

**From the holdings of the Department of Special Collections , Princeton University Library,
Princeton, NJ**

These documents are either in the public domain, or can only be used for educational and research purposes (Fair use) as per U.S. Copyright law (text below). By accessing this file, all users agree that their use falls in the public domain or within fair use as defined by the copyright law. We do not charge any permission or usage fees for the publication of images of material in our collections, including those provided by Princeton University Library, via our website, catalog, or directly from staff. For more information regarding copyright, credit, and citation please review the information on the Copyright, Credit and Citation Guidelines page of our website.

<https://library.princeton.edu/special-collections/policies/copyright-credit-and-citation-guidelines>. If you have any questions, including how to order higher-quality images from the collection, contact Special Collections Public Services via our Ask Us! Form. <https://library.princeton.edu/special-collections/ask-us>

Master's Theses, 2022-2026

Computer Science, 2023

2024

Exploring Early Exiting Strategies for Deep Neural Networks

Xia, Feng

Princeton, NJ : Princeton University

<http://arks.princeton.edu/ark:/88435/dsp0179408155k>

U.S. Copyright law: The copyright law of the United States (Title 17, §108, United States Code) governs the making of photocopies or other reproductions of copyrighted material. Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be used for any purpose other than private study, scholarship, or research. If a user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of fair use that user may be liable for copyright infringement. This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve a violation of copyright law. <https://www.copyright.gov/title17/title17.pdf>

EXPLORING EARLY EXITING STRATEGIES FOR DEEP NEURAL NETWORKS

FENG (SHELLEY) XIA

SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE IN ENGINEERING
DEPARTMENT OF COMPUTER SCIENCE
PRINCETON UNIVERSITY

ADVISER: PROFESSOR THOMAS L. GRIFFITHS

READER: PROFESSOR MATTHEW WEINBERG

MAY 2024

© Copyright by Feng (Shelley) Xia, 2024.

All rights reserved.

Abstract

Deep neural networks (DNNs) are renowned for their accuracy across a spectrum of machine learning tasks but often suffer from prolonged inference times due to their depth. To address this, early exiting strategies have been proposed, allowing predictions to be made at intermediate layers, thus reducing inference time. This thesis explores several methodologies to enhance early exit mechanisms within a single multi-exit DNN architecture on an image classification task. First, we demonstrate that replacing the traditional confidence measure entropy with maximum probability achieves comparable accuracy while substantially reducing inference time. Second, we observe that the total uncertainty used by conventional confidence measures does not consistently reflect true model uncertainty, especially for ambiguous images. To address this, we utilize a Dirichlet framework for neural networks, which assumes the network outputs a Dirichlet distribution of class probabilities rather than a point estimate. We introduce a novel training mechanism that includes both original training data and artificially created ambiguous data by blending training images. This approach allows more ambiguous data to exit early compared to the previous approach. Lastly, we examine two alternative perspectives to formulate early exiting criteria. While our experiments focus on a specific neural network architecture and dataset, our methodologies are designed to be independent of architectures and tasks, rendering potential for wider applicability in deep neural networks.

Acknowledgements

I am immensely grateful to my advisor, Professor Thomas L. Griffiths, for his invaluable support and guidance throughout my research journey. His patience, support, and kindness have been deeply meaningful to me. Furthermore, his passion for research and the “optimal” way of thinking to approach problems, as exemplified in his book *Algorithms to Live By*, have profoundly influenced my perspectives and approaches in tackling challenges and making decisions in life.

I would like to thank the members of Computational Cognitive Science Lab (Co-Sci) at Princeton University. Conversations with them are always insightful and I learned a lot from every one of them. I owe a great debt of gratitude to my mentor, Dr. Jake Snell, for his generous help throughout this thesis. He is always knowledgeable, insightful, patient, supportive, and incredibly kind.

I would also like to thank Professor Matthew Weinberg for graciously agreeing to serve as the reader for this thesis. His wonderful lectures profoundly influenced my thinking in problem-solving.

I would like to express my sincere and heartfelt gratitude to my family for their unwavering support. Their unwavering belief in me has been a tremendous source of strength and inspiration, encouraging me to believe in myself even during the most challenging times. I am also deeply grateful for the wonderful friendships. Their presence has been a beacon of light.

Lastly, I want to thank myself, for persevering during challenging times.

Contents

Abstract	iii
Acknowledgements	iv
List of Tables	vii
List of Figures	viii
1 Introduction	1
2 Background	3
2.1 Deep Neural Networks	3
2.1.1 AlexNet	4
2.1.2 The CIFAR10 Dataset	4
2.2 Early Exiting	5
2.3 BranchyNet	6
2.3.1 Architecture	6
2.3.2 Training	8
2.3.3 Exiting Criterion	8
3 Improving Existing Confidence Measures	10
3.1 Maximum Probability	10
3.1.1 Motivation	11
3.1.2 Results	12
3.2 A Dirichlet Framework	13

3.2.1	Motivation	13
3.2.2	The Dirichlet Framework	15
3.2.3	Training	18
3.2.4	Results	22
4	Alternative Perspectives on Early Exiting Criteria	28
4.1	Stability-Based Criteria	28
4.2	Loss-Based Criteria	31
5	Conclusion	34
	Bibliography	36

List of Tables

3.1	Four possible combinations of model uncertainty and data uncertainty with examples. Total uncertainty is the sum of model uncertainty and data uncertainty.	14
4.1	Ensemble accuracy versus accuracies of individual branches	32

List of Figures

2.1	The AlexNet architecture: five convolutional layers followed by one fully connect layer. We removed two fully connected layers from the original AlexNet.	5
2.2	The Branchy-AlexNet architecture: five convolutional layers, each followed by one fully connected layer to enable intermediate exits. . . .	7
2.3	Entropy as the exiting criterion at an intermediate branch	8
3.1	Maximum probability as the exiting criterion at an intermediate branch	11
3.2	Classification accuracy versus runtime for B-AlexNet using two confidence measure compared with original AlexNet. Knee points represent the thresholds where the accuracy of B-AlexNet is comparable to the original AlexNet.	12
3.3	Model predictions for an unambiguous image versus an ambiguous image. For an unambiguous image, predictions converge to a single class probability vector and entropy gravitates towards zero. For an ambiguous image, predictions continuously assign comparable probabilities to multiple classes and entropy remains high throughout all exits. . . .	15
3.4	An unambiguous image versus an ambiguous image: a prototypical image of airplane and a deceiving image of bird that may be seen as an airplane	16

3.5	Four possible Dirichlet simplex outputs under different kinds of model and data uncertainty	17
3.6	Blending an image of “Dog” and an image of “Cat” into an ambiguous image of “Cat” and “Dog”	20
3.7	Training accuracies of Dir-B-AlexNet and Amb-Dir-B-AlexNet: Dir-B-AlexNet reaches a high training accuracy faster. Accuracy is measured on unambiguous data only.	21
3.8	Differential Entropy as the exiting criterion at an intermediate branch	22
3.9	A model comparison of uncertainty difference on Challenging Test compared to CIFAR10 testing dataset. A smaller difference indicates better performance in handling ambiguous data.	26
3.10	A model comparison of uncertainty difference on External Blend compared to CIFAR10 testing dataset. A smaller difference indicates better performance in handling ambiguous data.	26
3.11	A model comparison of uncertainty difference on Internal Blend compared to CIFAR10 testing dataset. A smaller difference indicates better performance in handling ambiguous data.	26
3.12	A model comparison of percentage of ambiguous samples exiting from exit 1 when 1, 25, 50, 75, 99% unambiguous samples exit from exit 1. A higher percentage indicates a better performance in early exiting. Percentages with the highest values are highlighted in bold.	27
4.1	Stability as the exiting criterion at an intermediate branch	29
4.2	Classification accuracy versus runtime for B-AlexNet using three distance metrics. Knee points represent the thresholds where the accuracy of B-AlexNet is comparable to the original AlexNet.	31
4.3	Entropy vs cross-entropy loss for training and testing data	33

Chapter 1

Introduction

Deep learning has been the golden machine learning paradigm over the past decade [31]. The remarkable success of deep neural networks (DNNs) is often credited to their ability to learn data representations with multiple levels of abstraction [22], made possible by their depths. However, their substantial depth presents a dual challenge: while it boosts the network’s learning capacity, it incurs significant costs in both training and inference time. The need for fast inference in DNNs is important for various reasons. First, the sequential nature of DNNs’ computations limits the potential for parallelization [41], thus dramatically increases the need for computational resources. Second, deploying DNNs on resource-constrained devices, such as mobile phones and embedded systems, requires efficient use of limited computational resources. Third, in real-world applications, rapid computation is essential to deliver real-time responses and a seamless user experience. Consequently, inference acceleration in DNNs becomes a crucial area of research.

Numerous efforts have been made to address the prolonged inference time in DNNs, among which early exiting aims to reduce average inference time by allowing “easy” samples to exit early in intermediate layers of the network [30, 10]. Implementing an early exiting strategy involves addressing three primary challenges: 1) the

placement of exits, 2) designing a training mechanism, and 3) setting a exiting criterion. Exiting criteria in prior work primarily rely on total uncertainty as confidence measure or utilize neural networks to directly predict the exiting probability or making the exiting decision. The former may not fully capture true model uncertainty for data with inherent complexity and the latter may lack efficiency and interpretability.

In this thesis, we explore various exiting criteria designed from explicit principles to enhance interpretability and efficiency, as well as the ability to handle different types of data. We evaluated their effectiveness using AlexNet and an early exiting variant of AlexNet on the CIFAR10 image classification task. Our results demonstrate that a measure simpler than entropy maintains the same level of accuracy while significantly reduces average inference time. Furthermore, we observe that the network may converge to a prediction with high total uncertainty for ambiguous images, despite high model confidence. By adopting a Dirichlet framework to directly measure model uncertainty, more ambiguous images are able to exit early in the network. We conclude with a discussion of two potentially fruitful alternative perspectives on the early exiting problem along with some preliminary evidence.

This paper begins by outlining the general problem of early exiting in DNNs and introducing baseline models and dataset in Chapter 2. Chapter 3 elaborates on two approaches aimed at improving existing confidence measure. Chapter 4 motivates two alternative perspectives on the problem. Chapter 5 provides a summary, discuss limitations, and suggests future directions.

Chapter 2

Background

This chapter aims to lay the groundwork for understanding the problem we are exploring: early exiting in deep neural networks (DNNs). Detailed discussions on the related work for each methodology we explored will be distilled in the subsequent chapters as motivation. This chapter is organized into three sections. First, we introduce the concept of DNNs and explain the necessity for accelerating their inference. Within the same section, we also provide a detailed description of AlexNet [20], a specific DNN architecture serving as our baseline model, as well as an image classification dataset CIFAR10 [19] for evaluation. Second, we introduce early exiting as an approach to reduce inference time along with an overview of related work. Finally, we delve into the details of BranchyNet [38], a popular early exiting network architecture, which will be utilized as our baseline early exiting model.

2.1 Deep Neural Networks

Deep neural networks (DNNs) are multi-layer neural network architectures that form the cornerstone of modern machine learning [22]. These networks have shown tremendous success across various domains: convolutional neural networks (CNNs) for image classification [23, 20, 35, 37, 11], transformers for language modeling [39, 7, 4], re-

current neural networks (RNNs) for sequential data processing [14, 32, 36], just to name a few. While the remarkable success is primarily attributed to their substantial depth, this often comes at the cost of prolonged inference time.

Due to its sequential nature, parallelizing computations within DNNs presents significant challenges [41]. As these networks become deeper, the demand for computational resources dramatically increases. Furthermore, the deployment of DNNs on small and low-power devices necessitates short inference times. Consequently, improving the efficiency of feed-forward inference has become crucial. Many approaches have been proposed to address this problem. Network compression reduces the network size by pruning, quantization, and low-rank factorization [9, 17, 5]. Knowledge distillation transfers knowledge from large models to smaller ones [13, 8]. Our discussion focuses on early exiting, a technique that allows early termination in the network under certain conditions [38, 28, 3].

2.1.1 AlexNet

AlexNet is a pioneering convolutional neural network (CNN) in image classification. Developed by Krizhevsky et al., AlexNet was first introduced at the ImageNet Large Scale Visual Recognition Challenge in 2012 [20], where it achieved groundbreaking success in decreasing top-5 error. It consists of five convolutional layers followed by three fully connected layers. To align with the early exiting architecture we will introduce later in this chapter, we modified the traditional AlexNet by incorporating only one fully connected layer. The modified architecture of AlexNet is illustrated in figure 2.1. This will serve as our baseline DNN.

2.1.2 The CIFAR10 Dataset

CIFAR-10 is a widely used dataset in machine learning for image classification benchmarking [19]. It contains 60,000 color images of size 32x32 from 10 classes (airplane,

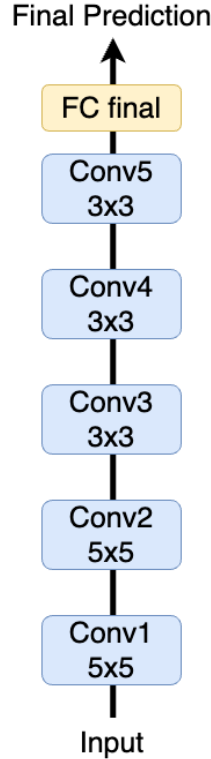


Figure 2.1: The AlexNet architecture: five convolutional layers followed by one fully connect layer. We removed two fully connected layers from the original AlexNet.

automobile, bird, cat, deer, dog, frog, horse, ship, truck), and it is divided into 50,000 training images and 10,000 testing images.

2.2 Early Exiting

Early exiting is an approach aimed to reduce inference time in deep neural networks (DNNs) by enabling “easy” samples to terminate early at intermediate exits to avoid full processing [30]. It has three core challenges: where to place the exits (architecture design), how to train the exits (training), and how to decide when to exit (exiting criterion). BranchyNet proposes to add side branches to the main network, enabling sample to early exit when predictions have high confidence [38]. Differentiable branching reformulates early exiting in a differentiable form and eliminates the

need for designing an explicit exiting criterion [29]. Adaptive Neural Networks solves a layer-by-layer weighted binary classification problem to trade off between future accuracy and computational cost [3]. PonderNet learns to predict the probability of halting at each step [2]. Conditional Deep Learning (CDL) identifies the input difficulty via confidence measure and conditionally activate deeper layers of the network [28]. Confident Adaptive Transformers (CAT) trains a meta consistency classifier on top of intermediate layers to decide when to stop [33].

Existing work generally falls into two categories in terms of exiting criteria: 1) using some function of classification entropy as confidence measure, or 2) using a neural network to learn and predict confidence measure or exiting probability directly. The first approach may not adequately represent true model uncertainty for certain kinds of data, while the second may lack interpretability and efficiency. In this thesis, we take a closer look at what constitutes a prediction’s confidence by exploring various confidence metrics, investigating the difference between model confidence and prediction confidence, and finally, considering alternative perspectives to target different goals of early exiting.

2.3 BranchyNet

BranchyNet [38] is one of the pioneering and most frequently referenced architectures for early exiting in deep neural networks (DNNs), making it an ideal baseline for our studies.

2.3.1 Architecture

BranchyNet adds side branches to the main DNN architecture to allow certain samples to exit early during inference. These side branches can range from simple fully connected layers to more complex configurations such as convolutional layers and

multilayer perceptrons. To minimize our modifications and maintain generality, we choose to add a fully connected layer as a side branch to each of the five convolutional layers in AlexNet. The architecture of Branchy-AlexNet (B-AlexNet) is shown in figure 2.2. Both AlexNet and B-AlexNet will serve as our baseline models for comparisons in the subsequent work.

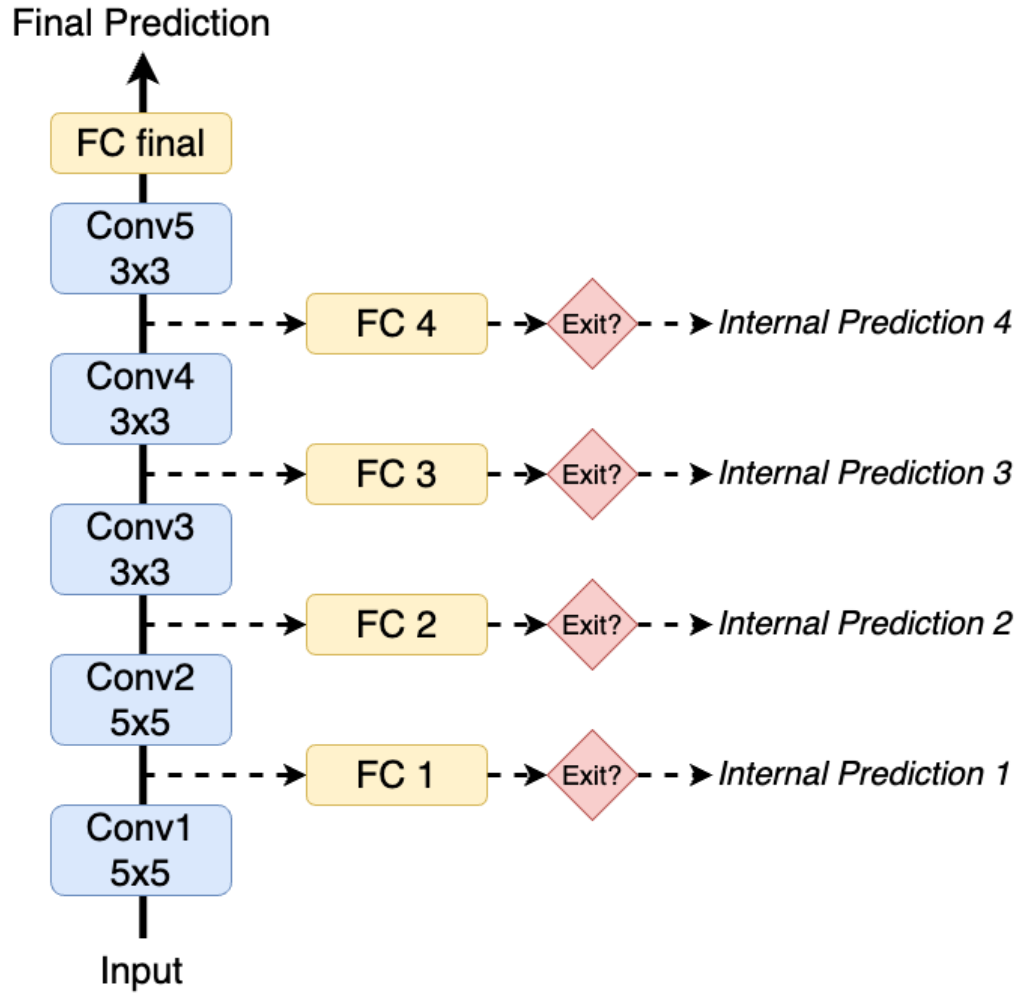


Figure 2.2: The Branchy-AlexNet architecture: five convolutional layers, each followed by one fully connected layer to enable intermediate exits.

2.3.2 Training

Let $\mathbf{y}$ be the one-hot encoded ground truth label vector, $\hat{\mathbf{y}}$ be the predicted softmax probability vector, then the softmax cross-entropy loss L is defined as:

$$L(\hat{\mathbf{y}}, \mathbf{y}) = - \sum_{c \in C} y_c \log \hat{y}_c$$

where C is the set of all possible labels. Weights are initialized from pretrained AlexNet on CIFAR10. All branches are then jointly trained by minimizing the sum of losses at each branch:

$$L_{\text{branchynet}}(\hat{\mathbf{y}}, \mathbf{y}) = \sum_{n=1}^N L(\hat{\mathbf{y}}_{\text{exit}_n}, \mathbf{y})$$

where $\hat{\mathbf{y}}_{\text{exit}_n}$ is the softmax probability vector predicted by branch n and N is the total number of branches.

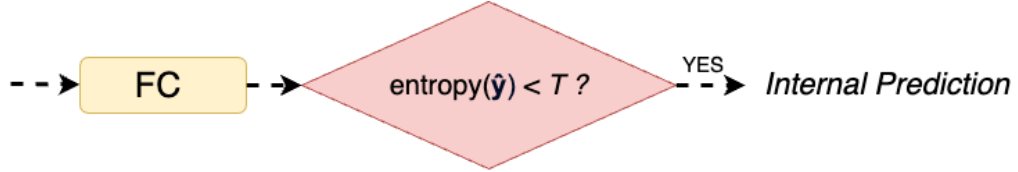


Figure 2.3: Entropy as the exiting criterion at an intermediate branch

2.3.3 Exiting Criterion

During inference, a sample begins with the lowest exit and iterates to the highest and final exit of the network. When the entropy of the predicted probability vector at any intermediate exit falls below a predetermined threshold T , the prediction at this exit is returned and the sample is not subjected to further processing by subsequent layers (see figure 2.3). Otherwise, the sample is continued to deeper layers for additional

processing. For a predicted probability vector $\hat{\mathbf{y}}$, entropy is defined as:

$$\text{entropy}(\hat{\mathbf{y}}) = - \sum_{c \in C} \hat{y}_c \log \hat{y}_c$$

Chapter 3

Improving Existing Confidence Measures

The previous chapter presents an outline of deep neural networks and discusses the early exiting approach to accelerate their inference. Specifically, it introduces a convolutional neural network, AlexNet, and its multi-exit variant, B-AlexNet. These two networks will serve as baseline models in the following chapters, where we evaluate methods aimed at improving the existing confidence measure entropy. In this chapter, we explore two approaches to refine entropy as the early exiting criterion: using maximum probability and applying a Dirichlet framework.

3.1 Maximum Probability

BranchyNet achieves accelerated inference with the entropy measure. We suggest using a simpler confidence metric, *maximum probability*, and show that it further reduces the average inference time.

3.1.1 Motivation

Maximum probability is one of the most fundamental and straightforward confidence measures used in out-of-distribution sample detection [12, 21, 24, 25]. It is defined as the maximum entry of a softmax probability vector $\hat{\mathbf{y}}$:

$$\text{maxProb}(\hat{\mathbf{y}}) = \max_{c \in C} \hat{y}_c$$

where C is the set of all possible labels. Neural networks generate higher softmax probabilities for in-distribution data and lower probabilities for out-of-distribution data. Although shown suboptimal compared to entropy-based confidence in handling certain distribution shifts [43], it remains more useful in misclassification detection [26]. Maximum probability as confidence measure has also been proven to be effective in other efficient inference objectives: network cascade [40] and neural network architecture optimization [15, 42]. We wish to investigate its efficacy as a confidence measure in early exiting scenarios with minimum network modifications.

Benefits of using maximum probability include its efficiency, as finding the maximum value among a set of probabilities is faster than entropy calculation. It also offers clear interpretability as it reflects the network’s confidence in its most likely prediction. The bounded range of 0 to 1 simplifies the selection of exit thresholds. Furthermore, when the network is highly confident in a prediction, maximum probability becomes a sufficient statistic for the overall entropy, as the entropy of other class probabilities vanishes.

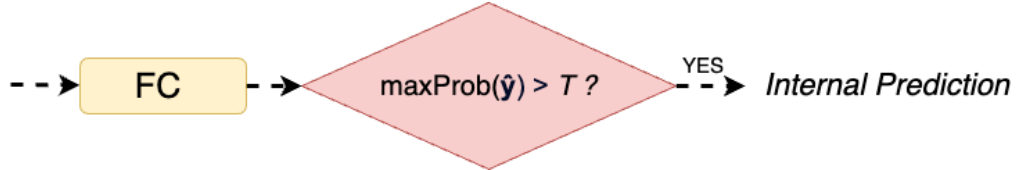


Figure 3.1: Maximum probability as the exiting criterion at an intermediate branch

3.1.2 Results

We substitute the entropy criterion with maximum probability (see figure 3.1). Thresholds were found via sweeping over a set of possible values, as for entropy. Experiments were run in Python 3.11 using an NVIDIA MIG GPU with 10 GB of GPU memory. Comparing it to the original Branchy-AlexNet (B-AlexNet) using entropy, we show that it maintains equivalent classification accuracy while notably reducing the average runtime per sample by 13.8%. Results are summarized in figure 3.2. We observe that B-AlexNet with maximum probability achieves the most significant efficiency gain without sacrificing accuracy.

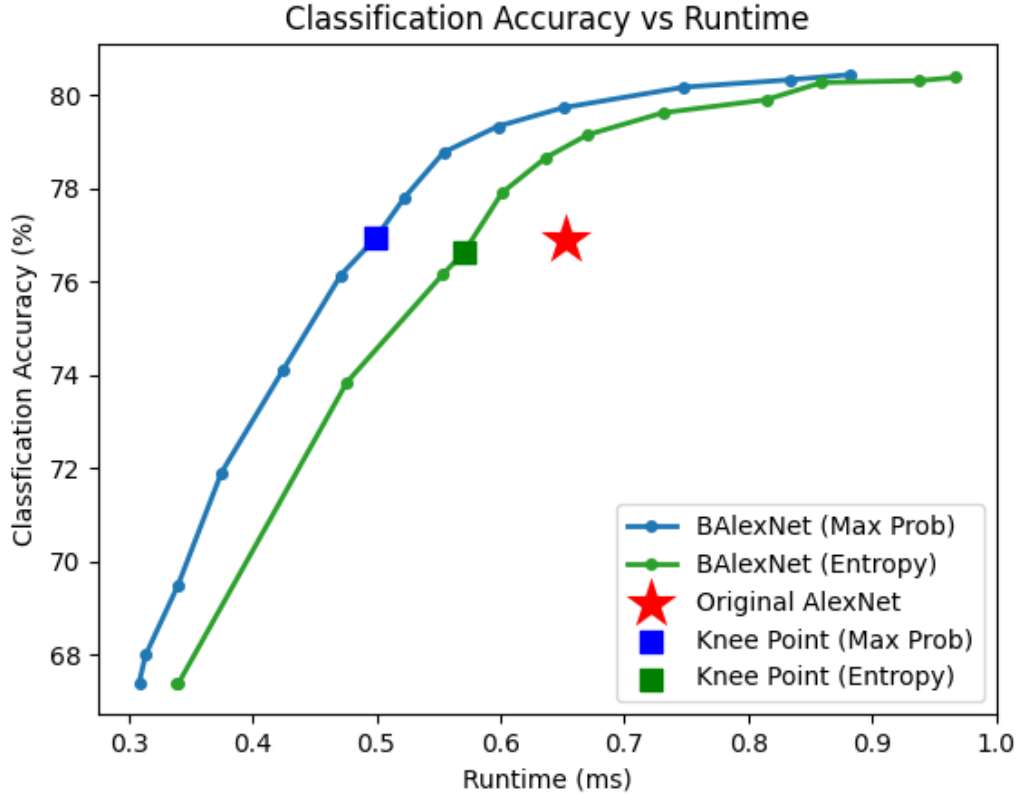


Figure 3.2: Classification accuracy versus runtime for B-AlexNet using two confidence measure compared with original AlexNet. Knee points represent the thresholds where the accuracy of B-AlexNet is comparable to the original AlexNet.

3.2 A Dirichlet Framework

The notion of uncertainty is of crucial importance in machine learning and forms a fundamental aspect of its methodologies. Though maximum probability and entropy both effectively serve as confidence measures to allow confident intermediate predictions to early exit, thereby reducing inference time, they are both measures of *total uncertainty*. Total uncertainty can be further broken down into two dimensions: *aleatoric uncertainty* and *epistemic uncertainty* [6, 16, 26]. Aleatoric uncertainty, also known as *data uncertainty*, usually refers to the irreducible uncertainty which arises from the natural complexity of data, such as class overlap and label noise. Epistemic uncertainty (we will refer to it as the *model uncertainty* in our context), on the other hand, is the uncertainty caused by a lack of training data or insufficient model complexity.

In this section, we start with the observation that measures of total uncertainty sometimes fail to reflect true model uncertainty, especially on data with high aleatoric uncertainty. We argue that for early exiting objectives, our goal is to identify true model uncertainty and allow samples to exit early if such uncertainty is sufficiently low. Based on the same model architecture as previous experiments, B-AlexNet, we propose using a Dirichlet framework to directly measure model uncertainty, utilizing two different training mechanisms. Our findings show that this framework allows us to successfully distinguish model uncertainty from total uncertainty and enables more data with high aleatoric uncertainty to exit early compared to B-AlexNet with total uncertainty measure.

3.2.1 Motivation

The underlying assumption of BranchyNet[38] and many other confidence-based early exiting frameworks [28, 40] is that if a neural network understands an image, its pre-

dictions should converge to a single class probability vector through successive layers. Therefore, total uncertainty such as entropy should be strictly decreasing towards zero. Based on this premise, predictions that exhibit low entropy are indicative of high model confidence. This assumption is generally accurate for unambiguous samples (see figure 3.3b). An unambiguous sample is one that clearly belongs to one class (see figure 3.4a). However, in the case of an ambiguous image (see figure 3.4b), even when the network correctly interprets the image, its predictions may converge to a probability vector that assigns comparable probabilities to multiple classes simultaneously, resulting in high entropy (see figure 3.3d). Thus, for ambiguous images, early exiting mechanisms that rely on total uncertainty may consistently route these images to the final exit, due to their overall high uncertainty.

Table 3.1 summarizes four possible combinations of data uncertainty and model uncertainty, with examples for each. Among these four cases, only case 1 has low total uncertainty, allowing potential early exits for samples from this category—unambiguous in-distribution data—when using total uncertainty measure. Given that data uncertainty is irreducible, our objective is to return an intermediate prediction as soon as the model uncertainty becomes low, regardless of data uncertainty. This will allow examples in the second case—ambiguous in-distribution data—to potentially exit early in the network, even when their network predictions have high entropy due to ambiguity.

Table 3.1: Four possible combinations of model uncertainty and data uncertainty with examples. Total uncertainty is the sum of model uncertainty and data uncertainty.

	Model Uncertainty	Data Uncertainty	Total Uncertainty	Examples
Case 1	low	low	low	Unambiguous In-Distribution Data
Case 2	low	high	high	Ambiguous In-Distribution Data
Case 3	high	low	high	Unambiguous Out-of-Distribution Data
Case 4	high	high	high	Ambiguous Out-of-Distribution Data

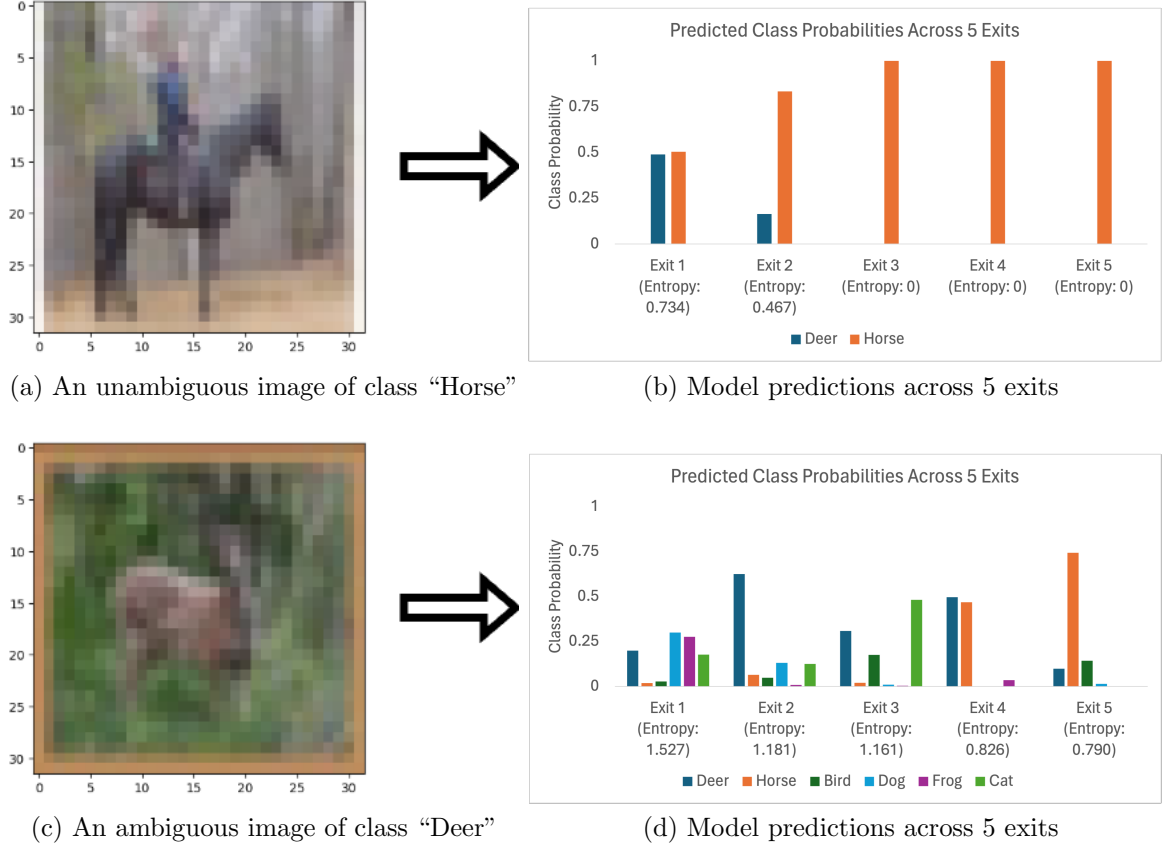


Figure 3.3: Model predictions for an unambiguous image versus an ambiguous image. For an unambiguous image, predictions converge to a single class probability vector and entropy gravitates towards zero. For an ambiguous image, predictions continuously assign comparable probabilities to multiple classes and entropy remains high throughout all exits.

3.2.2 The Dirichlet Framework

To differentiate between model uncertainty and data uncertainty from total uncertainty, which is calculated from the model’s output—a single probability vector—the assumption that neural networks produce a single point prediction seems inadequate. Given that the Dirichlet distribution is defined over a set of probability vectors, Dirichlet neural networks come in as a natural rescue [26, 34]. The Dirichlet framework posits that neural networks’ output is a *Dirichlet distribution over probability vectors* rather than a single probability vector. Model uncertainty can be quantified by the spread of the Dirichlet distribution. Under this assumption, we will show that

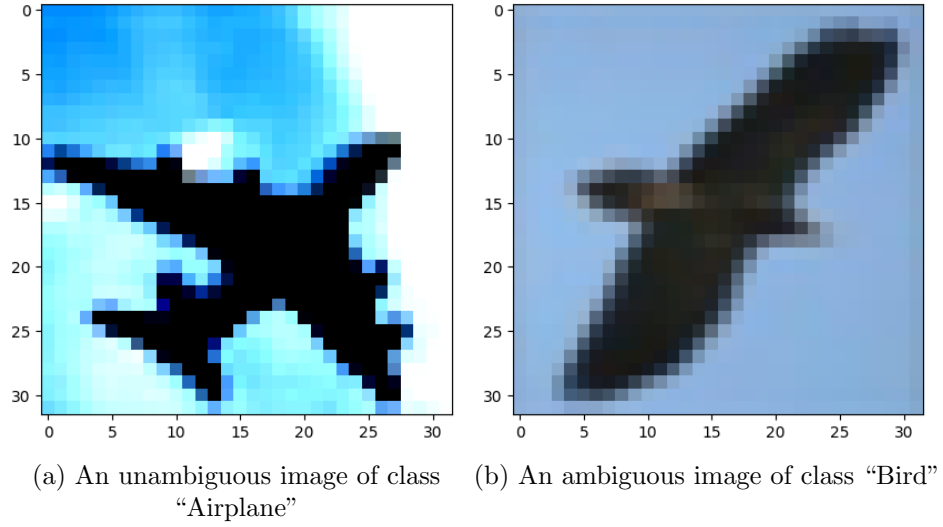


Figure 3.4: An unambiguous image versus an ambiguous image: a prototypical image of airplane and a deceiving image of bird that may be seen as an airplane

the neural network’s softmax output equals to the expected value of the Dirichlet distribution. Because both narrowly and widely scattered Dirichlet distributions may have the same expected values, this framework allows outputs with the same total uncertainty differ in model uncertainty. The Dirichlet distribution of probability vectors over the set of all possible classes C has dimension $|C| > 1$, is parameterized by $|C|$ concentration parameters $\alpha_1, \dots, \alpha_{|C|}$, and is defined as:

$$\text{Dir}(\hat{\mathbf{y}}|\boldsymbol{\alpha}) = \frac{\Gamma(\alpha_0)}{\prod_{c \in C} \Gamma(\alpha_c)} \prod_{c \in C} \hat{y}_c^{\alpha_c - 1}, \quad \alpha_c > 0, \quad \alpha_0 = \sum_{c \in C} \alpha_c$$

where $\hat{\mathbf{y}}$ is a predicted probability vector over C and $\Gamma(\cdot)$ is the gamma function:

$$\Gamma(z) = \int_0^\infty t^{z-1} e^{-t} dt$$

To visualize this, neural networks’ output can be seen as a $(|C| - 1)$ -simplex, where C is the set of all possible classes, and each point on the simplex corresponds to a proper probability vector over C . Figure 3.5¹ shows four examples of such simplex

¹Visualizations from <https://observablehq.com/@herbps10/dirichlet-distribution>.

over three classes. Mathematically, the posterior probability is then reparameterized as:

$$P(\hat{y} = c|x, \mathcal{D}) = \int p(\hat{y} = c|\alpha)p(\alpha|x, \mathcal{D}) d\alpha$$

where $\hat{y}$ is the predicted class, $c \in C$ is any class, x is the input, $\mathcal{D}$ is the set of training data, and α is the set of parameters of the output Dirichlet.

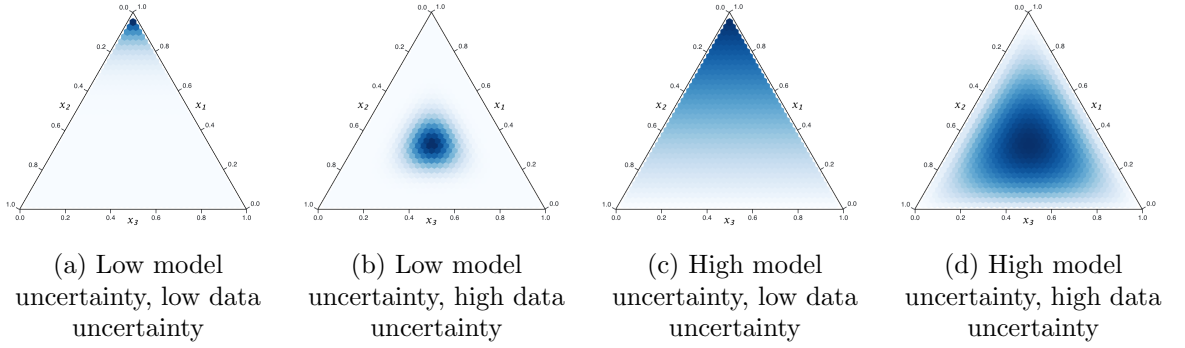


Figure 3.5: Four possible Dirichlet simplex outputs under different kinds of model and data uncertainty

Assuming that the trained model weights $\mathbf{w}$ is a sufficient statistic for the training dataset $\mathcal{D}$ and the neural network outputs a deterministic α^* (i.e. $\alpha^* = f(x, \mathbf{w})$, where f is the function that the neural network computes), the posterior probability simplifies to:

$$\begin{aligned}
P(\hat{y} = c|x, \mathbf{w}) &= \int p(\hat{y} = c|\alpha)p(\alpha|x, \mathbf{w}) d\alpha \\
&= \int p(\hat{y} = c|\hat{\mathbf{y}})p(\hat{\mathbf{y}}|\alpha)p(\alpha|x, \mathbf{w}) d\hat{\mathbf{y}}d\alpha \\
&= \int p(\hat{y} = c|\hat{\mathbf{y}})p(\hat{\mathbf{y}}|\alpha^*) d\hat{\mathbf{y}} \\
&= \int \hat{y}_c \cdot \text{Dir}(\hat{\mathbf{y}}|\alpha^*) d\hat{\mathbf{y}} \\
&= \mathbb{E}_{\text{Dir}(\hat{\mathbf{y}}|\alpha^*)}[\hat{y}_c] \\
&= \frac{\alpha_c^*}{\alpha_0^*}
\end{aligned}$$

where the last equation stems from the definition of expectation of a Dirichlet distribution. We observe that the posterior probability given by the softmax output is:

$$P(\hat{y} = c|x, \mathbf{w}) = \frac{e^{z_c(x)}}{\sum_{k \in C} e^{z_k(x)}}$$

where $z_c(x)$ is the neural network’s output before softmax. By equating the two, we have:

$$\frac{\alpha_c^*}{\alpha_0^*} = \frac{e^{z_c(x)}}{\sum_{k \in C} e^{z_k(x)}}$$

Under this formulation, the neural network is essentially predicting the parameters of the Dirichlet distribution. More precisely, it predicts the logarithmic of $\boldsymbol{\alpha}$, $\log(\boldsymbol{\alpha})$. We use differential entropy to measure the spread of the Dirichlet distribution, which quantifies model uncertainty:

$$\text{diffEntropy}(\boldsymbol{\alpha}) = \ln B(\boldsymbol{\alpha}) + (\alpha_0 - |C|)\psi(\alpha_0) - \sum_{c \in C} (\alpha_c - 1)\psi(\alpha_c)$$

3.2.3 Training

Training the neural network under this Dirichlet framework using cross entropy loss is degenerate because such loss encourages only the expected values of the Dirichlet to become closer to the ground truth labels. It is insensitive to arbitrary of scaling of $\boldsymbol{\alpha}$, which controls the spread of the Dirichlet. Hence, it is necessary to change the loss function to explicitly train the network to yield a sharp distribution (i.e. narrow spread) around the target labels. We define $\mathcal{D}_{\text{unamb}}$, $\mathcal{D}_{\text{amb}}$ as a set of unambiguous training data and a set of ambiguous training data, respectively. The construction of $\mathcal{D}_{\text{unamb}}$ and $\mathcal{D}_{\text{amb}}$ is elaborated later in this section. We borrowed ideas from Dirichlet Prior Network in out-of-distribution sample detection [26, 27] and define

two new losses based on Kullback-Leibler Divergence (KL Divergence):

$$L_1(\hat{\boldsymbol{\alpha}}) = \frac{1}{|\mathcal{D}_{\text{unamb}}|} \sum_{\mathcal{D}_{\text{unamb}}} KL[\text{Dir}(\hat{\mathbf{y}}|\hat{\boldsymbol{\alpha}}) || \text{Dir}(\hat{\mathbf{y}}|\boldsymbol{\alpha}_{\text{unamb}})] \quad (3.1)$$

$$\begin{aligned} L_2(\hat{\boldsymbol{\alpha}}) = & \frac{1}{|\mathcal{D}_{\text{unamb}}|} \sum_{\mathcal{D}_{\text{unamb}}} KL[\text{Dir}(\hat{\mathbf{y}}|\hat{\boldsymbol{\alpha}}) || \text{Dir}(\hat{\mathbf{y}}|\boldsymbol{\alpha}_{\text{unamb}})] \\ & + \frac{1}{|\mathcal{D}_{\text{amb}}|} \sum_{\mathcal{D}_{\text{amb}}} KL[\text{Dir}(\hat{\mathbf{y}}|\hat{\boldsymbol{\alpha}}) || \text{Dir}(\hat{\mathbf{y}}|\boldsymbol{\alpha}_{\text{amb}})] \end{aligned} \quad (3.2)$$

where $\hat{\boldsymbol{\alpha}}$ is the neural network’s prediction and $\boldsymbol{\alpha}_{\text{unamb}}, \boldsymbol{\alpha}_{\text{amb}}$ are the parameters of the target Dirichlets for unambiguous data and ambiguous data, respectively. The target concentration parameters $\boldsymbol{\alpha}_{\text{unamb}}, \boldsymbol{\alpha}_{\text{amb}}$ are set as follows:

$$\boldsymbol{\alpha}_{\text{unamb}}^{(c)} = \begin{cases} \beta + 1 & \text{if } c = k \\ 1 & \text{otherwise} \end{cases}$$

where k is the ground truth class of the unambiguous sample.

$$\boldsymbol{\alpha}_{\text{amb}}^{(c)} = \begin{cases} \beta + 1 & \text{if } c \in K \\ 1 & \text{otherwise} \end{cases}$$

where K is the set of classes the ambiguous sample may belong to. β is a hyperparameter; we used $\beta = 100$.

Under this setting, loss 1 in 3.1 guides the network to produce a sharp distribution centered on the ground truth class for unambiguous samples, as in figure 3.5a. Loss 2 in 3.2 shares the same incentive as loss 1 and additionally motivates the network to generate a sharp distribution centered around a specific region representing class intersections for ambiguous samples, as illustrated in figure 3.5b. Both losses encourage high confidence on the correct interpretation, for both unambiguous data and ambiguous data.

The Unambiguous Dataset

We use the original training dataset in CIFAR-10 as $\mathcal{D}_{\text{unamb}}$ to preserve the same assumption in cross-entropy-loss training that each image in the set has exactly one ground truth label. We will refer to B-AlexNet trained using loss 1 in 3.1 within the Dirichlet framework as *Dir-B-AlexNet*.

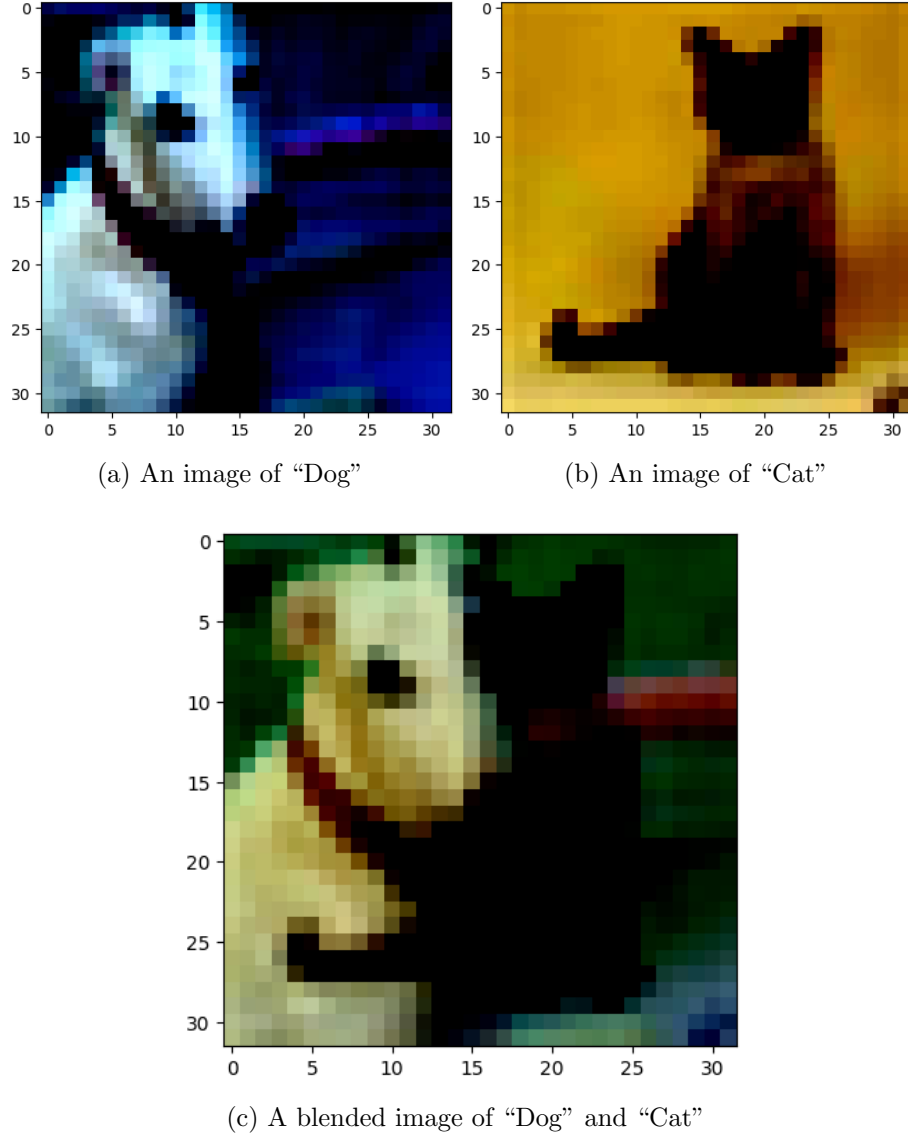


Figure 3.6: Blending an image of “Dog” and an image of “Cat” into an ambiguous image of “Cat” and “Dog”

The Ambiguous Dataset

Given the challenge to obtain a large set of naturally ambiguous data, $\mathcal{D}_{\text{amb}}$ is a set of artificial images generated by blending images from distinct classes in the CIFAR10 training dataset. These images represent the overlaps of the classes from which they are derived. Figure 3.6 illustrates an example of the blending process.

We generate 50,000 blended images to match with the size of CIFAR-10. Images were blended from six groups of common overlapping classes: 10,000 images of “Cat vs Dog”, 10,000 images of “Deer vs Horse”, 10,000 images of “Bird vs Airplane”, 10,000 images of “Automobile vs Truck”, 5,000 images of animals (Bird, Cat, Deer, Dog, Frog, Horse), 5,000 images of vehicles (Airplane, Automobile, Ship, Truck). B-AlexNet trained using loss 2 in 3.2 will henceforth be referred to as *Amb-Dir-B-AlexNet*.

Other training aspects are consistent with those of B-AlexNet. As expected, training Amb-Dir-B-AlexNet is slower than training Dir-B-AlexNet due to its more complex objective, as shown in figure 3.7. It is worth mentioning that both loss functions appear well-defined.

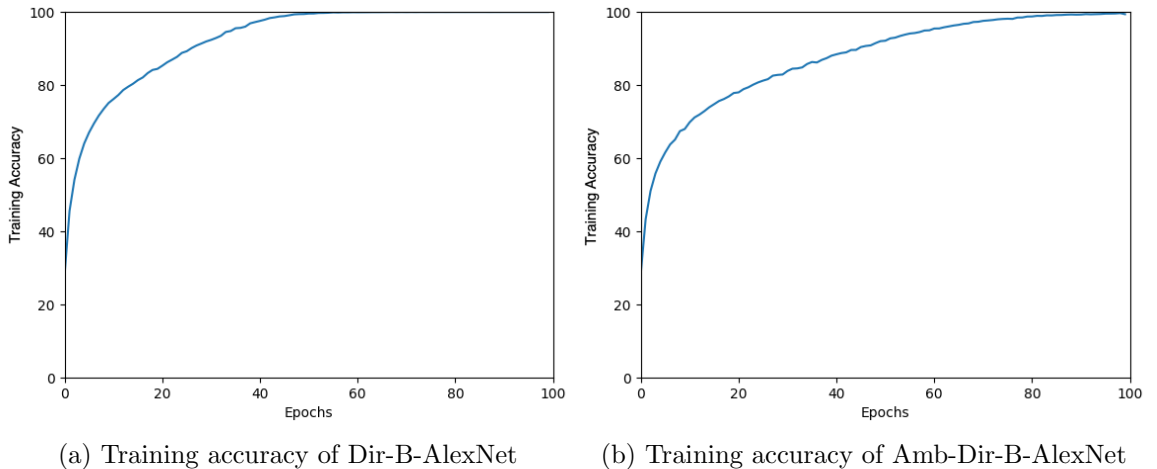


Figure 3.7: Training accuracies of Dir-B-AlexNet and Amb-Dir-B-AlexNet: Dir-B-AlexNet reaches a high training accuracy faster. Accuracy is measured on unambiguous data only.

3.2.4 Results

Once Dir-B-AlexNet and Amb-Dir-B-AlexNet are trained, we utilize the differential entropy metric mentioned earlier as our early exiting criterion (see figure 3.8). All experiments were run in Python 3.11 using an NVIDIA MIG GPU with 10 GB of GPU memory.

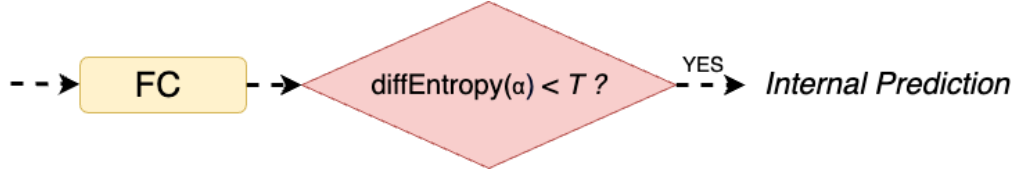


Figure 3.8: Differential Entropy as the exiting criterion at an intermediate branch

Evaluation Datasets

The original CIFAR10 testing dataset serves as the baseline unambiguous dataset for our evaluation. We aim to analyze how B-AlexNet, Dir-B-AlexNet, and Amb-Dir-B-AlexNet perform differently in handling ambiguous data compared to unambiguous data, represented by this baseline. For ambiguous data, we utilize three datasets: *Challenging Test*, *External Blend*, and *Internal Blend*. Challenging Test consists of 866 images found challenging by AlexNet, indicated by entropy values over 0.5. We confirm its ambiguity by observing that all branches in the three models achieve an accuracy around 50%. External Blend is generated by blending images from CIFAR10 testing data using a different distribution than the one used for training Amb-Dir-AlexNet. Specifically, it includes 2,500 images of “Bird vs Frog”, 2,500 images of “Ship vs Truck”, 2,500 images of “Automobile vs Airplane” and 2,500 images of “Deer vs Dog”. Internal Blend contains 10,000 images blended using the same distribution as the ambiguous training data for Amb-Dir-AlexNet.

Evaluation Metrics

We employ two evaluation metrics: 1) the difference in average uncertainty between ambiguous and unambiguous data, and 2) the percentage of ambiguous samples that can exit early through the first exit, when thresholds are set to allow 1, 25, 50, 75, 99% of unambiguous samples to exit from the first exit. The greater the difference in average uncertainty between ambiguous and unambiguous data, the more imbalanced the model is in handling these two types of data. The second metric directly assesses the practical utility of different uncertainty measures targeting ambiguous samples in early exiting.

Evaluation Methods

For the first evaluation metric, we first obtained the mean uncertainty for each model (entropy for B-AlexNet, differential entropy for Dir-B-AlexNet and Amb-Dir-B-AlexNet) on CIFAR10 testing dataset, as the baseline uncertainty measure for unambiguous data. Next, we computed the mean uncertainty across Challenging Test, External Blend, and Internal Blend. Given the difference in scale for entropy and differential entropy, we utilized min-max normalization to standardized the results:

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

To mitigate the impact of outliers, we selected the second highest and second lowest values as the maximum and minimum, respectively. This normalization process scales the results to a range between 0 and 1. We then calculated the difference between the normalized mean uncertainty for unambiguous data and that for ambiguous data. We report results in both relative and absolute difference. Relative difference is computed as the ratio of the normalized uncertainty for ambiguous data to the unambiguous

data, while absolute difference is obtained by subtracting the normalized uncertainty of unambiguous data from that of ambiguous data.

For the second evaluation metric, we manually identified model-specific thresholds that enabled 1, 25, 50, 75, 99% of the CIFAR10 testing samples to exit through the first exit by sweeping across a range of potential values. We then recorded the percentage of samples from Challenging Test, External Blend, Internal Blend that were able to exit through the first exit at these thresholds.

Evaluation Results

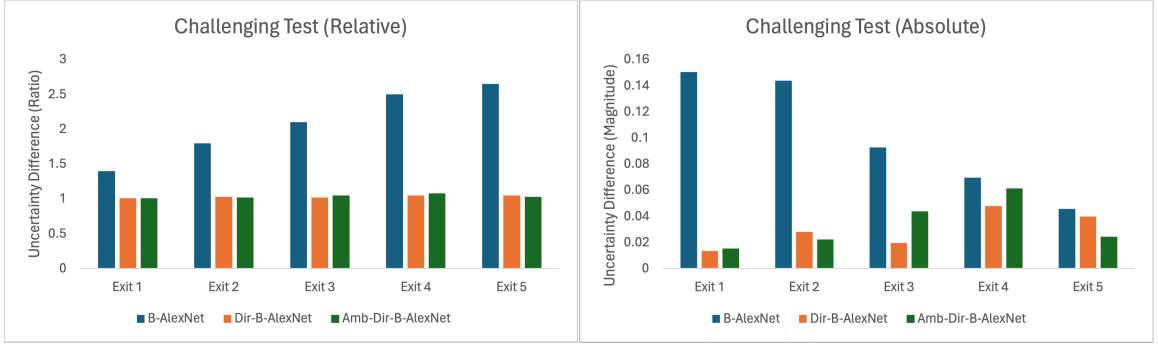
Figure 3.9, 3.10, 3.11 present a comparison of three models’ differences in average uncertainty between unambiguous data and ambiguous data across five exits for Challenging Test, External Blend, Internal Blend, respectively. Amb-Dir-B-AlexNet consistently demonstrates the smallest difference between the two types of data, followed closely by Dir-B-AlexNet. B-AlexNet exhibits the largest difference across all exits for all datasets. This suggests that our approach—using differential entropy within the Dirichlet framework—is effective in separating model uncertainty from total uncertainty, reducing the imbalance in handling ambiguous versus unambiguous data that typically arises in total uncertainty measure.

Both Amb-Dir-B-AlexNet and Dir-B-AlexNet maintain a consistently low relative difference (close to 1.0 ratio) across all five exits, while the relative difference for B-AlexNet experiences an increase from exit 1 to exit 5. We suspect this is because as B-AlexNet becomes more confident in its prediction, the entropy for unambiguous data approaches zero and the entropy for ambiguous data remains at a constant value greater than zero, leading to an amplified disparity in the uncertainty measure.

Amb-Dir-B-AlexNet and Dir-B-AlexNet exhibit comparable performances, except on the Internal Blend dataset. Amb-Dir-AlexNet shows a remarkably small difference on Internal Blend: its uncertainty in ambiguous data is even lower than unambiguous

data for exit 4 and 5. This aligns with our expectation since Amb-Dir-B-AlexNet is specifically trained on data with the same distribution as Internal Blend, which may enable the model to recognize images from this category faster, resulting in lower model uncertainty.

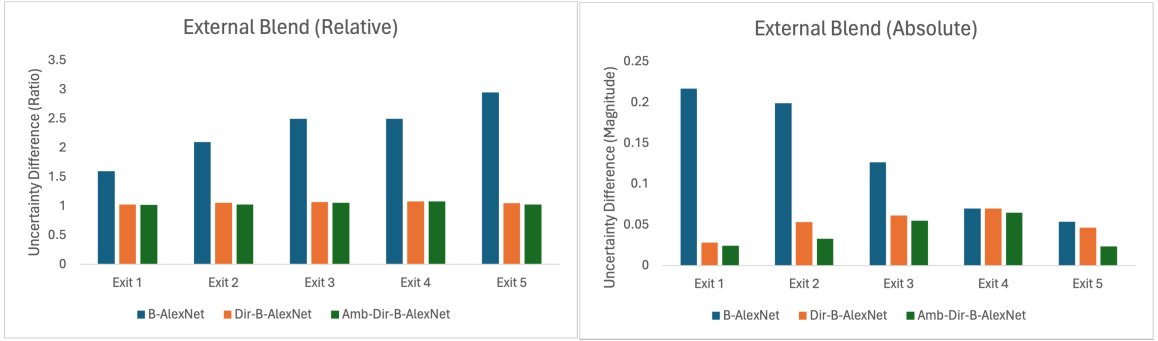
Figure 3.12 shows the results as per the second evaluation metric, capturing the fraction of ambiguous samples that can exit through exit 1 when 1, 25, 50, 75, 99% unambiguous samples can do so. We observe that Amb-Dir-B-AlexNet again outperforms the other two models, indicated by its consistently higher percentage values across all thresholds and datasets. However, this margin is less pronounced than we initially anticipated. It is also noteworthy that Dir-B-AlexNet, despite showing a comparable performance as Amb-Dir-B-AlexNet under the first evaluation metric, falls considerably behind both B-AlexNet and Amb-Dir-B-AlexNet with this metric. The reasons for this discrepancy is not immediate clear and need further investigation. We will leave such exploration to future work.



(a) Relative Difference on Challenging Test

(b) Absolute Difference on Challenging Test

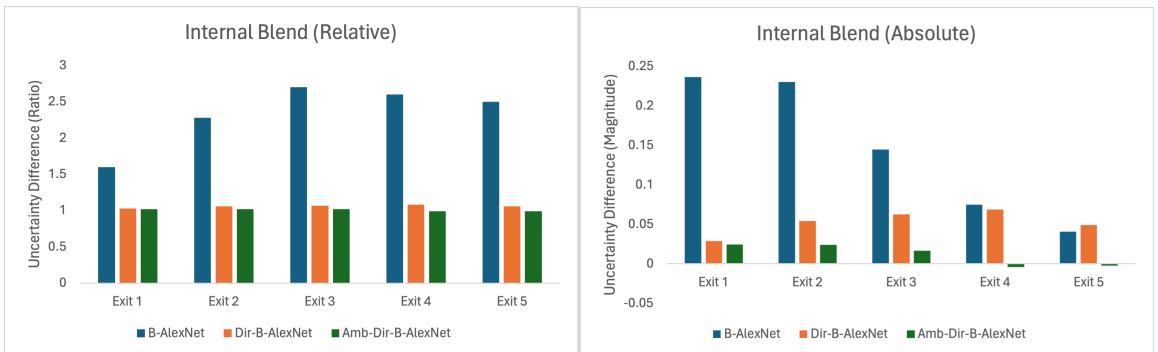
Figure 3.9: A model comparison of uncertainty difference on Challenging Test compared to CIFAR10 testing dataset. A smaller difference indicates better performance in handling ambiguous data.



(a) Relative Difference on External Blend

(b) Absolute Difference on External Blend

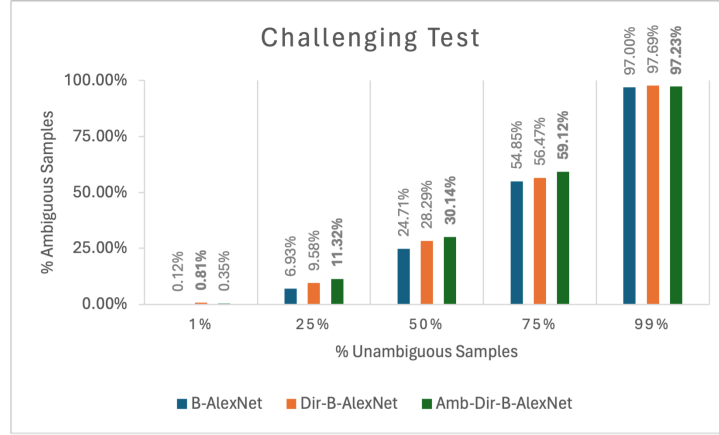
Figure 3.10: A model comparison of uncertainty difference on External Blend compared to CIFAR10 testing dataset. A smaller difference indicates better performance in handling ambiguous data.



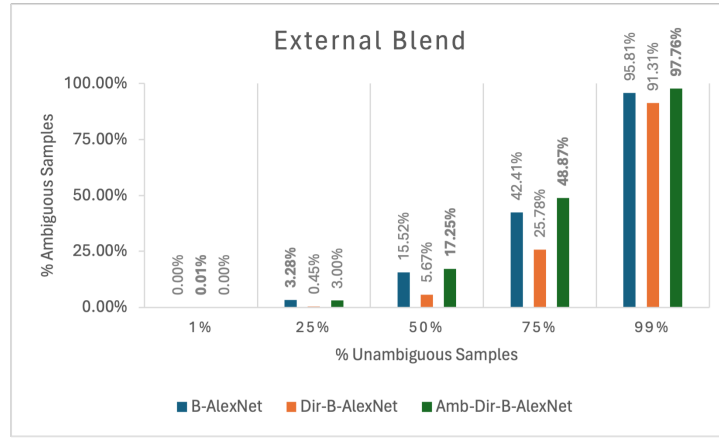
(a) Relative Difference on Internal Blend

(b) Absolute Difference on Internal Blend

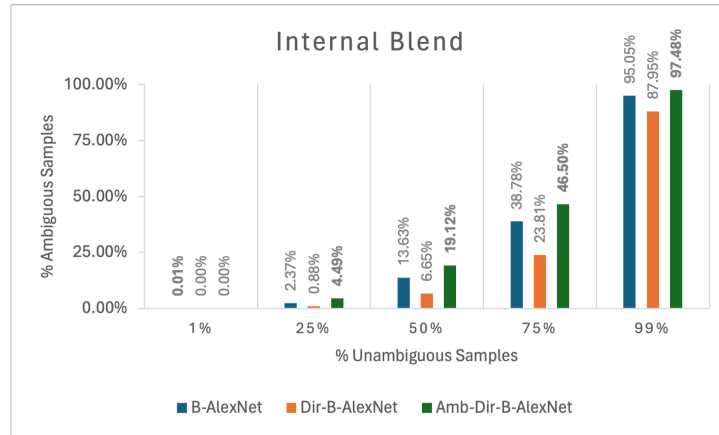
Figure 3.11: A model comparison of uncertainty difference on Internal Blend compared to CIFAR10 testing dataset. A smaller difference indicates better performance in handling ambiguous data.



(a) Percentage of ambiguous samples exiting from exit 1 in Challenging Test



(b) Percentage of ambiguous samples exiting from exit 1 in External Blend



(c) Percentage of ambiguous samples exiting from exit 1 in Internal Blend

Figure 3.12: A model comparison of percentage of ambiguous samples exiting from exit 1 when 1, 25, 50, 75, 99% unambiguous samples exit from exit 1. A higher percentage indicates a better performance in early exiting. Percentages with the highest values are highlighted in bold.

Chapter 4

Alternative Perspectives on Early Exiting Criteria

Previous chapters focus on confidence as the primary criterion for early exiting decisions in deep neural networks. This conventional perspective typically relies on a single point prediction for confidence computation. In addition, it is anchored solely on the model’s confidence for the prediction, often overlooking the accuracy of the prediction itself. In this chapter, we depart from this singular perspective and discuss two different yet interconnected angles to the early exiting decision problem: *stability-based criteria* and *loss-based criteria*. We focus our discussion on elucidating their motivations along with some preliminary analyses.

4.1 Stability-Based Criteria

Intuitively, if it is possible to identify and exit from the branch whose intermediate prediction aligns with those of all subsequent branches, inference time can be strictly reduced while ensuring that accuracy is at least equal to the accuracy of the final exit, which is assumed to be the highest among all branches. Hence, one effective early exiting strategy is to identify the branch from which predictions stabilize. We

refer to criteria used to assess whether a current prediction is stable as *stability-based criteria*.

Previous research has termed this crucial branch as *prediction depth* and shows it has many useful properties [1]. A sample’s prediction depth is closely related to its learning difficulty, classification accuracy, and prediction uncertainty. This measure is also consistent across different model architectures. Therefore, the ability to predict a sample’s prediction depth does not only help in early exiting but can potentially enhance other aspects in machine learning. The effectiveness of identifying the opposite of prediction consistency—the amount an internal prediction diverges from the final prediction—is further underscored by previous research in network confusion analysis [18]. Authors demonstrate that such inconsistency serves as a potent error indicator as a first step towards error mitigation.

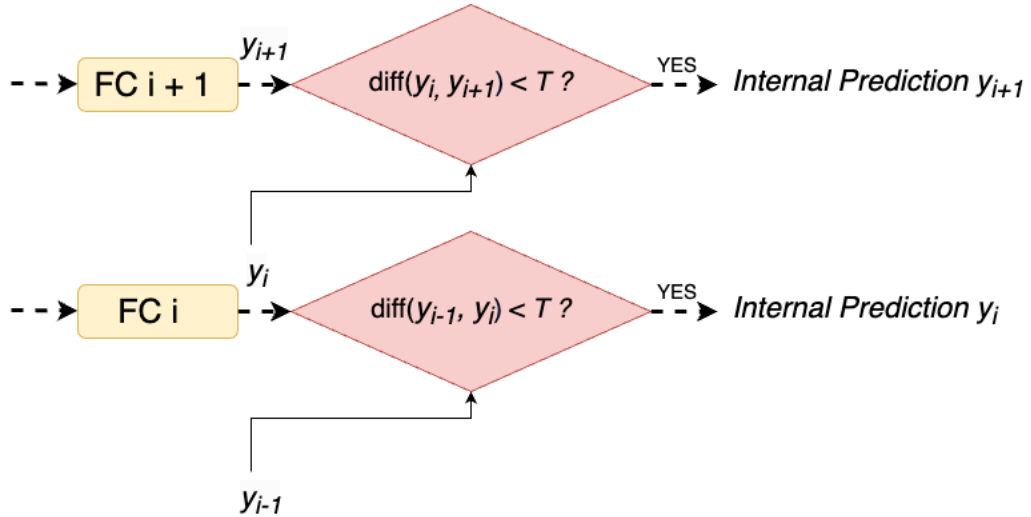


Figure 4.1: Stability as the exiting criterion at an intermediate branch

During inference, it is challenging to determine whether the prediction at the current branch is consistent with future branches due to a lack of information about the future. Since information is only available for preceding branches, we suggest using this information instead to estimate the current stability. Specifically, we pro-

pose to evaluate the discrepancy between the current prediction and the previous prediction: if it is sufficiently small, we infer that the current prediction is stable, thereby permitting early exit from the current branch (see figure 4.1). The underlying assumption of this approach is that predictions across layers generally converge. Hence, if a prediction has not changed significantly from the previous, it is likely to remain consistent in subsequent layers. Similar stability-based approaches have been applied to transformers on language data [33], where authors train a small binary multi-layer perceptron as a consistency estimator.

As an initial analysis, we tested three straightforward metrics to gauge the difference between the current predicted probability vector with one from the previous layer: 1) Kullback-Leibler Divergence (KL Divergence), 2) L1 Distance, and 3) L2 distance:

$$KL(\mathbf{y}_i \parallel \mathbf{y}_{i-1}) = \sum_{c \in C} y_{i,c} \log \left(\frac{y_{i,c}}{y_{i-1,c}} \right)$$

$$L_1(\mathbf{y}_i, \mathbf{y}_{i-1}) = \sum_{c \in C} |y_{i,c} - y_{i-1,c}|$$

$$L_2(\mathbf{y}_i, \mathbf{y}_{i-1}) = \sqrt{\sum_{c \in C} (y_{i,c} - y_{i-1,c})^2}$$

where $\mathbf{y}_i$ is the current prediction, $\mathbf{y}_{i-1}$ is the previous prediction, and C is the set of all possible classes.

Experiments were run in Python 3.11 using an NVIDIA MIG GPU with 10 GB of GPU memory. Figure 4.2 presents a comparative analysis of the performances of three metrics alongside the baseline AlexNet. Notably, all three metrics provide a potential 4% increase in accuracy under strict threshold settings. L1 and L2 distance exhibit closely aligned performance trajectories, surpassing KL Divergence in terms of both reduced inference time and accuracy. This preliminary evidence provides a foundation for further exploration into the utility of stability-based methods in early exiting.

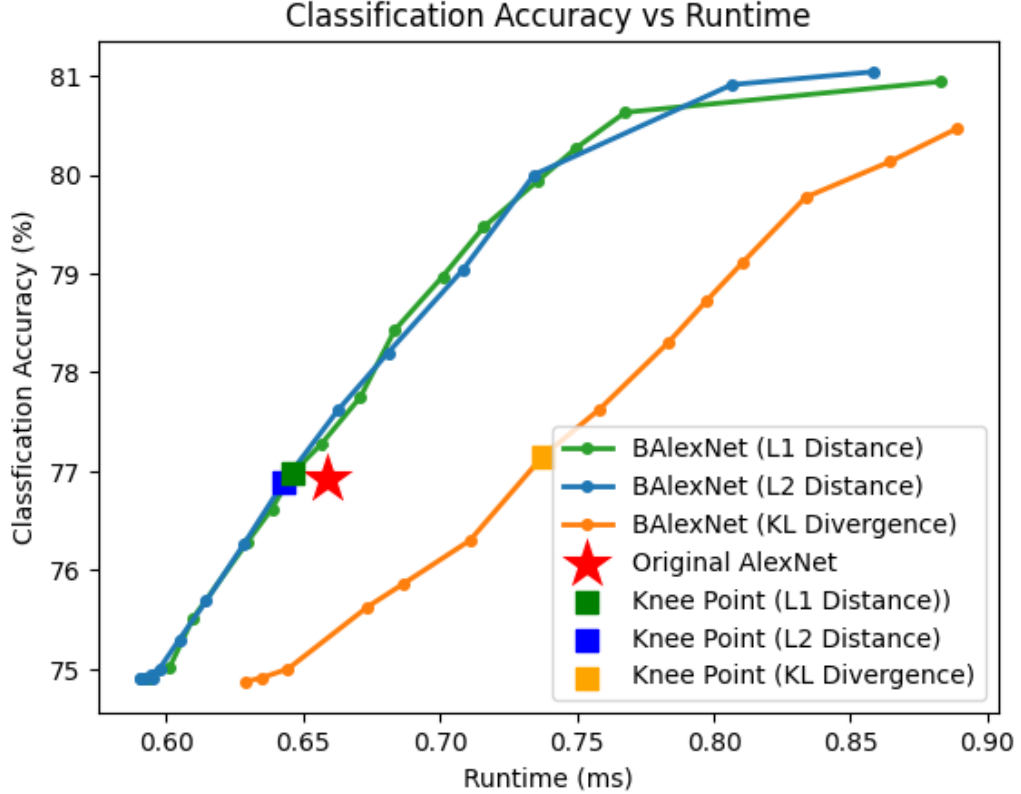


Figure 4.2: Classification accuracy versus runtime for B-AlexNet using three distance metrics. Knee points represent the thresholds where the accuracy of B-AlexNet is comparable to the original AlexNet.

4.2 Loss-Based Criteria

While stability-based methods aim to minimize inference time while preserving the accuracy of the original network, *loss-based* criteria, on the other hand, prioritize accuracy improvement without imposing excessive computational overhead. In this section, we motivate loss-based criteria by illustrating how an ensemble of branches across the network can collectively outperform each individual branch alone. Additionally, we showcase its connection with the confidence measure, entropy.

To demonstrate BranchyNet’s potential as an ensemble method for enhancing accuracy, we conducted an analysis on the CIFAR10 test dataset. For each sample in the dataset, we identified the branch with the lowest *cross-entropy loss* and computed

the accuracy of BranchyNet by routing each sample to its corresponding branch. Table 4.1 displays this highest achievable accuracy compared to accuracy of individual branch. Remarkably, the highest achievable accuracy of the BranchyNet ensemble exceeds the best branch (Exit 5) by 12.66%. Hence, with a reliable loss predictor, utilizing all branches can lead to a substantial improvement in accuracy. However, identifying such a predictor is not straightforward. We defer the task of finding such predictors and experiments to future work.

Table 4.1: Ensemble accuracy versus accuracies of individual branches

	Accuracy (%)
Exit 1	67.39
Exit 2	74.91
Exit 3	77.76
Exit 4	79.30
Exit 5	79.36
Ensemble	92.02

Entropy measure can be viewed as a special case of loss-based methods under a specific setting. Recall that the cross-entropy loss is defined as:

$$L(\hat{\mathbf{y}}, \mathbf{y}) = - \sum_{c \in C} y_c \log \hat{y}_c$$

and entropy is defined as:

$$\mathcal{H}(\hat{\mathbf{y}}) = - \sum_{c \in C} \hat{y}_c \log \hat{y}_c$$

where $\hat{\mathbf{y}}$ is the predicted probability vector and $\mathbf{y}$ is the one-hot encoded ground truth label vector. As $\hat{\mathbf{y}}$ approaches $\mathbf{y}$, entropy converges towards cross-entropy loss. Therefore, if a model effectively learns the data and consistently produces highly accurate predictions, then entropy can serve as a proxy for loss predictor.

For example, given a successful training procedure for B-AlexNet, $\hat{\mathbf{y}}$ for training data gradually approaches $\mathbf{y}$ from exit 1 to 5, so entropy converges to cross-entropy

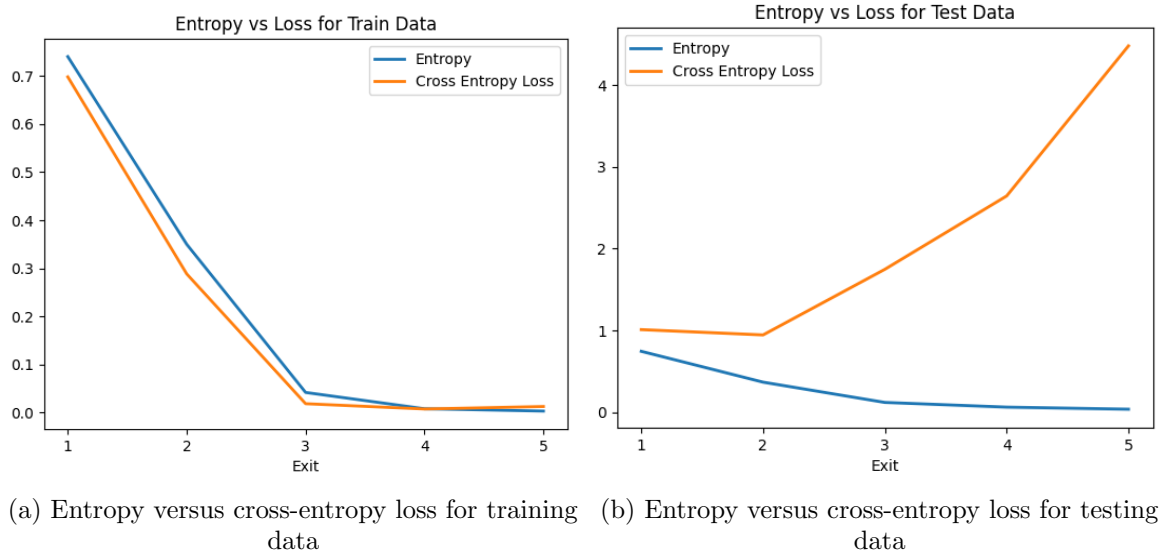


Figure 4.3: Entropy vs cross-entropy loss for training and testing data

loss, as illustrated in figure 4.3a. However, predictions may not consistently approach the ground truth. In fact, it may exhibit the opposite trend, as demonstrated in the CIFAR10 test dataset in figure 4.3b. We suspect this contrary behavior is due to the model getting overconfident in incorrect predictions across layers.

Chapter 5

Conclusion

Deep neural networks (DNNs) demonstrate impressive and robust performance in a wide range of machine learning tasks thanks to their depth [31], but this also comes at the cost of prolonged inference time. Early exiting is a strategy aimed at reducing average inference time by adaptively allowing “easy” samples to exit early in the network and “difficult” samples continue to leverage the full network capacity [30]. This thesis explores several exiting criteria within an early exiting DNN architecture on an image classification task. We show that a simple confidence measure using maximum probability achieves comparable performance to entropy measure while further accelerating inference. Existing exiting criteria that rely on total uncertainty measure do not consistently reflect true model uncertainty. To address this, we propose using a Dirichlet framework to directly gauge model uncertainty. Results suggests this approach facilitates early exits for ambiguous data. Finally, we broaden our exploration to discuss alternative principles for designing early exiting criteria: stability-based and loss-based approaches.

Our study has several limitations. First and foremost, our experiments focused on a single neural network architecture and a particular image classification dataset. This limits the ability to apply our conclusions to deep neural networks in general.

Further experiments on varied network architectures and diverse machine learning datasets and tasks are crucial to fully determine the applicability and generalizability of our results. Second, the technique we used to generate ambiguous datasets for training—by blending images from existing datasets—raises concerns about its validity: are blended images actually ambiguous? Blended images may contain less information than the combination of two individual images when their values cancel out; blending can produce overly noisy images; one image may overshadow the other during the blending process, resulting in an unambiguous image instead. Third, it remains unclear why Dir-B-AlexNet is outperformed by B-AlexNet under the second evaluation metric despite a clear advantage in the first metric. Last in our discussion, but certainly not the final limitation of our study, is the underutilization of the first exit in currently proposed stability-based methods. The first exit lacks a preceding layer as a reference to estimate its stability, which prevents samples from exiting at this stage. Future investigation is needed to better use this exit beyond its single role as the reference for the second exit. Addressing these limitations and questions will require extensive future work. We hope this thesis provides a foundation towards developing more interpretable, highly efficient, and robust early exiting strategies.

Bibliography

- [1] Robert Baldock, Hartmut Maennel, and Behnam Neyshabur. Deep learning through the lens of example difficulty. *Advances in Neural Information Processing Systems*, 34:10876–10889, 2021.
- [2] Andrea Banino, Jan Balaguer, and Charles Blundell. Pondernet: Learning to ponder. *arXiv preprint arXiv:2107.05407*, 2021.
- [3] Tolga Bolukbasi, Joseph Wang, Ofer Dekel, and Venkatesh Saligrama. Adaptive neural networks for efficient inference. In *International Conference on Machine Learning*, pages 527–536. PMLR, 2017.
- [4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.
- [5] Yu Cheng, Duo Wang, Pan Zhou, and Tao Zhang. A survey of model compression and acceleration for deep neural networks. *arXiv preprint arXiv:1710.09282*, 2017.
- [6] Armen Der Kiureghian and Ove Ditlevsen. Aleatory or epistemic? does it matter? *Structural safety*, 31(2):105–112, 2009.
- [7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- [8] Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6):1789–1819, 2021.
- [9] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.
- [10] Yizeng Han, Gao Huang, Shiji Song, Le Yang, Honghui Wang, and Yulin Wang. Dynamic neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11):7436–7456, 2021.

- [11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [12] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136*, 2016.
- [13] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.
- [14] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- [15] Gao Huang, Danlu Chen, Tianhong Li, Felix Wu, Laurens Van Der Maaten, and Kilian Q Weinberger. Multi-scale dense networks for resource efficient image classification. *arXiv preprint arXiv:1703.09844*, 2017.
- [16] Eyke Hüllermeier and Willem Waegeman. Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *Machine learning*, 110(3):457–506, 2021.
- [17] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016.
- [18] Yigitcan Kaya, Sanghyun Hong, and Tudor Dumitras. Shallow-deep networks: Understanding and mitigating network overthinking. In *International conference on machine learning*, pages 3301–3310. PMLR, 2019.
- [19] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.
- [20] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.
- [21] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.
- [22] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.
- [23] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

- [24] Kimin Lee, Honglak Lee, Kibok Lee, and Jinwoo Shin. Training confidence-calibrated classifiers for detecting out-of-distribution samples. *arXiv preprint arXiv:1711.09325*, 2017.
- [25] Shiyu Liang, Yixuan Li, and Rayadurgam Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. *arXiv preprint arXiv:1706.02690*, 2017.
- [26] Andrey Malinin and Mark Gales. Predictive uncertainty estimation via prior networks. *Advances in neural information processing systems*, 31, 2018.
- [27] Andrey Malinin and Mark Gales. Reverse kl-divergence training of prior networks: Improved uncertainty and adversarial robustness. *Advances in Neural Information Processing Systems*, 32, 2019.
- [28] Priyadarshini Panda, Abhronil Sengupta, and Kaushik Roy. Conditional deep learning for energy-efficient and enhanced pattern recognition. In *2016 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 475–480. IEEE, 2016.
- [29] Simone Scardapane, Danilo Comminiello, Michele Scarpiniti, Enzo Baccarelli, and Aurelio Uncini. Differentiable branching in deep networks for fast inference. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4167–4171. IEEE, 2020.
- [30] Simone Scardapane, Michele Scarpiniti, Enzo Baccarelli, and Aurelio Uncini. Why should we add early exits to neural networks? *Cognitive Computation*, 12(5):954–966, 2020.
- [31] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.
- [32] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11):2673–2681, 1997.
- [33] Tal Schuster, Adam Fisch, Tommi Jaakkola, and Regina Barzilay. Consistent accelerated inference via confident adaptive transformers. *arXiv preprint arXiv:2104.08803*, 2021.
- [34] Murat Sensoy, Lance Kaplan, and Melih Kandemir. Evidential deep learning to quantify classification uncertainty. *Advances in neural information processing systems*, 31, 2018.
- [35] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
- [36] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27, 2014.

- [37] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.
- [38] Surat Teerapittayanon, Bradley McDanel, and Hsiang-Tsung Kung. Branchynet: Fast inference via early exiting from deep neural networks. In *2016 23rd international conference on pattern recognition (ICPR)*, pages 2464–2469. IEEE, 2016.
- [39] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
- [40] Xin Wang, Yujia Luo, Daniel Crankshaw, Alexey Tumanov, Fisher Yu, and Joseph E Gonzalez. Idk cascades: Fast deep learning by learning not to overthink. *arXiv preprint arXiv:1706.00885*, 2017.
- [41] Weizheng Xu, Youtao Zhang, and Xulong Tang. Parallelizing dnn training on gpus: Challenges and opportunities. In *Companion Proceedings of the Web Conference 2021*, pages 174–178, 2021.
- [42] Le Yang, Yizeng Han, Xi Chen, Shiji Song, Jifeng Dai, and Gao Huang. Resolution adaptive networks for efficient inference. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2369–2378, 2020.
- [43] Gal Yona, Amir Feder, and Itay Laish. Useful confidence measures: Beyond the max score. *arXiv preprint arXiv:2210.14070*, 2022.