images (correctly classified as animals) are correctly classified as cats. Therefore, the overall accuracy for cats is:

$$\text{Overall Accuracy (Cats)} = \frac{\text{Number of correctly classified cats}}{\text{Total number of cats}} = \frac{5}{10} = 50\%$$

For automobiles, 15 out of 20 images are correctly classified as vehicles in the first stage.

In the second stage, 6 out of 8 automobile images (correctly classified as vehicles) are correctly classified as automobiles. Therefore, the overall accuracy for automobiles is:

$$\text{Overall Accuracy (Automobiles)} = \frac{\text{Number of correctly classified automobiles}}{\text{Total number of automobiles}} = \frac{6}{10} = 60\%$$

For trucks, 15 out of 20 images are correctly classified as vehicles in the first stage. In the second stage, 5 out of 7 truck images (correctly classified as vehicles) are correctly classified as trucks. Therefore, the overall accuracy for trucks is:

$$\text{Overall Accuracy (Trucks)} = \frac{\text{Number of correctly classified trucks}}{\text{Total number of trucks}} = \frac{5}{10} = 50\%$$

The overall accuracy for each specific category in a hierarchical model is influenced by the correct classifications at each stage. The overall accuracy is not simply the product of the stage accuracies but rather the fraction of the total images that are correctly classified through all stages. The hierarchical structure ensures that the overall accuracy reflects the performance of the model across all classification stages.

5. Experiments

This chapter describes the experimental methodologies used to establish a baseline CNN model for initial performance benchmarking and iteratively build hierarchical models, progressing from a two-stage to a four-stage hierarchy. Each stage is designed based on insights from the previous model's confusion matrix analysis, focusing on refining classification accuracy.

5.1 Baseline Model

5.1.1 Analysis of Vehicle Classification

1. **Automobile:** The model achieves an accuracy of 81.0%, with an average confusion rate of 2.1%. The classes that exceed this average confusion rate are trucks (10.4%), ships (2.3%), and airplanes (2.3%). The confusion between automobiles and trucks is notably high, indicating a potential area for model improvement.

2. **Truck:** With the highest accuracy among vehicle classes at 84.8%, the truck model has

an average confusion rate of 1.7%. The vehicle classes most commonly confused with trucks, surpassing this average confusion rate, are automobiles (4.3%), ships (3.6%), airplanes (2.4%), and cats (2.0%). Interestingly, trucks are seldom confused with other animals: 0.3% with birds, 0.5% with deers, and 0.7% with dogs, but they are obviously confused with cats.

3. **Ship:** The ship model has a high accuracy of 84.3% and an average confusion rate of 1.7%. The classes with confusion rates exceeding this average are airplanes (5.9%), trucks (2.1%), and automobiles (1.8%).

4. **Airplane:** The airplane model has the lowest accuracy among vehicle types at 73.5%, with an average confusion rate of 2.9%. The classes most frequently confused with airplanes, above this average confusion rate, are ships (8.2%), birds (5.2%), and trucks (4.8%). Notably, 'Bird' is the most frequently confused animal category within the vehicle classification (not just for airplanes but across all vehicle types).

All vehicle types except for airplanes achieved accuracies exceeding the average accuracy of 80.9%. Trucks have the highest accuracy at 84.8%, followed by ships at 84.3%, and automobiles at 81.0%. In contrast, airplanes have a noticeably lower accuracy of 73.5%. The average confusion rate among vehicles is 2.1%. The most frequent confusion pairs are ('Airplane', 'Ship') and ('Automobile', 'Truck'), contributing to an average confusion rate of

7.1% and 7.4%, respectively. Specifically, 8.2% of airplanes are misclassified as ships, and 5.9% of ships are misclassified as airplanes. For the ('Automobile', 'Truck') pair, 10.4% of automobiles are misclassified as trucks, and 4.3% of trucks are misclassified as automobiles.

5.1.2 Analysis of Animal Classification

1. **Frog:** The model achieves the highest accuracy among animals at 85.7%, with an average confusion rate of 1.6%. The classes with confusion rates above this average are cats (5.4%), dogs (3.1%), and birds (3.0%).

2. **Cat:** The model has the lowest accuracy among animals at 52.6%, with an average confusion rate of 5.3%. The classes with confusion rates above this average are dogs (20.7%), frogs (8.4%), and birds (6.1%).

3. **Dog:** The model achieves an accuracy of 71.7%, with an average confusion rate of 3.1%. The classes with confusion rates above this average are cats (14.1%), horses (3.6%), and frogs (3.0%).

4. **Bird:** The model achieves an accuracy of 55.2%, with an average confusion rate of 5.0%. The classes with confusion rates above this average are frogs (10.8%), dogs (7.9%), airplanes (7.3%), deers (7.2%), and cats (6.2%). Notably, birds are rarely mistaken for other vehicles, such as automobiles (0.0%), trucks (0.9%), and ships (1.3%). However,

they are often confused with airplanes, with a rate of 7.3%, making this the most frequent confusion between a vehicle and an animal category.

5. **Deer:** The model achieves an accuracy of 64.0%, with an average confusion rate of 4.0%. The classes with confusion rates above this average are frogs (8.1%), cats (7.3%), birds (6.3%), dogs (5.9%), and horses (4.9%).

6. **Horse:** The model achieves an accuracy of 74.5%, with an average confusion rate of 2.8%. The classes with confusion rates above this average are dogs (8.7%), cats (4.9%), deers (4.7%), and birds (3.1%).

As seen in Figure 5.1, the average accuracy of animal classes in the CIFAR-10 dataset is 67.3%. Frogs (85.7%), horses (74.5%), and dogs (71.7%) exhibit higher accuracies, while deers (64.0%), birds (55.2%), and cats (52.6%) show lower accuracies. This indicates that frogs, horses, and dogs have more salient features that are easier for the current CNN to recognize than other animal categories.

The average confusion rate among animals is 3.6%. The most frequent confusion pairs are ('Bird', 'Frog'), ('Cat', 'Dog'), and ('Deer', 'Horse'), with average confusion rates of 6.9%, 17.4%, and 4.8%, respectively. Specifically, 10.8% of birds are misclassified as frogs, and 3.0% of frogs are misclassified as birds. Additionally, 20.7% of cats are misclassified as dogs, and 14.1% of dogs are misclassified as cats. For the ('Deer', 'Horse') pair, 4.9% of deers are

misclassified as horses, and 4.7% of horses are misclassified as deers.

Overall Model Performance

The confusion matrix result (Figure 5.1) of the baseline model shows an average accuracy of 72.7% with a time cost of 704 seconds. The model demonstrates strong performance with vehicle images in CIFAR-10, where all four vehicle categories achieved above-average accuracy, and only 'Frog' and 'Horse' in the animal categories achieved above-average accuracy. The range of the accuracy of vehicles falls from 73.5% to 84.8%, and the range of the accuracy of animals falls from 52.6% to 85.7%. The range of accuracies for animals is more extensive at both the high and low ends of the range. This enormous variance in accuracy indicates that animals have a higher degree of intra-class variability, which leads to challenges in achieving consistent classification accuracy.

Overall, vehicles are most often confused with other vehicles, and animals are most often confused with other animals, except for birds, which are also frequently confused with airplanes (7.3%). This pattern indicates that common features within these super-categories could benefit from a hierarchical model. By using a hierarchical model, the classification process can be broken down into stages. Initially, the model can differentiate between broad categories (such as vehicles and animals) and then focus on finer distinctions within each category (such as cars, trucks, ships, and airplanes). The smaller sub-problems, once the

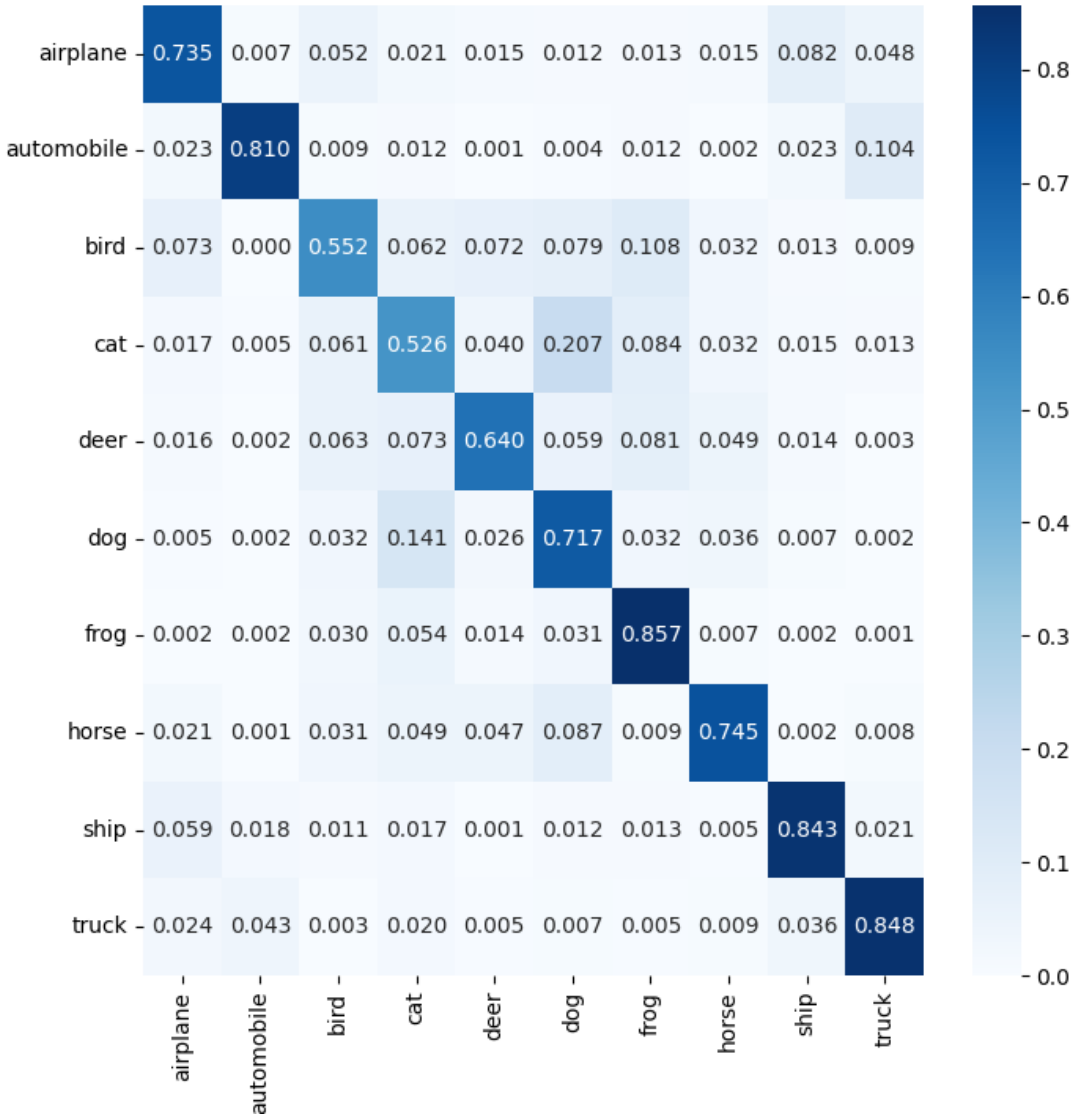

Figure 5.1: Confusion matrix showing the accuracy of each class for the baseline model. The vertical axis represents the true classes, while the horizontal axis represents the predicted labels. The values in each row sum to 100%, indicating the distribution of predictions for each true class. Values in each column may not sum to 100%, reflecting the nature of misclassification errors.

initial broad classification is made, should be easier to solve, potentially leading to better overall performance. The subsequent analysis of hierarchical models directly addresses this hypothesis.

5.2 Hierarchical Classification Models

5.2.1 Two-Stage Hierarchical Classification for Broad Visual Distinction

Our analysis of the baseline model's confusion matrix revealed that vehicles are often confused with other vehicles and animals with other animals, indicating that vehicles and animals have distinct features within their super-categories. This observation supports the use of a hierarchical model to leverage these distinct features to improve overall classification accuracy.

By applying the two-stage hierarchical classification, we aim to improve the model's ability to distinguish between visually similar classes within each broad category, thereby enhancing overall classification accuracy. The two-stage hierarchical classification framework applies the divide-and-conquer strategy to improve differentiation between visually distinct categories, which often causes confusion among the classes within each category in the baseline model. This approach aims to break down the complex classification problem into simpler, more

manageable sub-problems. By dividing the problem into two stages, each sub-problem should be easier to solve, leading to better overall results.

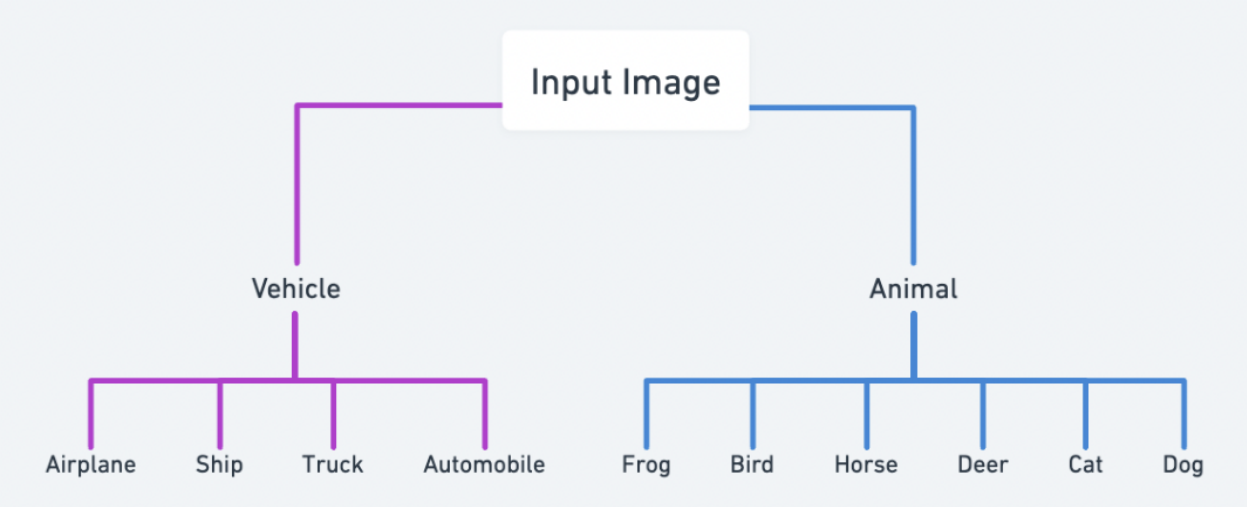

Figure 5.2: Diagram of the Two-Stage Hierarchical Classification framework

In the first stage (Figure 5.2), the model determines whether an item belongs to the broad category of vehicles or animals. This broad classification is crucial because it reduces the problem space, allowing the model to focus on more specific features in the subsequent stage. If this initial classification is incorrect, it will lead to a nonsensical classification later, as the model will then try to identify the item within the wrong broad category. For example, an animal mistakenly classified as a vehicle will lead to incorrect identification of the type of vehicle, resulting in fundamentally flawed outcomes.

Analysis of Vehicle Classification

Before analyzing the change in the performance of superclasses ('Vehicle' or 'Animal') categorization from the baseline model (Table 5.1) to a two-stage model, we observed that the average accuracy for vehicle categorization in the baseline model is 93.1%. Among the vehicle classes, 'Airplane' faces more challenges in being classified correctly into 'Vehicle' in the baseline model, where 12.8% is misclassified into 'Animal' as shown in Table 5.1, primarily because 'Bird' is the most commonly confused animal category with 'Airplane'. This trend suggests that the features used to identify airplanes overlap with those used to identify birds.

Class	**Vehicle (%)**	**Animal (%)**
Airplane	87.2	12.8
Automobile	96.0	4.0
Ship	94.1	5.9
Truck	95.1	4.9
Mean	93.1	6.9

Table 5.1: Classification percentages of each class into vehicle and animal categories by the baseline model. These percentages are calculated by summing the relevant rates allocated to vehicle and animal types, as illustrated in Figure 5.1.

In the two-stage model, the average accuracy for vehicle categorization improves to 93.8% (Table 5.2). Despite this improvement, the relatively high misclassification rate of 'Airplane' as an 'Animal' in both the baseline and hierarchical models suggests that the features used to identify airplanes overlap with those of certain animals. In the baseline model, 'Bird' is the most commonly confused animal category for 'Airplane'. It is likely that 'Bird' remains the

most confusing animal category for 'Airplane' in the two-stage model. The following analysis of the two-stage hierarchical model and the final overall confusion matrix will provide further insights and confirm whether this trend persists.

Class	**Vehicle (%)**	**Animal (%)**
Airplane	**88.2**	11.8
Automobile	**97.2**	2.8
Ship	**94.6**	5.4
Truck	95.0	5.0
Mean	**93.8**	6.3

Table 5.2: Classification percentages of each class into vehicle and animal categories by the two-stage hierarchical classification model. These percentages are calculated by summing the relevant rates allocated to vehicle and animal types, as illustrated in Figure 5.5.

The comparison between the baseline model and the two-stage hierarchical classification model reveals distinct trends in classification accuracy across different classes (Table 5.3). Most vehicle classes saw an improvement in accuracy when correctly classified into the 'Vehicle' superclass from the baseline model to the hierarchical model. The newly added stage, which is intended to help the model classify vehicles from animals more correctly, indeed improves the overall 'Vehicle' superclass classification. Specifically, the 'Airplane' class improved by 1.0% to correctly be classified into the 'Vehicle' superclass (Table 5.3), increasing its vehicle classification accuracy to 88.2% (Table 5.2). The 'Automobile' class saw a 1.2% improvement (Table 5.3) to correctly be classified into the 'Vehicle' superclass (Table 5.3), achieving a classification accuracy of 97.2% (Table 5.2). On the other hand, the 'Truck' class did not benefit from the hierarchical structure. Its classification accuracy as

a vehicle decreased by 0.1% (Table 5.3). This indicates that the newly added superclasses, which were intended to differentiate vehicles from animals, instead caused 0.1% more 'Truck' images to be incorrectly classified as 'Animal' (Table 5.3).

Class	**Change in Vehicle (%)**	**Change in Animal (%)**
Airplane	1.0	-1.0
Automobile	1.2	-1.2
Ship	0.5	-0.5
Truck	-0.1	0.1

Table 5.3: Difference in classification accuracy between the baseline model and the two-stage hierarchical model for vehicle and animal categories. The values represent the change in classification percentages, calculated by subtracting the baseline model values Table 5.1 from the two-stage model values for each category 5.2.

The average accuracy of vehicle categories in the second stage improved to 85.7% (Figure 5.3) from the baseline model's counterparts, which was 80.9% (Figure 5.1). This improvement indicates that the hierarchical model has the potential to improve the subsequent stage's accuracy of the subclasses. However, the same key confusion pairs exist, demonstrating a fundamental limit of the current system. For example, if this neural net architecture thinks a truck image is an automobile, it will be inclined to do so no matter where that classification occurs in the hierarchy.

The detailed confusion matrix for the second stage within the 'Vehicle' superclass (Figure 5.3) shows the most frequently confused pair combination is ('Airplane', 'Ship') and ('Automobile', 'Truck'). This leads to an increased average confusion rate for 'Airplane' and 'Ship' at 7.6%, while the confusion rate for 'Automobile' and 'Truck' decreases to 6.5%. The

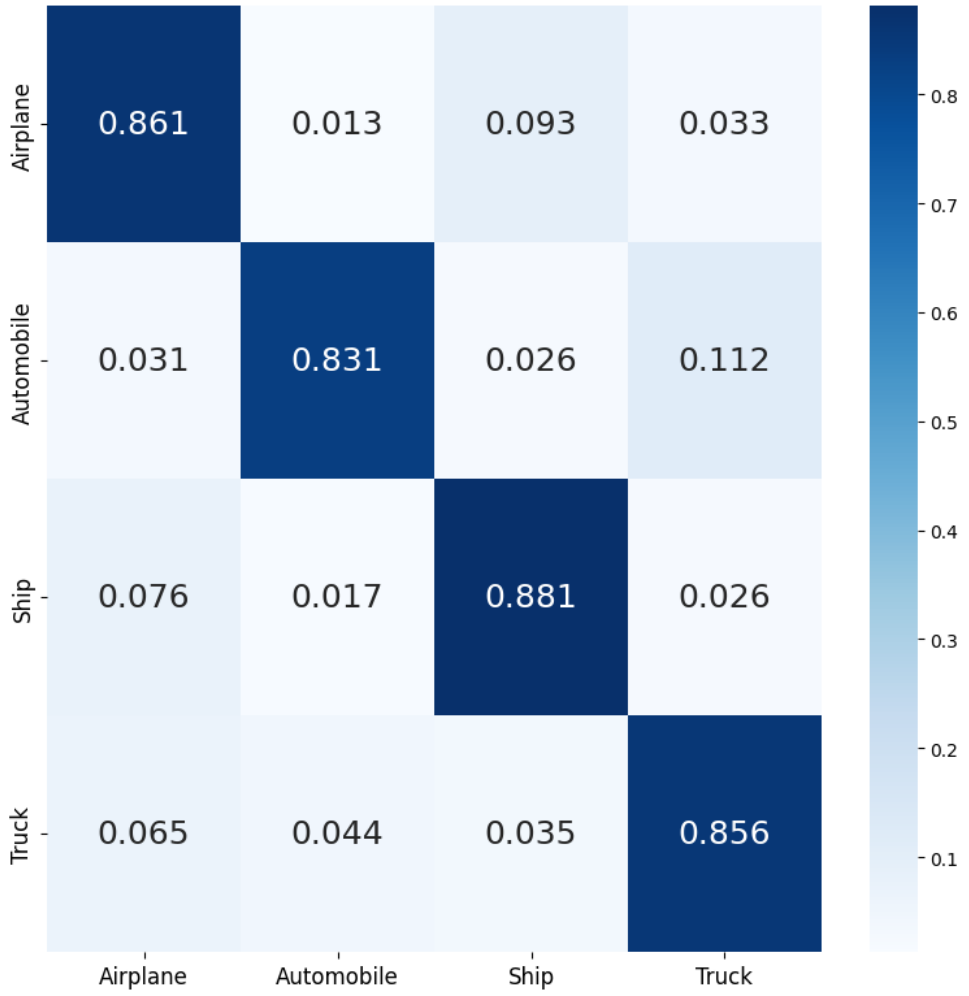

Figure 5.3: Confusion matrix for the second stage within 'Vehicle' superclass in the two-stage hierarchical classification model. The vertical axis represents the true classes, while the horizontal axis represents the predicted labels.

7.6% rate is the average of the sum of 7.6% (the confusion rate of 'Ship' for 'Airplane') and 7.6% (the confusion rate of 'Airplane' for 'Ship'). Similarly, the 6.5% rate is the average of the sum of 4.2% (the confusion rate of 'Automobile' for 'Truck') and 8.8% (the confusion rate

of 'Truck' for 'Automobile'). These rates are higher compared to the baseline model, where the corresponding average confusion rates were 7.1% and 7.4%, respectively (Figure 5.1). This trend indicates that the hierarchical approach of isolating vehicles from animals has made the features of 'Automobile' and 'Truck' more distinct from each other, reducing their confusion rate. However, it has also made the features of 'Airplane' and 'Ship' more similar, increasing their confusion rate. This suggests that while the hierarchical model can enhance classification accuracy for certain pairs by making features more distinguishable, it can also inadvertently increase confusion for other pairs by making their features more similar.

Comparing the overall accuracy of each vehicle class in the two-stage hierarchical model (Figure 5.5) with the baseline model (Figure 5.1), it is evident that the 'Airplane' and the 'Automobile' are the only categories that show an improvement in classification accuracy within the hierarchical model. Both categories successfully reduced the confusion with their most frequently confused classes from the baseline model ('Ship' for 'Airplane' and 'Truck' for 'Automobile'). These improvements underline the model's potential effectiveness in refining classification processes for specific vehicle types with distinctive visual features.

For 'Airplane' in the overall confusion matrix (Figure 5.5), the accuracy increased from 73.5% in the baseline model to 76.9% in the current hierarchical model, reducing the confusion with the 'Bird' from 5.2% to 4.2%. Despite this improvement, 'Bird' remains the most frequently confused animal category for 'Airplane' and the second most confused class

after 'Ship' for 'Airplane'. This persistence indicates that while the hierarchical approach can improve classification accuracy for certain vehicle types, it does not fully eliminate misclassification with certain visually similar animal classes.

The aforementioned observations underscore that the current hierarchical approach does not alter the misclassification trends observed in the baseline model for the most commonly confused pairs. It motivates us to add a new grouping layer focusing on the most confused pairs of ('Airplane', 'Ship') and ('Automobile', 'Truck') into the current hierarchical model to observe the further implications of the refined categorization strategy. The next section will discuss the results of this new three-stage hierarchical model.

Analysis of Animal Classification

Before analyzing the change in the performance of superclasses ('Vehicle' or 'Animal') categorization from the baseline model (Table 5.4) to a two-stage model, we observed that the average accuracy for animal categorization in the baseline model is 96.1%. Among the animal classes, 'Bird' faces more challenge in being classified correctly into 'Animal' in the baseline model, with 9.5% misclassified into 'Animal'. This result corresponds to the animal analysis of the baseline model, which found 'Airplane' was the most frequent confusion between a vehicle and an animal category.

When moving to the two-stage model to analyze the performance of superclasses ('Vehicle'

Class	Vehicle (%)	Animal (%)
Bird	9.5	90.5
Cat	5.0	95.0
Deer	3.5	96.5
Dog	1.6	98.4
Frog	0.7	99.3
Horse	3.2	96.8
Mean	3.9	96.1

Table 5.4: Classification percentages of each class into vehicle and animal categories by the two-stage hierarchical classification model. These percentages are calculated by summing the relevant rates allocated to vehicle and animal types, as illustrated in Figure 5.5.

or 'Animal') categorization, we observe that the average accuracy for animal categories drops to 92.7% (Table 5.5). The misclassification rate for 'Bird' into Vehicle remains high at 12.8%, same as the misclassification rate observed for 'Airplane' in the vehicle superclass (Table 5.1). This suggests that 'Airplane' likely remains the most confused vehicle category for 'Bird' in the two-stage model, as it was in the baseline model. This trend is confirmed by the final overall confusion matrix of the two-stage model (Figure 5.5), indicating that the hierarchical approach has not fully resolved the misclassification issues between these categories.

Class	Vehicle (%)	Animal (%)
Bird	12.8	87.2
Cat	8.6	91.4
Deer	5.0	95.0
Dog	4.4	95.6
Frog	5.9	94.1
Horse	7.3	92.7
Mean	7.3	92.7

Table 5.5: Classification percentages of each class into vehicle and animal categories by the two-stage hierarchical classification model

The comparison between the baseline model and the two-stage hierarchical classification

model reveals a consistent downward trend in classification accuracy across different classes (Table 5.6). Every animal class saw a reduction in accuracy when classified into the 'Animal' superclass from the baseline model to the hierarchical model (Table 5.3). This indicates that the features used to differentiate these superclasses overlap greatly for animal categories, which causes the newly added superclass differentiation, intended to separate vehicles from animals, instead cause more 'Animal' images to be incorrectly classified as 'Vehicle' (Table 5.6).

Class	**Change in Vehicle (%)**	**Change in Animal (%)**
Bird	3.3	-3.3
Cat	3.6	-3.6
Deer	1.5	-1.5
Dog	2.8	-2.8
Frog	5.2	-5.2
Horse	4.1	-4.1

Table 5.6: Difference in classification accuracy between the baseline model and the two-stage hierarchical model for vehicle and animal categories. The values represent the change in classification percentages, calculated by subtracting the baseline model values Table 5.4 from the two-stage model values for each category 5.5.

The confusion matrices for the second stage within the 'Animal' superclass in Figure 5.4 reveal ongoing difficulties in differentiating closely related animal pair combination: ('Bird', 'Frog'), ('Cat', 'Dog'), and ('Deer', 'Horse') while the reduction in misclassification rates within the ('Bird', 'Frog') and ('Cat', 'Dog') pairs are suggesting that the hierarchical approach can be effective in specific areas where confusion occurs. Notably, the pairs ('Bird', 'Frog') and ('Cat', 'Dog') demonstrated reduced average confusion rates of 5.1% and 14.5%, respectively,

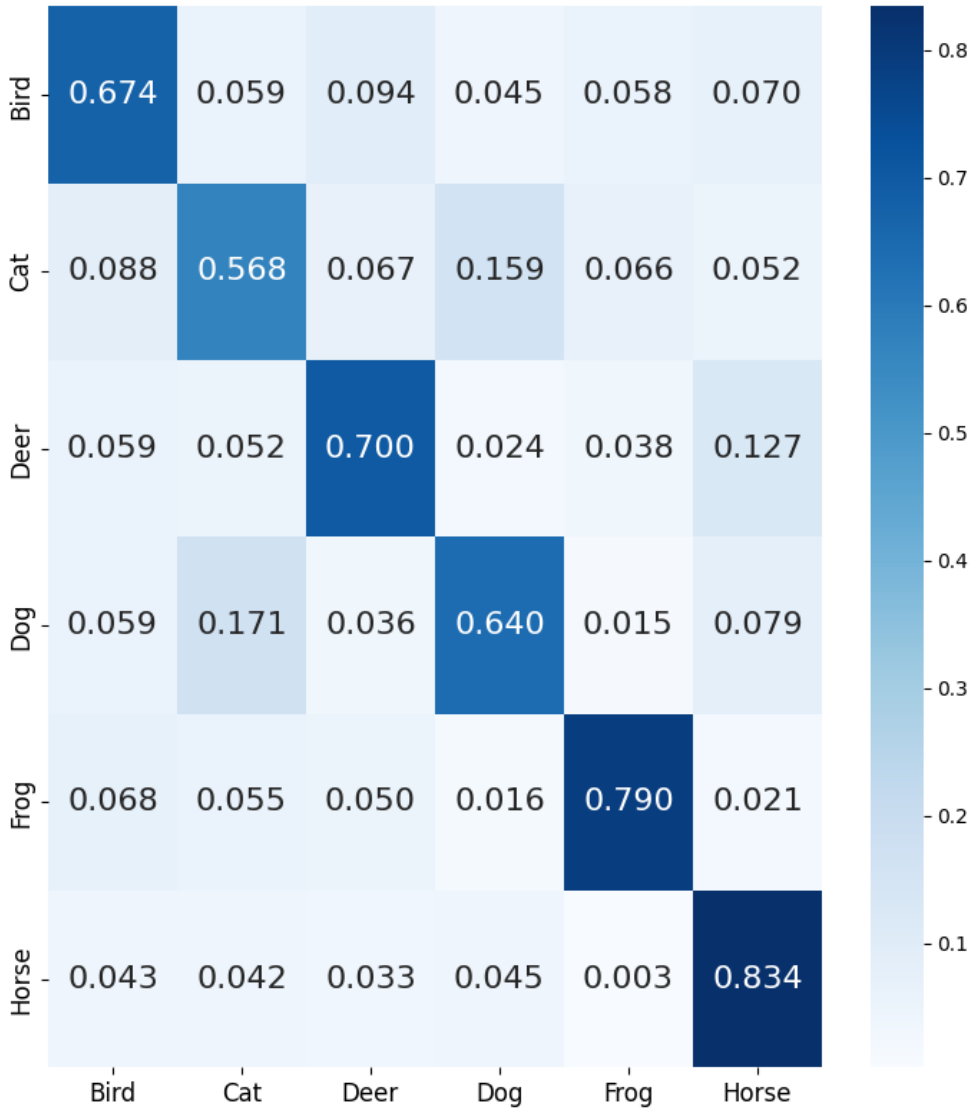

Figure 5.4: Confusion matrix for the second stage within the 'Animal' superclass in the two-stage hierarchical classification model. The vertical axis represents the true classes, while the horizontal axis represents the predicted labels.

compared to the baseline model's rates of 6.9% and 17.4%. Conversely, the pair ('Deer', 'Horse') showed an increased confusion rate of 7.3%, up from the baseline model's rate of 4.8%. For the ('Birds', 'Frogs') pair in the two-stage hierarchical model, 4.6% of birds are misclassified as frogs, and 5.5% of frogs are misclassified as birds. Additionally, 14.1% of cats are misclassified as dogs, and 14.9% of dogs are misclassified as cats. For the ('Deer', 'Horse') pair, 11.6% of deers are misclassified as horses, and 2.9% of horses are misclassified as deers. Similar to the average accuracy improvement in vehicle subclasses from the hierarchical model, the simpler sub-problem of classifying animal images also leads to improvements. The average accuracy of animal categories in the second stage improved to 70.1% (Figure 5.4) from the baseline model's counterparts, which was 67.3% (Figure 5.1). What is special from the insights gained from the animal images is the 'Bird' class' biggest confusion becomes the 'Deer' class (7.9%) from the 'Frog' class of the baseline model (10.8%), showing the breakthrough of not only reducing the confusion the subsequent stage but also diverting the confusion to other classes, which can be potentially beneficial for aiming to reduce a specific easily confused area.

The observed improvements in classification accuracy (Formula 4.13) from the baseline model's confusion matrix (Figure 5.1) to the hierarchical model's confusion matrix (Figure 5.5) for 'Bird', 'Deer', and 'Horse' categories indicate that the hierarchical structure is particularly effective at refining feature distinctions for these animals. Specifically, the accuracy for 'Bird'

increased from 55.2% (Figure 5.1) to 59.0% (Figure 5.5), for 'Deer' from 64.0% (Figure 5.1) to 67.0% (Figure 5.5), and for 'Horse' from 74.5% (Figure 5.1) to 77.9% (Figure 5.5). These enhancements can be attributed to the hierarchical model's ability to better focus on distinguishing features unique to each category by reducing the interference of non-relevant features from other classes.

Overall Model Performance

The implementation of the hierarchical classification model led to a mixed performance, where overall average accuracy decreased to 72.0% (Figure 5.5) with a slightly longer time cost of 806 seconds, which is composed of 404 seconds from the first stage, 165 seconds from the vehicle subclasses in the second stage, and 238 seconds from the animal subclasses. The hierarchical model's structure shows promise in refining classification tasks for specific vehicle and animal categories such as airplanes, automobiles, birds, deers, and horses that benefit from focused feature extraction and classification strategies. Additionally, vehicle classification accuracy experienced a slight improvement, increasing from the baseline's 93.1% to 93.8% (Table 5.7).

Conversely, the accuracy for animal classifications declined from 96.1% to 92.7% (Table 5.7). This disparity in performance suggests that hierarchical classification models can improve performance in specific areas, such as vehicle classification, but in the meantime, they can also

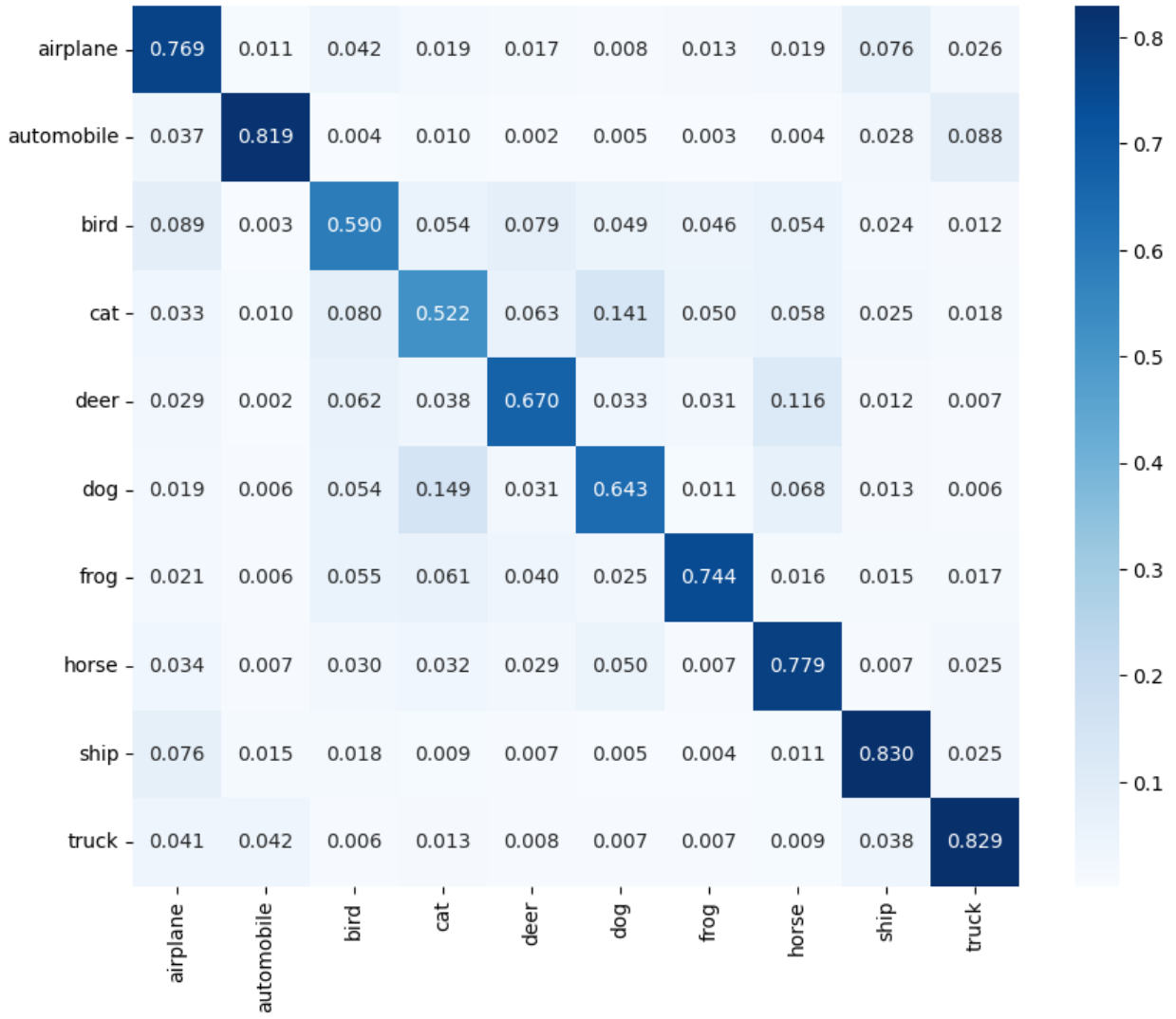

Figure 5.5: Confusion matrix for each class' accuracy of the two-stage hierarchical model. The vertical axis represents the true classes, while the horizontal axis represents the predicted labels.

have no distinctive feature in other areas despite the implications of the previous confusion rates. This indicates a fundamental limitation in the current neural network architecture. Specifically, the hierarchical model doesn't always correct initial misclassifications. For instance, if a bird was previously misclassified as a plane, it is likely still misclassified as a

Baseline Model		
Acc %	**Vehicle**	**Animal**
Vehicle	93.1	6.9
Animal	3.9	96.1
First-Stage of Hierarchical Classification		
Acc %	**Vehicle**	**Animal**
Vehicle	93.8	6.2
Animal	7.3	92.7

Table 5.7: Confusion matrices (Formula 4.12) for 'Vehicle' and 'Animal' superclasses comparison between the baseline model and the hierarchical classification model.

Note: In confusion matrices, rows are actual classes, columns are predicted classes and each row's percentages sum to 100%.

vehicle (Figure 5.1, 5.5). A much later analysis can highlight the need for each classifier in the hierarchy to be specifically designed for the sub-problem at hand (the most frequently confused pairs) so that the same error/weakness doesn't appear in a different stage.

5.2.2 Three-Stage Hierarchical Classification for Detailed Subgroup Discrimination

The newly added intermediate stage in the three-stage hierarchical classification model targets the most confused pair combinations identified in the second stage of the previous model: 'Airplane' and 'Ship', 'Automobile' and 'Truck', 'Bird' and 'Frog', 'Cat' and 'Dog' and 'Deer' and 'Horse' (Table 5.8 and Table 5.12).

Specifically, the model shown in Figure 5.6 is designed as:

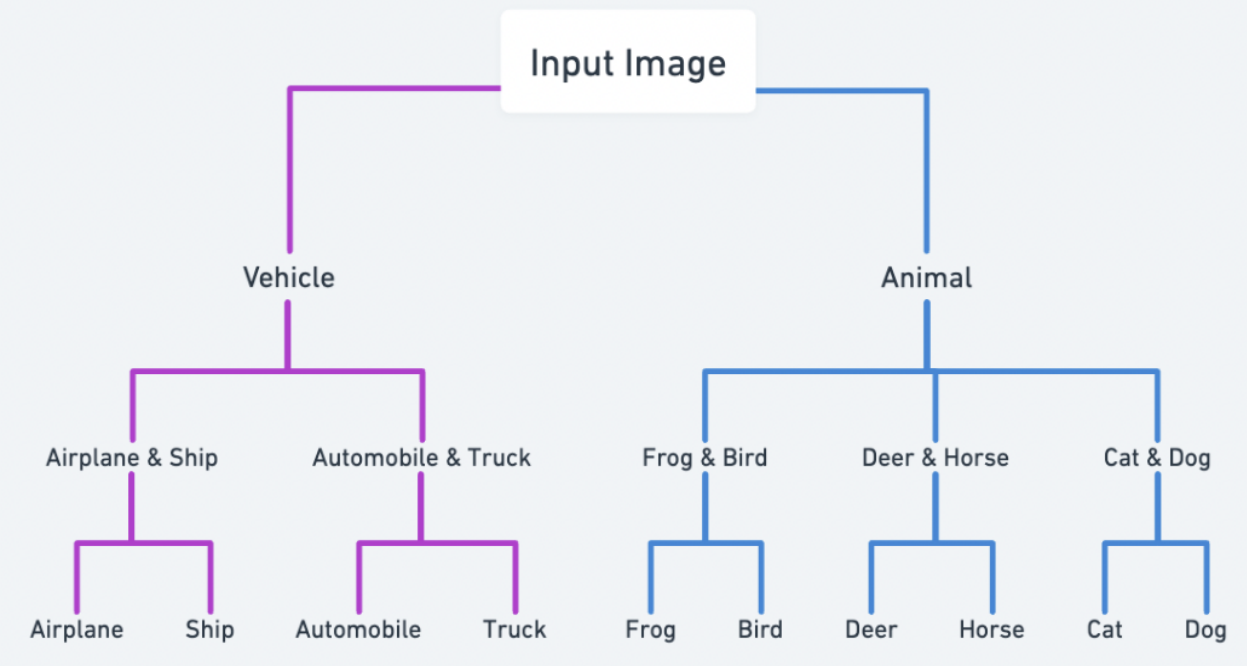

Figure 5.6: Diagram of the Three-Stage Hierarchical Classification framework

1. **Stage 1:** This superclass stage is identical to Stage 1 in the two-stage hierarchical classification approach, categorizing images broadly into 'Vehicle' and 'Animal' groups.

2. **Stage 2 (Newly Introduced Categorization):** This intermediate stage addresses problematic pairs identified in the baseline model by further subdividing the primary groups. The selected subgroups are:

    - *Within Animals:* The pairs ('Bird', 'Frog'), ('Cat', 'Dog'), and ('Deer', 'Horse') were chosen for further classification due to the high confusion rates observed in the baseline model.

    - *Within Vehicles:* ('Airplane', 'Ship') and ('Automobile', 'Truck') subgroups were

identified as focus areas, given their higher misclassification rates.

3. **Stage 3:** The final stage aims to resolve the classification of individual classes within the previously identified subgroups.

Analysis of Vehicle Classification

The subsequent stage under the 'Vehicle' superclass in the three-stage model shows a higher accuracy of 95.1% for the ('Airplane', 'Ship') subclass and a lower accuracy of 92.4% for the ('Automobile', 'Truck') subclass (Table 5.8). This indicates more distinct visual cues between airplanes and ships compared to automobiles and trucks, which show features similar to those of the neural net.

Acc %	('Airplane', 'Ship')	('Automobile', 'Truck')
('Airplane', 'Ship')	95.1	4.9
('Automobile', 'Truck')	7.5	92.4

Table 5.8: Confusion matrices for the second stage of the vehicles' three-stage model.

Initially, the accuracies for 'Airplane' and 'Ship' in the baseline model were 73.5% and 84.3%, respectively (Figure 5.1), with corresponding confusion rates of 26.5% for airplanes and 15.7% for ships. As shown in Figure 5.7, assuming a 50-50 distribution in the binary classification of these misclassified rates, one would expect an accuracy of 86.8% for 'Airplane' and 92.2% for 'Ship'. However, when analyzing deeper into the third stage, the actual accuracies were 91.0% for 'Airplane' and 83.0% for 'Ship' (Table 5.9). This suggests that

only the 'Airplane' class met the expected improvement, whereas the 'Ship' class did not achieve the anticipated accuracy based on the initial confusion rates.

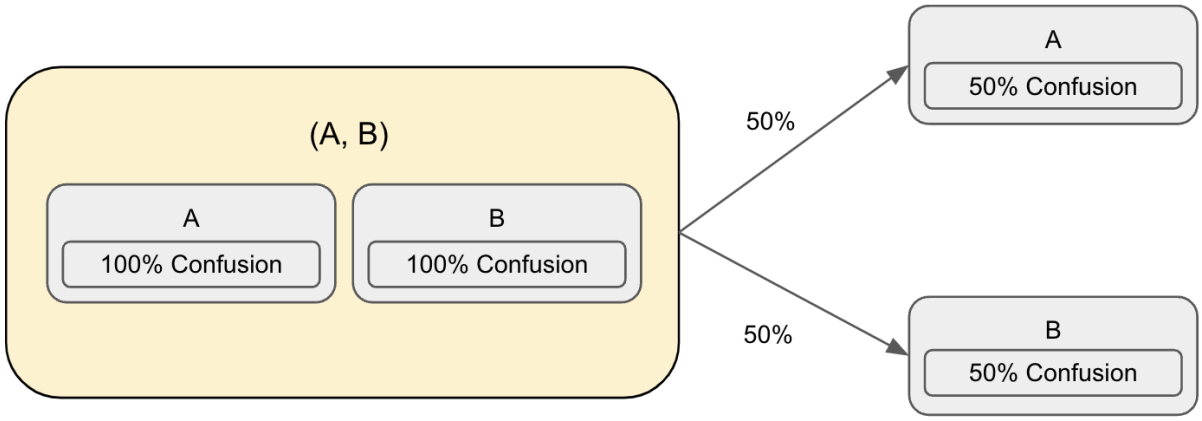

Figure 5.7: If a confused pair in the second stage consists of Class A and Class B, we assume the original confusion rate splits equally between the two classes because of the added stage, leading to the expected binary classification accuracies.

Acc %	Airplane	Ship
Airplane	91.0	9.0
Ship	17.0	83.0

Table 5.9: Confusion matrices for airplanes and ships in the third stage of the vehicles' three-stage model.

Initially, automobile and truck accuracies were 81.0% and 84.8% in the baseline model respectively (Figure 5.1); their confusion rates were 19.0% and 15.2% respectively (Figure 5.1). If their 19.0% and 15.2% confusion rates that are misclassified were even 50-50 for the binary classification, there should be 90.5% accuracy for 'Automobile' and 92.4% accuracy for the 'Truck' binary classification problem. This assumption did not happen as shown in Table 5.11: 'Automobile' vs. 'Truck' records accuracies of 85.9% for automobiles and 90.5% for

trucks (Table 5.11), which didn't improve their overall accuracy (Formula 4.13) but dropped it accordingly (Figure 5.8). The findings highlight the effectiveness of the hierarchical model designed to address specific sub-problems in improving classification accuracy for classes of the most frequently confused pairs with distinct visual features, such as airplanes. However, the drop in accuracy for 'Ship', 'Automobile' and 'Truck' indicates that divide and conquer of the hierarchical model has failed (Tab 5.10). The smallest and ideally easiest classification problem still leads to large error rates.

Acc %	Baseline Accuracy	Expected Binary Accuracy	Achieved Accuracy
Airplane	73.5	86.8	**91.0**
Automobile	81.0	90.5	85.9
Ship	84.3	92.2	83.0
Truck	84.8	92.4	90.5

Table 5.10: Baseline and expected vs. achieved binary classification accuracies in the hierarchical model

Notably, the combined average accuracy for this pair 'Automobile' vs. 'Truck' in the third stage is 88.2%, which surpasses the accuracy of the 'Airplane' vs. 'ship' pair at 87.0%. This is a reversal from the previous stage, where the accuracy for the ('Automobile', 'Truck') pair was lower than for the ('Airplane', 'Ship') pair (Table 5.8). This shift demonstrates the variability in classification performance at different levels of the hierarchy and suggests that the model's effectiveness can fluctuate based on the characteristics of the categories involved. It highlights that while hierarchical classification can improve performance for

certain categories, it may introduce challenges for others, particularly those with visually similar features.

Acc %	Automobile	Truck
Automobile	85.9	14.1
Truck	9.5	90.5

Table 5.11: Confusion matrices for automobiles and trucks in the third stage of the vehicles' three-stage model.

Analysis of Animal Classification

The second stage within the animal superclass (Table 5.12) introduces a greater difficulty in classifying animals into their respective subclasses than the previous stage, having an accuracy of 92.7%. In this second stage (Table 5.12), the classification accuracies drop to 80.3% for ('Bird', 'Frog'), 78.9% for ('Cat', 'Dog'), and 74.3% for ('Deer', 'Horse') (Table 5.12). These results suggest that distinguishing between these closely related animal subclasses is more challenging than making broader classifications. Specifically, the second stage confusion matrix shows that the ('Bird', 'Frog') pair is primarily misclassified as the ('Cat', 'Dog') pair compared to the ('Deer', 'Horse') pair(Table 5.12). The ('Cat', 'Dog') pair is mainly misclassified as the ('Bird', 'Frog') pair compared to the ('Deer', 'Horse') pair (Table 5.12). The ('Deer', 'Horse') pair is primarily misclassified as the ('Cat', 'Dog') pair compared to the ('Bird', 'Frog') pair (Table 5.12). These trends are consistent with the baseline model,

indicating that the inherent visual similarities between these subclasses maintain the same order of difficulty.

Acc %	(‘Bird’, ‘Frog’)	(‘Cat’, ‘Dog’)	(‘Deer’, ‘Horse’)
(‘Bird’, ‘Frog’)	80.3	12.4	7.2
(‘Cat’, ‘Dog’)	12.9	78.9	8.2
(‘Deer’, ‘Horse’)	12.7	13.0	74.3

Table 5.12: Confusion matrices for the second stage of the animals’ three-stage model.

In the third stage of the hierarchical model, we observed notable improvements despite previous challenges. Specifically, although the ‘Deer’ vs. ‘Horse’ subclass had the lowest classification accuracy in the second stage at 74.3%, this pair showed a great turnaround in the third stage. Here, the classification accuracy for ‘Deer’ and ‘Horse’ reached 91.4% and 86.1%, respectively, marking the highest average accuracy observed in the third stage for animal categorization (Table 5.13). Despite that it has the lowest accuracy in the second stage (Table 5.12) compared to other pairs, this demonstrates a subgroup having better performance in the previous stage of the hierarchical model is not a must to also have a better performance in its next stage classification.

Acc %	Deer	Horse
Deer	91.4	8.6
Horse	13.9	86.1

Table 5.13: Confusion matrices for deers and horses in the third stage of the animals’ three-stage model.

The third stage results for the other pairs, such as (‘Bird’, ‘Frog’) and (‘Cat’, ‘Dog’), show

varying levels of success. For the 'Bird' vs. 'Frog' pair, the model achieves an accuracy of 84.8% for 'Bird' and 89.6% for 'Frog' (Table 5.14), both increases from the 80.3% accuracy observed in the second stage (Table 5.12). This indicates that the model is becoming more adept at distinguishing between these two classes as it progresses through the hierarchical stages.

Acc %	Bird	Frog
Bird	84.8	15.2
Frog	10.4	89.6

Table 5.14: Confusion matrices for birds and frogs in the third stage of the animals' three-stage model.

Conversely, the 'Cat' vs. 'Dog' classification presents a greater challenge. Initially, cat and dog accuracies were 52.6% and 71.7% in the baseline model respectively (Figure 5.1); their total confusion rates were 47.4% and 28.3% respectively (Figure 5.1). If these confusion rates were split evenly for a binary classification problem, one might expect accuracies of 76.3% accuracy for 'Cat' and 85.9% for 'Dog'. However, this assumption does not hold as shown in Table 5.15: The accuracies actually decrease to 64.3% for 'Cat' and 78.6% for 'Dog' (Table 5.15) in the three-stage model. Both also fall below the 78.9% accuracy achieved in the previous stage (second stage) (Table 5.12). This decline explains that the hierarchical model for the particular challenging pair has more potential for its high-level group classification than its specific category classification. In the final overall confusion

matrix of the three-stage model (Figure 5.8), it was observed that the confusion rates between 'Cat' and 'Dog' increased, as did their confusion with 'Frog'. This suggests that while the hierarchical model was designed to refine classifications by addressing the most confused pair combinations, it inadvertently made it more difficult to distinguish between classes with subtle differences. This is especially true when these classes were grouped closely together based on their initial confusion patterns.

Acc %	Cat	Dog
Cat	64.3	35.7
Dog	21.4	78.6

Table 5.15: Confusion matrices for cats and dogs in the third stage of the animals' three-stage model.

To fully understand the performance changes and verify if they meet the expected improvement based on a 50-50 distribution of the misclassified rates, we evaluate the initial accuracies of 'Deer', 'Horse', 'Bird', 'Frog', 'Cat' and 'Dog' in the baseline model, which were 64.0%, 74.5%, 55.2%, 85.7%, 52.6%, and 71.7%, respectively. Based on these, we expected the binary classification accuracies to be 82.0%, 87.3%, 77.6%, 92.9%, 76.3%, and 85.9%, respectively. These expectations are derived by adding half of their confusion rates (36.0%, 25.5%, 44.8%, 14.3%, 47.4%, and 28.3%) to their baseline model accuracies. The results in the third stage of the three-stage model show that 'Deer' and 'Bird' classes outperformed their expected accuracies (Table 5.16). This indicates that the hierarchical model's deeper

layers succeeded in refining the classification accuracy of these specific categories beyond the anticipated binary classification performance.

Acc %	Baseline Accuracy	Expected Binary Accuracy	Achieved Accuracy
Deer	64.0	82.0	**91.4**
Horse	74.5	87.3	86.1
Bird	55.2	77.6	**84.8**
Frog	85.7	92.9	89.6
Cat	52.6	76.3	64.3
Dog	71.7	85.9	78.6

Table 5.16: Baseline and expected vs. achieved binary classification accuracies in the hierarchical model

Overall Model Performance

The average overall accuracy further dropped to 68.8% (Figure 5.8) with an increased time cost of 2,152 seconds, which is added by 538 seconds of the third stage of the vehicle model and 807 seconds of the third stage of the animal model into the previous 806 seconds of the two-stage model. This result shown in Figure 5.8 suggests the additional stages can lead to error propagation and impact overall performance, where misclassifications at one level affect subsequent levels, a common challenge in hierarchical models [56]. Furthermore, the increased model complexity resulted in longer training times and higher computational costs, detracting from the overall performance.

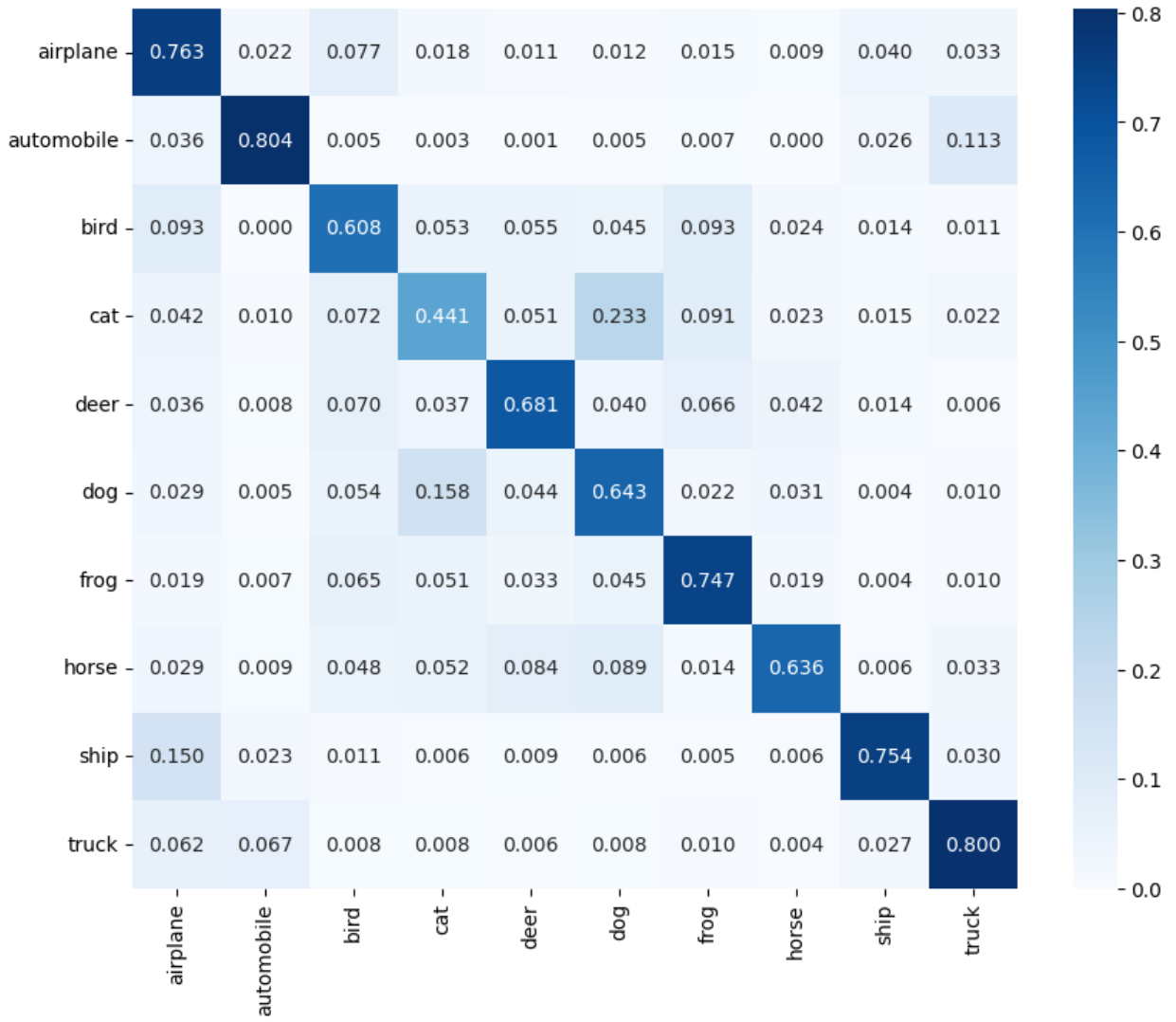

Figure 5.8: Confusion matrix for each class' accuracy of the three-stage model. The vertical axis represents the true classes, while the horizontal axis represents the predicted labels.

5.2.3 Four-Stage Hierarchical Classification for Animals

In the earlier hierarchical models, the overall accuracy for classifying animal superclasses did not improve from the first stage when compared to the vehicle classes (Table 5.7). However, there was a notable improvement in the overall accuracy calculated by Formula 4.13 for the

'Bird' category. The accuracy for 'Bird' increased from 55.2% in the baseline model to 59.0% in the two-stage hierarchical model and further to 60.8% in the three-stage hierarchical model. This enhancement suggests that intermediate stages in the hierarchy effectively isolate and refine features unique to birds, thereby reducing their confusion with other classes. Given this success, further refining the hierarchy to segregate the ('Bird', 'Frog') pair from other animal pairs is hypothesized to enhance the accuracy for birds even more.

In response to this hypothesis, the four-stage hierarchical classification model introduces a separation of 'Visually Two-Legged' animals from 'Four-Legged' ones at an earlier stage. This is done before specifically distinguishing ('Bird', 'Frog') from ('Cat', 'Dog'), and ('Deer', 'Horse') as shown in Figure 5.9. This additional layer aims to simplify subsequent classification stages by assigning the ('Bird', 'Frog') pair to an independent branch, allowing the ('Cat', 'Dog'), and ('Deer', 'Horse') pairs to be classified separately in their respective branches.

Acc %	Two-legged	Four-legged
Two-legged	83.9	16.2
Four-legged	18.6	81.4

Table 5.17: Confusion matrices for the second stage of the animals' four-stage model.

Acc %	('Cat', 'Dog')	('Deer', 'Horse')
('Cat', 'Dog')	88.4	11.6
('Deer', 'Horse')	16.8	83.2

Table 5.18: Confusion matrices for ('Cat', 'Dog') and ('Deer', 'Horse') pairs in the third stage of the animals' four-stage model.

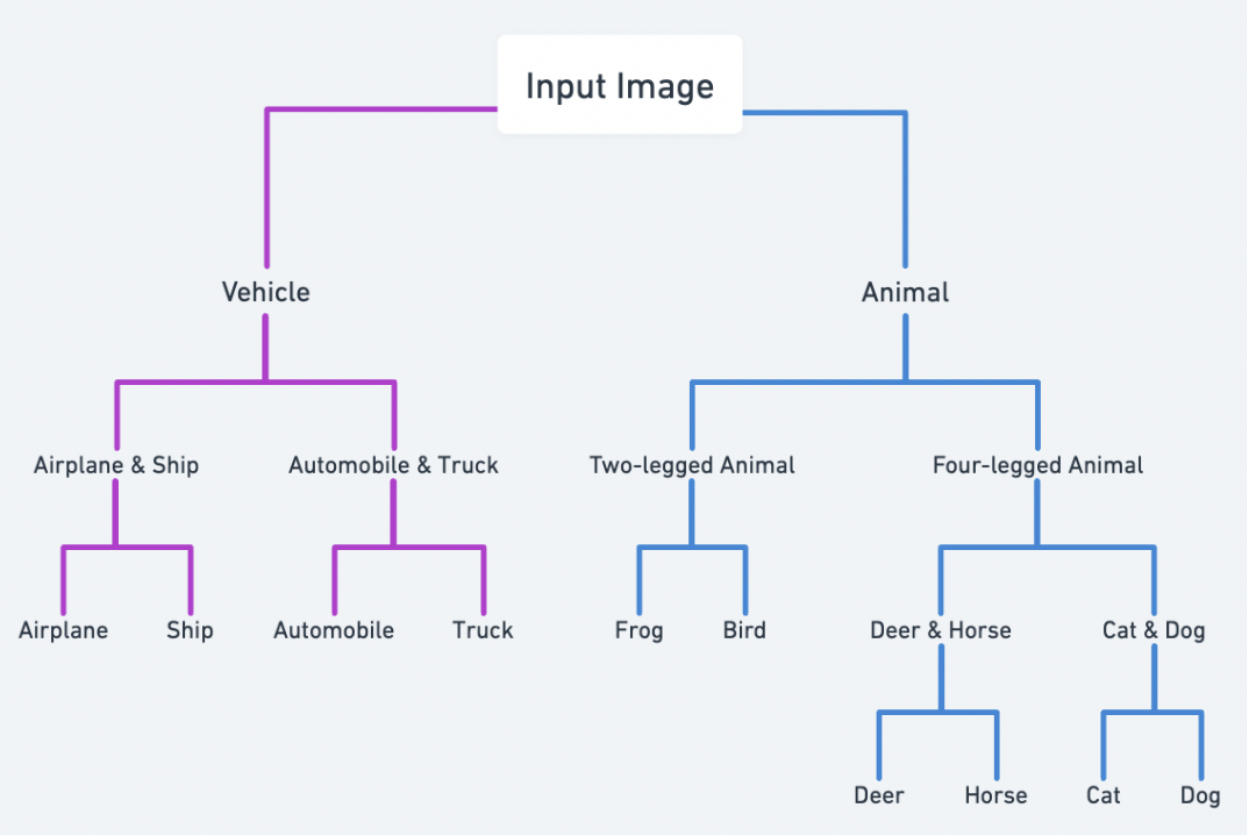

Figure 5.9: Diagram of the four-stage hierarchical classification framework

Table 5.17 presents the confusion matrix for the second stage, which categorizes animals into two broad groups: two-legged and four-legged. The model achieves an accuracy of 83.9% for the two-legged animal group and 81.4% for the four-legged animal group. The two-legged group shows higher accuracy than its counterpart in the second stage of the previous three-stage model (80.3% accuracy for the ('Bird', 'Frog') pair, which is the counterpart of the two-legged group from Table 5.12. Furthermore, Table 5.18 shows the performance of the ('Cat', 'Dog') and ('Deer', 'Horse') subgroups within the four-legged group in the third stage. The accuracies are 88.4% for ('Cat', 'Dog') and 83.2% for ('Deer', 'Horse'), which both show

improved accuracy compared to their performance in the second stage of the three-stage model, which were 78.9% and 74.3%, respectively (Table 5.12). These results suggest that the added level of granularity improves the model's ability to initially sort animal images effectively. The initial separation into more homogeneous groups helps the model focus on more distinct features relevant to each group, thus enhancing classification accuracy at this stage.

As discussed in the previous section, we expected the binary classification accuracies of 'Bird', 'Frog', 'Cat', 'Dog', 'Deer', and 'Horse' to be 77.6%, 92.9%, 76.3%, 85.9%, 82%, and 87.3%, respectively. When evaluating the confusion matrices of the last stage of the two-legged and four-legged groups (Table 5.19, Table 5.20, Table 5.21), only 'Bird' (93.4% improved from 84.8% of the three-stage model (Table 5.14)), 'Cat' (82.7% from 64.3% of the three-stage model (Table 5.15)), and 'Deer' (90.0% dropped from 91.4% of the three-stage model (Table 5.13) met the assumption this time. The results suggest that while adding more stages to the hierarchical model can refine classification for certain categories by focusing on similar features, the increased complexity can also lead to more confusion for classes with overlapping features.

Acc %	Bird	Frog
Bird	93.4	6.6
Frog	18.3	81.7

Table 5.19: Confusion matrices for birds and frogs in the third stage of the animals' four-stage model.

Acc %	Cat	Dog
Cat	82.7	17.3
Dog	31.8	68.2

Table 5.20: Confusion matrices for cats and dogs in the fourth stage of the animals' four-stage model.

Acc %	Deer	Horse
Deer	90.0	10.0
Horse	14.5	85.5

Table 5.21: Confusion matrices for deers and horses in the fourth stage of the animals' four-stage model.

Overall, the average accuracy of the four-stage hierarchical classification model decreased to 67.6% (Figure 5.10) because of the error propagation, and the time required to complete the classification also increased to 2,346 seconds, added 194 seconds into the previous three-stage model. While the average accuracy of the animal categories in the hierarchical model declined to 60.6% (Figure 5.10) from the baseline model's 67.3% (Figure 5.1), 'Bird' experienced notable improvements. In the four-stage hierarchical model, the accuracy for 'Bird' surged to 67.6% from 60.8% of the three-stage model, an impressive rise from the baseline accuracy of 55.2%. This reflects the model's enhanced ability to isolate and focus on the distinguishing characteristics of birds.

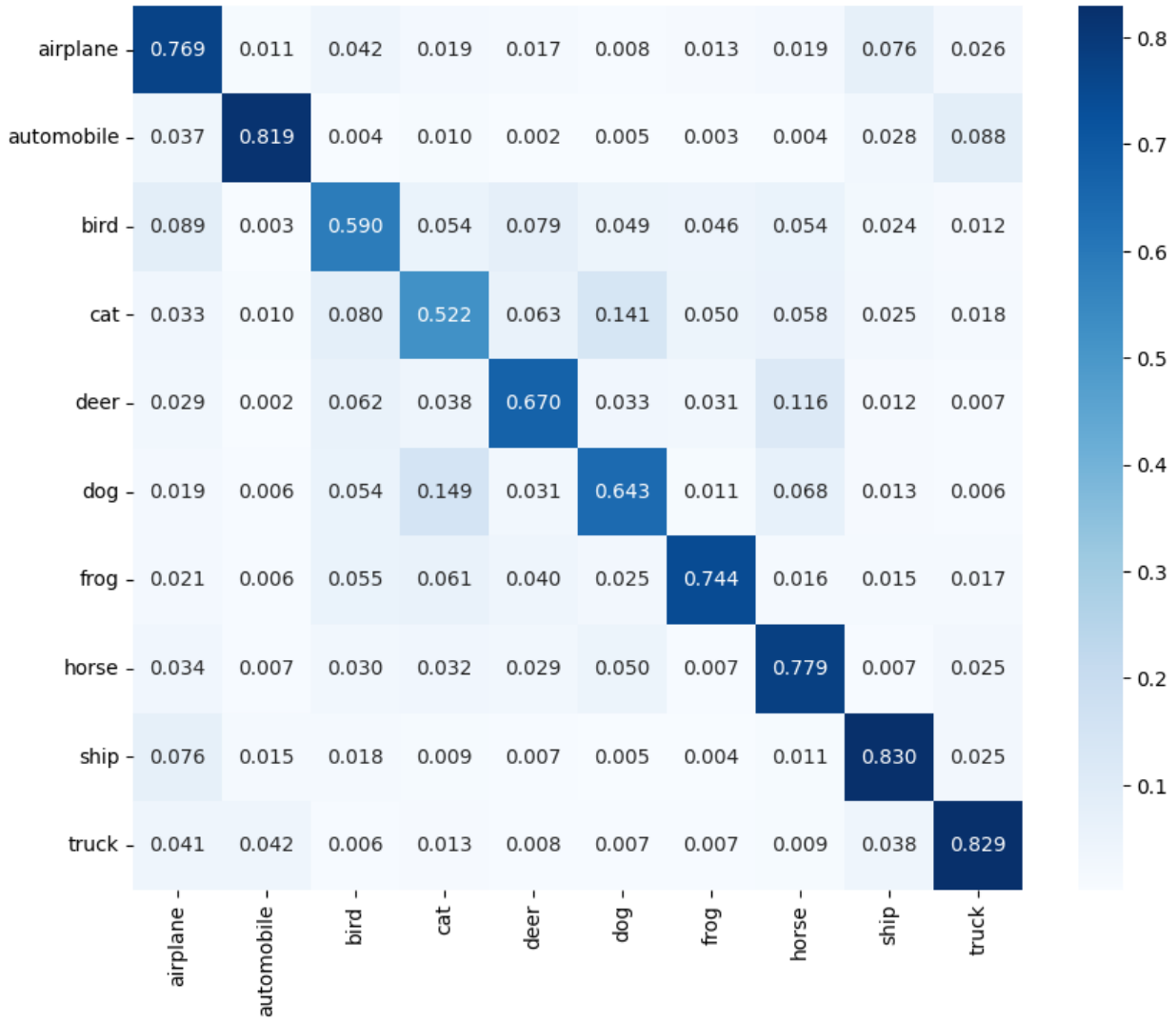

Figure 5.10: Confusion matrix for each class' accuracy of the two-stage hierarchical model. The vertical axis represents the true classes, while the horizontal axis represents the predicted labels.

6. Summary

Analyzing the confusion matrix results for the baseline model, we observed challenges such as high intra-class variation and similarities between different classes leading to confusion, a common issue in flat classification models noted in various studies [18, 76, 77]. The hierarchical model was implemented to mitigate these issues by refining classification through a multi-stage process that narrows the focus progressively, aiming to reduce the potential for inter-class confusion.

The hierarchical approach successfully enhanced the model's accuracy in certain areas, notably increasing the accuracy for the 'Bird' category from 55.2% in the baseline model to 67.6% in the four-stage hierarchical model (Table 6.1) and subgroups of the most confused pairs compared to the average of their respective initial accuracy in the baseline model as calculated by Formula 4.13. This improvement illustrates the hierarchical approach's ability to focus on distinguishing features critical to specific classes and groups/pairs. Additionally, the hierarchical approach improved the accuracy of vehicle classification in the first-stage

hierarchical model, as shown in Table 5.7.

Furthermore, the hierarchical models provide enhanced transparency over the baseline model by allowing a granular examination of classification performance across multiple stages. This structure not only aids in pinpointing where misclassifications occur but also helps track how classification accuracy evolves through each stage, which is essential for diagnosing errors and developing mitigation strategies.

Acc %	Baseline Model	Two-Stage Model	Three-Stage Model	Four-Stage Animal Model
Airplane	73.5	**76.9**	**76.3**	**76.3**
Automobile	81.0	**81.9**	80.4	80.4
Bird	55.2	**59.0**	**60.8**	**67.6**
Cat	52.6	52.2	44.1	51.4
Deer	64.0	**67.0**	**68.1**	52.8
Dog	71.7	64.3	64.3	53.9
Frog	85.7	74.4	74.7	72.4
Horse	74.5	**77.9**	63.6	65.3
Ship	84.3	83.0	75.4	75.4
Truck	84.8	82.9	80.0	80.0
Mean	72.7	72.0	68.8	67.6

Table 6.1: The overall accuracy of each class calculated by Formula 4.13.

Also, training time can be reduced when focusing only on specific superclasses. For example, the baseline model took 704 seconds for training. If we only need to fine-tune vehicle classes, we could focus on training the model specifically for vehicles by first training the initial stage of differentiating vehicles and animals in 404 seconds, and then training the sub-model for categorizing vehicle categories in 165 seconds. This would reduce the total training time to 569 seconds.

However, while the hierarchical model introduced improvements, it also brought forth new challenges, particularly the dependency of overall model accuracy on the success of the initial categorization stages. This vulnerability was highlighted as the average accuracy across most image classes in the CIFAR-10 dataset declined from 72.7% in the baseline to 67.6% in the four-stage model. The decline underscores the issue of error propagation, where initial misclassifications can exacerbate errors in subsequent stages, emphasizing the need for robust initial classification to ensure model scalability and efficiency [78, 79]. This complexity suggests that non-homogeneous hierarchical classifiers, which utilize different classification approaches at various stages, could be a crucial area for further research to optimize performance across diverse scenarios.

Additionally, the comparison between the baseline and the two-stage hierarchical models revealed a consistent decline in accuracy of animal classifications (Table 5.6). This trend indicates that the broad categorization into 'Animal' and 'Vehicle' superclasses may have been too generalized, leading to great feature overlap and resulting in animals being misclassified as vehicles more frequently. The CNN architecture employed may not have been adequately tailored to address the diverse features within the broad 'Animal' category, highlighting the necessity for more sophisticated hierarchical structures.

7. Future Work

Given these insights, it is evident that each classifier in the hierarchy needs to be precisely tailored to the specific sub-problems it addresses to prevent error propagation. Future research should focus on several key areas to further optimize hierarchical classification models:

- Non-Homogeneous Hierarchical Classifiers: The hierarchical approach in this research shows that the same key confusion pairs exist throughout the hierarchy. For example, if a neural net architecture misclassifies an automobile as a truck, it will likely continue this error across stages. This limitation underscores the need for non-homogeneous hierarchical classifiers, which use different classification methods or algorithms at various stages.

- Dynamic Confidence Thresholds: Implementing dynamic confidence thresholds within the classification process offers a way to reassess and adjust predictions that fall below a certain confidence level. For example, if a classifier returns a confidence score of less than

50% for a given prediction, the system could flag this as uncertain and either revert to a previous stage to re-predict the result using a different classifier or adjust the classifier architecture to better handle such cases. This iterative process would continue until the prediction confidence surpasses the threshold, or the model architecture is sufficiently adapted to make a more confident prediction. This approach could help mitigate error propagation by allowing the model to dynamically adjust its decision-making process based on real-time performance metrics, ensuring that it remains responsive to the classification challenges encountered.

- Refinement of Superclasses: The hierarchical model's approach to classifying broad categories like 'Animal' reduced accuracy for every animal class compared to the baseline model (Table 5.6). The baseline model (Figure 5.1) also exhibited a wider accuracy range for animal categories, indicating their overlapping features might not be sufficiently distinctive. This suggests the 'Animal' superclass is too broad, leading to considerable feature overlap and misclassification. We can reduce feature overlap and improve classification accuracy by breaking these broad categories into narrower categories (e.g., two-legged vs. four-legged). This refinement can capture more distinct features within the animal categories, leading to better performance.

Addressing these areas will allow future work to build on the foundational insights gained

from this research to develop more effective and efficient hierarchical classification models, enhancing their applicability across various domains and datasets.

8. Contributions

This research contributes to the broader field of computer vision by establishing a foundational framework for the adoption and optimization of hierarchical classification models. These models are designed not only to enhance targeted classification accuracy but also to improve the interpretability of the decision-making process. By addressing class confusion through a structured, multi-stage approach, this work advances the understanding of how hierarchical models can be effectively utilized to manage and reduce errors in specific contexts, such as distinguishing closely related subclasses.

The insights gained from this study are particularly valuable for applications requiring high reliability and precision, such as in automobile safety systems and medical diagnostics [80]. In these domains, the ability to increase specificity in distinguishing between critical categories while providing a transparent and interpretable decision-making process is crucial. Moreover, the hierarchical approach facilitates the involvement of professionals at different levels of the decision-making process, enabling them to understand and intervene where necessary, which

is especially important in fields like healthcare where expert judgment is critical.

Our progressive hierarchical classification model has shown that, despite challenges in maintaining consistent accuracy across diverse categories, the advantages of enhanced interpretability and targeted error correction offer a compelling case for further refinement and broader adoption of this approach. The study highlights the importance of developing hierarchical structures that are finely tuned to specific sub-problems, ensuring that the model delivers balanced performance without compromising robustness.

This work moves towards a system-level view of image classification, where hierarchical models play a key role in addressing targeted class confusion and ensuring that high-stakes applications can achieve the necessary levels of reliability and transparency. Additionally, the research underscores the value of hierarchical models in empowering professionals to engage more effectively with the decision-making process, thereby improving the overall efficacy of the system. The findings and proposed future work offer a clear direction for continued advancements in hierarchical classification models, with the potential to greatly benefit real-world applications.

Bibliography

[1] Chenyi Chen et al. "Deepdriving: Learning affordance for direct perception in autonomous driving". In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 2722–2730.

[2] Loganathan Agilandeeswari and S Divya Meena. "SWIN transformer based contrastive self-supervised learning for animal detection and classification". In: *Multimedia Tools and Applications* 82.7 (2023), pp. 10445–10470.

[3] Mohammed AS Ali et al. "Evaluating very deep convolutional neural networks for nucleus segmentation from brightfield cell microscopy images". In: *SLAS DISCOVERY: Advancing the Science of Drug Discovery* 26.9 (2021), pp. 1125–1137.

[4] Yomna Safaa El-Din, Mohamed N Moustafa, and Hani Mahdi. "Deep convolutional neural networks for face and iris presentation attack detection: Survey and case study". In: *IET Biometrics* 9.5 (2020), pp. 179–193.

[5] M Kiani. "Accuracy assessment of crop classification in hyperspectral imagery using very deep convolutional neural networks". In: *Second International Congress on Science and Engineering, Paris*. 2020.

[6] Logambal Madhuanand et al. "Deep convolutional neural networks for surface coal mines determination from sentinel-2 images". In: *European Journal of Remote Sensing* 54.1 (2021), pp. 296–309.

[7] Ali Jamali and Masoud Mahdianpari. "Swin transformer and deep convolutional neural networks for coastal wetland classification using sentinel-1, sentinel-2, and LiDAR data". In: *Remote Sensing* 14.2 (2022), p. 359.

[8] Keno K Bressem et al. "Comparing different deep learning architectures for classification of chest radiographs". In: *Scientific reports* 10.1 (2020), p. 13590.

[9] Anabel Gómez-Ríos et al. "Towards highly accurate coral texture images classification using deep convolutional neural networks and data augmentation". In: *Expert Systems with Applications* 118 (2019), pp. 315–328.

[10] Lucky Indra Kesuma et al. "ELREI: Ensemble Learning of ResNet, EfficientNet, and Inception-v3 for Lung Disease Classification based on Chest X-Ray Image." In: *International Journal of Intelligent Engineering & Systems* 16.5 (2023).

[11] Waseem Rawat and Zenghui Wang. "Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review". In: *Neural Computation* 29.9 (2017), pp. 2352–2449. DOI: 10.1162/neco_a_00990.

[12] Ahmad Chaddad et al. "Survey of explainable AI techniques in healthcare". In: *Sensors* 23.2 (2023), p. 634.

[13] Patrick Weber, K Valerie Carl, and Oliver Hinz. "Applications of explainable artificial intelligence in finance—a systematic review of finance, information systems, and computer science literature". In: *Management Review Quarterly* 74.2 (2024), pp. 867–907.

[14] Jürgen Fritsch and Michael Finke. "Applying divide and conquer to large scale pattern recognition tasks". In: *Neural networks: Tricks of the trade*. Springer, 2002, pp. 315–342.

[15] Yu Zheng et al. "Hierarchical convolutional neural network via hierarchical cluster validity based visual tree learning". In: *Neurocomputing* 409 (2020), pp. 408–419.

[16] Shuying Liu and Weihong Deng. "Very deep convolutional neural network based image classification using small training sample size". In: *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*. 2015, pp. 730–734. DOI: 10.1109/ACPR.2015.7486599.

[17] Yogesh Pant et al. "Comparison of CNN Architecture of Image Classification Using CIFAR10 Datasets". In: *International Journal on Engineering Technology* 1.1 (Dec.

2023), pp. 37–52. DOI: 10.3126/injet.v1i1.60898. URL: https://www.nepjol.info/index.php/injet/article/view/60898.

[18] Danupon Chansong and Siriporn Supratid. "Impacts of Kernel Size on Different Resized Images in Object Recognition Based on Convolutional Neural Network". In: *2021 9th International Electrical Engineering Congress (iEECON)*. 2021, pp. 448–451. DOI: 10.1109/iEECON51072.2021.9440284.

[19] Joao Carreira et al. "Human pose estimation with iterative error feedback". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 4733–4742.

[20] Max Kuhn, Kjell Johnson, et al. *Applied predictive modeling*. Vol. 26. Springer, 2013.

[21] Gareth James et al. *An introduction to statistical learning*. Vol. 112. Springer, 2013.

[22] Saeed S Alahmari et al. "Challenges for the repeatability of deep learning models". In: *IEEE Access* 8 (2020), pp. 211860–211868.

[23] Mohammad Momeny et al. "A noise robust convolutional neural network for image classification". In: *Results in Engineering* 10 (2021), p. 100225.

[24] Qinghe Zheng et al. "A full stage data augmentation method in deep convolutional neural network for natural image classification". In: *Discrete Dynamics in Nature and Society* 2020 (2020).

[25] SH Shabbeer Basha et al. "Impact of fully connected layers on performance of convolutional neural networks for image classification". In: *Neurocomputing* 378 (2020), pp. 112–119.

[26] Navdeep Singh and Hiteshwari Sabrol. "Convolutional neural networks-an extensive arena of deep learning. A comprehensive study". In: *Archives of Computational Methods in Engineering* 28.7 (2021), pp. 4755–4780.

[27] Vijaypal Singh Dhaka et al. "A survey of deep convolutional neural networks applied for prediction of plant leaf diseases". In: *Sensors* 21.14 (2021), p. 4749.

[28] Pavlo M Radiuk. "Impact of training set batch size on the performance of convolutional neural networks for diverse datasets". In: *Information Technology and Management Science* 20.1 (2017), pp. 20–24.

[29] Priya Goyal et al. "Accurate, large minibatch sgd: Training imagenet in 1 hour". In: *arXiv preprint arXiv:1706.02677* (2017).

[30] Leiyu Chen et al. "Review of image classification algorithms based on convolutional neural networks". In: *Remote Sensing* 13.22 (2021), p. 4712.

[31] Pai Peng et al. "Gas classification using deep convolutional neural networks". In: *Sensors* 18.1 (2018), p. 157.

[32] Vasily Derbentsev et al. “Deep learning approach for short-term forecasting trend movement of stock indeces”. In: *2021 IEEE 8th International Conference on Problems of Infocommunications, Science and Technology (PIC S&T)*. IEEE. 2021, pp. 607–612.

[33] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems* 25 (2012).

[34] Li Deng and Dong Yu. *Deep Learning: Methods and Applications (Foundations and Trends in Signal Processing Series Book 20)*. 2014.

[35] Matthew D Zeiler and Rob Fergus. “Visualizing and understanding convolutional networks”. In: *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13*. Springer. 2014, pp. 818–833.

[36] Li Wan et al. “Regularization of neural networks using dropconnect”. In: *International conference on machine learning*. PMLR. 2013, pp. 1058–1066.

[37] Karen Simonyan and Andrew Zisserman. “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556* (2014).

[38] Liran Chen and Greg Shakhnarovich. “Learning ensembles of convolutional neural networks”. In: *The University of Chicago* (2014).

[39] Ryo Takahashi, Takashi Matsubara, and Kuniaki Uehara. "Multi-stage convolutional neural networks for robustness to scale transformation". In: *International Symposium on Nonlinear Theory and Its Applications*. 2017, pp. 4–7.

[40] Suchisrit Gangopadhyay and Anthony Zhai. "CGBNet: A Deep Learning Framework for Compost Classification". In: *IEEE Access* 10 (2022), pp. 90068–90078.

[41] Rajasekhar Nennuri et al. "A Multi-stage Deep Model for Crop Variety and Disease Prediction". In: *International Conference on Soft Computing and Pattern Recognition*. Springer. 2022, pp. 52–59.

[42] Alina Roitberg et al. "Cnn-based driver activity understanding: Shedding light on deep spatiotemporal representations". In: *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*. IEEE. 2020, pp. 1–6.

[43] Amina Adadi. "A survey on data-efficient algorithms in big data era". In: *Journal of Big Data* 8.1 (2021), p. 24.

[44] Khin Yadanar Win et al. "Ensemble deep learning for the detection of Covid-19 in unbalanced chest X-ray dataset". In: *Applied Sciences* 11.22 (2021), p. 10528.

[45] Siyu An et al. "Is Seeing Still Not Necessarily Believing?" In: (2023).

[46] Justine Boulent et al. "Convolutional neural networks for the automatic identification of plant diseases". In: *Frontiers in plant science* 10 (2019), p. 464450.

[47] Pratinav Seth, Akshat Bhandari, and Kumud Lakara. "Analyzing Effects of Fake Training Data on the Performance of Deep Learning Systems". In: *arXiv preprint arXiv:2303.01268* (2023).

[48] Burak Ekim, Elif Sertel, and M Erdem Kabadayı. "Automatic road extraction from historical maps using deep learning techniques: A regional case study of turkey in a German World War II Map". In: *ISPRS International Journal of Geo-Information* 10.8 (2021), p. 492.

[49] Laith Alzubaidi et al. "Review of deep learning: concepts, CNN architectures, challenges, applications, future directions". In: *Journal of big Data* 8 (2021), pp. 1–74.

[50] Khurram Hameed, Douglas Chai, and Alexander Rassau. "Class distribution-aware adaptive margins and cluster embedding for classification of fruit and vegetables at supermarket self-checkouts". In: *Neurocomputing* 461 (2021), pp. 292–309.

[51] Subhankar Banerjee and Shayok Chakraborty. "Budgeted subset selection for fine-tuning deep learning architectures in resource-constrained applications". In: *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2020, pp. 1–10.

[52] Enrico Barbierato and Alice Gatti. "Towards Green AI. A methodological survey of the scientific literature". In: *IEEE Access* (2024).

[53] Victor Quétu, Zhu Liao, and Enzo Tartaglione. "The Simpler The Better: An Entropy-Based Importance Metric To Reduce Neural Networks' Depth". In: *arXiv preprint arXiv:2404.18949* (2024).

[54] Brenda Cinthya Solari Berno et al. "A Framework for Analyzing Book Covers and Co-purchases using Object Detection and Data Mining Methods". In: *2019 IEEE Latin American Conference on Computational Intelligence (LA-CCI)*. 2019, pp. 1–6. DOI: 10.1109/LA-CCI47412.2019.9037060.

[55] Abhinav Goel et al. "Tree-Based Unidirectional Neural Networks for Low-Power Computer Vision". In: *IEEE Design & Test* 40.3 (2022), pp. 53–61.

[56] Puneet Mathur et al. "LayerDoc: layer-wise extraction of spatial hierarchical structure in visually-rich documents". In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2023, pp. 3610–3620.

[57] Brenda Cinthya Solari Berno et al. "A Framework for Analyzing Book Covers and Co-purchases using Object Detection and Data Mining Methods". In: *2019 IEEE Latin American Conference on Computational Intelligence (LA-CCI)*. IEEE. 2019, pp. 1–6.

[58] Zhicheng Yan et al. "HD-CNN: hierarchical deep convolutional neural networks for large scale visual recognition". In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 2740–2748.

[59] Manhua Liu et al. “Hierarchical fusion of features and classifier decisions for Alzheimer’s disease diagnosis”. In: *Human brain mapping* 35.4 (2014), pp. 1305–1319.

[60] Alex Krizhevsky and Geoff Hinton. “Convolutional deep belief networks on cifar-10”. In: *Unpublished manuscript* 40.7 (2010), pp. 1–9.

[61] Asifullah Khan et al. “A survey of the recent architectures of deep convolutional neural networks”. In: *Artificial intelligence review* 53 (2020), pp. 5455–5516.

[62] Nebojša Bačanin Džakula et al. “Convolutional neural network layers and architectures”. In: *Sinteza 2019-International Scientific Conference on Information Technology and Data Related Research*. Singidunum University. 2019, pp. 445–451.

[63] Andrej Krenker, Janez Bešter, and Andrej Kos. “Introduction to the artificial neural networks”. In: *Artificial Neural Networks: Methodological Advances and Biomedical Applications. InTech* (2011), pp. 1–18.

[64] Abhinav Agrawal and Namita Mittal. “Using CNN for facial expression recognition: a study of the effects of kernel size and number of filters on accuracy”. In: *The Visual Computer* 36.2 (2020), pp. 405–412.

[65] Ajala Sunday Adeyinka. “Convolutional Neural Network Implementation for Image Classification using CIFAR-10 Dataset”. In: (2021).

[66] Fahad Alrasheedi, Xin Zhong, and Pei-Chi Huang. "Padding module: Learning the padding in deep neural networks". In: *IEEE Access* 11 (2023), pp. 7348–7357.

[67] Nils Bjorck et al. "Understanding batch normalization". In: *Advances in neural information processing systems* 31 (2018).

[68] Vinod Nair and Geoffrey E Hinton. "Rectified linear units improve restricted boltzmann machines". In: *Proceedings of the 27th international conference on machine learning (ICML-10)*. 2010, pp. 807–814.

[69] Sagar Sharma, Simone Sharma, and Anidhya Athaiya. "Activation functions in neural networks". In: *Towards Data Sci* 6.12 (2017), pp. 310–316.

[70] Chigozie Nwankpa et al. "Activation functions: Comparison of trends in practice and research for deep learning". In: *arXiv preprint arXiv:1811.03378* (2018).

[71] Muhammad Shafiq and Zhaoquan Gu. "Deep residual learning for image recognition: A survey". In: *Applied Sciences* 12.18 (2022), p. 8972.

[72] Hongwei Yong et al. "Momentum batch normalization for deep learning with small batch size". In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XII 16*. Springer. 2020, pp. 224–240.

[73] Léon Bottou. *Large scale machine learning with stochastic gradient descent*. Springer, 2010.

[74] Saeed Mohsen. "Recognition of human activity using GRU deep learning algorithm". In: *Multimedia Tools and Applications* 82.30 (2023), pp. 47733–47749.

[75] Mohammadreza Heydarian, Thomas E Doyle, and Reza Samavi. "MLCM: Multi-label confusion matrix". In: *IEEE Access* 10 (2022), pp. 19083–19095.

[76] Zhi Zhang et al. "Bag of Freebies for Training Object Detection Neural Networks". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 1281–1290.

[77] Liang Chen and Vishal M Patel. "Learning Robust Representations by Projecting Superficial Statistics Out". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 1601–1610.

[78] Manish Bhatt et al. "Hierarchy-based file fragment classification". In: *Machine Learning and Knowledge Extraction* 2.3 (2020), pp. 216–232.

[79] Juan Carlos Gomez and Marie-Francine Moens. "Hierarchical classification of web documents by stratified discriminant analysis". In: *Multidisciplinary Information Retrieval: 5th Information Retrieval Facility Conference, IRFC 2012, Vienna, Austria, July 2-3, 2012 Proceedings 5*. Springer. 2012, pp. 94–108.

[80] Geert Litjens et al. "A survey on deep learning in medical image analysis". In: *Medical image analysis* 42 (2017), pp. 60–88.