

# Robust and Hierarchy-Aware Classification

by

Nicholas Pellegrino

A thesis  
presented to the University of Waterloo  
in fulfillment of the  
thesis requirement for the degree of  
Doctor of Philosophy  
in  
Systems Design Engineering

Waterloo, Ontario, Canada, 2025

© Nicholas Pellegrino 2025

## **Examining Committee Membership**

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner: Steven Lake Waslander  
Professor, Institute for Aerospace Studies,  
University of Toronto

Supervisor: Paul Werner Fieguth  
Professor, Dept. of Systems Design Engineering,  
University of Waterloo

Internal Members: Arash Arami  
Associate Professor, Dept. of Mechanical and Mechatronics  
Engineering, cross-appointed to the Dept. of Systems  
Design Engineering,  
University of Waterloo

Linlin Xu  
Assistant Professor, Dept. of Geomatics Engineering,  
University of Calgary  
Adjunct Faculty, Dept. of Systems Design Engineering,  
University of Waterloo

Internal-External Member: Jeff Orchard  
Associate Professor, Cheriton School of Computer Science,  
University of Waterloo

## **Author's Declaration**

This thesis consists of material all of which I authored or co-authored: see [Statement of Contributions](#) included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

## Statement of Contributions

Nicholas Pellegrino is the sole author of [Chapters 3, 6 and 8](#), which were not written for publication. All chapters were authored under the supervision Dr. Paul W. Fieguth. This thesis consists in part of several manuscripts written for publication. Exceptions to sole authorship of material are as follows:

- Components of [Chapters 1, 2 and 4](#) were presented and published in the proceedings of the *Datasets and Benchmarks Track of the Thirty-seventh and Conference on Neural Information Processing Systems (NeurIPS 2023)* [\[1\]](#) and the *Datasets and Benchmarks Track of the Thirty-eighth Conference on Neural Information Processing Systems (NeurIPS 2024)* [\[2\]](#).
- Components of [Chapter 5](#) and [Chapter 6](#) were either presented and/or published at the *9<sup>th</sup>* [\[3\]](#) and *10<sup>th</sup> Annual Conference on Vision and Intelligent Systems (CVIS)*. [Section 5.3](#) is intended for publication [\[4\]](#).
- [Chapter 7](#) was prepared for and will be presented at *Beyond Euclidean: Hyperbolic and Hyperspherical Learning for Computer Vision* [\[5\]](#), a workshop of the *2025 International Conference on Computer Vision (ICCV 2025)*.
- [Figure 2.1](#) was based upon work by Peter Halasz [\[6\]](#), [Figure 6.6](#) is the work of T. Ruen and R. Nelson [\[7\]](#), and [Figure 6.1](#) is the work of Sam Derbyshire [\[8\]](#). All of these graphics were made available through Creative Commons licences.

The following table summarizes the chapters in which significant components were drawn from publications and a brief description of the author’s contribution to each. In the citation column, the full author list is provided to demonstrate the highly collaborative nature of the BIOSCAN research and other projects.

| Chapter             | Citation                                                                                                                                                                                                                                                                                                                                                                                                                          | Author’s Contributions                                                                                                                                                                |
|---------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Chapters 1, 2 and 4 | [1] Gharaee, Z. <sup>†</sup> , Gong, Z. <sup>†</sup> , Pellegrino, N. <sup>†</sup> , Zarubiieva, I., Bruslund Haurum, J., Lowe, S.C., McKeown, J.T.A., Ho, C.C.Y., McLeod, J., Wei, Y.C., Agda, J., Ratnasingham, S., Steinke, D., Chang, A.X., Taylor, G.W., Fieguth, P. “A Step Towards Worldwide Biodiversity Assessment: The BIOSCAN-1M Insect Dataset”. <i>Advances in Neural Information Processing Systems</i> , 36, 2024. | Background and literature review, organization of the dataset, gathering of statistics of the dataset, and writing the manuscript.                                                    |
| Chapters 2 and 4    | [2] Gharaee, Z. <sup>†</sup> , Lowe, S.C. <sup>†</sup> , Gong, Z. <sup>†</sup> , Millan Arias, P. <sup>†</sup> , Pellegrino, N., Wang, A.T., Haurum, J.B., Eyriay, I., Kari, L., Steinke, D. and Taylor, G.W., Fieguth, P., Chang, A.X. “BIOSCAN-5M: A Multimodal Dataset for Insect Biodiversity”. <i>Advances in Neural Information Processing Systems</i> , 37, 2025.                                                          | Contributed to dataset curation and pre-processing, assisted conducting zero-shot transfer-learning clustering experiments, and co-authored the manuscript text and figures.          |
| Section 5.1         | [3] Pellegrino, N. <sup>†</sup> , Zhao, N. <sup>†</sup> , and P. Fieguth. “The Effects of Label Errors in Training Data on Model Performance and Overfitting”. <i>Journal of Computational Vision and Imaging Systems</i> , vol. 9, 2023.                                                                                                                                                                                         | Project leadership, concept ideation and formulation, initial implementation of method, design of experiments and interpretation of results, co-authorship and editing of manuscript. |
| Section 5.3         | [4] Pellegrino, N. <sup>†</sup> , Szczecina, D. <sup>†</sup> , and P. Fieguth. Manuscript to be submitted for publication, available on arXiv at <a href="https://arxiv.org/abs/2511.16512">arXiv:2511.16512</a> .                                                                                                                                                                                                                | Project leadership, concept ideation and formulation, design of experiments and interpretation of results, co-authorship and editing of manuscript.                                   |
| Chapter 7           | [5] Gong, Z., Tang, C., Huo, X., Pellegrino, N., Wang, A., Taylor, G.W., Chang, A.X., Lowe, S.C., Bruslund Haurum, J. “Hyperbolic Multimodal Representation Learning for Biological Taxonomies”. In <i>2nd Beyond Euclidean Workshop: Hyperbolic and Hyperspherical Learning for Computer Vision</i> . Available on arXiv at <a href="https://arxiv.org/abs/2508.16744">arXiv:2508.16744</a> .                                    | Concept ideation and formulation, assisted in initial implementation of method, co-authorship and editing of manuscript.                                                              |

<sup>†</sup> indicates equal contribution, joint first-authorship.

## Abstract

The BIOSCAN project, led by the International Barcode of Life (iBOL) Consortium, is an international, multi-year, and multidisciplinary effort seeking to catalogue all multicellular life on Earth by 2045 to enable the global-scale study of changes in biodiversity, species interactions, and species dynamics. Access to this information has the potential to inform strategies to mitigate the damaging ecological effects of climate change. In the near term, the goal is to catalogue all insects. Each sample is imaged, genetically barcoded, and taxonomically classified by domain experts, a time- and resource-intensive process that is becoming increasingly impractical as collection rates surpass five million samples annually. Addressing such needs is among the foundational motivations for the research of this thesis.

This thesis presents several contributions motivated by the challenges of the BIOSCAN project. Over five million insect samples were organized into a machine-learning-ready dataset, and a deep neural network classifier was developed to establish a baseline for image-to-taxonomy classification performance. To mitigate the harmful impacts of mislabelled samples in training data, a study of neural network architecture robustness was conducted alongside the development of two novel loss functions: Blurry and Piecewise-zero loss. Blurry loss de-weights and reverses the gradient of samples likely to be mislabelled, while Piecewise-zero loss disregards these samples. These improvements strengthen model robustness and enhance label error detection, enabling the referral of suspicious samples for expert review and correction. Additional work investigates the hierarchical structure of biological data and its integration into classification models, specifically through Hyperbolic neural networks, and measures the benefits of doing so in comparison to using conventional architectures. Finally, this thesis explores aligning image, genetic, and taxonomic representations in a hierarchy-aware manner to improve retrieval across modalities.

The contributions of this thesis advance the application of machine learning to facilitate the ongoing global-scale cataloguing of insect life. As challenges such as label errors, hierarchical structures in data, and incomplete annotations are present across many domains, the contributions are valuable to both the machine learning community and the global network of BIOSCAN collaborators.

## Acknowledgements

I would like to thank my supervisor, Professor Paul W. Fieguth, for his ongoing support and guidance throughout my PhD, but most of all, for giving me the freedom and encouragement to pursue many interests and opportunities available to me as a graduate student (lecturing, TA'ing, university governance, service to the GSA, mentorship of URAs, organization of the CVIS conference, auditing math courses, organizing lab intramural teams, and more!). I would also like to thank my committee members, Dr. Arash Arami, Dr. Linlin Xu, Dr. Jeff Orchard, and Dr. Steven L. Waslander, for reviewing my thesis and partaking in my defence.

I would like to thank all my friends and colleagues at the Vision and Image Processing (VIP) Lab for creating an amazing, constructive, and enjoyable environment that I had the pleasure of being part of for the past three and a half years. A long list of lab-mates that contributed to my experience includes Amir Nazemi, Chris (Krzysztof) Czarnecki, M. Hadi Sepanj, Nima Abbasi Firoozjah, Ken Mangouh Nsiempba, Eddie (Jinman) Park, Auguste L. W. Koh, Pranav Venkatesan, Soyeon Jang, Saed Moradi, Zahra Gharaee, Amy Tai, Emily (Zhixuan) Zeng, Chang Liu, Kiernan McGuigan, Muhammed Patel, Neil Brubacher, Jayden Hsiao, Fernando Jose Pena Cantu, Hayden Gunraj, Kimathi Kaai, Vasyl Chomko, Ksusha (Kseniia) Buzko, Adrian Ramlal, Daniel Mao, Frank (Fakher) Mokadem, and many more. I would also like to express my gratitude to Dr. Yuhao Chen for his efforts in organizing lab activities and always being available to chat about research directions and ideas.

My special thanks goes to several Undergraduate Research Assistants (URAs), David Szczecina, Nolen Zhao, David Wu, Marcus Chong, Isha Sharma, Utku Çiçek, Yousef Sahawneh, and Vikram Chandramohan, with whom I conducted much of my research.

I would like to thank Spencer Small and Engineering Counselling as a whole. You helped me throughout my degree and especially in times when I needed someone to listen with sensitivity, but also provide realistic and actionable advice. Additionally, I would like to thank Sinem Ozkardas of the Writing and Communication Centre for facilitating the

Grad Writing Café, a service that created a space in which I could focus on my writing and made consistent progress over an entire term to complete my thesis.

Further special thanks goes to my flatmate and close friend throughout graduate school, Saad Rasheed Abbasi. We've shared many late night discussions of our pains and passions, and you've always been there for me.

I would like to thank my family and loved ones for supporting and being with me throughout all of this, the ups and downs, and successes, and the times when things weren't going too well. Through all of this, you believed in me, and that helped.



This research was enabled in part by support provided by Calcul Québec ([calculquebec.ca](http://calculquebec.ca)) and the Digital Research Alliance of Canada ([alliancecan.ca](http://alliancecan.ca)).

We acknowledge the support of the Government of Canada's New Frontiers in Research Fund (NFRF), [NFRFT-2020-00073], and the support of the Natural Sciences and Engineering Research Council of Canada (NSERC) via NSERC-CGS D.

Nous remercions le Fonds Nouvelles Frontières en Recherche du gouvernement du Canada de son soutien (FNFR), [FNFRT-2020-00073], et le soutien du Conseil de Recherches en Sciences Naturelles et en Génie du Canada (CRSNG), CRSNG-BESC D.



Government  
of Canada    Gouvernement  
du Canada



Natural Sciences and Engineering  
Research Council of Canada

Conseil de recherches en sciences  
naturelles et en génie du Canada

**Canada**

# Table of Contents

|                            |      |
|----------------------------|------|
| Examining Committee        | ii   |
| Author's Declaration       | iii  |
| Statement of Contributions | iv   |
| Abstract                   | vi   |
| Acknowledgements           | vii  |
| List of Figures            | xiv  |
| List of Tables             | xvii |
| List of Symbols            | xix  |
| Graphic                    | xx   |
| Quote                      | xxi  |

|          |                                                                 |          |
|----------|-----------------------------------------------------------------|----------|
| <b>1</b> | <b>Introduction</b>                                             | <b>1</b> |
| 1.1      | Overarching Motivation: The BIOSCAN Project . . . . .           | 1        |
| 1.2      | Thesis Contributions . . . . .                                  | 3        |
| 1.3      | Thesis Layout . . . . .                                         | 4        |
| <b>2</b> | <b>Background</b>                                               | <b>5</b> |
| 2.1      | Motivation: BIOSCAN Project . . . . .                           | 5        |
| 2.1.1    | Taxonomic Classification of Organisms . . . . .                 | 6        |
| 2.1.2    | DNA Barcoding and Barcode Index Numbers . . . . .               | 6        |
| 2.2      | Related Work on Biological and Insect Datasets . . . . .        | 7        |
| 2.3      | Deep Learning and Convolutional Neural Networks . . . . .       | 9        |
| 2.3.1    | Computer Vision . . . . .                                       | 10       |
| 2.3.2    | Learning Paradigms . . . . .                                    | 11       |
| 2.3.3    | Classification and Regression . . . . .                         | 12       |
| 2.4      | Machine Learning Challenges . . . . .                           | 13       |
| 2.4.1    | Long-tailed Distributions (Class Imbalance) . . . . .           | 14       |
| 2.4.2    | Hierarchical & Fine-grained Classification . . . . .            | 16       |
| 2.4.3    | Other Challenges and Considerations . . . . .                   | 18       |
| 2.5      | Background on Label Errors in Training Data . . . . .           | 19       |
| 2.5.1    | Challenges of Label Errors in Training Data . . . . .           | 20       |
| 2.5.2    | Label Error Detection . . . . .                                 | 21       |
| 2.5.3    | Robust Loss Functions . . . . .                                 | 22       |
| 2.6      | Background on Hyperbolic Neural Networks . . . . .              | 24       |
| 2.6.1    | Related Multimodal and Hyperbolic Representation Work . . . . . | 25       |

|          |                                                                         |           |
|----------|-------------------------------------------------------------------------|-----------|
| <b>3</b> | <b>Formulation of the Insect Classification Problem</b>                 | <b>27</b> |
| <b>4</b> | <b>The BIOSCAN Dataset</b>                                              | <b>30</b> |
| 4.1      | Construction and Statistical Analysis of the BIOSCAN-1M Dataset . . . . | 31        |
| 4.1.1    | Subset sampling and split mechanism . . . . .                           | 32        |
| 4.1.2    | Image Cropping . . . . .                                                | 33        |
| 4.1.3    | Statistical Analysis . . . . .                                          | 34        |
| 4.2      | Classification Results on the BIOSCAN-1M Dataset . . . . .              | 36        |
| 4.3      | The BIOSCAN-5M Dataset . . . . .                                        | 41        |
| 4.3.1    | Dataset Details . . . . .                                               | 42        |
| 4.3.2    | Data Partitioning . . . . .                                             | 49        |
| 4.4      | Chapter Conclusion . . . . .                                            | 51        |
| <b>5</b> | <b>Improving Robustness to Label Errors</b>                             | <b>52</b> |
| 5.1      | Label Error Sensitivity . . . . .                                       | 53        |
| 5.1.1    | Preliminaries . . . . .                                                 | 53        |
| 5.1.2    | Introduction . . . . .                                                  | 55        |
| 5.1.3    | Framework for Evaluating Model Robustness . . . . .                     | 56        |
| 5.1.4    | Discussion . . . . .                                                    | 61        |
| 5.1.5    | Conclusions of Study on Label Error Sensitivity . . . . .               | 63        |
| 5.2      | Robust Model Architectures . . . . .                                    | 63        |
| 5.2.1    | Methods . . . . .                                                       | 64        |
| 5.2.2    | Results of Study on Robust Model Architectures . . . . .                | 66        |
| 5.2.3    | Conclusions of Study on Robust Model Architectures . . . . .            | 74        |

|          |                                                                             |            |
|----------|-----------------------------------------------------------------------------|------------|
| 5.3      | Robust Loss Functions . . . . .                                             | 75         |
| 5.3.1    | Introduction . . . . .                                                      | 75         |
| 5.3.2    | Method . . . . .                                                            | 76         |
| 5.3.3    | Experiments . . . . .                                                       | 79         |
| 5.3.4    | Results of Study of Robust Loss Functions . . . . .                         | 82         |
| 5.3.5    | Conclusion of Study of Robust Loss Functions . . . . .                      | 94         |
| 5.4      | Chapter Conclusion . . . . .                                                | 94         |
| <b>6</b> | <b>Improved Hierarchical Classification with Hyperbolic Neural Networks</b> | <b>96</b>  |
| 6.1      | Introduction & Problem Definition . . . . .                                 | 97         |
| 6.2      | Method . . . . .                                                            | 99         |
| 6.3      | Experiments . . . . .                                                       | 99         |
| 6.3.1    | Experiment Design . . . . .                                                 | 100        |
| 6.3.2    | Experiment Results . . . . .                                                | 104        |
| 6.4      | Chapter Conclusion . . . . .                                                | 110        |
| <b>7</b> | <b>Hierarchical Embeddings and Multi-modal Retrieval</b>                    | <b>112</b> |
| 7.1      | Introduction . . . . .                                                      | 113        |
| 7.2      | Approach . . . . .                                                          | 115        |
| 7.2.1    | Input and Output Specification . . . . .                                    | 115        |
| 7.2.2    | Encoders . . . . .                                                          | 116        |
| 7.2.3    | Model Details . . . . .                                                     | 116        |
| 7.2.4    | Stacked Entailment Loss . . . . .                                           | 118        |
| 7.3      | Experiments and Results . . . . .                                           | 121        |

|          |                                                                        |            |
|----------|------------------------------------------------------------------------|------------|
| 7.3.1    | Metrics and Datasets . . . . .                                         | 123        |
| 7.3.2    | Results . . . . .                                                      | 123        |
| 7.4      | Chapter Conclusion . . . . .                                           | 124        |
| <b>8</b> | <b>Conclusion and Future Work</b>                                      | <b>125</b> |
| 8.1      | Future Work . . . . .                                                  | 126        |
| 8.1.1    | Future Work Contending with Label Errors . . . . .                     | 127        |
| 8.1.2    | Future Work on the Study of Hyperbolic Neural Networks . . . . .       | 129        |
| 8.1.3    | Semi-Supervised Learning for Deeper Taxonomic Classification . . . . . | 129        |
| 8.1.4    | Insect Biomass Estimation . . . . .                                    | 130        |
|          | <b>References</b>                                                      | <b>132</b> |

# List of Figures

|     |                                                                                      |    |
|-----|--------------------------------------------------------------------------------------|----|
| 2.1 | The hierarchical structure used to classify life . . . . .                           | 6  |
| 2.2 | Graphical demonstration of the Lorenz curve and Gini coefficient . . . . .           | 15 |
| 3.1 | Diagram of basic image classification model training scheme . . . . .                | 28 |
| 4.1 | Example dataset record showing image and associated metadata . . . . .               | 31 |
| 4.2 | Class distribution for each taxonomic rank . . . . .                                 | 35 |
| 4.3 | Per-class Top-1 test accuracy for Order- and Family-level classification . . .       | 39 |
| 4.4 | Confusion matrices for order-level classification . . . . .                          | 40 |
| 4.5 | Overview of the multiple modalities in BIOSCAN-5M . . . . .                          | 43 |
| 4.6 | Samples of original full-size images in the BIOSCAN-5M dataset . . . . .             | 43 |
| 4.7 | Geographical locations where samples of BIOSCAN-5M were collected . . .              | 45 |
| 4.8 | Global distribution of BIOSCAN-5M sample collection . . . . .                        | 47 |
| 4.9 | Examples of original and masked images of BIOSCAN-5M . . . . .                       | 48 |
| 5.1 | Outliers <i>vs.</i> Label errors arise from different causes . . . . .               | 57 |
| 5.2 | Accuracy as a function of training data corruption rate . . . . .                    | 59 |
| 5.3 | Accuracy <i>vs.</i> training data corruption rate, for 10-, 6-, and 2-class problems | 60 |
| 5.4 | Training and testing loss as a function of corruption rate . . . . .                 | 61 |

|      |                                                                                                                                               |     |
|------|-----------------------------------------------------------------------------------------------------------------------------------------------|-----|
| 5.5  | Accuracy <i>vs.</i> Corruption rate on 10-class datasets, evaluated across architecture types . . . . .                                       | 68  |
| 5.6  | Accuracy <i>vs.</i> Corruption rate on CIFAR-100 (100-class) dataset, 12 epochs, evaluated across architecture types . . . . .                | 69  |
| 5.7  | Accuracy <i>vs.</i> Corruption rate evaluated across datasets, for various architecture types . . . . .                                       | 71  |
| 5.8  | Accuracy <i>vs.</i> Corruption rate on MNIST dataset, evaluated for KNN classifier for various $K$ . . . . .                                  | 72  |
| 5.9  | Accuracy <i>vs.</i> $K$ for KNN classifier on MNIST dataset, evaluated for various corruption rates . . . . .                                 | 73  |
| 5.10 | KNN results on the Fashion-MNIST dataset . . . . .                                                                                            | 74  |
| 5.11 | The two proposed loss functions: Blurry Loss (a) and Piecewise-zero Loss (b). . . . .                                                         | 78  |
| 5.12 | Detection performance of BL and PZ Loss as functions of their parameters . . . . .                                                            | 84  |
| 5.13 | Gradient distributions per epoch . . . . .                                                                                                    | 87  |
| 5.14 | Predicted probability distributions per epoch . . . . .                                                                                       | 88  |
| 5.15 | Heatmap of F1 score as function of loss function parameters . . . . .                                                                         | 90  |
| 5.16 | Detection F1 score <i>vs.</i> Delay, $d$ , for PZ loss on CIFAR-100 . . . . .                                                                 | 92  |
| 5.17 | AUM distributions for artificially corrupted and clean samples in CIFAR-100 with corruption at $\eta = 0.1$ for CE, BL, and PZ loss . . . . . | 93  |
| 6.1  | Cantor's Middle Third Set, with the implicit hierarchy made explicit . . . .                                                                  | 100 |
| 6.2  | Repeated bisection of the real line gives rise to hierarchical structure . . . .                                                              | 102 |
| 6.3  | The layer structure for both ENNs and HNNs . . . . .                                                                                          | 103 |
| 6.4  | Accuracy for 1D, 2D, and 3D experiments for ENNs and HNNs . . . . .                                                                           | 105 |
| 6.5  | Embeddings of ENN <i>vs.</i> HNN . . . . .                                                                                                    | 106 |

|     |                                                                             |     |
|-----|-----------------------------------------------------------------------------|-----|
| 6.6 | Triangular tiling of the Poincaré disk . . . . .                            | 107 |
| 6.7 | Embeddings for ENNs with dead ReLUs . . . . .                               | 108 |
| 6.8 | MNIST Embeddings of ENN <i>vs.</i> HNN . . . . .                            | 109 |
| 7.1 | Diagram of Contrastive loss, Entailment loss, and Stacked entailment loss . | 114 |
| 8.1 | Measures of model robustness . . . . .                                      | 128 |
| 8.2 | Distribution in insect masses . . . . .                                     | 131 |

# List of Tables

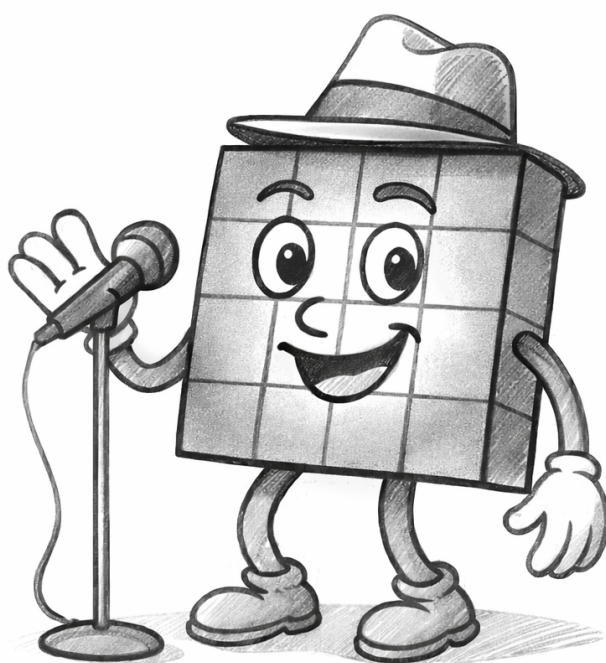
|     |                                                                                                                                               |     |
|-----|-----------------------------------------------------------------------------------------------------------------------------------------------|-----|
| 2.1 | Summary of fine-grained and long-tailed biological datasets . . . . .                                                                         | 8   |
| 4.1 | Summary of the number of samples within sub-datasets and partitions . .                                                                       | 33  |
| 4.2 | Comparison of the BIOSCAN-1M dataset to existing biological datasets . .                                                                      | 36  |
| 4.3 | Detailed hyperparameter settings of the BIOSCAN-1M experiments . . . .                                                                        | 37  |
| 4.4 | Micro- and Macro-averaged Top-K accuracy test scores . . . . .                                                                                | 38  |
| 4.5 | Summary statistics of dataset records by taxonomic rank . . . . .                                                                             | 44  |
| 4.6 | Statistics for geographical locations of sample collection . . . . .                                                                          | 46  |
| 4.7 | Statistics and purpose of BIOSCAN-5M data partitions. . . . .                                                                                 | 49  |
| 5.1 | Hyperparameters for label error sensitivity experiments . . . . .                                                                             | 56  |
| 5.2 | Label error detection performance (F1) compared across loss functions . .                                                                     | 83  |
| 5.3 | Label error detection performance (BA) compared across loss functions . .                                                                     | 85  |
| 5.4 | Label error detection performance (F1) on the CIFAR-100 cleaned test set                                                                      | 85  |
| 5.5 | Label error detections on the CIFAR-10N / -100N datasets with non-uniform<br>corruption . . . . .                                             | 88  |
| 5.6 | Label error detection using AUM framework and CIFAR-100 dataset . . .                                                                         | 89  |
| 6.1 | Rate of occurrence of Dead ReLUs by Dimensionality, $n$ , Depth of Hierarchy,<br>$d$ , and Model Type for Synthetic data experiments. . . . . | 109 |

|     |                                                                                                                                |     |
|-----|--------------------------------------------------------------------------------------------------------------------------------|-----|
| 6.2 | Rate of Occurrence of Dead ReLUs, Median, and Maximal Test Accuracy reported by Model Type for MNIST data experiments. . . . . | 110 |
| 7.1 | Example of taxonomic labels and their full-text concatenation. . . . .                                                         | 118 |
| 7.2 | Macro top-1 accuracy (%) comparison of different training objectives across taxonomic levels on BIOSCAN-1M . . . . .           | 121 |
| 7.3 | Micro top-1 accuracy (%) comparison of different training objectives across taxonomic levels on BIOSCAN-1M . . . . .           | 122 |

# List of Symbols

|                                           |                                                   |
|-------------------------------------------|---------------------------------------------------|
| $\mathcal{X}$                             | Feature Space (normally images)                   |
| $\mathcal{Y}$                             | Label Space (categorical for classification)      |
| $x \in \mathcal{X}$                       | Feature (normally image)                          |
| $y \in \mathcal{Y}$                       | Label (correct)                                   |
| $D$                                       | Dataset                                           |
| $N$                                       | Size of dataset                                   |
| $K$                                       | Number of classes in dataset                      |
| $k \in \{1, \dots, K\}$                   | Class label                                       |
| $M$                                       | Batch size                                        |
| $f : \mathcal{X} \rightarrow \mathcal{Y}$ | Classification function                           |
| $\theta$                                  | Model parameters                                  |
| $\mathcal{L}$                             | Loss function                                     |
| $R_{\mathcal{L}}$                         | Empirical risk                                    |
| $p(k x)$                                  | Probability (predicted) of $x$ being of class $k$ |
| $y^*$                                     | Predicted label                                   |
| $\hat{y}$                                 | Corrupted label                                   |
| $\tilde{y}$                               | Possibly corrupt, as-labelled class               |
| $\eta \in [0, 1]$                         | Corruption rate                                   |
| $\eta'$                                   | Cut-off point                                     |
| $\eta_{y\hat{y}}$                         | Probability of corruption from $y$ to $\hat{y}$   |
| $D_{\eta}$                                | Corrupted dataset                                 |

## Graphic



*"No need to ask, he's a smooth operator"*

This caricature of a Gaussian Smoothing Kernel was inspired by Karaoke Nights at the Vision and Image Processing (VIP) Lab, my most enjoyed extracurricular lab activity, in which a favourite song to sing was "*Smooth Operator*" by Sade.

## Quote

Sometimes, the only way *past* your obstacle is *through* it.

— Spencer Small

# Chapter 1

## Introduction<sup>1</sup>

### 1.1 Overarching Motivation: The BIOSCAN Project

Global change is restructuring ecosystems on a planetary scale, creating an increasingly urgent need to track impacts on biodiversity. Such tracking is exceptionally challenging because life is highly diverse: the biosphere comprises more than 10 million multicellular species [9]. Until recently, this complexity has meant that an Earth observation system for biodiversity was inconceivable, however the increased power of DNA sequencing and the recognition that living organisms can be discriminated by short stretches of DNA have revealed a way forward, which has become the central focus of the International Barcode of Life (iBOL) Consortium.

Discriminating organisms by DNA sequences [10, 11] can revolutionize our understanding of biodiversity, not only by providing a reliable species proxy for known and unknown species, but also by revealing their interactions and assessing their responses to changes in the ecosystem. This is essential to mitigate a looming mass extinction, where an *eighth*

---

<sup>1</sup>Note that significant portions of this chapter, particularly those pertaining to the BIOSCAN project, were originally written for a submission to the NeurIPS 2023 conference [1]. See the [Statement of Contributions](#) for details.

of *all species* may become extinct by 2100 unless there is a significant change in human behaviour [12, 13].

The BIOSCAN project [14], lead by iBOL, has the following three main goals:

1. species discovery,
2. studying the interactions between species, and
3. tracking and modelling species dynamics over geography and time.

To that end, BIOSCAN collects samples of multicellular life from around the world. Each sample is individually imaged, has its mitochondrial DNA sequenced and barcoded [10], and then is classified by expert taxonomists. Of particular interest to the BIOSCAN project are *insects*, which constitute a great proportion of the Earth’s species and many of which remain unknown. Indeed, it is estimated that 5.5 M insect species exist worldwide, of which only roughly one million have been identified [15, 16]. The rate of insect collection is increasing as the BIOSCAN project progresses, such that at the time of starting this PhD in 2022, only roughly 1.5 M insect specimens had been collected, whereas now in 2025, that number has grown to beyond 10 M.

Using high-resolution photographs, human taxonomists can accurately classify insects from within their domain of expertise. However, human annotation cannot scale to the volume of samples needed to measure and track global biodiversity. Moreover, many taxonomists with highly specialized knowledge will leave the practice over time and will likely not be replaced. Thus, the use of artificial intelligence and machine learning to process visual information collected by the BIOSCAN project is crucial to the success of a planet-scale observation system. Classification of the insect images to their taxonomic labels is especially useful in regions of the world where the facilities required to perform DNA barcoding are not available. Indeed, even beyond this project, there are opportunities for computer vision to transform entomology [17].

## 1.2 Thesis Contributions

The overarching research goal motivating this thesis is to develop and make use of machine vision and machine learning technology to assist in classifying insect images to their taxonomic labels. This goal was approached through several steps and independent research directions. The first was the curation and restructuring of a large-scale insect dataset for use in machine learning. Then, to establish baseline classification performance for this dataset, and to better understand the specific challenges associated with it, a baseline classifier was developed.

Through creating this dataset and baseline classifier, it was learned that the real-world, biological insect data contains many label errors which negatively impact the training and predictive performance of machine learning classifiers trained on this data. Thus, research towards improving robustness to label errors within training data was pursued, leading to investigations of model robustness to label errors (in terms of architectures), and the development of robust loss functions, which can be used synergistically to both train models in the presence of label errors and to identify data likely to be mislabelled. Next, to better model the inherent hierarchies, such as those present in the biological data (related to insect taxonomy), the use of hyperbolic neural networks was explored, and in particular, a comparison was made between Hyperbolic and standard Euclidean models. Finally, hyperbolic neural networks were applied to the BIOSCAN data for a multimodal retrieval task involving taxonomic labels (text), insect images, and DNA barcode sequences.

Thus, the main contributions of this thesis are the:

1. Creation of two large-scale insect image datasets (BIOSCAN-1M and BIOSCAN-5M) and implementation of a baseline classifier, in [Chapter 4](#),
2. Evaluation of robust model architectures and the development of robust loss functions for use in label error detection, in [Chapter 5](#),
3. In the study of Hyperbolic neural networks, a dying ReLU problem was found to be mitigated by the use of Hyperbolic neural networks, and across all experiments, the Hyperbolic neural networks consistently matched or outperformed their Euclidean counterpart on classification tasks involving hierarchical data, in [Chapter 6](#), and

4. Hyperbolic neural networks were found to effectively capture hierarchical structures in the BIOSCAN data, in [Chapter 7](#).

## 1.3 Thesis Layout

[Chapter 2](#) provides additional motivation for the research in terms of the goals of the BIOSCAN project, but mainly gives relevant background on the fields of computer vision, and in particular, image classification, and then taxonomic classification. Further background is given on addressing label errors in training data, and hyperbolic neural networks. Then, [Chapter 3](#) frames the overall insect classification problem in a more technical manner, defines notation and formulates the problem mathematically in terms of this notation.

[Chapter 4](#) describes the development of the insect dataset, initially with 1 M samples, and then with 5 M samples, termed the BIOSCAN-1M and BIOSCAN-5M datasets, respectively. Experimental results from the baseline classifier are also provided.

[Chapter 5](#) is about addressing label errors in training data. This is explored through both the comparative evaluation of model architectures and the development of robust loss functions, suitable for use with downstream label error detection frameworks.

[Chapter 6](#) explores the use of hyperbolic neural networks to better learn from and embed hierarchical data, and compares classification performance between hyperbolic models and standard Euclidean models.

[Chapter 7](#) applies hyperbolic neural networks to the BIOSCAN data for a multimodal retrieval task involving taxonomic labels (text), insect images, and DNA barcode sequences.

Finally, this thesis is concluded by [Chapter 8](#), which summarizes the research contributions and provides insights towards future work.

# Chapter 2

## Background<sup>1</sup>

In this chapter, motivation for the research goal of using machine vision to taxonomically classify insects is given, based on needs and importance of the BIOSCAN project. Information needed to understand taxonomic classification in biology as well as the use of DNA barcoding is provided. Additional background related to machine learning, convolutional neural networks, and image classification is given, followed by a brief exploration of several pertinent machine learning challenges associated with the research in this thesis. Further background is given on the subject of label errors in training data, including label error detection and robust loss functions. Finally, background on Hyperbolic neural networks is given.

### 2.1 Motivation: BIOSCAN Project

The International Barcode of Life Consortium (iBOL), through their BIOSCAN project [14], has the broad goal of cataloguing all multicellular life on Earth in order to study species dynamics and interactions, and ultimately to provide valuable guidance and insights for

---

<sup>1</sup>Note that significant portions of this chapter, particularly [Sections 2.1](#) and [2.2](#), were originally written for submissions to the NeurIPS 2023 conference [1] and NeurIPS 2024 conference [2]. See the [Statement of Contributions](#) for details.

biodiversity-preserving and climate change mitigation strategies. One major component of the project focuses on insects, one of the most abundant and diverse classes of animal life [16]. The diversity of insects them a highly valuable class upon which for iBOL to focus much of their research efforts.

### 2.1.1 Taxonomic Classification of Organisms

In biology, taxonomic classification is the study of the hierarchical categorization of lifeforms based on shared characteristics. In particular, Linnean taxonomy [18–21] forms the basis for the modern (generally accepted) system of taxonomy, of which the main hierarchical ranks are Domain, Kingdom, Phylum, Class, Order, Family, Genus, and Species, as illustrated in Figure 2.1, where the taxonomic ranks are listed. All insect life is part of the class *Insecta*.

Conventionally, expert taxonomists classify organisms based on their appearance and behaviour [19]. However, this approach is susceptible to misclassification and lacks consensus throughout the community of taxonomists, since it is difficult to prove with certainty that a given classification is absolutely *correct*. This shortcoming of traditional taxonomy has prompted the use of classification heuristics, based on fairly concrete evidence in the form of genetic codes, that are sensitive to species identity.

### 2.1.2 DNA Barcoding and Barcode Index Numbers

DNA barcoding [10, 11] employs large-scale screening of one or a few reference genes for assigning unknown individuals to a species label, as well as increasing the discovery of new species [22]. Barcoding is commonly used in several fields including taxonomy, ecology, conservation biology, diet analysis and food safety [23, 24]. It is faster and more accurate than traditional methods, which rely on the judgment of experts [25].

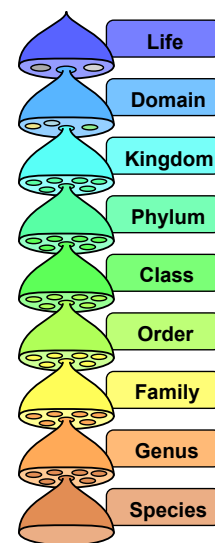


Figure 2.1: The hierarchical structure used to classify life. Image remixed based on original by P. Halasz [6] (CC0).

Barcoding is based on the use of a short, standardized segment of mitochondrial DNA, typically a portion of the *mitochondrial cytochrome c oxidase subunit I (COI) gene* [10, 22], which is nearly always unique for different species. Once the DNA sequence is obtained, it can be compared to a reference library of known sequences to identify the species. DNA barcodes have been successfully applied to taxonomic identification and classification, ecology, conservation, diet analysis, and food safety [23, 24], offering faster and more accurate results than traditional methods [25].

The concept of DNA barcoding can be taken a step further by mapping barcodes to clusters of organisms (characterized by their barcodes) with *highly* similar genetic code, known as operational taxonomic units (OTU) [26, 27]. OTUs act as a proxy for species based on the high degree of genetic similarity exhibited by their members. To enable indexing, each OTU is assigned a uniform resource identifier (URI), commonly referred to as the Barcode Index Number (BIN) [28], which offers a unique representation such that genetically identical taxa will be assigned the same BIN, and each BIN is registered in the Barcode Of Life Data system (BOLD) [29], a platform created to store and analyze data using four modules: (1) a data portal, (2) an educational portal, (3) a registry of BINs (putative species), and (4) a data collection and analysis workbench. BINs additionally provide an alternative to the use of Linnean names, offering a DNA-based classification for organisms.

## 2.2 Related Work on Biological and Insect Datasets

There are a wide variety of related large-scale biological datasets that can act as a reference on which to compare the BIOSCAN-1M/5M datasets developed for this research, in Chapter 4. Biological datasets are essential for advancing our understanding of the natural world, with uses in genomics [30], proteomics [31], ecology [32], evolutionary biology [33], medicine [34], and agriculture [35–38]. Table 2.1 compares biological datasets used for taxonomic classification. Many of these datasets feature fine-grained classes and exhibit a long-tailed class distribution, making the recognition task challenging for machine learning

Table 2.1: **Summary of fine-grained and long-tailed biological datasets.** The “Taxa” column indicates the taxonomic scope of each dataset. The “IR” column is the class imbalance ratio, computed as the ratio of the number of samples in the largest category to the smallest category.

| Dataset            | Reference | Year | Images  | IR     | Taxa       | Rank    | Categories | Taxon | BIN | DNA | Geography | Size |
|--------------------|-----------|------|---------|--------|------------|---------|------------|-------|-----|-----|-----------|------|
| LeafSnap           | [42]      | 2012 | 31 k    | 8      | Plants     | Species | 184        | ×     | ×   | ×   | ×         | ×    |
| NA Birds           | [43]      | 2015 | 48 k    | 15     | Birds      | Species | 400        | ×     | ×   | ×   | ×         | ×    |
| Urban Trees        | [44]      | 2016 | 80 k    | 7      | Trees      | Species | 18         | ×     | ×   | ×   | ×         | ×    |
| DeepWeeds          | [45]      | 2019 | 17 k    | 9      | Plants     | Species | 9          | ×     | ×   | ×   | ✓         | ×    |
| IP102              | [41]      | 2019 | 75 k    | 14     | Insects    | Species | 102        | ✓     | ×   | ×   | ×         | ×    |
| Pest24             | [46]      | 2020 | 25 k    | 494    | Insects    | Species | 24         | ×     | ×   | ×   | ×         | ×    |
| Pl@ntNet-300K      | [47]      | 2021 | 306 k   | 3,604  | Plants     | Species | 1,000      | ×     | ×   | ×   | ×         | ×    |
| iNaturalist (2021) | [48]      | 2021 | 2,686 k | 2      | All        | Species | 10,000     | ✓     | ×   | ×   | ×         | ×    |
| iNaturalist-Insect | [48]      | 2021 | 663 k   | 2      | Insects    | Species | 2,526      | ✓     | ×   | ×   | ×         | ×    |
| Species196-L       | [38]      | 2023 | 19 k    | 351    | Various    | Mixed   | 196        | ✓     | ×   | ×   | ×         | ×    |
| CWD30              | [39]      | 2023 | 219 k   | 61     | Plants     | Species | 30         | ✓     | ×   | ×   | ×         | ×    |
| BenthicNet         | [49]      | 2024 | 1,429 k | 22,394 | Aquatic    | Mixed   | 791        | ✓     | ×   | ×   | ✓         | ×    |
| Insect-1M          | [50]      | 2024 | 1,017 k | N/A    | Arthropods | Species | 34,212     | ✓     | ×   | ×   | ×         | ×    |
| BIOSCAN-1M         | [1]       | 2023 | 1,128 k | 12,491 | Insects    | BIN*    | 90,918     | ✓     | ✓   | ✓   | ×         | ×    |
| BIOSCAN-5M         | [2]       | 2024 | 5,150 k | 35,458 | Arthropods | BIN*    | 324,411    | ✓     | ✓   | ✓   | ✓         | ✓    |

\* For datasets that include Barcode Index Numbers (BINs) annotations, BINs are presented, which serve as a (sub)species proxy for organisms and offer a viable alternative to Linnean taxonomy.

(ML) methods that do not account for these properties. While many datasets provide images, they do not include other attributes such as DNA barcode, or geographical locations. In addition, labels for multi-level taxonomic ranks can offer valuable insights into the evolutionary relationships between organisms; however, datasets with hierarchical taxonomic annotations [1, 38–41] are relatively scarce.

Perhaps the best known and largest biological dataset is iNaturalist [48], containing 2.6 M images from over 10,000 different species of plants and animals, and containing 2,526 categories of insects with ~663 k annotated images. Many insect-specific image datasets focus on insect as pests found in agricultural settings [41, 46, 51–56]; the most prominent of which, the IP102 [41] dataset, contains roughly 75 k insect images, 19 k of which are annotated by agricultural experts, with over 102 classes of insects. In the space of plants, the Pl@ntNet-300K [47] dataset has 306 k images and was constructed by sampling the larger Pl@ntNet database [57]. Most relevant to this thesis work are the BIOSCAN-1M

and BIOSCAN-5M Insect Datasets, developed in [Chapter 4](#). These datasets constitute 1 M and 5 M, respectively, insect images paired with DNA barcodes and taxonomic labels.

Image-based insect classification [\[58\]](#) most often finds use in agricultural settings, where Integrated Pest Management (IPM) systems are used to identify and count harmful insect pests [\[59, 60\]](#) and often additionally identify plant diseases through computer vision [\[61–63\]](#).

Recently, DNA sequences have been analyzed [\[64\]](#) using tools from the field of Natural Language Processing [\[65\]](#), and in particular, through the application of bidirectional encoder representations from transformers (BERT) [\[66\]](#). Indeed, BERT-based models have been used to taxonomically classify DNA sequences [\[67, 68\]](#). Other recent work has used DNA barcodes as “side information” to perform zero-shot species-level recognition from images, albeit at a much smaller scale than BIOSCAN-1M [\[69\]](#).

## 2.3 Deep Learning and Convolutional Neural Networks

It is assumed that the reader already has a reasonable degree of familiarity with deep learning and artificial neural networks (for which [\[70, 71\]](#) are excellent resources). Therefore, the main goals of this section are to provide a brief overview of deep learning with convolutional neural networks while also providing notation to be used in this thesis.

At the forefront of much of modern machine learning is the use of deep artificial neural networks [\[70–72\]](#). These network models approximate some unknown underlying function that best solves a given inverse problem. In particular, when working with image data, for example in image classification and object detection, discussed below in [Section 2.3.1](#), the use of *convolutional* neural networks (CNNs) [\[73–75\]](#) has become widespread. The key difference between CNNs and pre-existing deep neural networks (DNNs), for example those with fully-connected layers, is the use of *convolutional* layers, whereby sets of learned filters are convolved with the layer input, thereby dramatically reducing the number of network parameters and effectively regularizing the model.

For an image input,  $x \in \mathbb{R}^{W \times H \times 3}$ , with width,  $W$ , height  $H$ , and three colour channels (RGB), the network model,  $f$ , maps to predicted probabilities,  $p(k|x)$ , from which the (possibly vectoral, i.e., having more than a single value) output is selected as the most probable class,  $y^* = f(x) = \arg_k \max p(k|x)$ , approximating the ground truth solution,  $y$  to the inverse problem. Network models are parameterized by a set of weights,  $\theta$ , such that

$$y^* = f(x) = \arg_k \max p(k|x_i; \theta). \quad (2.1)$$

Finding an optimal selection of parameters, i.e., connection weights and biases, is performed through training of the network, in general by minimizing some loss function,  $\mathcal{L}$ , penalizing differences between predicted probabilities,  $p(k|x)$ , and ground truth,  $y$ , across a set of training data. For a set of  $N$  training examples, with ground truth input-output pairs,  $\{(x_i, y_i)\}_{i=1}^N$ , optimal weights are found through minimizing

$$\theta^* = \arg \min_{\theta} \frac{1}{N} \sum_{i=1}^N \mathcal{L}(p(k|x_i; \theta), y_i). \quad (2.2)$$

In practice, Stochastic Gradient Descent (SGD) [76] and derivative methods such as Adam [77] are most commonly used to optimize parameters during training [72, 76]. Typical loss functions depend on the application, the most common of which are Mean-squared-error (MSE) [72] (for regression) and Cross-entropy loss [78, 79] (for classification).

### 2.3.1 Computer Vision

The field of computer vision is highly active and extreme broad. Familiarizing oneself would at least require reading several textbooks, such as [80, 81]. However, here, as background for the work presented in this thesis, succinct discussion is given for only two sub-fields: image classification and object detection.

## Image Classification

Image classification is the computer vision task of assigning labels or categories to images based on their content. Classification in general was introduced in [Section 2.3.3](#). Following that discussion, image classification is simply the case where images are used as input. In general, for each image, a single label is predicted by the model. Several popular models, now considered standard, include AlexNet [\[82\]](#), VGGNet [\[83\]](#), InceptionNet (GoogLeNet) [\[84\]](#), and ResNet [\[85\]](#). Image classification is used in the research of this thesis for taxonomically classifying insect images.

## Object Detection

Object detection [\[86–88\]](#) can be thought of an extension of image classification; however, object detection models may detect *several* objects in each picture, and will predict the probabilities of each object’s class and location within the image (most often, a bounding box). Several popular object detection models include Region-based Convolutional Neural Network (R-CNN) [\[89\]](#) and its derivatives (Fast R-CNN [\[90\]](#), and Faster R-CNN [\[91\]](#)), YOLO (You only look once) [\[92\]](#) and its loose derivatives (YOLO9000 / YOLOv2 [\[93\]](#), YOLOv3 [\[94\]](#), YOLOv4 [\[95\]](#), YOLOv6 [\[96\]](#), YOLOv7 [\[97\]](#), YOLOv8 [\[98\]](#)), and RetinaNet [\[99\]](#). In the research of this thesis (specifically [Chapter 4](#)), object detection is used in the pre-processing step of image cropping, whereby the insect is detected and located within the image, providing the necessary information for cropping.

### 2.3.2 Learning Paradigms

The above introduction on CNNs has assumed that ground truth (i.e., labelled) data are available; however, this is certainly not always the case in all applications. In cases where ground truth data are available, *supervised* learning [\[100, 101\]](#) is used, whereby predictions are compared to ground truth in order to optimize model parameters, as was shown in [Equation \(2.2\)](#). When ground truth data are *not* available, *unsupervised* learning [\[102, 103\]](#) must be used, whereby the model attempts to uncover inherent structures of the data

or organize the data into clusters. Examples of such tasks include clustering [104] and dimensionality reduction [105, 106]. Cases also exist where only part of the training data has ground truth. In this case, *semi-supervised* learning [107–110] may be used. Two main types of unsupervised learning include inductive learning [111, 112], whereby a model is trained to optimally provide inference on new data, and transductive learning [113, 114], whereby the outcome is predicted labels for the previously unlabelled portion of the training data [110], similar to clustering methods of unsupervised methods. A special case of unsupervised learning is known as *self-supervised* learning [115–120] whereby the model is trained on a specially-designed task using only the data itself to generate supervisory signals (leveraging relationships in the data, pairing similar or dissimilar data) through which to learn representations [120]. The learned representations can be fine-tuned for downstream tasks like classification, detection, or segmentation, often achieving performance close to fully supervised methods while drastically reducing the need for labelled data. Self-supervised learning is used in Chapter 7 to train a multi-model retrieval [121] model whereby images, text, or DNA can be used to query for related data points in the other modalities.

### 2.3.3 Classification and Regression

Tasks approached using machine learning tend to fall into two main categories: classification and regression. Concise details of each are provided here along with the differences in network architecture (mainly, treatment at the final layers).

#### Classification

Classification is the task of categorizing input data,  $x \in \mathcal{X}$ , to a set of predefined classes,  $\mathcal{Y} = \{1, \dots, K\}$  for a  $K$ -class classification problem [70, 71, 79]. In this case, the labels,  $y$ , and predictions,  $y^*$ , are *categorical*. Architecturally, this is often achieved in CNNs by pooling feature outputs from the convolutional layers, passing through fully-connected layers, the last of which must have an output size matching the number of predefined classes,

and finally, normalization is applied via a SoftMax [122] layer (or alternatives [123]) to produce normalized scores, representative of the predicted probability,  $p(k|x)$ , for each class,  $k$ , and where values sum to one, i.e.,  $\sum_{k=1}^K p(k|x) = 1$ , and each value lies between zero and one, i.e.,  $p(k|x) \in [0, 1]$ . Cross-entropy loss [78, 79] and its derivatives are most commonly used for training classifiers. Classification is used throughout this thesis: taxonomic classification of insect images in Section 4.2, studying label errors and model robustness in Chapter 5, and for comparing the ability for models to learn from hierarchical data in Chapters 6 and 7.

## Regression

Regression is the task of predicting continuous numerical values, i.e., relating one or more independent explanatory variables (inputs),  $x \in \mathcal{X}$ , to a numerical dependent variable (output),  $\mathcal{Y} = \mathbb{R}^r$ , for an  $r$ -dimensional regression problem [70, 71, 79]. Note that in this case, the labels,  $y$ , and estimates (predictions),  $y^*$ , are continuous-valued numbers. Architecturally, regression is achieved in CNNs by passing extracted features from convolutional layers through a set of fully-connected layers, here, with the last fully-connected layer having an output size matching the dimension of the model output. In general, the final fully-connected layer may be used as the model output; however, in cases where target values are restricted to a certain range, activation functions may be used to re-scale the output, so as to bound it. Loss functions including MSE and mean absolute error (MAE) [72] are popular for regression tasks. In the research of this thesis, regression is used to estimate bounding boxes for use in cropping insect images, touched upon in Chapter 4.

## 2.4 Machine Learning Challenges

As will be further explored in Section 4.1, the BIOSCAN insect dataset of Chapter 4 exhibits several characteristics corresponding to open problems in the field of machine learning: Class imbalance, Fine-grainedness, and a Hierarchical label structure. Details of these challenges and existing work is outlined below.

### 2.4.1 Long-tailed Distributions (Class Imbalance)

The degree to which the expected quantity of instances varies between classes is known as the class imbalance. In the context of a closed dataset (one in which all classes are known), the broad notion of class imbalance describes the disparity in size between classes [124, 125]. As will be described in Section 4.1 (and in particular through the class distribution for each taxonomic rank, shown in Figure 4.2), the insect image dataset exhibits a long-tailed class distribution whereby the sizes of classes closely follow a power-law, meaning that there is a substantial class imbalance. This represents a challenge due to the disproportionate amounts of available training data for majority *vs.* minority classes.

There exist several metrics of class imbalance [126], outlined here.

- **The Imbalance Ratio** [127], IR, is defined as the ratio of the number of samples in the largest to the smallest class. Although imbalance is widely used [127, 128], it is highly subject to bias due to its dependence on only the two most extreme classes, and therefore does not reflect the overall characteristics of the dataset.
- **Standard deviation**,  $\sigma$ , of sample abundance [126, 129] between classes is an alternative metric, but is not specifically sensitive to long-tailedness.
- **The mean/median ratio** of class sample abundances [126],  $\gamma$ . Motivation for this metric comes from the fact that median is relatively insensitive to outliers in comparison to mean, such that their ratio is indicative of the skew in the distribution.  $\gamma$  close to 1 indicates a balanced dataset, whereas larger values indicate imbalance.
- **The Gini coefficient** [130], commonly used in economics, is an indicator of distribution inequality. It is based on the Lorenz curve [131], a probability plot (P-P plot), whereby the cumulative portion of the total variable of interest (wealth or income normally, but in this case, dataset samples) is plotted against the cumulative portion of the population (in this case, classes). This line is curved for imbalanced distributions. The same process can be performed for a perfectly balanced distribution, producing a straight line (i.e., cumulative dataset samples increase exactly in proportion with the cumulative number of classes), denoted the Line of Equality. Finally, Figure 2.2 is used as a demonstrative aid whereby the distribution under

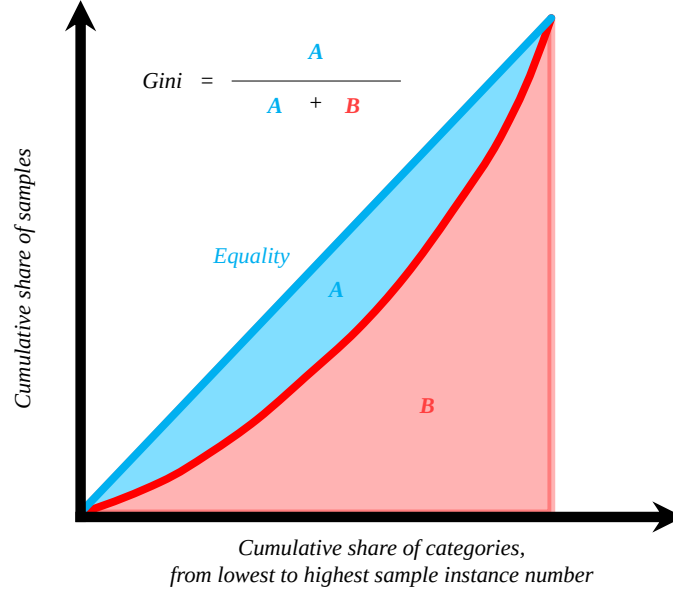


Figure 2.2: Graphical demonstration of the Lorenz curve and Gini coefficient. The **Lorenz curve** is shown in red, whereas the **Line of Equality** is shown in blue. The area between the **Line of Equality** and the **Lorenz curve** is denoted  $A$ , and the area beneath the **Lorenz curve** is denoted  $B$ .

analysis is characterized by the red **Lorenz curve**, the area between the blue **Line of Equality** and the **Lorenz curve** is denoted  $A$ , and the area beneath the **Lorenz curve** is denoted  $B$ . The Gini coefficient is computed as

$$\text{Gini} = \frac{A}{A + B}.$$

A Gini coefficient of 0 indicates perfect equality (i.e., balanced distribution), whereas coefficients tending towards 1 indicate inequality (i.e., imbalanced distributions).

These metrics will be used in [Section 4.1](#) to evaluate the imbalance of the insect dataset, and for comparison to other existing datasets.

## 2.4.2 Hierarchical & Fine-grained Classification

Classification problems involving data with labels arising from an underlying tree-structured hierarchy such that the similarity between categories is governed by their proximity within that hierarchy, present a unique challenge in comparison to simpler “flat” classification problems [132–134]. The outputs of hierarchical classification algorithms are defined over tree-like class taxonomies, where the relationship between parent and child nodes is given by the asymmetric “is-a” relationship. A basic example of this is the relationship that “all dogs are canines, but not all canines are dogs”, whereby “dogs” would be a child node of the parent node “canines”, which itself may be a child of “mammals”. The task of categorizing biological organisms to the Linnean taxonomy, discussed above in Section 2.1.1, is an example of a hierarchical classification problem.

The task of classifying objects or images to subordinate (leaf-node level) categories is known as Fine-grained classification [135–139]. Thus fine-grained problems can be seen as a special case of hierarchical classification focused on the lowest-level classes. Fine-grained classification is specifically challenging due to the small inter-class variation inherent to the hierarchical categorization structure.

Hierarchical classification [132–134] can broadly be categorized into two main types of approaches: *Local* and *Global*.

- **Local Approaches:**

Local approaches are characterized by the use of many local classifiers, responsible for various components of the hierarchy [132–134]. Local approaches can further be broken down into *local (binary) classification per node (LCN)* [140], *local classification per level (LCL)* [141], and *local classification per parent node (LCPN)* [142]. Of these approaches, local classification per parent node is most viable and avoids both the requirement to train many classifiers and the challenges of handling inconsistent or incompatible predictions (e.g., multiple positive predictions within the same level, or incompatible parent-child predictions, which must carefully be addressed in LCN and LCL approaches) [132, 133]. In local approaches, the placement of the local classifiers directly encodes the hierarchy.

- **Global Approach:**

In the global approach to hierarchical classification, a single complex model is learned which considers the entire hierarchy at once [143–145]. In this case, the model’s representations (e.g., visible through embeddings) reflect the hierarchy. The global classification approach requires only one classifier (*vs.* many with local approaches) and avoids inconsistent predictions [132, 133].

Note that some methods require leaf-node prediction (known as mandatory leaf node prediction), whereby instances are associated to labels from the root to the leaf-node of the hierarchy (either implicitly, where leaf-nodes imply all parents, or explicitly, where labels at all levels are given) [133, 142]. The alternative is non-mandatory leaf-node prediction [133, 146] (also known as internal or partial-path prediction), in which predictions can ‘stop’ at non-leaf-node levels of the hierarchy.

For additional clarity and to avoid possibly misconceptions, note that although deep neural networks are themselves composed of layered graphs of nodes, hierarchical classification does not refer to the leveraging of this graphical architecture to perform coarse-to-fine classification at different layers. While earlier layers in a DNN often encode low-level features and deeper layers encode higher-level, more discriminative features [70, 147], outputs from specific layers are not used to perform classification at particular levels of the hierarchy, e.g., classifying broad categories early and fine-grained categories later. Instead, hierarchical classification refers to the task of classifying data where labels are arranged in a taxonomy or hierarchy that is external to the graphical structure of the deep neural network used to perform this classification.

The BIOSCAN insect dataset developed in this thesis (Chapter 4) perfectly matches the hierarchical classification paradigm and may be used to study novel approaches for handling the hierarchical classification problem. In Chapter 6, Hyperbolic neural networks [148] (see Section 2.6) are compared to Euclidean neural networks, following a global approach to hierarchical classification wherein leaf-node predictions imply all parents. In Chapter 7, Hyperbolic neural networks are applied again in a study exploring whether embeddings in hyperbolic space can provide more accurate classification in a multimodal setting, again, using a global (single model) approach.

### 2.4.3 Other Challenges and Considerations

So far, challenges specifically surrounding the insect dataset used in the research of this thesis have been discussed. However, class-imbalance and fine-grained classification are by no means the only challenges faced within the field of machine learning. Here, as background, a brief overview of several other challenges, not specific to the research, are presented.

Many of the key challenges in deep learning stem from the following groups

- Model adjustments (hyperparameter selection / tuning, architecture selection choices),
- Implementation considerations for real-world computer hardware (contending with memory constraints, GPU and hardware selection), and
- Getting the most out of the model and data (the use transfer learning, data augmentation, and model validation).

Hyperparameter tuning [149, 150] involves selecting the optimal values for parameters that are *not* learned during the training process. This includes choices such as learning rate, batch size, network depth, and regularization parameters. Determining the best set of hyperparameters often requires extensive experimentation and can significantly impact model performance. Similar to hyperparameter selection, network architecture selection involves considering the number of layers, type of layers (e.g., convolutional, recurrent), and connectivity patterns. This challenge has spawned the area of research known as Neural Architecture Search (NAS) [151–153]. The model architecture should be well-suited for the task being approached, while balancing model complexity and performance.

Memory size limitations pose a practical challenge when training deep learning models, as having large number of parameters entails large memory requirements. Having access to an appropriate GPU for deep learning tasks is necessary for both timely model training and inference. Factors to consider include the GPU’s memory capacity, memory bandwidth, and processing power.

The use of transfer learning [154–157] addresses the challenges of having limited labelled data and limited computational resources for training by leveraging models pre-trained on

large-scale datasets. Fine-tuning pre-trained models or using them as feature extractors can help improve performance. Data augmentation [158–161] is often used for preventing overfitting when training data are limited. Data augmentation effectively increases the diversity of the training data. Techniques such as random cropping, rotation, scaling, and flipping are applied to generate additional training samples, enhancing the model’s ability to generalize to unseen data. Model validation is crucial for assessing the performance of deep learning models [70]. Techniques such as cross-validation [71, 79] are used to detect and prevent overfitting.

Each of these challenges requires careful consideration and understanding to effectively train deep learning models. While these problems are not specifically the subject of this thesis, all must be considered.

## 2.5 Background on Label Errors in Training Data

This background section is most relevant to [Chapter 5](#), which addresses the issues associated with having label errors in training data, through the study of the robustness of various deep neural network architectures and the development of robust loss functions implemented within label error detection schemes.

The background information is divided into three main categories: The Challenges of Label Errors in Training Data, Label Error Detection, and Robust Loss Functions. Because the label error detection frameworks of Confident Learning CL [162, 163] and Area Under the Margin (AUM) [164] are used in this thesis to evaluate the proposed loss functions, more detail is given to them than to other frameworks. Similarly, the robust loss functions Generalized Cross Entropy (GCE) [165] and Active Negative Loss (ANL) [166] are used in this thesis and presented in more detail.

### 2.5.1 Challenges of Label Errors in Training Data

Supervised learning assumes that training labels are accurate; however, label errors are pervasive even in well-established benchmark datasets, typically at rates of at least 5% [163], including in classic benchmark datasets such as MNIST [167], CIFAR [168], and ImageNet [169]. Label errors (also referred to as label *noise* in the literature) degrade model performance, limit generalizability [164, 170, 171], and mislead validation metrics. Note the distinction between label errors — the categorical permutation of sample labels — and feature measurement noise — a stochastic processes (normally) added to underlying feature values, such as images. The prevalence of label errors is especially problematic in domains with hierarchical and fine-grained classes, such as biological datasets [1, 2, 38, 41, 47, 48, 50], where mislabelling often occurs among highly similar categories.

The issues associated with label errors can affect nearly any domain, but is imperative to the BIOSCAN project where insects are hand-labelled by taxonomic experts who make their assessment based on captured images. The main difficulty here, ignoring the requirement for a high level of expertise, is the lack of consensus and certainty about the taxonomy of life itself (i.e., the locations and numbers of branches / subcategories within the tree-like hierarchy). Fundamentally, the taxonomic categorization of life is based on theory more so than an observable underlying structure. Indeed, much controversy may be found within the community of taxonomists! Nonetheless, it is accepted that a hierarchical structure does exist and may eventually be largely uncovered. Therefore, the notion of what should be considered an error is somewhat vague. Errors may arise as a result of human error (e.g., labelling two examples of the same species as being of different taxa), or as a result of simply not knowing in which category a given example belongs (e.g., labelling an example (or entire group of examples) as being part of a given category, when in fact it would better fit elsewhere). In the BIOSCAN-1M and BIOSCAN-5M Insect Datasets of Chapter 4, the error rate is unknown; however, there is no doubt that some errors are present. Efficient automatic error detection could significantly streamline curation and the flagging of samples for expert review, reducing costs and improving dataset quality.

## 2.5.2 Label Error Detection

The prevalence and negative impacts on training models caused by errors in training data have motivated the development of label error detection frameworks. Two prominent approaches in this space are Confident Learning (CL) [162] and Area Under the Margin (AUM) [164], both of which rely on training surrogate models on corrupt datasets in order to detect samples likely to have label errors. Other recent frameworks include MentorNet [172], SELFIE [173], and SelNLPL (Selective Negative Learning and Positive Learning) [174].

Confident Learning [162] uses k-fold cross-validation to train models on all folds of the mislabelled dataset, wherein, for each fold, the joint distribution of observed,  $\tilde{y}$ , and underlying true labels,  $y$ , termed the Confident joint,  $Q_{\tilde{y},y} := p(\tilde{y}|y)$ , is estimated based on the model’s predicted probabilities. Proposal methods for sample pruning available in the framework include Prune by Class (PBC), Prune by Noise Rate (PBNR), and “both”. PBC detects examples from each class that the model is least confident about, whereas PBNR detects examples that are most likely to be mislabelled as a *different* class. With PBC, for each class  $k$ , examples with the lowest self-confidence (i.e.,  $p(k|x)$ , for inputs  $x$  and labelled class  $k$ ) are proposed for pruning. With PBNR, examples from each class  $k$  having the largest margin, (i.e.,  $p(j|x) - p(k|x)$ ,  $j \neq k$ ), are proposed for pruning. If the “both” method is selected, examples must both be proposed by PBC and PBNR to be detected.

The Area Under the Margin [164] method identifies mislabelled samples based on their training dynamics. AUM measures the difference between the logit values,  $z_{\tilde{y}}(x)$ , for a training example’s as-labelled class,  $\tilde{y}$ , and its highest other class,  $\max_{k \neq \tilde{y}} z_k$ , and averages over the course of training. The margin is defined as

$$M(x, \tilde{y}) = z_{\tilde{y}}(x) - \max_{k \neq \tilde{y}} z_k, \quad (2.3)$$

and the Area Under the Margin is calculated over  $T$  training epochs as

$$\text{AUM}(x, \tilde{y}) = \frac{1}{T} \sum_{t=1}^T M_t(x, \tilde{y}). \quad (2.4)$$

Correctly labelled samples tend to exhibit positive AUM values, while mislabelled samples tend to have significantly lower AUM values due to conflicting gradient updates from other samples of the same class. To determine at which threshold of the AUM statistic samples should be detected as being likely mislabelled, a small portion of the training data,  $D_{\text{Threshold}} \subset D_{\text{Train}}$ , are re-labelling to a new *fake* class, numbered  $K + 1$  for a  $K$ -class problem, and correspondingly one additional output neuron is added to the DNN. The AUM for this set of ‘threshold samples’, i.e.,  $\text{AUM}(x_i, \hat{y}_i)$  for  $(x_i, \hat{y}_i) \in D_{\text{Threshold}}$ , is monitored, and the 99th percentile AUM from this set is used as the threshold at which correctly- and mislabelled data are separated.

### 2.5.3 Robust Loss Functions

The choice of loss function largely determines the training dynamics and the generalization properties of models trained under supervision. For classification problems, Categorical Cross Entropy (CE) [175] loss is standard. For predicted probabilities  $p_k$ , as-labelled class  $\tilde{y}$ , Cross Entropy loss is defined as

$$\text{CE}(p_k, \tilde{y}) = -\log(p_{\tilde{y}}). \quad (2.5)$$

Focal loss (FL) [99] builds upon Cross Entropy loss by placing additional weight on difficult-to-classify samples, through the inclusion of an additional factor,  $(1 - p_{\tilde{y}})^\gamma$ , with weighting parameter  $\gamma$  (with a standard setting of  $\gamma = 2$ ). Focal loss is defined as

$$\text{FL}(p_k, \tilde{y}) = -(1 - p_{\tilde{y}})^\gamma \log(p_{\tilde{y}}). \quad (2.6)$$

Note that if  $\gamma = 0$ , Focal loss reduces simply to Cross Entropy. Focal loss primarily sees use in classification problems where some classes are more difficult to learn than others, for

example in highly imbalanced datasets where some classes have far fewer training examples from which to learn. Despite their widespread use, standard loss functions are sensitive to label errors [170, 176, 177] as a result of the strong penalization of confident mispredictions, wherein  $p_{\tilde{y}}$  is low for erroneous  $\tilde{y} = \hat{y}$ .

Robustness to label errors has motivated several alternative loss functions. Mean Absolute Error (MAE) is theoretically robust to symmetric label corruption [176, 178], but has been shown to perform poorly on complicated datasets [165]. Generalized Cross Entropy (GCE) [165] combines Cross Entropy with MAE to obtain the positive characteristics of both losses. The GCE loss function, officially termed  $\mathcal{L}_q$  loss, is defined as the negative Box-Cox transformation[179],

$$\text{GCE}(p_k, \tilde{y}) = \mathcal{L}_q(p_k, \tilde{y}) = \frac{(1 - p_{\tilde{y}}^q)}{q}, \quad (2.7)$$

where  $q \in (0, 1]$ . The parameter  $q$  controls the transition between these two losses, such that in the limit as  $q \rightarrow 0$ , GCE becomes Cross Entropy (via L'Hôpital's rule), and for  $q = 1$ , GCE becomes MAE. The recommended parameter setting is  $q = 0.7$  for most problems [165].

Active Negative Losses (ANL) [166] set the current state-of-the-art and are a class of loss functions that combine Normalized Loss Functions [180],  $\mathcal{L}_{\text{norm}}$ , (originally introduced to be robust to label errors; normalizes by dividing loss for as-labelled class with loss summed over all classes) and Normalized Negative Loss Functions (NNLFs),  $\mathcal{L}_{\text{nn}}$ , (introduced in [166]) through weighting parameters,  $a, b > 0$ ,

$$\text{ANL}(p_k, \tilde{y}) = \alpha \cdot \mathcal{L}_{\text{norm}} + \beta \cdot \mathcal{L}_{\text{nn}}. \quad (2.8)$$

Any existing loss function, such as Cross Entropy or Focal loss, can be converted to an ANL through the substitution of normalized and normalized negative forms. The inclusion of L1 regularization is suggested to avoid overfitting, which commonly occurs for ANL without regularization.

Other notable robust loss functions include Symmetric Cross Entropy [181] (combines Reverse Cross Entropy [182] with CE), PHuber-CE [183] (composite loss-based gradient clipping applied to CE), Active Passive Loss (APL) [180] (designed to maximize  $p_{\tilde{y}}$  while minimizing  $p_k$ ,  $k \neq \tilde{y}$ ), Asymmetric Loss Functions (ALFs) [184], and Curriculum Loss [185]. These precursors set the stage for continued research into robust loss design.

## 2.6 Background on Hyperbolic Neural Networks

This section provides background on Hyperbolic neural networks which are used in Chapters 6 and 7. Hyperbolic representations have been used both for compact visualizations of large taxonomies [186], and more recently, as feature embeddings within the field of machine learning. Their usage has been found in tasks involving hierarchical, taxonomic, and entailment data [187–189], resulting in performance exceeding that of conventional Euclidean embeddings.

In 2018, Hyperbolic Neural Networks [148] were introduced as a means of incorporating a hyperbolic representations into deep neural networks, motivated by the fact that many complex data exhibit a highly non-Euclidean latent anatomy [190] and cannot be embedded with arbitrarily low distortion (in any number of dimensions) in Euclidean space by conventional models [188]. This seminal work [148] primarily focused on language-based applications. Therefore, in 2020, the work on HNNs was extended to work with images [191], defining several analogous operations in hyperbolic space and publishing PyTorch [192] compatible code<sup>2</sup>. In this framework, the bulk of the neural network remains the same, and it is only at the penultimate layer that features are mapped to hyperbolic space, whereby a final (hyperbolic-space) fully-connected layer is employed to perform classification and in which conventional loss functions, such as Cross-Entropy Loss [175], may be used during model training. The transformation to hyperbolic space is achieved via an exponential mapping, given in Equation (2.9), whereby an element of a Euclidean space  $v \in \mathbb{R}^n$  (a latent embedding, typically from the penultimate layer of a conventional neu-

---

<sup>2</sup>Available at <https://github.com/leymir/hyperbolic-image-embeddings>

ral network backbone) is mapped to the Poincaré disk (the hyperbolic space) of constant negative curvature  $-c$  through

$$\text{Exp}_0^c(v) := \tanh(\sqrt{c}\|v\|) \frac{v}{\sqrt{c}\|v\|}. \quad (2.9)$$

Building further upon HNNs, Hyperbolic Vision Transformers were introduced in [193], combining the transformer architecture with hyperbolic representations. Additionally, a vanishing gradient [194, 195] problem was found to exist in HNNs [196, 197] and was solved [198] through clipping features of the Euclidean sub-network prior to exponential mapping to hyperbolic space. This innovation dramatically boosted the performance of HNNs, and in particular allowed HNNs to perform just as well as ENNs on non-hierarchical data, while consistently further outperforming ENNs on hierarchical data.

While the preceding background has focused mainly on image-based innovations, HNNs are used across many domains. For example, there has been work on Hyperbolic Graph Neural Networks [199–201], NHHs for Natural Language Processing (NLP) [148, 202, 203], and HNNs for visualization / dimensionality-reduction [204].

### 2.6.1 Related Multimodal and Hyperbolic Representation Work

Multimodal contrastive learning [118, 205] trains models to align representations from different data modalities (e.g., images and text) by bringing matching pairs closer together in a shared embedding space while pushing non-matching pairs apart. This enables the model to learn cross-modal associations without requiring explicit labels, i.e., through self-supervision, supporting tasks like image–text retrieval and zero-shot classification. Background here is relevant to Chapter 7, where advancements to multimodal learning are made for biological data.

**Euclidean Multimodal Learning** is the norm for recent advances in the multimodal contrastive learning domain, both in general vision-language frameworks such as CLIP [118] and SigLIP [206] and domain-specific ones, including those for biodiversity applications [207–209]. These biodiversity models embed images, textual data, and optionally DNA barcodes

into a shared Euclidean embedding space using modality-specific encoders and contrastive learning objectives. CLIBD [209] in particular demonstrates zero-shot classification on the BIOSCAN-1M dataset of Section 4.1, achieving superior accuracy to unimodal baselines.

**Hyperbolic Multimodal Learning** combines multimodal learning and the use of hyperbolic geometry to co-align embeddings from different modalities in a hierarchical representation space. Liu *et al.* [210] showed how to align images and text embeddings in a Poincaré hyperbolic space, while MERU [211] uses contrastive learning to align images and text in Lorentzian space. Following MERU, HyCoCLIP [212] incorporated compositional constraints to strengthen fine-grained alignment between parts and wholes in visual concepts. These works show hyperbolic geometry can enhance the structural consistency and interpretability of multimodal models, particularly in settings with implicit or weakly defined hierarchies.

# Chapter 3

## Formulation of the Insect Classification Problem

In [Section 2.3](#), background was given on the use of convolutional neural networks in the fields of machine learning and computer vision. Here, notation specific to the research is introduced. With this notation, the research problem of taxonomic insect classification is mathematically formulated.

The research presented in this thesis centres around taxonomically classifying insects based on images. The model input therefore is an RGB image of an insect,  $x \in \mathcal{X} \subset \mathbb{R}^{W \times H \times 3}$ . Possible “side information” for use during training could include attributes such as BINs, described in [Section 2.1.2](#),  $\xi$ , which act as a proxy for species labels and which are available for all training data (whereas species labels are not). The model is a function,  $f$ , that maps input,  $x$ , to predicted classes by selecting the class,  $k$ , having the maximal predicted probability,  $y^* = f(x) = \arg_k \max p(k|x)$ . The predicted probabilities,  $p(k|x)$ , are compared against ground truth labels,  $y$ , during training. For classification tasks at a specific taxonomic rank, there is only one label, rather than the general case for which  $y$  includes categorical labels for each taxonomic rank. Building upon [Equation \(2.2\)](#) for the most general case, network parameters,  $\theta$ , are learned by training on a dataset of  $N$

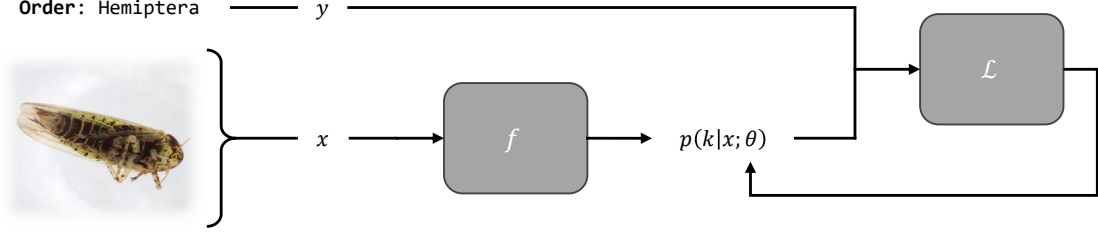


Figure 3.1: The image classification model training scheme is demonstrated graphically here. Training data consisting of an insect image,  $x$ , and its order-level taxonomic label,  $y = \text{Hemiptera}$ , are provided as inputs. The model,  $f$ , produces predicted probabilities,  $p(k|x; \theta)$ , which are then fed into the loss function,  $\mathcal{L}$ , to be compared to the ground truth label,  $y$ . This is performed in batches over the entire training set. The gradient of the loss is then be fed back to the model to update weights,  $\theta$ , tending towards optimal values.

images, minimizing the empirical risk,

$$\theta^* = \arg \min_{\theta} \frac{1}{N} \sum_{i=1}^N \mathcal{L} (p(k|x_i; \theta), (y_i, \xi_i)) . \quad (3.1)$$

While this formulation is suitable for the most general case, the taxonomic insect classification research of this thesis begins in [Chapter 4](#) with classification, individually, at the order- and family-levels of taxonomy and does not incorporate the use of any BIN “side information”. Therefore, predicted probabilities are compared only to the taxonomic labels,  $y$ , and [Equation \(3.1\)](#) becomes

$$\theta^* = \arg \min_{\theta} \frac{1}{N} \sum_{i=1}^N \mathcal{L} (p(k|x_i; \theta), y_i) . \quad (3.2)$$

This basic training scheme is demonstrated graphically in [Figure 3.1](#), where an insect image,  $x$ , and its order-level taxonomic label,  $y = \text{Hemiptera}$ , are provided as training data. Model parameters,  $\theta^*$ , (i.e., convolutional filters, connection weights, and biases) are learned via optimizing the loss function,  $\mathcal{L}$ , over the entire training dataset.

This chapter formulated the overarching insect classification problem addressed by this thesis, and established the notation to be used. This formulation is applied in [Chapter 4](#), which describes the curation of the BIOSCAN dataset and sets a baseline for classification performance upon it. Additionally, the notation is extended in [Chapter 5](#) to the subject of label errors.

# Chapter 4

## The BIOSCAN Dataset<sup>1</sup>

The International Barcode of Life (iBOL) Consortium has the primary goal of cataloguing all multicellular life in an effort to study global biodiversity. Further background on iBOL’s overarching goals and motivation for the BIOSCAN project are discussed in [Section 2.1](#). Through the BIOSCAN project, iBOL and its international partners have collected a vast number of insect images, while also DNA barcoding and taxonomically classifying these samples. However, the BIOSCAN data was neither organized for nor available to the machine learning community. With current advancements in computer hardware, the main bottle-neck in machine learning is now the availability of large curated datasets. Simultaneously, the BIOSCAN team wishes to incorporate machine learning techniques into their insect classification process. Therefore, a curated insect dataset is of mutual benefit to both the machine learning community and the BIOSCAN team. Further background on insect classification and datasets is given in [Section 2.2](#).

This chapter describes the curation of two large-scale BIOSCAN datasets (BIOSCAN-1M [\[1\]](#) and BIOSCAN-5M [\[2\]](#), produced in consecutive years as the BIOSCAN collection grew), and presents baseline classification results associated with BIOSCAN-1M. First, [Section 4.1](#) discusses the construction of a 1 million sample dataset, BIOSCAN-1M, and

---

<sup>1</sup>Note that significant portions of this chapter were originally written for submissions to the NeurIPS 2023 [\[1\]](#) and 2024 [\[2\]](#) conferences. See the [Statement of Contributions](#) for details.

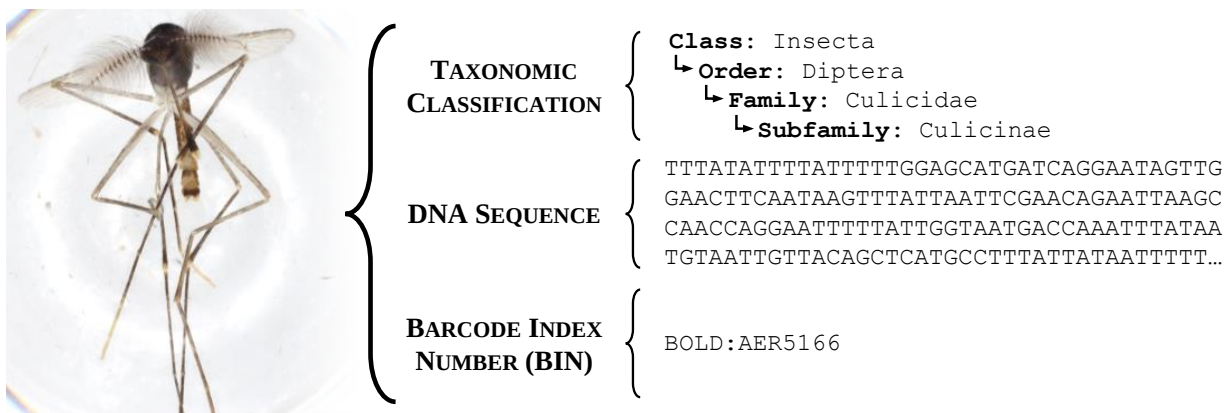


Figure 4.1: Dataset records contain high-quality microscope images of insects and labels including the taxonomic classification, raw DNA sequences, and Barcode Index Number (BIN). Pictured here is a mosquito of the subfamily *Culicinae*, the most populous subfamily of mosquitoes with species found around the world.

provides a statistical analysis of its characteristics. Next, [Section 4.2](#) provides classification results for the BIOSCAN-1M dataset, setting a classification performance baseline for the ML community. Finally, [Section 4.3](#) extends the BIOSCAN-1M dataset through the creation of the BIOSCAN-5M dataset, which includes 5 million samples and new metadata not originally included in the BIOSCAN-1M dataset.

## 4.1 Construction and Statistical Analysis of the BIOSCAN-1M Dataset

The curated BIOSCAN insect image dataset contains approximately 1.1 M high-quality microscope images, each of which is annotated with the insect’s taxonomic ranking and accompanied by its raw DNA sequences and Barcode Index Number (BIN) [28], an example of which is shown in [Figure 4.1](#).

### 4.1.1 Subset sampling and split mechanism

For use in the baseline classifier experiments of [Section 4.2](#), the full-sized BIOSCAN-1M Insect dataset was sampled to produce the BIOSCAN-Diptera dataset. Both the BIOSCAN-1M Insect and BIOSCAN-Diptera datasets also needed to be split into training, validation, and testing partitions, which was done in a 70-10-20 ratio (selected to balance the need for large amounts of training data, while still including sufficient validation and testing data). When partitioning the BIOSCAN-1M Insect dataset, orders with fewer than three samples were discarded (since it would be impossible for each of training, validation, and testing partitions to contain even one sample). Sufficient samples are available for 16 of the 19 orders included in the original database. The BIOSCAN-Diptera dataset was constructed by specifically selecting samples of the order Diptera from the main dataset. The purpose of the BIOSCAN-Diptera dataset is for use in Family-level classification experiments. Only the 40 most populous families are retained for this dataset.

Next, Small and Medium subsets are generated by sampling 50 k and 200 k data samples, respectively, from both the train, validation, and test sets of the full BIOSCAN-1M Insect and BIOSCAN-Diptera datasets. Note that sampling is performed *within* each split, such that samples of a particular split (e.g., training) in the full BIOSCAN dataset can only be found in the *same* split of the Small and Medium subsets. In all classification experiments, class-based stratified sampling is used to split the dataset into train, validation and test sets. To this end, 70% of the samples of each class are randomly selected as training, 10% as validation, and 20% as test samples, as shown in [Table 4.1](#).

A comprehensive metadata file is created alongside the images, and includes taxonomic annotations, DNA barcode sequences, and data sample indexes and labels. The metadata file also contains image names and IDs to locate the corresponding images within the dataset packages. Additionally, it identifies the images associated with the training, validation, and test splits.

Table 4.1: The total number of samples used in the BIOSCAN-Insect dataset and its subsets. The number of data samples in the train, validation, and test sets, as well as the number of classes for Order-level (16 orders) and Diptera family-level (40 families) experiments are specified.

| Dataset                       | Total     | Train   | Validation | Test    | Classes |
|-------------------------------|-----------|---------|------------|---------|---------|
| BIOSCAN-1M-Insect             | 1,128,313 | 789,813 | 112,835    | 225,660 | 16      |
| BIOSCAN-Diptera               | 891,338   | 623,937 | 89,135     | 178,266 | 40      |
| BIOSCAN-Insect/Diptera Medium | 200,000   | 140,000 | 20,000     | 40,000  | 16/40   |
| BIOSCAN-Insect/Diptera Small  | 50,000    | 35,000  | 5000       | 10,000  | 16/40   |

### 4.1.2 Image Cropping

To improve computational efficiency during model training, images are cropped and resized to be 256 px on the smaller dimension. Furthermore, reducing the resolution to 256 px helps to reduce the size of the dataset from 2.3 TB down to 26 GB for the original uncropped images, and from 151 GB down to 7 GB for cropped images, thereby improving the accessibility of the data to the public. The BIOSCAN-1M image dataset has insects with varying size, pose, color and shape. Due to these variations, cropping is not a simple task.

A cropping tool is developed by fine-tuning a DETR [213] model with a ResNet-50 [85] backbone (pretrained on MSCOCO [214]) on a small set of 2,000 insect images annotated using the Toronto Annotation Tool Suite [215]. In DETR, the CNN-based feature extractor extracts a set of image features that are fed into a transformer-based encoder-detector. The detector takes a set of learned positional embeddings as object queries and uses them to attend to the encoder outputs. Each of the output decoder embeddings is then passed to a shared FFN which predicts whether there is an “insect” or “no object” and regresses the bounding box. The DETR model is further trained for 10 epochs with the AdamW optimizer with a learning rate of 0.0001, weight decay of 0.0001 and a batch size of 8. To crop the image, the fine-tuned DETR model is first applied, and the predicted bounding box with the highest confidence score is selected. The finalized crop is determined as the selected bounding box, extended equally in width and height by a factor of 0.4 of the maximum dimension.

### 4.1.3 Statistical Analysis

A statistical analysis of the overall BIOSCAN-1M dataset is presented here. [Figure 4.2](#) shows the class distribution of labelled records at each taxonomic rank, categorized by taxonomic order. The top panel shows the quantity of labelled records at each taxonomic rank (log-scaled), whereas the second panel shows the relative frequency or proportion (i.e., normalized by the number of samples in each particular order) of labelled records at each rank. While a great proportion of records are labelled at the family-level, dramatically fewer are labelled at the genus- and species-levels. Indeed,  $\sim 1.11$  M samples are labelled at the family-level, but only  $\sim 254$  k at the genus-level, and  $\sim 84$  k at the species-level, out of  $\sim 1.12$  M records in total. Furthermore, these lower-rank labelled samples are concentrated within the most abundant orders, e.g., Diptera contains  $\sim 47$  k samples with species labels.

A trend line for order-level counts is shown in the top panel of [Figure 4.2](#). The trend line follows a power law,  $\hat{c} = (8.96 \times 10^5)k^{-3.33}$ , for approximate sample counts in each order,  $\hat{c}$ , indexed by  $k$ . This type of long-tailed class distribution is typical of biological data, as was discussed in [Section 2.4.1](#). An exponential fit was also explored, given that sample counts tend to decrease by a factor of approximately 2 from one order to the next when sorted in descending order. However, between Diptera and Hymenoptera, there is a decrease in sample count by a factor of approximately 10, which leads to a relatively poor exponential fit and indicates the power law (long-tailed distribution) is more likely.

Building upon [Table 2.1](#) presented in [Chapter 2](#), [Table 4.2](#) includes rows for the BIOSCAN-Insect and BIOSCAN-Diptera datasets for comparison to the other existing datasets already present in the original table, and includes the imbalance ratio, IR, in the rightmost column. Observe that while there is significant variety in the imbalance ratio among datasets, the imbalance of the BIOSCAN-Insect is *orders of magnitude* greater than that of all the other datasets. The Pl@ntNet-300K [\[47\]](#) dataset also has remarkably high class imbalance, exceeding that of the BIOSCAN-Diptera dataset. While a high imbalance ratio was expected based on prior discussion of the dataset, the imbalance ratio metric is also highly subject to bias due to its dependence on only the two most extreme classes, e.g., one can imagine a dataset where the smallest class has only one sample and

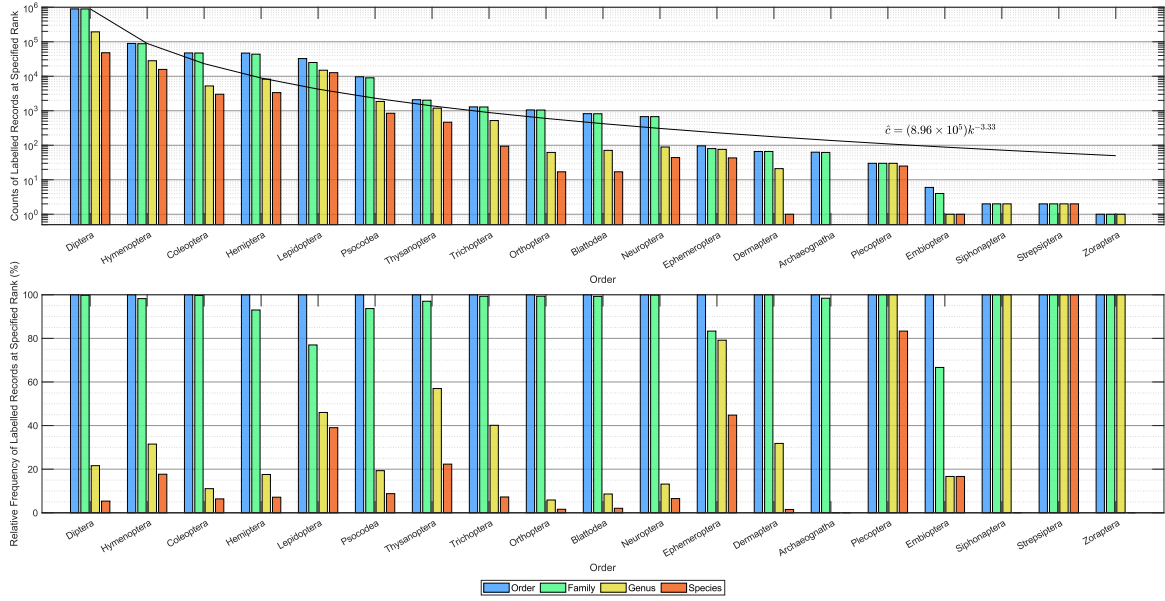


Figure 4.2: Class distribution of labelled records at each taxonomic rank, categorized by taxonomic order and sorted in descending order of sample abundance. The top panel shows the quantity of labelled records at each taxonomic rank (log-scaled), whereas the second panel shows the relative frequency or proportion (i.e., normalized by the number of samples in each particular order) of labelled records at each taxonomic rank. Note that because this dataset contains only samples that are labelled at least at the order-level, the relative frequencies of labelled records at the order-level are all 100%. Observe that while a great proportion of records are labelled at the family-level, dramatically fewer are labelled at the genus- and species-levels. In the top panel, a trend line for order-level counts is shown. The trend line follows a power law,  $\hat{c} = (8.96 \times 10^5)k^{-3.33}$ , for approximate sample counts in each order,  $\hat{c}$ , indexed by  $k$ .

Table 4.2: Comparison of the BIOSCAN-1M Insect and BIOSCAN-Diptera datasets to existing biological datasets

| Name / Citation        | Domain           | Images  | Classes         | IR         |
|------------------------|------------------|---------|-----------------|------------|
| iNaturalist [216]      | Plants & Animals | 859 k   | 5,089           | 435.44     |
| Pl@ntNet-300K [47]     | Plants           | 306 k   | 1,000           | 3,604.00   |
| Urban Trees [44]       | Trees            | 80 k    | 18              | 7.51       |
| IP102 [41]             | Insect           | 75 k    | 102             | 13.63      |
| NA Birds [43]          | Birds            | 48 k    | 555             | 15.00      |
| LeafSnap [42]          | Plants           | 31 k    | 184             | 8.00       |
| Pest24 [46]            | Insect           | 25 k    | 24              | 493.95     |
| Flowers 102 [217]      | Flowers          | 8 k     | 102             | 1.00       |
| IP-FSL [54]            | Insect           | 7 k     | 142             | 2.94       |
| <b>BIOSCAN-Insect</b>  | Insect           | 1,128 k | <sup>†</sup> 16 | 156,856.75 |
| <b>BIOSCAN-Diptera</b> | Insect           | 891 k   | *40             | 1,092.61   |

<sup>†</sup>= Orders. \* = Families.

the largest has 100 samples, so  $IR = 100$ ; however, this does not offer any information about the intermediate class sizes.

Going beyond the imbalance ratios shown in Table 4.2, the Gini coefficient [130], described in Section 2.4.1 and illustrated graphically in Figure 2.2, is computed for the BIOSCAN datasets. The Gini coefficient provides a more holistic measure of imbalance, owing to the fact that it depends on all classes, not only the most extreme. The Gini coefficient of the BIOSCAN-Insect dataset is 0.81, and for the BIOSCAN-Diptera dataset is 0.80. Note that Gini coefficients close to 1 indicate maximal imbalance. Therefore, the degree of class imbalance appears to be high.

## 4.2 Classification Results on the BIOSCAN-1M Dataset

Following the insect classification problem formulation presented in Chapter 3, two sets of experiments were run: order-level and family-level classification (within the order Diptera). Each experiment involves two models (ResNet-50 and ViT-Patch-16-224) trained using two

Table 4.3: Detailed hyperparameter settings of the BIOSCAN-1M experiments. The model ResNet-50 is abbreviated to R50 here.

| Parameters             | Settings            | Parameters                   | Settings |
|------------------------|---------------------|------------------------------|----------|
| Model                  | R50/ViT-P16-224     | Batch-Size                   | 32       |
| Loss                   | Cross-Entropy/Focal | Epoch                        | 100      |
| Optimizer              | SGD                 | Num-Workers                  | 4        |
| Weight Decay ( $\mu$ ) | 0.0001              | Image-Size (Train/Val.)      | 256      |
| Learning rate          | 0.001               | Crop-Size (Train)            | 224      |
| Momentum               | 0.9                 | Rand-Horizontal-Flip (Train) | Yes      |
| K                      | [1, 3, 5, 10]       | Centre-Crop (Val.)           | 224      |
| group-level            | order/family        | Dataset size                 | L/M/S    |

loss functions (Cross-Entropy and Focal loss) on Small and Medium subsets and the full-sized dataset. Here, results of these experiments are provided and discussed. In total, 24 experiments were run (2 models  $\times$  2 loss functions  $\times$  6 subsets).

Detailed hyperparameter settings of these 24 experiments are shown in Table 4.3. For Small and Medium subsets, the models were trained for 100 epochs; for the full dataset, the models were trained for fewer epochs considering the convergence was met in the validation set.

The performance of the classification models was evaluated using top-K accuracy, which evaluates whether the ground-truth label is among the top-K predicted classes. From the 24 experiments described above, test results are reported for only the *best* model based on validation performance for the micro-averaged (i.e., averaged equally over all samples) and macro-averaged (i.e., averaged per category, then averaged again) top-K accuracy at  $K \in [1, 3, 5, 10]$  for each data subset, as shown in Table 4.4. Figure 4.3 shows the per-class top-1 test accuracy for the Order and Family classification of the full dataset. Accuracy is quite high in relation to that of Pl@ntNet-300K [47] (the experimental setup of which guided that of this experiment), above 90%, for most classes, decreasing mainly for classes with little training data (those further to the right in Figure 4.3).

Table 4.4: Micro- and Macro-averaged Top-K accuracy based on the test sets of Insect-Order and Diptera-Family subsets using the Small and Medium subsets and the full-sized dataset.

| Metric      | Dataset | Insect-Order |       |       |        | Diptera-Family |       |       |        |
|-------------|---------|--------------|-------|-------|--------|----------------|-------|-------|--------|
|             |         | Top-1        | Top-3 | Top-5 | Top-10 | Top-1          | Top-3 | Top-5 | Top-10 |
| Micro Top-K | Small   | 98.10        | 99.50 | 99.79 | 99.93  | 93.65          | 97.35 | 98.06 | 98.65  |
|             | Medium  | 99.10        | 99.77 | 99.89 | 99.99  | 73.50          | 80.00 | 83.76 | 89.51  |
|             | Full    | 99.68        | 99.96 | 99.98 | 99.99  | 97.48          | 99.01 | 99.43 | 99.78  |
| Macro Top-K | Small   | 85.99        | 91.74 | 99.33 | 99.82  | 91.95          | 96.21 | 97.19 | 98.04  |
|             | Medium  | 83.87        | 96.50 | 97.17 | 99.59  | 83.83          | 90.34 | 92.25 | 94.98  |
|             | Full    | 80.85        | 88.87 | 91.00 | 93.66  | 89.67          | 95.77 | 96.63 | 97.71  |

For the Small dataset, the Vit-P16-224 model with Focal loss produced the best validation results. For the Medium dataset, Vit-P16-224 with Focal loss for Order classification and with Cross Entropy for the Family classification experiments were best. Finally, for the full dataset, best results were produced using ResNet-50 with Cross Entropy and ViT-P16-224 with Cross Entropy for Order and Family experiments, respectively. Therefore, it is not obvious that a particular model type or loss is necessarily universally better, and further exploration involving more architectures and loss functions will be required.

Figure 4.4 shows representative confusion matrices for order-level classification, evaluated on training data (top row) and validation data (bottom row) from the Medium subset. In both cases, the left column shows the normalized confusion matrices, whereas the right column shows the non-normalized confusion matrices, illustrating the actual quantity of correctly / incorrectly classified images. Rows and columns of the confusion matrices are sorted in descending order of class size. Observe that the patterns found in the confusion matrices for validation closely match those found in the training confusion matrices, indicating that over-fitting is likely not the issue here leading to poor performance in orders of low sample abundance. Additionally, as was seen already in Figure 4.3, model accuracy is remarkably high for nearly all classes and diminishes only for orders with very few available training samples (fewer than 100, with two cases, Embioptera and Odonata, having only *one* sample).

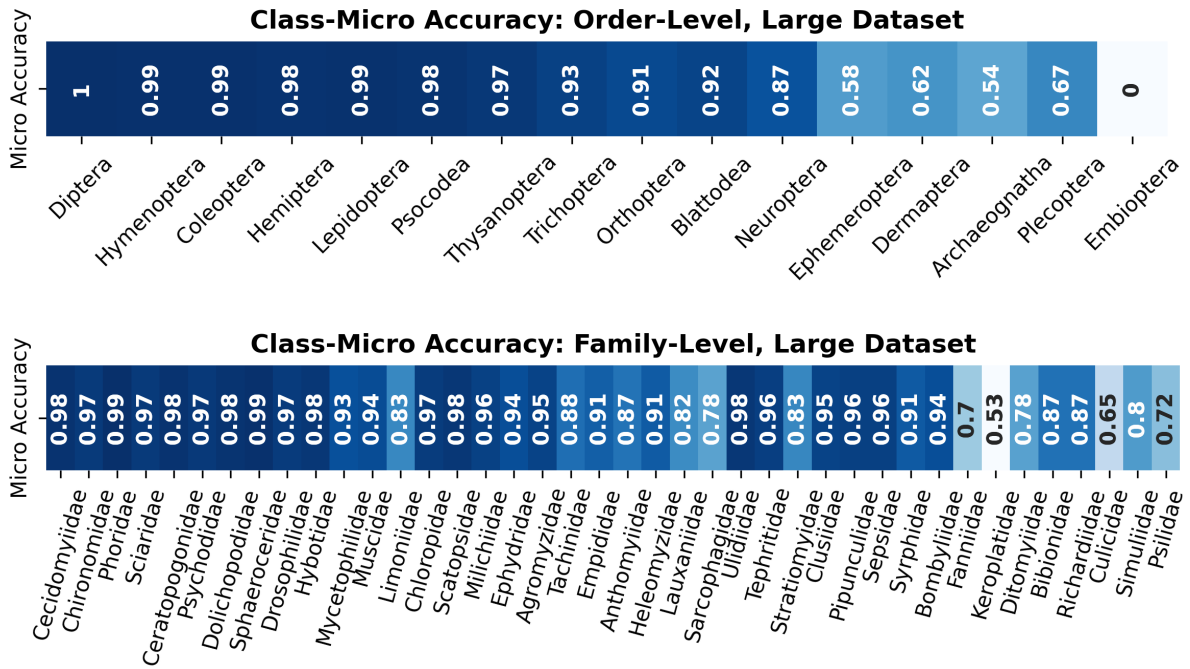
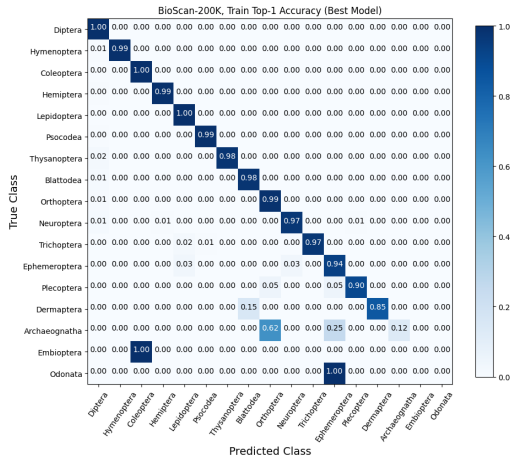
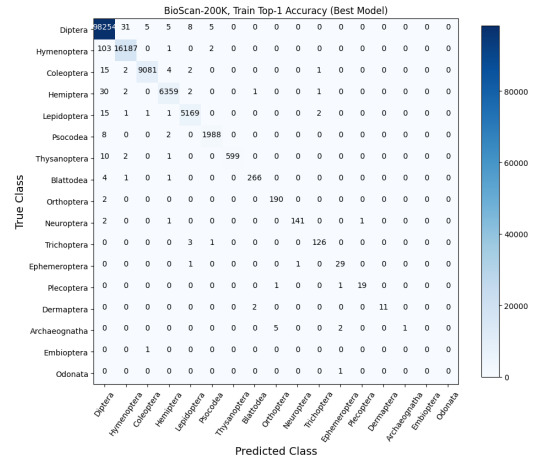


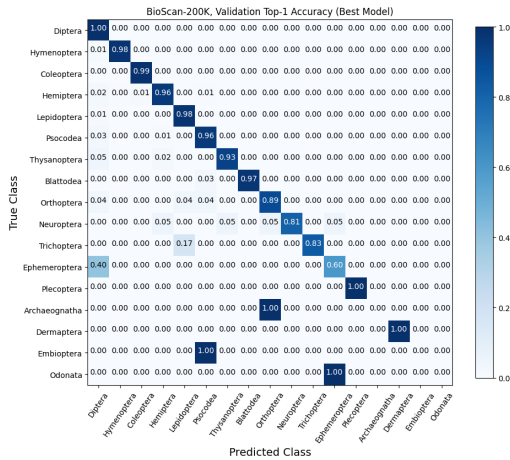
Figure 4.3: Per-class Top-1 test accuracy of the Order- and Family-level classification experiments using the full-sized dataset. Classes are listed in descending order with respect to their number of samples. Figure produced for and taken from [1].



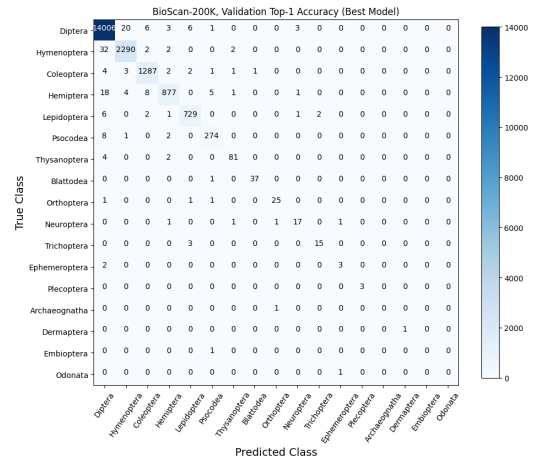
(a) Normalized



(b) Non-normalized



(c) Normalized



(d) Non-normalized

Figure 4.4: Confusion matrices for order-level classification, evaluated on training data from the Medium subset. In the top row, results are evaluated on the training partition, whereas in the bottom row, results are evaluated on the validation partition. The left column shows the normalized confusion matrices, whereas the right column shows the non-normalized confusion matrices, illustrating the actual quantity of correctly / incorrectly classified images. Figure produced for and taken from [1].

### 4.3 The BIOSCAN-5M Dataset

The BIOSCAN-1M dataset of [Section 4.1](#) was created to understand and monitor biodiversity. The dataset paired DNA with images as a stepping stone to developing AI tools for automatic classification of organisms. However, that work only investigated image classification down to the family-level, focusing on the family Diptera, and did not fully utilize the multimodal nature of the dataset. In addition, BIOSCAN-1M was limited to the *Insecta* class.

The BIOSCAN-5M dataset expands upon the original BIOSCAN-1M dataset by including dramatically more samples and metadata. It is a comprehensive repository of multi-modal information (see [Figure 4.5](#)) on over 5 million arthropod specimens (98% insects), with 1.2 million labelled to **genus** or **species** taxonomic ranks. Compared to its predecessor, the BIOSCAN-5M dataset offers a significantly larger volume of high-resolution microscope images and DNA barcodes along with critical annotations, including taxonomic ranks, size, and geographical information. Additionally, data cleaning is performed to resolve inconsistencies and provide more reliable labels.

The multimodal characteristics of BIOSCAN-5M are not only essential for biodiversity studies, but also facilitate further innovation in machine learning and AI. As part of the research [\[2\]](#) not presented in this thesis, collaborators conducted experiments that leverage the multimodal aspects of BIOSCAN-5M, extending its application beyond the image-only modality in the BIOSCAN-1M experiments of [Section 4.2](#). These experiments [\[2\]](#) are briefly outlined here for the purpose of demonstrating the usage of the multimodal aspects of BIOSCAN-5M. One experiment entailed training the masked language model (MLM) proposed in BarcodeBERT [\[218\]](#) on the DNA barcodes of the BIOSCAN-5M dataset and demonstrating the impact of using this large reference library on species- and genus-level classification. Accuracy higher than that of state-of-the-art models pretrained on more general genomic datasets is achieved, especially in the 1NN-probing task of assigning samples from unseen species to seen genera. Another experiment studied a zero-shot transfer learning task [\[219\]](#), through zero-shot clustering representation embeddings obtained from encoders trained with self-supervised paradigms. This zero-shot transfer learning approach

demonstrates the effectiveness of pretrained embeddings in clustering data, even in the absence of ground-truth. Lastly, as in CLIBD [209], a shared embedding space is learned across three modalities in the dataset — high-quality RGB images, textual taxonomic labels, and DNA barcodes — for fine-grained taxonomic classification.

### 4.3.1 Dataset Details

The BIOSCAN-5M dataset is derived from [220] and comprises 5,150,850 **arthropod** specimens, with insects accounting for about 98% of the total. BIOSCAN-5M is a superset of the BIOSCAN-1M Insect dataset, providing more samples and additional metadata such as geographical location. Unlike existing image-based datasets, BIOSCAN-5M integrates taxonomic labels, raw nucleotide barcode sequences, assigned barcode index numbers, geographical data, and specimen size information, making it one of the most comprehensive biological datasets available. Figure 4.5 illustrates the data modalities available in the dataset for one sample record.

**Images.** The BIOSCAN-5M dataset provides specimen images at  $1024 \times 768$  pixels, captured using a Keyence VHX-7000 microscope. Figure 4.6 showcases the diversity in organism morphology across the dataset. The images are accessed via the `processid` field of the metadata as `{processid}.jpg`. Following the method described in Section 4.1.2 for BIOSCAN-1M, the images are cropped and resized to  $341 \times 256$  pixels to facilitate model training. A DETR [213] (End-to-End Object Detection with Transformers) model is fine-tuned for image cropping. For BIOSCAN-1M, the DETR cropping model was trained using 2k insect images. Building on the BIOSCAN-1M cropping tool checkpoint, the model is fine-tuned for BIOSCAN-5M using the same 2k images and an additional 837 images that were not well-cropped previously. The bounding box of the cropped region is provided as part of the dataset release.

**DNA-based indexing.** The DNA information of the BIOSCAN-5M dataset described in Section 2.1.2 is represented as the raw nucleotide barcode sequence, under the `dna_barcode`

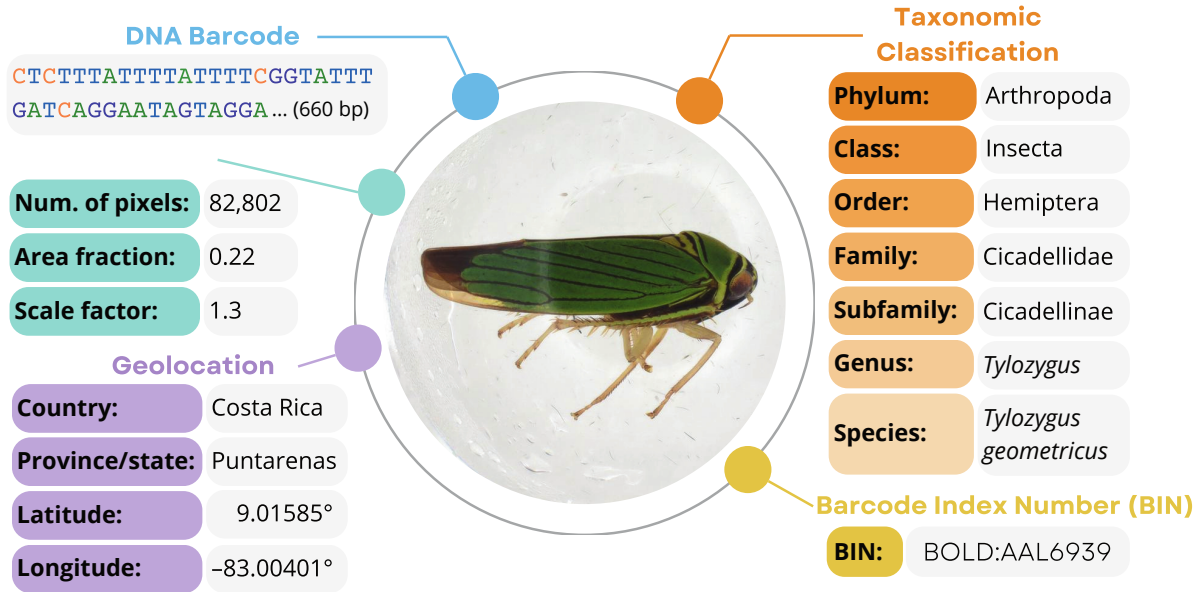


Figure 4.5: The BIOSCAN-5M Dataset provides taxonomic labels, a DNA barcode sequence, barcode index number, a high-resolution image along with its cropped and resized versions, as well as size and geographic information for each sample. Figure produced for and taken from [2].



Figure 4.6: Samples of original full-size images of distinct organisms in the BIOSCAN-5M dataset. Figure produced for and taken from [2].

Table 4.5: **Summary statistics** of dataset records by taxonomic rank: imbalance ratio (IR) between most and least common labels, number of unique labels, and number of labelled samples.

| Attributes         | BIOSCAN-5M |            |           |              | BIOSCAN-1M |           |              |
|--------------------|------------|------------|-----------|--------------|------------|-----------|--------------|
|                    | IR         | Categories | Labelled  | Labelled (%) | Categories | Labelled  | Labelled (%) |
| <b>phylum</b>      | 1          | 1          | 5,150,850 | 100.0        | 1          | 1,128,313 | 100.0        |
| <b>class</b>       | 719,831    | 10         | 5,146,837 | 99.9         | 1          | 1,128,313 | 100.0        |
| <b>order</b>       | 3,675,317  | 55         | 5,134,987 | 99.7         | 16         | 1,128,313 | 100.0        |
| <b>family</b>      | 938,928    | 934        | 4,932,774 | 95.8         | 491        | 1,112,968 | 98.6         |
| <b>subfamily</b>   | 323,146    | 1,542      | 1,472,548 | 28.6         | 760        | 265,492   | 23.5         |
| <b>genus</b>       | 200,268    | 7,605      | 1,226,765 | 23.8         | 3,441      | 254,096   | 22.5         |
| <b>species</b>     | 7,694      | 22,622     | 473,094   | 9.2          | 8,355      | 84,397    | 7.5          |
| <b>dna_bin</b>     | 35,458     | 324,411    | 5,137,441 | 99.7         | 91,918     | 1,128,313 | 100.0        |
| <b>dna_barcode</b> | 3,743      | 2,486,492  | 5,150,850 | 100.0        | 552,629    | 1,128,313 | 100.0        |

field, and the Barcode Index Number under **dna\_bin** field. Independently, the field **processid** is a unique number assigned by BOLD [29] to each record, and **sampleid** is an identifier given by the collector.

**Biological taxonomic classification.** The dataset samples undergo taxonomic classification using a hybrid approach involving an AI-assisted tool based on the original BIOSCAN-1M [1] work and human taxonomic experts. After DNA barcoding and sequence alignment, the taxonomic levels derived from both the AI tool and DNA sequencing are compared. Any discrepancies are then reviewed by human experts. Importantly, assignments to deeper taxonomic levels, such as **family** or lower, rely entirely on human expertise. Labels at seven taxonomic ranks are used to represent individual specimens, denoted by fields **phylum**, **class**, **order**, **family**, **subfamily**, **genus**, and **species**.

In the source data, cases of identical DNA nucleotide sequences labelled differently at some taxonomic levels were identified, which were likely due to human error (e.g., typos) or disagreements in the taxonomic labelling. To address this, all of the inconsistent taxonomic labels were checked and corrected to address typos and ensure consistency across DNA

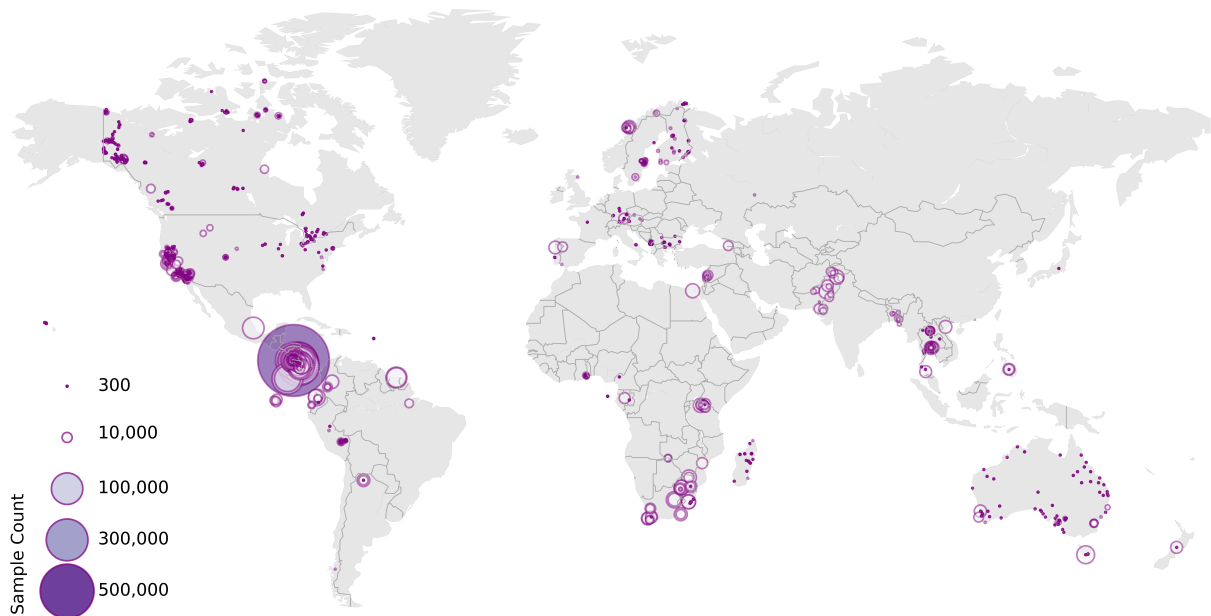


Figure 4.7: Geographical locations obtained from latitude and longitude coordinates of the regions where the samples of the BIOSCAN-5M dataset were collected. Figure produced for and taken from [2].

barcodes. Note that some of the inconsistent species labels are placeholder labels that do not correspond to well-established scientific taxonomic species names. These placeholder labels are used by taxonomists to indicate when they believe two records are from the same species but do not have well-established species names. In the BIOSCAN data, the placeholder **species** labels are identified by **species** labels that begin with a lowercase letter, contain a period, contain numerals, or contain “malaise”.

Statistics for BIOSCAN-5M are given in Table 4.5 for the seven taxonomic ranks along with the BIN and DNA nucleotide barcode sequences. For each group, the number of categories, and the count and fraction labelled are reported. The class imbalance ratio (IR) is also reported, and is computed as the ratio of the number of samples in the largest category to the smallest category, reflecting the class distribution within each group.

Table 4.6: The statistics for the columns indicating geographical locations where the specimens are collected.

| Geo locations       | Categories | Labelled  | Labelled (%) | Unlabelled | Unlabelled (%) | IR          |
|---------------------|------------|-----------|--------------|------------|----------------|-------------|
| country             | 47         | 5,150,842 | 100.00       | 8          | 0.00           | 325,631.6   |
| province_state      | 102        | 5,058,718 | 98.21        | 92,132     | 1.79           | 1,243,427.0 |
| coord-lat           | 1,394      | 5,149,019 | 99.96        | 1,831      | 0.04           | 556,352.0   |
| coord-lon           | 1,489      | 5,149,019 | 99.96        | 1,831      | 0.04           | 618,931.0   |
| Location (lat, lon) | 1,650      | 5,149,019 | 99.96        | 1,831      | 0.04           | 520,792.0   |

**Geographic location.** The BIOSCAN-5M dataset includes geographic location information, detailing the country and province or state where each specimen is collected, along with the latitude and longitude coordinates of each collection site. This information is detailed in the fields `country`, `province_state`, `coord-lat` and `coord-lon`. The distribution of specimen collection sites are shown on a world map in [Figure 4.7](#).

The detailed statistical analysis of the geographical locations where the organisms were collected is presented in [Table 4.6](#). This table indicates the number of distinct regions represented by country, province or state, along with their corresponding latitude and longitude. Additionally, [Table 4.6](#) provides the count of labelled versus unlabelled records, as well as the class imbalance ratio (IR) for each location group within the dataset.

The latitude and longitude coordinates indicate that the dataset comprises 1,650 distinct regions with unique geographical locations shown by [Table 4.6](#). However, a significant portion of the organisms — approximately 73.36% — were collected from the top 70 most populated regions (by insects, not humans), which represent only 4.24% of the total regions identified by their coordinates.

[Figure 4.8](#) shows the distinct countries where the organisms were collected on the world map. The majority of the organisms, over 62%, were collected from Costa Rica.

**Size information** Tracking organism size offers insights into environmental shifts vital for biodiversity conservation [221]. In the case of insects, it can signal shifts in various

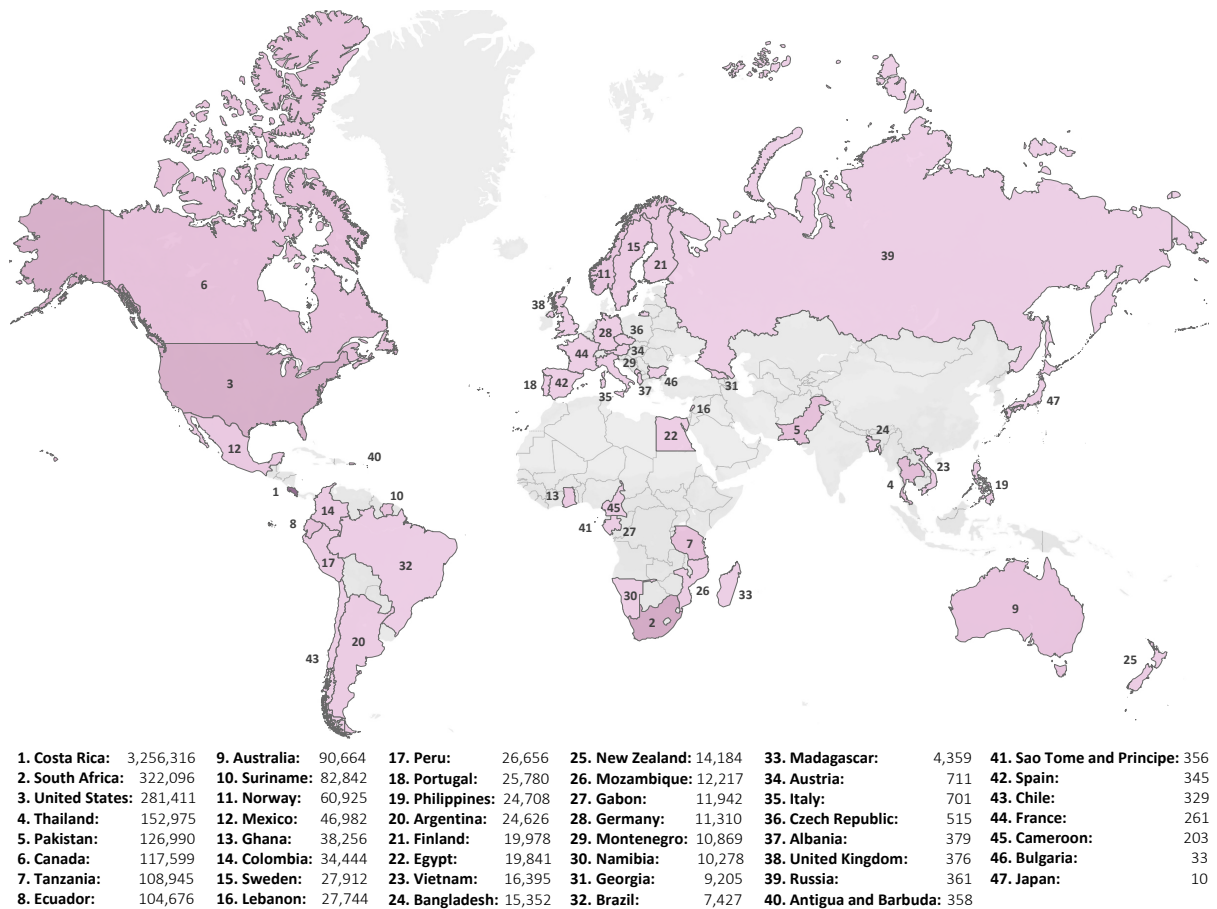


Figure 4.8: Global distribution of sample collection efforts. The countries are ranked by the number of samples collected. Figure produced for and taken from [2].

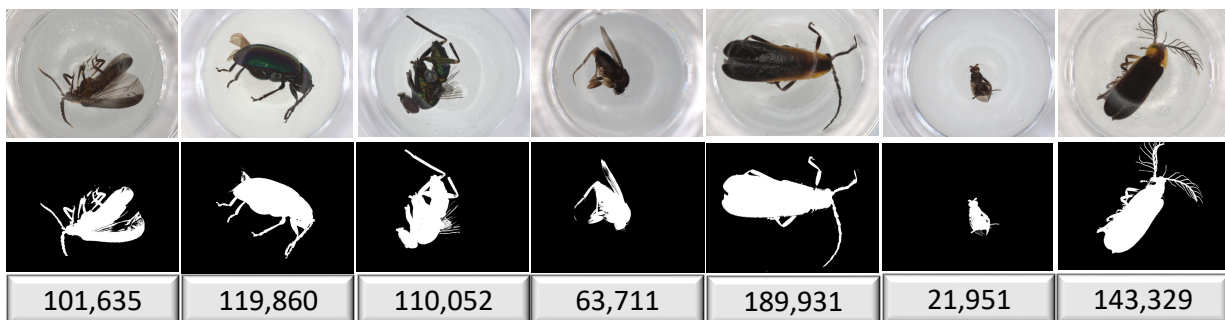


Figure 4.9: Examples of original images of the BIOSCAN-5M dataset, along with their respective total number of pixels (size) that occupy the image. The top row shows original images and the bottom row shows masks. Figure produced for and taken from [2].

biotic and abiotic factors affecting their lives, including food access, nutrition, and climate change [222]. For instance, in humans, limited access to nutrition correlates with a decrease in average height over generations [223], reflecting environmental and economic changes.

**Pixel count.** The raw dataset provides information about each organism’s size by quantifying the total number of pixels occupied by the organism. This information is provided in the `image_measurement_value` field. Since the image capture settings are consistent for all images, irrespective of scale, as indicated by the organism’s distance to the camera, the number of pixels occupied by the organism should approximate its size. Less than 1% of samples of the BIOSCAN-5M dataset do not have this information.

To provide a clearer understanding of the content in the `image_measurement_value` field, Figure 4.9 displays examples of original images along with their corresponding masks, highlighting the total number of pixels occupied by an organism. To determine the real-world size of the organism based on the number of pixels, it is also important to have the pixel to metric scaling factor. For the original full-sized images, most of the images are captured using a Keyence imaging system with a known pixel to millimetre scaling.

**Challenges.** Two challenges associated with the BIOSCAN-5M dataset include a sampling bias as a result of the locations where and the methods through which specimens are

Table 4.7: Statistics and purpose of BIOSCAN-5M data partitions.

| Species set     | Split                | Purpose                       | # Samples | # Barcodes | # Species |
|-----------------|----------------------|-------------------------------|-----------|------------|-----------|
| unknown<br>seen | <b>pretrain</b>      | self- and semi-sup. training  | 4,677,756 | 2,284,232  | —         |
|                 | <b>train</b>         | supervision; retrieval keys   | 289,203   | 118,051    | 11,846    |
|                 | <b>val</b>           | model dev; retrieval queries  | 14,757    | 6,588      | 3,378     |
|                 | <b>test</b>          | final eval; retrieval queries | 39,373    | 18,362     | 3,483     |
| unseen          | <b>key_unseen</b>    | retrieval keys                | 36,465    | 12,166     | 914       |
|                 | <b>val_unseen</b>    | model dev; retrieval queries  | 8,819     | 2,442      | 903       |
|                 | <b>test_unseen</b>   | final eval; retrieval queries | 7,887     | 3,401      | 880       |
| heldout         | <b>other_heldout</b> | novelty detector training     | 76,590    | 41,250     | 9,862     |

collected, and that the number of labelled records sharply declines at deeper taxonomic levels, especially beyond the family rank, which makes fine-grained classification tasks more challenging.

### 4.3.2 Data Partitioning

In real-world insect biodiversity monitoring, it is common to encounter both species which are already known to science, and samples whose species is novel. Thus, to excel in biodiversity monitoring, a model must correctly categorize instances of known species, and identify novel species outside the existing taxonomy. Two settings are established for this data: closed-world and open-world. In the closed-world setting, the task is to accurately identify species from a predefined set of existing labels. In the open-world setting, the task is to group together samples of novel species.

**Species sets.** Records are first partitioned based on their species label into one of four categories, with all samples bearing the same species label being placed in the same species set. *Seen*: all samples whose species label is an established scientific name of a species. *Unseen*: labelled with an established scientific name for the genus, and a uniquely identifying placeholder name for the species. *Heldout*: labelled with a placeholder genus and species name. *Unknown*: samples without a species label (note: these may truly belong in any of the other three categories).

**Splits.** Using the above species sets, four partitions are established (Table 4.7). The *unknown* samples are all placed into a **pretrain** split for use in self-supervised pretraining and/or semi-supervised learning. As some DNA barcodes are common to multiple samples, for *seen* and *unseen* sets, records are split by placing all samples with the same barcode in the same partition, to ensure there is no repetition of barcodes across splits.

For the closed-world setting, the *seen* records are used to establish **train**, **val**, **test** splits. To ensure that the **test** set is not too imbalanced in species distribution, samples are placed in the **test** set with a flattened distribution. Records are sampled from species with at least two unique barcodes and eight samples, and the number of samples placed in the **test** set scales linearly with the total number of samples for the species, until reaching a cap of 25 samples. Five percent of the remaining *seen* data are sampled to form the **val** partition, but in this case, they match the imbalance of the overall dataset. The remaining samples then form the **train** split, with a final split distribution of 84.2:4.3:11.5 (again, balancing the need for much training data while also having sufficient validation and testing data). Following standard practice, the **val** set is for model evaluation during development and hyperparameter tuning, and the **test** set is for final evaluation. In the retrieval setting, the **train** split should additionally be used as a database of *keys* to retrieve over, and the **val** and **test** split as queries.

For the open-world scenario, a similar procedure is used to establish **val\_unseen** and **test\_unseen** over the *unseen* records. After creating **test\_unseen** with the same methodology as **test**, 20% of remaining *unseen* species records are sampled to create **val\_unseen**. The remaining *unseen* species samples form the **keys\_unseen** set. In the retrieval setting, **keys\_unseen** is used to form the database of *keys* to retrieve, and the **val\_unseen** and **test\_unseen** splits act as queries. The *heldout* species samples form a final **other\_heldout** partition. As these species are in neither *seen* nor *unseen*, this split can be used to train a novelty detector without using any *unseen* species.

## 4.4 Chapter Conclusion

The curation and publication of these insect datasets (see the [Statement of Contributions](#)) represents both a meaningful step towards reaching the research goal, but also a significant contribution to the fields of conservation biology and image classification, as the availability of the BIOSCAN datasets will spur interest from other researchers in the field, leading to further innovation. At the time of their publication, the BIOSCAN-1M and BIOSCAN-5M datasets were the largest publicly available insect datasets with all images annotated.

Looking ahead, iBOL’s ongoing efforts will lead to further advancements in several aspects. The rate of insect sample collection is already increasing, resulting in future datasets that are not only larger in terms of the number of records but also more comprehensive, with additional taxa at lower taxonomic levels such as genera and species. Moreover, the dataset will expand to encompass diverse life forms beyond insects (some non-insects are already present in BIOSCAN-5M). While a more comprehensive dataset would be useful, making the most of the data that is currently available has become the focus for the BIOSCAN AI/ML team, of which the author is a part. One direction forward is addressing the presence of label errors within the data through the design and application of robust classification models and robust loss functions. These techniques, developed in [Chapter 5](#), may lead to both more accurate classifiers and the detection of mislabelled samples. Another direction is in the exploration of model architectures that are better suited to learning embeddings of the hierarchy present in the insect data, such as Hyperbolic Neural Networks, studied in comparison to standard Euclidean networks in [Chapter 6](#), and applied to a multimodal retrieval problem on the BIOSCAN data in [Chapter 7](#). Thus, curating and publishing the BIOSCAN dataset has been an important enabler of future research and it represents just the beginning of what lies ahead.

# Chapter 5

## Improving Robustness to Label Errors<sup>1</sup>

Section 2.5 gives background on the issues associated with label errors in training data, label error detection, and on existing work towards robust loss functions for training on mislabelled data. Briefly, in supervised learning problems, a set of labelled data, known as training data, are required to optimize / train the model [100, 101]. Deep neural networks, including convolutional neural networks (CNNs) [70, 72, 82, 224], consist of layers of interconnected artificial neurons with associated weights which must be optimized in order to train the model. Machine learning engineers normally assume that the “ground truth” training data are labelled *correctly*; however, this is not necessarily the case, and in fact is often *not* the case! Indeed, in many benchmark datasets, label errors are present in rates of up to roughly 5% [163], for example, in MNIST [167], CIFAR [168], and ImageNet [169].

To address the presence of label errors in benchmark datasets, in 2021, Northcutt *et al.* [162, 163] developed a method for automatically detecting and correcting errors in

---

<sup>1</sup>Note that significant portions of the work presented in this chapter were carried out in collaboration with Nolen Zhao (Sections 5.1 and 5.2) and David Szczecina (Section 5.3), two Undergraduate Research Assistants. Components of Section 5.1 were originally prepared for a submission [3] to the 9<sup>th</sup> Annual Conference on Vision and Intelligent Systems (CVIS). Components of Section 5.3 were originally presented [4] at the 10<sup>th</sup> Annual Conference on Vision and Intelligent Systems (CVIS). See the Statement of Contributions for details.

training data, known as Confident Learning. In doing so, benchmark datasets including MNIST [167], CIFAR [168], ImageNet [169], and more were examined, and possible mislabelled examples were identified. Crowd-sourcing (Mechanical Turk) was then used both to verify which selected examples were indeed incorrectly labelled, and to propose a corrected label through consensus. The collection of samples identified being falsely labelled are available at [labelerrors.com](http://labelerrors.com) and provide a valuable resource for those in the field. While this work [163] provides one possible path forward in contending with label errors in training data, little is known about the behaviour and robustness of specific deep neural network architectures in terms of handling label errors.

In this chapter, the challenge of label errors is addressed by studying model sensitivity to mislabelled training data, comparing the robustness of different deep neural network architectures, and through the development of specialized loss functions robust to the presence of label errors. Firstly, a motivating, though simplified, overview of model sensitivity to label errors is given in Section 5.1. This demonstration of model robustness for a simple case motivates the further analysis of model robustness in terms of deep neural network architectures in Section 5.2. Finally, two robust loss functions are designed and analysed in Section 5.3, where they are used to improve the downstream detection of label errors. These approaches are presented separately; however, they may also be applied synergistically to improve overall model robustness and generalizability, as well as the detection (and even possible cleaning) of label errors in datasets.

## 5.1 Label Error Sensitivity

### 5.1.1 Preliminaries

The notation from the problem formulation given in Chapter 3 is extended here to fit the subject of Label Errors. Building upon [165, 166], this work is set in the context of a  $K$ -class classification problem based with  $\mathcal{X} \subset \mathbb{R}^r$  as the  $r$ -dimensional feature space (typically images) and  $\mathcal{Y} = \{1, \dots, K\}$  as the label space (categorical). A clean (ideal,

error-free) dataset is denoted by  $D = \{(x_i, y_i)\}_{i=1}^N$  for sets containing  $N$  examples and where  $(x_i, y_i) \in (\mathcal{X}, \mathcal{Y})$ . A classifier is a function,  $f : \mathcal{X} \rightarrow \mathcal{Y}$ , that maps input features to predicted classes by selecting the class,  $k$ , having the maximal predicted probability,  $f = \arg_k \max p(k|x)$ . The predicted probabilities,  $p(k|x)$ , for each class  $k \in \mathcal{Y}$ , are inferred through a trainable deep neural network (DNN) with a softmax output layer, such that  $\sum_{k=1}^K p(k|x) = 1$ . In practice, classifiers are trained through the strategy of attempting to minimize the empirical risk,  $R_{\mathcal{L}} = \frac{1}{N} \sum_{i=1}^N \mathcal{L}(p(k|x_i), y_i)$  for a given loss function  $\mathcal{L} : [0, 1]^K \times \mathcal{Y} \rightarrow \mathbb{R}^+$ .

Label errors are often present within training data, as discussed in [Section 5.3.1](#). The proportion of data that are mislabelled is termed the corruption rate,  $\eta \in [0, 1]$ , and the corrupted dataset is denoted  $D_\eta = \{(x_i, \tilde{y}_i)\}_{i=1}^N$  for possibly erroneous labels  $\tilde{y}_i$ . The probability that a correct label  $y$  is mislabelled (corrupted) as label  $\hat{y}$  is  $\eta_{y\hat{y}}$ , such that the corruption rate of specific label  $y$  is given by  $\eta_y = \sum_{\hat{y} \neq y} \eta_{y\hat{y}}$ . Taking the common assumption that the label corruption (i.e., mislabelling) process is conditionally independent of input features [\[225\]](#),  $x$ , labels are given by

$$\tilde{y} = \begin{cases} y & \text{with probability } (1 - \eta_y) \\ \hat{y}, \hat{y} \in \mathcal{Y} \setminus \{y\}, & \text{with probability } \eta_{y\hat{y}} \end{cases}. \quad (5.1)$$

Corruption processes may be categorized based on their nature. In the most general case of *asymmetric* corruption, the corruption process is class-dependent and conditioned on both correct labels,  $y$ , and corrupted labels,  $\hat{y}$ . Corruption is *symmetric* if the process is *identically* conditioned on both correct and incorrect labels. Lastly, the corruption process is *uniform* if it is not conditioned on either correct or incorrect labels.

For the sake of notational simplicity throughout the remainder of this chapter, predicted probabilities,  $p(k|x)$ , are stated in terms of an *implied* input  $x$ , such that the notational simplification is made, from  $p(k|x)$  to  $p_k$  (indicating predicted probability for class  $k$ ).

### 5.1.2 Introduction

The challenges associated with mislabelled training data are discussed both in the introduction to this chapter and in [Section 2.5.1](#). In cases where training data label errors exist, one must ask the question of how model performance ought to be evaluated, and what it means to achieve a particular percentage accuracy when some (likely unknown) fraction of labels are incorrect. The research in this section seeks to create a framework through which to study the sensitivity of models to the presence of label errors in training data.

For illustration purposes, two versions of labelled data from a simple 2-class problem are pictured in the top row of [Figure 5.1](#). In the first column, the data contain outliers, and in the second column, there are label errors. Data point shape (circle *vs.* triangle) indicates the true class, and colour (red *vs.* blue) indicates the ground truth (training) label. Note that while outliers and mislabelled data may appear to be similar, the two arise from completely different causes and will impact classification models differently. Assuming data are clustered with high density surrounding some prototypical center point, outlier points are those that are far from their true class’s center, whereas mislabelled data may appear *anywhere* but actually are often (due to the assumption of high density) near their true class’s center. The presence of mislabelled data may lead to local classification error, especially in cases of overfitting. Indeed, a more *local* classifier, such as a nearest-neighbour scheme (shown in middle row of [Figure 5.1](#)), would be highly susceptible to overfitting and local errors, whereas a more *global* classifier, for example a 5-nearest-neighbour scheme (shown in bottom row of [Figure 5.1](#)), would be less susceptible to overfitting and local classification errors due to its reliance on the consensus of multiple training data points.

While simple nearest neighbour classification schemes may be easy to envision and intuitively understood for simple problems such as that of [Figure 5.1](#), the behaviours of deep-neural-network-based classifiers on real-world problems are not. This section studies the impact of having mislabelled training data by artificially corrupting the training data from a familiar benchmark dataset, MNIST [\[167\]](#), and then training and evaluating a simple CNN model. By setting the corruption rate, evaluations of model overfitting are made in a very controlled environment. Techniques shown here are re-applied in [Section 5.2](#)

Table 5.1: Hyperparameters used for experiments.

| Parameter     | Setting           |
|---------------|-------------------|
| Loss function | Cross-Entropy     |
| Optimizer     | SGD with momentum |
| Learning rate | 0.01              |
| Momentum      | 0.9               |
| Batch-Size    | 64                |
| Num. Epochs   | 12                |

to compare the robustness of several deep-neural-network model architecture *types* to the presence of training label errors.

### 5.1.3 Framework for Evaluating Model Robustness

Demonstrational experiments are conducted upon the MNIST dataset, known to have a *very* low error rate (0.15%) [163] due to its simplicity. To evaluate the impact of having increased error rates on model accuracy, the training partition of the dataset is artificially corrupted. Data are re-labelled according to a specified *corruption rate*,  $\eta \in [0, 1]$ . Whether any given example is re-labelled is determined randomly according to whether a random number drawn (from a uniform distribution) is less than  $\eta$ . In this manner, over large quantities of data, the proportion of re-labelled data approximates  $\eta$ . Note that if selected, an example’s label is *necessarily* changed, i.e., made incorrect (different than its original label). Throughout all experiments, model and training hyperparameters are set according to values specified in Table 5.1. To keep experiments simple, a minimalistic model based on an introductory example from PyTorch [226] capable of achieving  $> 99\%$  accuracy on the MNIST dataset was selected. The model used is a CNN consisting of two convolutional layers, followed by max pooling, dropout, a fully connected layer, dropout, and a final fully connected layer. In total, the model has only 1.2 M trainable parameters.

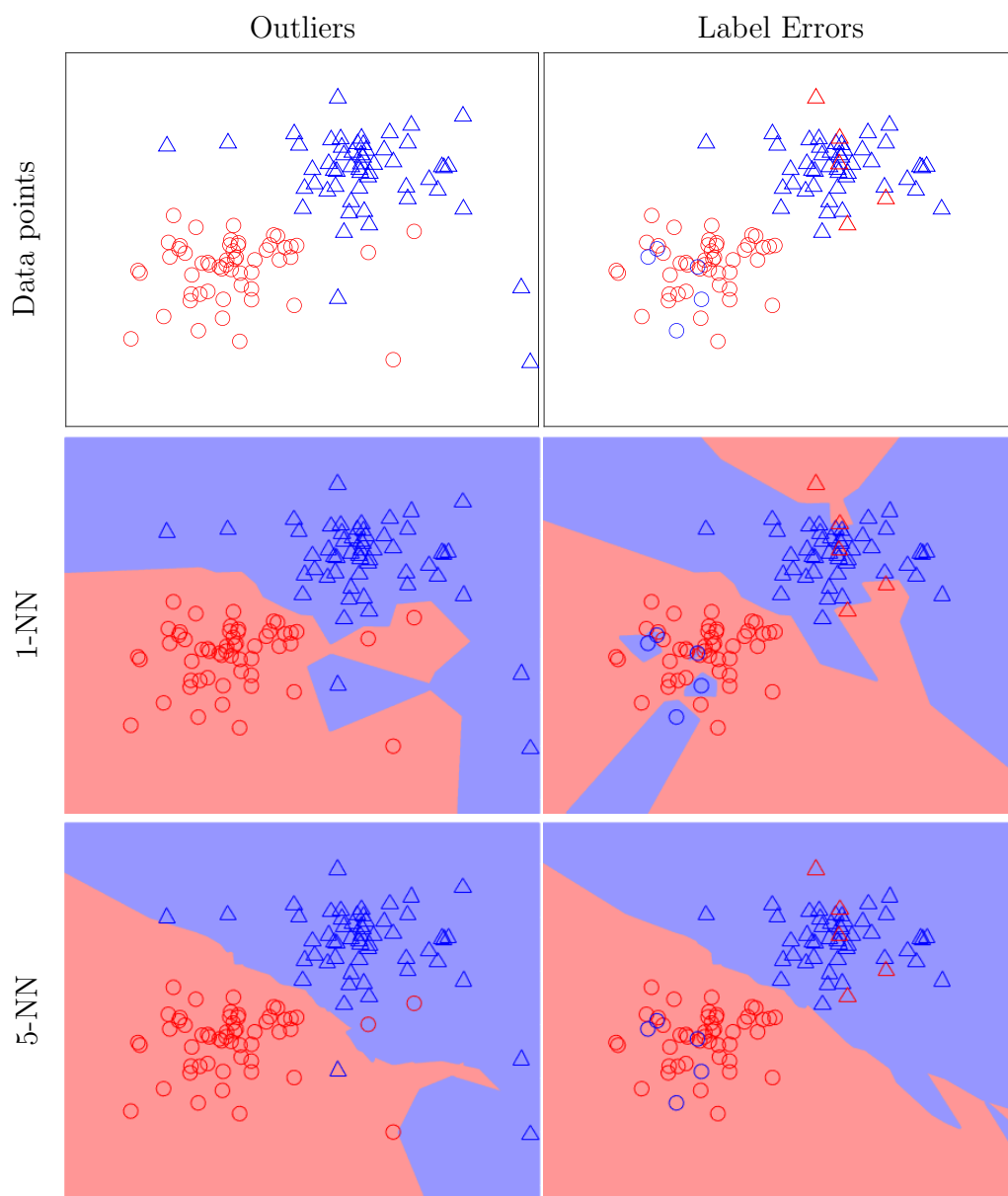


Figure 5.1: Training data may contain both outliers and label errors. The two columns include versions of a 2-class dataset: one with outliers and the other with label errors. The first row shows the data points, while the subsequent rows show nearest-neighbour (1-NN) and 5-nearest-neighbour (5-NN) classification regions. Data point shape (circle *vs.* triangle) indicates *true* class and colour (red *vs.* blue) indicates ground truth *label*. Outliers and mislabelled data may appear to be similar, but arise from completely different causes.

Firstly, the model validation accuracy is examined as a function of training data corruption rate, shown in [Figure 5.2](#). Observe that accuracy remains approximately steady and high (over 95%!) until a corruption rate of approximately  $\eta = 0.9$ , where an abrupt downward change occurs, before settling-out once again. This finding is quite remarkable, given that the model continues to be accurate even when *most* training data are mislabelled! The abrupt change seems to correspond to the transition point at which for any class, the number of labels indicating the *correct* class equals the number of labels for any other, *incorrect*, class, which would cause the model to learn to randomly guess. Before this point, the model still tends to learn the correct class-label association, and performs quite well. After this point, *fewer* training samples of each given class are labelled correctly than *all other* classes, and the model learns to *not* estimate the correct class, fitting to the mislabelled data, and performs poorly during testing. In terms of the number of classes within the dataset ( $K = 10$ , for MNIST), the relationship determining the location of this catastrophic change in model behaviour appears to be  $\eta' = 1 - 1/K$ .

To verify the relationship between the location of the abrupt change and the number of classes, a similar experiment over which the number of classes is artificially reduced is conducted. Here, accuracy results for the original 10-class problem are shown alongside those of a 6-class and 2-class problem. In each case, for the general  $K$ -class problem, examples from the first  $K$  classes of MNIST are retained, omitting the remainder. [Figure 5.3](#) shows the results of this experiment which indeed confirm that the point at which the catastrophic change occurs is related to the number of classes through  $\eta' = 1 - 1/K$ . Note that this relationship only holds in the case of *balanced* datasets, where all classes have equal numbers of samples.

To gain further insight, training and testing loss are examined in [Figure 5.4](#). Notice that while both losses do increase with increasing corruption rate, the testing loss remains below the training loss until a cross-over point at  $\eta = 0.9$ , demonstrating for this range of corruption rates that the model performs better during testing than it does during training and is able to generalize quite well (i.e., not overfit) in spite of large quantities of mislabelled data. At  $\eta = 0.9$ , the cross-over point, for each true class, there are approximately equal numbers of training samples labelled as all ten classes, and the model learns to randomly

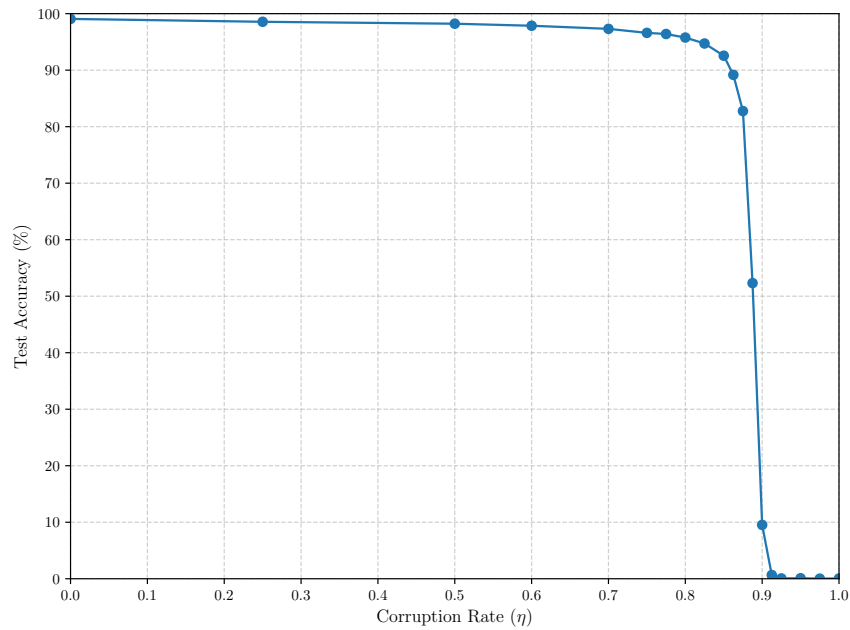


Figure 5.2: Model accuracy as a function of training data corruption rate, for MNIST, a balanced 10-class dataset. Accuracy remains remarkably high even when *most* training data are mislabelled! Until the corruption rate nears 0.9 ( $= 1 - 1/10$ , for a balanced 10-class dataset), model performance is hardly affected. Beyond this point, there becomes fewer labels of the *correct* class than any other, *incorrect*, class, and accuracy plummets towards zero.

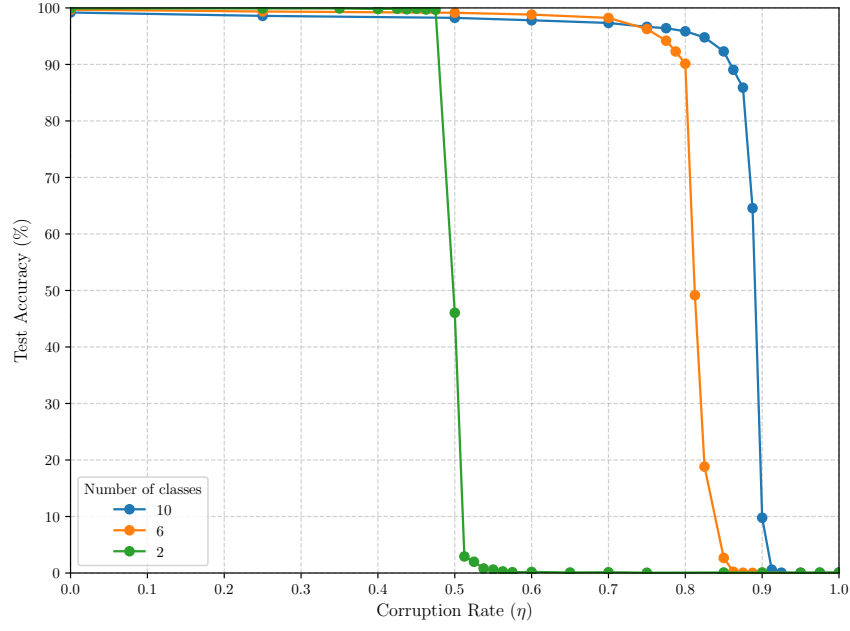


Figure 5.3: Similar to the accuracy *vs.* corruption rate plot of Figure 5.2, model accuracy is evaluated for a 10-class, 6-class, and 2-class problem. For the 10-class problem, the abrupt change occurs at a corruption rate of approximately 0.9, whereas for the 6-class problem, the abrupt change occurs at only 0.83, and for the 2-class problem, already at 0.5. This location follows a trend specified by  $\eta' = 1 - 1/K$ , where  $K$  is the number of classes.

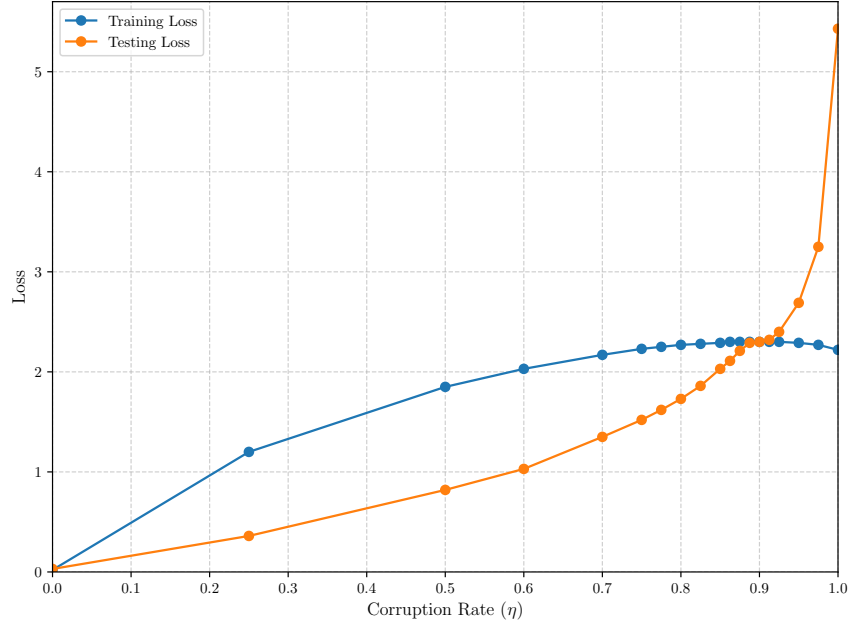


Figure 5.4: Training and testing loss as a function of corruption rate. Observe the cross-over point at  $\eta = 0.9$ , whereby testing loss begins to exceed training loss. Training loss tends to plateau as false-labels tend towards being fully uncorrelated and then anti-correlated with the data itself, i.e., random but *not* correct. Testing loss initially is below training loss, as the model is still able to partially learn the correct class-label relationships (given that most data are still correctly labelled); however, beyond the cross-over point, most data are *not* labelled correctly, the model learns to *not* estimate the correct class, and testing loss spikes.

guess, thereby resulting in equal loss during training and testing. Beyond the cross-over point, *fewer* training samples of each given class are labelled correctly than *all other* classes, the model learns to *not* estimate the correct class (i.e., has overfit to mislabelled data), training loss plateaus, and testing loss spikes.

#### 5.1.4 Discussion

In Figure 5.2, accuracy tapers quite gradually for modest (i.e., realistic) corruption rates, for example  $0.05 < \eta < 0.3$ . This insensitivity to corruption rate indicates that the

model is able to generalize well, and may be a feature useful as a point of comparison between classifier types (neural networks, KNN [227], SVM [228], *etc.*) or neural network architectures (MLP [229], CNN, ResNet [85], *etc.*). Models for which accuracy decreases at a greater rate would have a greater tendency to overfit and would generalize more poorly than those models for which accuracy decreases more gradually.

Comparing loss with accuracy, in Figure 5.4, loss increases gradually with corruption rate, when meanwhile in Figure 5.2, the accuracy is almost invariant to corruption rate until a point at which there is a catastrophic and large change. This behaviour in accuracy seems to contradict what is seen in the loss:

*Why is it that loss changes by only a small amount (specifically surrounding the  $\eta = 0.9$  point) when meanwhile accuracy rapidly plummets from near 100% to near 0%?*

This is a result of how inference is performed and how cross-entropy loss is defined. The model outputs (after running through SoftMax) a set of predicted class probabilities,  $\{p_k\}_{k \in [1,10]}$ . The class,  $k$ , with the highest predicted probability is selected as the inferred class for a given input, i.e.,

$$\text{predicted class} = y^* = \arg_k \max p_k. \quad (5.2)$$

So long as the predicted probability for the correct class,  $y$ , is slightly higher than that of all others  $p_k$ ,  $k \neq y$ , the network will infer the correct class. As corruption rates increase towards  $\eta'$ , fewer and fewer samples are correctly labelled, and the predicted class probabilities tend towards that of a uniform random distribution. Just prior to  $\eta'$ , the amount of correctly labelled data slightly exceeds the amount of incorrectly labelled data for each label, the predicted class probability for the correct class,  $p_y$ , generally slightly exceeds that of all others, and the model tends to still correctly classify testing data correctly. However, cross-entropy loss computes the negative natural log of the predicted correct class probability,  $p_y = p(y|x)$ , averaged over all samples in a batch of size  $M$ ,

$$J_{\text{CE}} = -\frac{1}{M} \sum_{i=1}^M \ln(p(y_i|x_i)). \quad (5.3)$$

Notice that for the 10-class example,  $-\ln(0.1) \approx 2.3026$ , which is almost exactly the loss seen at the cross-over point, at  $\eta' = 0.9$ . The negative log of the predicted correct class probability,  $-\ln(p_y)$ , is smooth and does not exhibit large change surrounding the point  $p_y = 0.1$ , whereas the highly non-linear class selection method of Equation (5.2), which simply selects the class with highest predicted probability, abruptly changes as  $p_y$  decreases below 0.1, leading to a near instantaneous loss in accuracy.

While it is totally unrealistic to assume that models are being trained with data having error rates towards  $\eta' = 1 - 1/K$  in practice, the resulting observed trends in accuracy *vs.* corruption rate do reveal a great deal about the *robustness* of a particular model to the presence of label errors. Having robustness to mislabelled data indicates that a model is better able to generalize, and not overfit to mislabelled data.

### 5.1.5 Conclusions of Study on Label Error Sensitivity

This study investigated the impacts of the presence of label errors in training data on model accuracy, and training and testing loss. A simple CNN model was used, with data artificially corrupted in the MNIST dataset. Remarkably, the model continued to perform with high accuracy (over 95%) even when *most* training data was mislabelled! While cases of data with large error rates are highly unlikely in practice, a further investigation of which model architectures tend to generalize better and can be used to avoid overfitting to mislabelled data would be useful both for the BIOSCAN program and the ML field as a whole. In Section 5.2, this investigation is conducted, whereby a variety of models (architectures) and classification problems (datasets) are explored.

## 5.2 Robust Model Architectures

Following on Section 5.1, where a method was developed for evaluating the sensitivity of a given model to the presence of label errors within training data, here, a variety of model architectures are evaluated and compared, such that insights can be drawn related to which

attributes or types of architectures lead to improved robustness. Motivation for this study comes from the desire to select a suitably robust architecture for training on the BIOSCAN insect data, where label errors are present. By analyzing many architecture types over a range of data corruption rates and data sets, overall trends may be observed such that one could understand the general trade-offs between model architectures and either select or construct a model for the task of learning to correctly classify insects in spite of errors being present in the training data.

A search for the types of general deep neural network architectures that are resistant to the presence of label errors is desired. Past research [230, 231] has shown that simpler models, at least classically, tend to be more generalizable (i.e., maintain their level performance on previously unseen data). This study explores models of varying *types*, rather than complexity. The scope is limited to classification-based tasks, leaving regression-based tasks to further research.

### 5.2.1 Methods

Building on [Section 5.1](#), this section extends that study by exploring a variety of model architectures on a wider range of datasets. Data are randomly corrupted to simulate the natural presence of mislabelled data within training datasets. For simplicity, the corruption process is uniform, such that the label transition probability is equal between classes and samples of all classes have equal probability of being corrupted. Models are then trained on the corrupted data, over a range of corruption rates,  $\eta$ , before being evaluated in terms of test accuracy, such that comparisons of robustness between architectures can be made. Models exhibiting higher accuracy scores at higher corruption rates are generally considered to be more robust to label errors in training data than models that suffer lower accuracy at the same corruption rates. To ensure fairness, models are kept similar in size to each other, such that the number of learnable parameters is roughly equal. Further details of the model architectures and datasets used are given below.

## Model Architectures

A wide variety of models were explored, as the goal of the research is to identify model attributes or architectures that lead to improved robustness to label errors. Model types explored include

- Multilayer-Perceptron [229] (MLP),
- Convolutional Neural Network [82, 224] (CNN; the same as is used in Section 5.1),
- Residual Neural Network [85] (Specifically ResNet-18),
- Vision Transformer [232] (ViT), and
- Data Efficient Image Transformer [233] (DEiT).

Additionally, the K-nearest neighbours [227] (KNN) algorithm was also implemented to act as a point of comparison, even though it is not a deep-neural-network model. Models in the above list are ordered in terms of date of creation, with simpler models being near the top and more complex architectures being lower. The MLP is the simplest form of deep neural network and has little associated inductive bias [234, 235]. CNNs build on MLPs by effectively sharing weights in convolution layers, creating a regularization effect whereby the associated inductive bias is increased [234, 235]. ResNets are an advanced form of CNNs incorporating residual (skip) connections, making training deeper networks possible while avoiding vanishing gradients [85]. The ResNet-18 model is deeper than the CNN model used here. Finally, the vision transformers take an entirely different approach than the MLP and CNNs, utilizing the attention mechanism and tokenizing image inputs [232]. The DEiT uses knowledge distillation from a teacher CNN during training to inherit some inductive bias from CNNs [233]. The explored models incorporate a wide range of features such a convolutional layers, skip connections, and tokenization. To ensure a fair comparison between model types, the number of learnable parameters (model degrees of freedom) was kept approximately constant at  $\sim 125\text{k}$ . Note that KNN predates all neural network architectures and is a classical method in machine learning. The ability to vary the number of nearest neighbours,  $K$ , allows KNN to be tuned to act as a highly local (small  $K$ ) or more global (high  $K$ ) classifier.

## Datasets

Building upon the work of [Section 5.1](#), where only the MNIST dataset was used, here, more complex datasets are explored. While MNIST includes only small grey-scale images, other datasets explored include larger images and those with three colour channels. Datasets used include

- MNIST [[167](#)],
- Fashion-MNIST [[236](#)],
- Kuzushiji-MNIST [[237](#)] (K-MNIST),
- CIFAR-10 [[168](#)], and
- CIFAR-100 [[168](#)].

These datasets are ordered in terms of increasing complexity (difficulty) while still maintaining the same format as MNIST in terms of number of classes (10), image dimensions ( $28 \times 28$  pixels), and number of images (60,000 training, 10,000 test), the exception being the CIFAR datasets, which are in colour (three channels), and have only 50,000 training and 10,000 testing images. A further exception is CIFAR-100, which has 100 classes. As the motivation for this research is based on the classification of insects, a complex task, results corresponding to the more complicated colour-image datasets are more directly relevant.

### 5.2.2 Results of Study on Robust Model Architectures

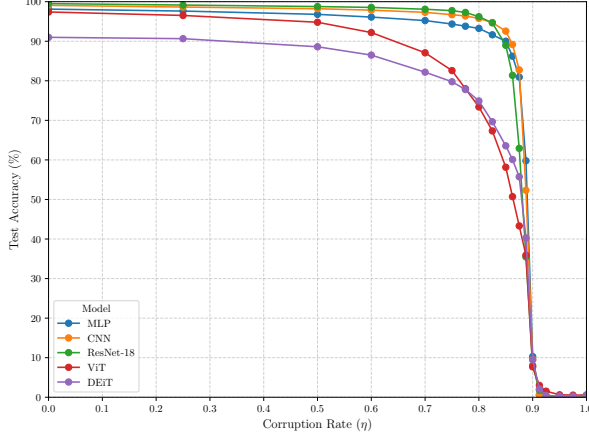
Building on [Figure 5.2](#) from [Section 5.1](#), which shows accuracy *vs.* corruption rate on the MNIST dataset for only a CNN model, [Figure 5.5](#) compares a variety of architectures, including MLP, CNN, ResNet-18, ViT, and DEiT, on corrupted versions of the 10-class datasets, MNIST, Fashion-MNIST, K-MNIST, and CIFAR-10 (ordered in terms of increasing complexity). Results are averaged over five trials. On the least complex datasets, all models perform fairly well and similarly to what was seen in the original results of [Figure 5.2](#), with the same sharp cut-off occurring at  $\eta' = 0.9$ , as expected, though there is a notable separation between the more conventional neural network architectures (MLP,

CNN, ResNet-18) and the transformer-based models (ViT, DEiT), with the transformer-based models suffering a greater loss in accuracy. This trend is exaggerated as problem difficulty / dataset complexity is increased (lack of easily separable classes), as expected, such that the gap widens between the two broad categories of models, and the MLP becomes more comparable to the transformer-based architectures.

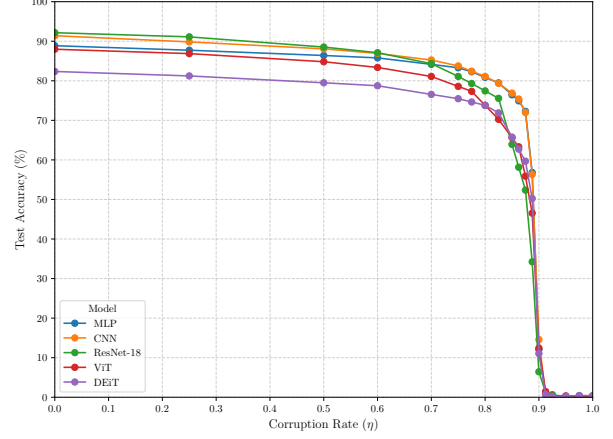
The accuracy for the transformer-based models is both lower at zero corruption ( $\eta = 0$ ) and tapers-off earlier (rounded trend line; less robust) than the more conventional architectures (which show a flatter trend with a sharp cut-off; more robust). This trend, whereby the conventional models have both greater zero-corruption accuracy and less sensitivity to increasing corruption rate, is hypothesized to be a result of the amount of inductive bias built into the models. The CNN and ResNet-18 are both convolutional neural network architectures, and are known to have high inductive bias and are able to learn on reduced quantities of data, whereas the MLP is less constrained (no weight sharing in convolutional filters), and the transformer-based models have even less inductive bias and require high amounts of data to train [234, 235].

Building upon the previous experiments for the 10-class datasets (MNIST, Fashion-MNIST, K-MNIST, and CIFAR-10), shown in Figure 5.5, accuracy *vs.* corruption rate is evaluated on the 100-class CIFAR-100 dataset, with results plotted in Figure 5.6. Note that for CIFAR-100, there are 100 classes, and thus the expected threshold, as was found in Figure 5.3, is at  $\eta' = 0.99$ . The results of this experiment align with that of 10-class data, where the convolutional architectures exhibit more robustness (greater accuracy sustained over greater corruption rates) than the MLP (which further drops in performance and robustness) and the transformer-based models.

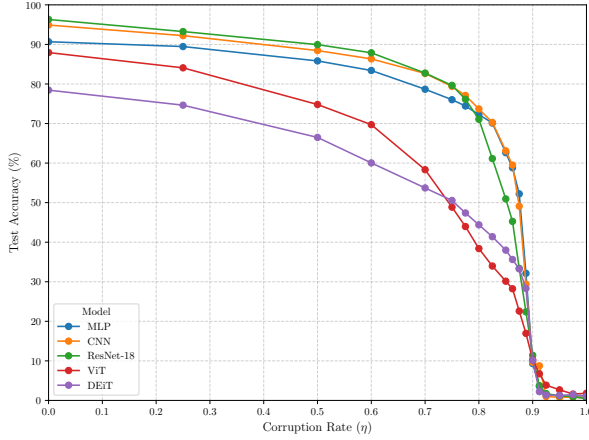
Because the CIFAR-100 dataset is much larger and represents a more difficult classification problem than the other datasets explored, an additional experiment was conducted whereby the number of training epochs was extended tenfold, from 12 epochs to 120 epochs, to ensure the models were not underfitting on this data as a result of having insufficient training time. Results from that experiment are almost indiscernible from those of the original 12 epoch experiment of Figure 5.6, indicating that the insufficient training time is not causing underfitting in this case, and that instead, it is likely that these models simply



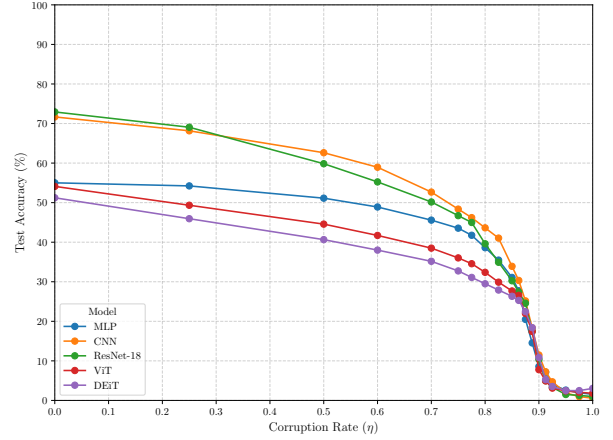
(a) MNIST



(b) Fashion-MNIST



(c) K-MNIST



(d) CIFAR-10

Figure 5.5: Accuracy *vs.* Corruption rate on the 10-class datasets MNIST, Fashion-MNIST, K-MNIST, and CIFAR-10, evaluated for model architectures MLP, CNN, ResNet-18, ViT, and DEiT. Results averaged over five trials. The same  $\eta' = 0.9$  cut-off is seen, expected for 10-class data. Observe a downward trend in model performance and a decrease in robustness (roundening of trend lines) with increased dataset complexity. The convolutional architectures (CNN and ResNet-18) tend to exhibit greater robustness (flatter trend lines) than the MLP and transformer-based models, which correlates with a decrease in the inductive bias present in the models.

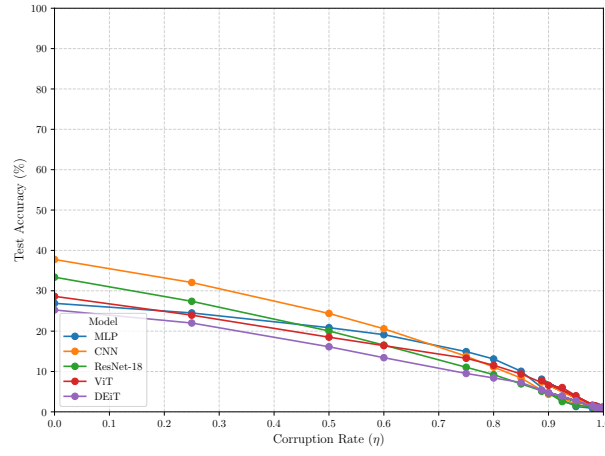


Figure 5.6: Extending the 10-class experiments of Figure 5.5, Accuracy *vs.* Corruption rate on the 100-class CIFAR-100 dataset is evaluated for model architectures MLP, CNN, ResNet-18, ViT, and DEiT. Results averaged over five trials. Observe that the zero-corruption accuracy is lowest on this dataset (most complex); however, the trend whereby the convolutional architectures (CNN, ResNet-18) exhibit greater robustness than the MLP and transformer-based architectures continues in these results and aligns with the 10-class experiments. As expected, the cut-off for this 100-class problem is at  $\eta' = 0.99$ .

have too few trainable parameters to adequately fit to this data. Later in this chapter, in [Section 5.3.3](#), a ResNet-34 [\[85\]](#) model is trained on CIFAR-10 and CIFAR-100 datasets (for the purpose of evaluating robust *loss functions*) and achieves 93.5% test accuracy on CIFAR-10 and 74.5% on CIFAR-100 with uncorrupted training data, indicating again that the limitation in the robustness experiments for CIFAR-100 ([Figure 5.6](#)) is a result of the models having too few trainable parameters.

To better observe the differences in architectures across datasets, the MLP, CNN and ViT were evaluated on all explored datasets, with results shown in [Figure 5.7](#). These architectures were selected as being representative of one convolutional architectures (CNN), the MLP, and a transformer-based architecture (ViT). Consistent with the above results of this section, [Figures 5.5](#) and [5.6](#), this view of the experimental results more clearly demonstrate model robustness as a function of dataset complexity. Indeed, the CNN has the least change in performance between datasets (smaller decrease in accuracy, similarly flat trends, indicating more consistent robustness), whereas the MLP and ViT change far more (large decrease in accuracy, rounding of trends, indicating less consistent robustness). While the accuracy of the MLP is substantially reduced, the trend lines remain fairly similar in shape across datasets, whereas the trends for the ViT progressively become more round (less robust) with increasing dataset complexity. These results further confirm that convolutional architectures tend to be more robust than the MLP and transformer-based alternatives.

As a point of comparison to the deep neural network models, a set of KNN classifiers is used to explore the impacts of having a more local *vs.* global classifier, controlled by the number of nearest neighbours,  $K$ , used for classification. To establish a set of reasonable models,  $K$  is selected to range from  $K = 1$  (maximally local) to  $K = 6000$  (highly global), since MNIST-type datasets explored in this section have 6000 training samples per class and extending the range beyond this point would certainly not improve accuracy.  $K$  is sampled to be approximately log-spaced, with the lower end of the range set to include  $K = 1$  and 3 before increasing logarithmically to  $K = 6000$ .

KNN results showing Accuracy *vs.* Corruption rate are given in [Figure 5.8](#), and Accuracy *vs.*  $K$  in [Figure 5.9](#). The most local classifier,  $K = 1$ , results in a linear trend from

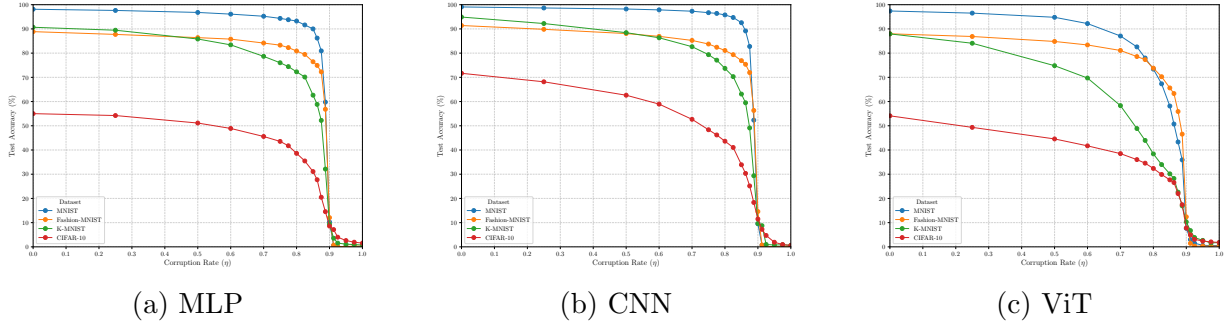


Figure 5.7: Accuracy *vs.* Corruption rate evaluated across 10-class datasets (MNIST, Fashion-MNIST, K-MNIST, and CIFAR-10), for the MLP, CNN, and ViT architecture types. Results averaged over five trials. Observe that the trends for the CNN change least (indicating high robustness), whereas the trends show greater decreases in accuracy for the MLP and ViT model (indicating lower robustness). The ViT model especially shows a rounding effect with increased dataset complexity, further indicating a lack of robustness.

near-perfect accuracy at zero-corruption to zero accuracy at total corruption ( $\eta = 1$ ). This matches intuition, whereby the nearest neighbour to any given test data point has a probability of being corrupted of exactly  $\eta$  (meaning the probability of incorrect classification of the test point is also  $\eta$ ). Therefore, averaged over the test set, this results in a negative linear trend with  $\eta$ . In the case of the more global classifiers, large  $K$ , trends become increasingly flat for low corruption rates and with increasingly sharp cut-offs, indicating high robustness. For the most global classifiers, underfitting occurs, and the zero-corruption accuracy is diminished. Therefore, there is a trade-off between robustness and baseline accuracy. Note that in cases of corruption beyond  $\eta' = 0.9$ , all KNN classifiers have accuracy lower than 0.1, which is worse than random guessing. This is consistent with the deep neural network models from earlier in this section.

From Figure 5.9, showing Accuracy *vs.*  $K$ , a trend emerges whereby the optimal selection of  $K$  (that which leads to best accuracy) increases with corruption rate. Although the most global of the classifiers (large  $K$ ) never reach the high accuracy that is possible with more local classifiers (small  $K$ ) under the ideal conditions of low corruption, only these global classifiers are capable of achieving high accuracy when corruption rates are high.

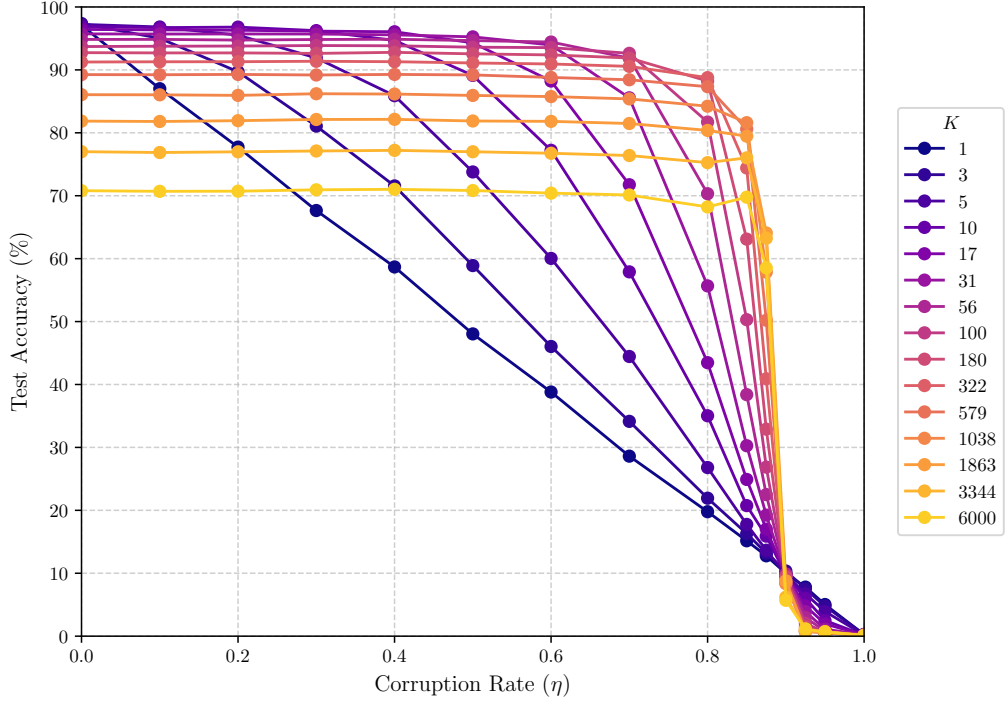


Figure 5.8: Accuracy *vs.* Corruption rate on the MNIST dataset, evaluated for KNN classifiers for various  $K$ . Observe continuous variation in trend line shape between those of local (low  $K$ ) classifiers to more global classifiers (high  $K$ ). The accuracy trend line of the most local classifier ( $K = 1$ ) is linear with  $\eta$  and the model is not robust, whereas the more global classifiers (high  $K$ ) tend to have more consistent performance with increasing corruption and are therefore more robust.

These findings may be useful for selecting a model type (balance of local *vs.* global) in cases where the corruption rate,  $\eta$ , is approximately known: If corruption is low, then a more local classifier would learn best from the data, indicated by high classification scores, whereas if corruption is quite high, only a more global classifier will be robust enough to learn from the highly mislabelled data and be able to produce reasonable classification results.

The KNN experiments of Figures 5.8 and 5.9 for the MNIST dataset are repeated in Figure 5.10 for the Fashion-MNIST dataset, to verify that the observed trends and

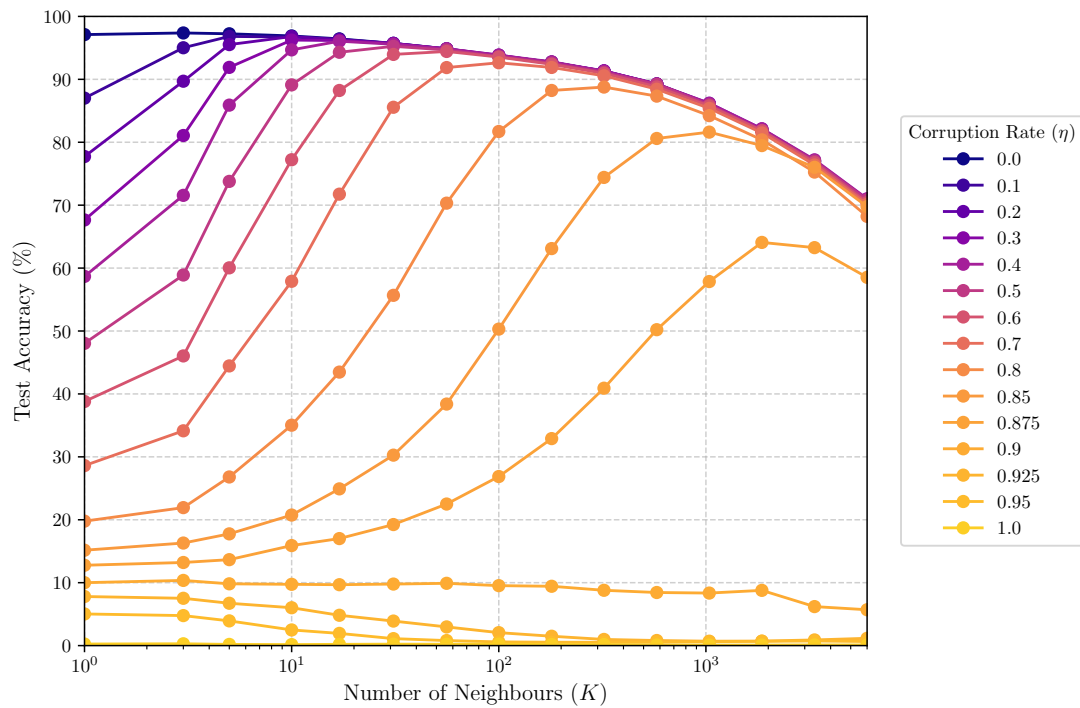
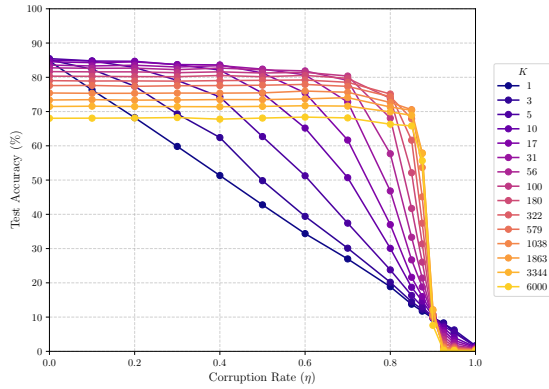
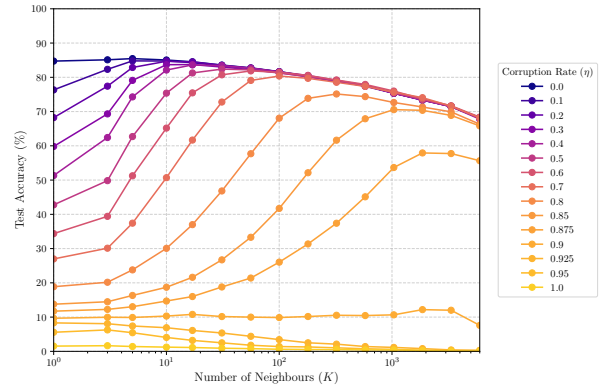


Figure 5.9: Accuracy *vs.*  $K$  for KNN classifiers on the MNIST dataset, evaluated for various corruption rates. Observe the trend in maximal accuracy with increasing  $\eta$ . As corruption increases, an increasingly global classifier (increasing  $K$ ) is required to achieve maximal accuracy, indicating the global classifiers are more robust to label errors than their local counterparts.



(a) Accuracy *vs.* Corruption rate



(b) Accuracy *vs.*  $K$

Figure 5.10: Building on the MNIST results of Figures 5.8 and 5.9, Accuracy *vs.* Corruption rate and Accuracy *vs.*  $K$  are plotted for the Fashion-MNIST dataset, evaluated for KNN classifiers for various  $K$  and various corruption rates. These are nearly identical as those from the MNIST data; however, across all models, the accuracy is reduced, consistent with the earlier neural network experiments on these datasets.

conclusions are consistent across multiple datasets. Indeed, the trends from the Fashion-MNIST experiments are nearly identical as those from the MNIST data; however, across all models, the accuracy is reduced, consistent with the earlier neural network experiments on these datasets.

### 5.2.3 Conclusions of Study on Robust Model Architectures

In this section, experiments were conducted to investigate the robustness of various model architectures. Specifically, the robustness to label errors was studied across several datasets, with artificially corrupted labels at various corruption rates,  $\eta$ . Five types of neural network architectures were investigated, including a MLP, two convolutional, and two transformer-based architectures. Results indicate that the convolutional architectures tend to be more robust than the MLP and transformer-based architectures, characterized by having accuracy *vs.* corruption rate trend lines that are fairly flat across early corruption rates and have a steep cut-off. An additional experiment using KNN classifiers explored the relationship

between a classifier’s global-*vs.*-local nature and its robustness to label errors. The KNN experiments indicate that more global classifiers tend to be more robust, but also underfit in cases of low corruption, diminishing their baseline accuracy, so a trade-off exists. More local classifiers learn best from data with low corruption rates, whereas only more global classifiers are robust enough to learn from highly mislabelled data. These KNN findings may be applied in practice as a guide when selecting neural network architectures (selecting a balance of local *vs.* global) in cases where the corruption rate,  $\eta$ , is approximately known.

## 5.3 Robust Loss Functions

The preceding two sections, [Sections 5.1](#) and [5.3](#), study model robustness to label errors. This section addresses the challenge of detecting such errors by developing robust loss functions that enhance model robustness and downstream label error detection performance.

### 5.3.1 Introduction

Existing work on detecting label errors [[162](#), [164](#), [172–174](#)] such as Confident Learning (CL) [[162](#)] and Area Under Margin (AUM) [[164](#)] typically rely on training surrogate models, intended to be robust or resistant to overfitting, such that they are well-generalized and not fit to erroneous labels in the training data. Crucially, the effectiveness of these methods depends on the models producing statistically distinguishable predicted probabilities,  $p(k|x)$ , for erroneously *vs.* correctly labelled samples. However, when trained with standard loss functions such as Cross Entropy [[175](#)] or Focal Loss [[99](#)], models tend to inadvertently fit to erroneous labels, reducing detection effectiveness. Modification of the loss function used when training such models is one avenue for improving their robustness and downstream utility for detecting label errors.

This section introduces and rigorously investigates two novel robust loss functions — *Blurry Loss* and *Piecewise-zero Loss* — specifically designed to improve model robustness

by explicitly de-weighting samples likely to be mislabelled. In datasets with label errors, erroneous samples are likely to be difficult-to-classify. Inspired by Focal loss [99], which places emphasis on difficult-to-classify samples, these novel loss functions reverse this idea, assigning less weight to samples with low predicted probability in their as-labelled class (likely to be mislabelled), thereby avoiding detrimental overfitting to erroneous data. These loss functions result in a more robust training scheme and models that are better suited to detecting label errors than those trained with existing standard or robust loss functions.

Comprehensive experiments are performed to validate these loss functions’ effectiveness in improving label error detection, exploring several artificially corrupted datasets, a range of artificial corruption rates, uniform and non-uniform error statistics, multiple frameworks for label error detection (CL and AUM), and gradients and predicted probability distributions of corrupted and clean training samples. In particular, the results of Figures 5.13 and 5.14, showing trends in gradients and predicted probability distributions during training, demonstrate that the proposed loss functions operate as intended to improve robustness to mislabelled data. The proposed Blurry and Piecewise-zero loss functions outperform existing loss functions, Generalized Cross Entropy [165] (GCE) and Active-Negative Loss [166] (ANL), across nearly all experiments, highlighting their utility for practical error detection and correction.

### 5.3.2 Method

Two novel loss functions are introduced that are designed to be robust to label errors. During training on corrupt data,  $D_\eta = \{(x_i, \tilde{y}_i)\}_{i=1}^N$ , mislabelled samples  $(x, \hat{y})$  are likely difficult to classify and thus have low predicted probabilities,  $p_{\tilde{y}} = p_{\hat{y}}$ , for their as-labelled class,  $\hat{y}$ . With CE loss [175], having low predicted probability,  $p_{\tilde{y}}$ , means the *gradient* of the loss is *large and negative*, imparting a strong signal to the optimizer causing fitting to these samples. The proposed loss functions achieve their robustness by reducing the loss for samples likely to be mislabelled, either resulting in a *positive* or *zero* gradient and no longer incorrectly steering the optimizer towards fitting to these erroneous samples. This

mechanism is experimentally verified in the ablation studies of [Figures 5.13](#) and [5.14](#) in [Section 5.3.4](#).

The proposed loss functions are the main contribution to the field; however, equally important is the thorough comparative evaluation against other state-of-the-art losses and ablation studies used to provide a deeper understanding of their mechanisms. These experiments involve the use of these loss functions within existing label error detection frameworks, Confident Learning (CL) and Area Under the Margin (introduced in [Section 2.5.2](#)), to correctly identify mislabelled data in artificially corrupted datasets, and compare the effectiveness resulting from the use of these proposed loss functions *vs.* existing state-of-the-art robust loss functions. The proposed loss functions are outlined here, whereas the experimental details are outlined in [Section 5.3.3](#).

## Blurry Loss

*Blurry Loss* is closely related to Focal loss [\[99\]](#) (described in [Equation \(2.6\)](#) of [Section 2.5.3](#)) but *de-weights* difficult-to-classify samples, likely to be mislabelled, through a multiplicative factor. For predicted probability  $p_k$ , as-labelled class  $\tilde{y}$ , and a weighting parameter  $\gamma \in \mathbb{R}$ , Blurry Loss is defined as

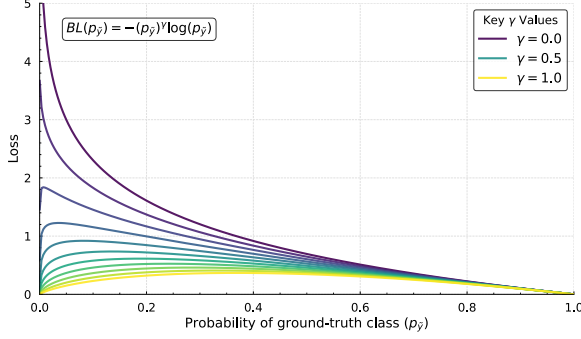
$$\text{BL}(p_k, \tilde{y}) = -(p_{\tilde{y}})^\gamma \log(p_{\tilde{y}}). \quad (5.4)$$

[Figure 5.11a](#) shows the this loss function for several variations of parameter  $\gamma$ .

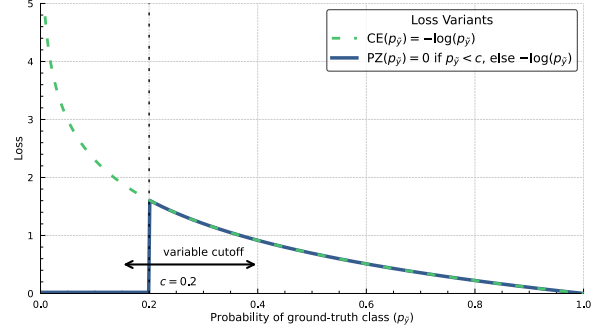
Notice that this function is non-monotonic and features a section of *positive* gradient for  $p_{\tilde{y}} < e^{-1/\gamma}$ , encouraging training *against* the as-labelled class if  $p_{\tilde{y}}$  is low. At  $\gamma = 0$ , Blurry Loss is equivalent to Cross Entropy Loss.

## Piecewise-zero Loss

*Piecewise-zero Loss* is a variant of CE loss [\[175\]](#) and is designed to ignore (zero) difficult-to-classify samples, which are likely to be mislabelled. Examples with predicted probability



(a) The Blurry Loss function of Equation (5.4), plotted for a range of parameter  $\gamma$ , controlling the shape. Note that at  $\gamma = 0$ , this is equivalent to CE.



(b) The Piecewise-zero Loss function of Equation (5.5), illustrating the effect of adjusting the cutoff position,  $c$ . Note that at  $c = 0$ , this is equivalent to CE.

Figure 5.11: The two proposed loss functions: Blurry Loss (a) and Piecewise-zero Loss (b).

beneath some cutoff,  $p_{\tilde{y}} \leq c \in [0, 1]$ , are assigned a loss of zero through a piecewise definition of the function. Also note that the *gradient* within the cutoff region is also zero, causing these examples with low  $p_{\tilde{y}}$  not to affect the training process (weights are not updated by zero-gradient samples). Piecewise-zero Loss is defined as

$$\text{PZ}(p_k, \tilde{y}) = \begin{cases} 0 & p_{\tilde{y}} \leq c, \\ \text{CE}(p_k, \tilde{y}) = -\log(p_{\tilde{y}}) & p_t > c. \end{cases} \quad (5.5)$$

Figure 5.11b shows this loss function for several variations of the cutoff-position parameter  $c$ . At  $c = 0$ , Piecewise-zero Loss is equivalent to Cross Entropy Loss.

## Loss Scheduling

DNN model parameters are typically randomly initialized at the start of training. As a result, the model's outputs at the early stages of training bear no meaningful relationship to the training labels. Applying the proposed loss functions immediately — especially the Piecewise-zero Loss with a high cutoff — is likely to lead to many correctly labelled samples being de-weighted or ignored, dramatically reducing the ability to train. Instead, a scheme

through which training begins with Cross Entropy loss is proposed, to allow the model to sufficiently generalize prior to the use of these novel loss functions. A delay parameter,  $d \in \mathbb{N}$ , determines the epoch at which the training switches to one of the proposed loss functions. With this scheduling approach, for example with PZ loss, the loss,  $\mathcal{L}_d$ , used is given by

$$\mathcal{L}_d(p_k, \tilde{y}) = \begin{cases} \text{CE}(p_k, \tilde{y}) & \text{Epoch} \leq d, \\ \text{PZ}(p_k, \tilde{y}) & \text{Epoch} > d. \end{cases} \quad (5.6)$$

### 5.3.3 Experiments

The following paragraphs outline the datasets, method of artificial corruption, loss functions used for comparison, models and training scheme, and metrics used for the experiments.

**Datasets and Artificial Corruption** Synthetic uniform label noise is injected into standard benchmark datasets following the same method used in [3, 165, 166, 225]. For a given corruption rate  $\eta \in [0, 1]$ , a fraction  $\eta$  of the training labels are randomly flipped. Corruption rates in  $\eta \in \{0.1, 0.2, 0.3, 0.4\}$  are tested. Datasets include MNIST [167], Fashion-MNIST [236], CIFAR-10, and CIFAR-100 [168], under the assumption that their existing label error rates are negligible compared to the induced corruption,  $\eta$ .

To assess robustness under realistic, non-uniform noise, the CIFAR-10N and CIFAR-100N [238] datasets are employed, which are versions of the CIFAR-10 and CIFAR-100 datasets, re-annotated using Amazon Mechanical Turk. By design, the re-annotations do *not* correct labels but rather induce abundant, real-world, non-uniform human annotation errors. The original CIFAR labels are considered ground truth in the CIFAR-N datasets. For CIFAR-10N, three re-annotations are given, whereas for CIFAR-100, only one is given (with  $\eta = 0.4020$ ). With CIFAR-10N, the ‘Aggregate’ version (labels used are the majority vote of the three re-annotations;  $\eta = 0.0903$ ) and the ‘Worst’ version (re-annotations *not* matching ground truth are always used when available;  $\eta = 0.4021$ ) are used for experiments.

**Baseline Loss Functions** Categorical Cross-Entropy [175] and Focal Loss [99] are the primary baseline loss functions, as these serve as the *de facto* gold standard for most supervised classification problems. Focal Loss is used with  $\gamma = 2$  as recommended in [99]. To assess robustness under label errors, state-of-the-art *robust* loss functions serve as a baseline. These include Generalized Cross-Entropy [165] (GCE) loss, defined in Equation (2.7), Active Negative Cross-Entropy [166] (ANL-CE) and Active Negative Focal Loss [166] (ANL-FL), defined in Equation (2.8).

Consistent with the original formulation, the GCE hyperparameter  $q$  is fixed at 0.7 as is recommended in [165]. For the ANL variants, the dataset-specific hyperparameters  $\alpha$ ,  $\beta$ , and  $\delta$ , as well as the focal-loss scaling factor  $\gamma$  for ANL-FL, are set exactly as prescribed in [166]. However, since the Fashion-MNIST dataset was not used in [166], no parameter specification was provided and thus the MNIST parameters of [166] are used for the Fashion-MNIST experiments here. Each loss function is evaluated using these recommended settings to ensure a fair comparison.

**Models and Hyperparameters** For MNIST and Fashion-MNIST, a minimal convolutional neural network is adopted, consisting of two convolutional layers followed by two fully connected layers interleaved with dropout, totalling approximately 1.2 million trainable parameters. Under a ten-epoch training schedule, this architecture reliably exceeds 99% test accuracy on clean data (near state-of-the-art) [239].

For CIFAR-10, CIFAR-100, and their noisy variants (CIFAR-10N/CIFAR-100N), an ImageNet-pretrained ResNet-34 [85], gently modified for lower resolution images (replaced its initial  $7 \times 7$ , stride-2 convolution with a  $3 \times 3$ , stride-1 convolution and removed the following max-pool layer, preserving spatial resolution on  $32 \times 32$  inputs), is fine-tuned. Training is performed for ten epochs under identical conditions using the Adam optimizer [77] with an initial learning rate of  $1 \times 10^{-4}$ ; a scheduler reduces the learning rate by a factor of 0.1 at epoch 5 for CIFAR-10 (factor 0.2 for CIFAR-100). Under this setup, the model attains 93.5% test accuracy on CIFAR-10 and 74.5% on CIFAR-100. All experiments utilize the Adam optimizer, leveraging its adaptive learning capabilities without requiring additional tuning.

To explore the parameter-sensitivity of the proposed losses, a grid search is performed over the Blurry Loss exponent,  $\gamma$ , and the Piecewise-Zero Loss cutoff,  $c$ , thereby characterizing their influence on label error detection efficacy and allowing optimal values to be identified for each problem. As demonstrated in ablation Figure 5.16 of Section 5.3.4, introducing a one-epoch delay yields the best performance for PZ loss; accordingly,  $d = 1$  is used for all subsequent PZ loss experiments.

**Label Error Detection and Performance Metrics** To evaluate each loss function’s efficacy for the detection of label errors, the Confident Learning (CL) [162] and AUM [164] frameworks, both described in Section 2.5.2, are applied. The CL pruning method used in all cases is “both”, in which detected samples must both be unlikely to be their as-labelled class and likely to be of another different class. Five cross-validation folds are used with 80% train / 20% predict splits. The use of AUM serves as an ablation study of label error detection, to demonstrate the efficacy of the proposed loss functions across multiple detection frameworks. Following the methodology of [164], experiments on CIFAR-100 are replicated, using a ResNet-34 [85] model trained for 150 epochs with a learning rate of 0.1 and weight decay of  $10^{-4}$ ; however, with Cross-Entropy (CE) [175] loss replaced with Blurry and Piecewise Zero Loss for a direct comparison.

Performance is primarily quantified using the F1 score [240] — a gold-standard metric defined as the harmonic mean of precision and recall. The F1 score penalizes situations in which either precision or recall fall, which is not always reflective of one’s goals when attempting to identify label errors. To address this limitation with low corruption rates, the Balanced Accuracy score [241], defined as the average of sensitivity (true positive rate) and specificity (true negative rate), which provides a more robust measure of detector performance under class imbalance, is also reported.

**Compute Resource Requirements** All experiments were conducted on a compute cluster using an NVIDIA Tesla V100-SXM2-16GB GPU. The main Confident Learning experiments required 6 GB of memory, and each loss function and parameter combination required on average 10 minutes (5 minutes for MNIST and Fashion-MNIST, 15 minutes

for CIFAR-10 and CIFAR-100). Experiments on the cleaned CIFAR-100 test set followed the same process, requiring only 3 GB memory. When additionally storing the per-epoch gradient information, for MNIST 3 GB of memory and 1 hour was required, and 6 GB of memory and 4 hours for CIFAR-100. The CIFAR-N experiments required 15 minutes and 6 GB of memory. The AUM experiments required 7 GB of memory as well as 2.5 hours for each loss function and parameter test conducted. The complete set of experiments required approximately 1800 hours of compute, with an additional  $\sim 400$  hours for preliminary experimentation.

The Confident Learning experiments were conducted using the Python library `cleanlab` v2.5.0, from <https://github.com/cleanlab/cleanlab>, as introduced in [162] under an AGPL-3.0 license. The AUM experiments were conducted using the python library `aum` v1.0.2, from <https://github.com/asappresearch/aum> as introduced in [164] under an MIT license.

### 5.3.4 Results of Study of Robust Loss Functions

Results for the main experiment and additional studies are presented here. All tables are formatted such that best scores are **bolded and underlined**, and second best are simply **bolded**. The first ablation study, shown in Figures 5.13 and 5.14, examining training dynamics through gradients and predicted probability distributions is a key result of this chapter and thesis and demonstrates that the proposed loss functions operate as intended.

#### Main Findings

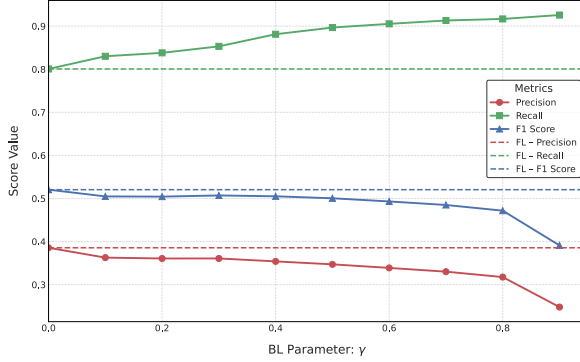
The proposed loss functions are compared against existing standard and robust loss functions, including CE, FL, GCE, ANL-CE, and ANL-PZ. This comparison is performed on several datasets and corruption rates, as described in Section 5.3.3. Results from 20 random trials are summarized in Table 5.2. The proposed loss functions are evaluated over a range of their parameters,  $\gamma$  and  $c$ ; however, only the best performing cases are shown

Table 5.2: Label error detections, averaged over 20 random trials, measured using the F1 score. The parameters ( $\gamma$  and  $c$ ) resulting in the best performance for each dataset and corruption rate are given for the proposed loss functions, Blurry Loss (BL) and Piecewise-zero Loss (PZ).

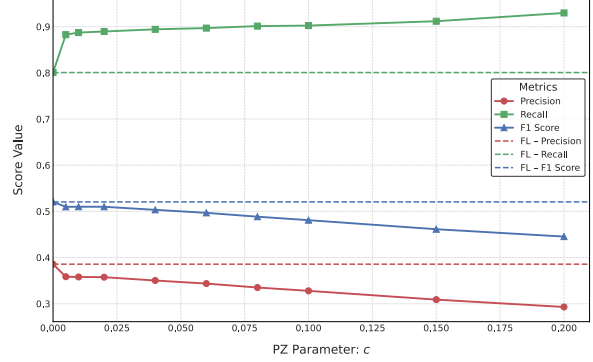
| Dataset       | $\eta$ | CE                | FL                                | GCE               | ANL-CE            | ANL-FL            | BL                                | $\gamma$ | PZ                                | $c$   |
|---------------|--------|-------------------|-----------------------------------|-------------------|-------------------|-------------------|-----------------------------------|----------|-----------------------------------|-------|
| MNIST         | 0.10   | 0.969 $\pm$ 0.002 | 0.960 $\pm$ 0.004                 | 0.957 $\pm$ 0.002 | 0.941 $\pm$ 0.002 | 0.940 $\pm$ 0.002 | <b>0.979<math>\pm</math>0.002</b> | 0.3      | <b>0.974<math>\pm</math>0.001</b> | 0.005 |
|               | 0.20   | 0.975 $\pm$ 0.001 | 0.967 $\pm$ 0.002                 | 0.975 $\pm$ 0.001 | 0.970 $\pm$ 0.001 | 0.969 $\pm$ 0.001 | <b>0.984<math>\pm</math>0.001</b> | 0.4      | <b>0.983<math>\pm</math>0.001</b> | 0.010 |
|               | 0.30   | 0.975 $\pm$ 0.001 | 0.967 $\pm$ 0.001                 | 0.983 $\pm$ 0.001 | 0.980 $\pm$ 0.001 | 0.980 $\pm$ 0.001 | <b>0.988<math>\pm</math>0.001</b> | 0.4      | <b>0.986<math>\pm</math>0.001</b> | 0.020 |
|               | 0.40   | 0.973 $\pm$ 0.001 | 0.963 $\pm$ 0.002                 | 0.987 $\pm$ 0.001 | 0.985 $\pm$ 0.000 | 0.985 $\pm$ 0.001 | <b>0.990<math>\pm</math>0.000</b> | 0.5      | <b>0.988<math>\pm</math>0.000</b> | 0.040 |
| Fashion-MNIST | 0.10   | 0.824 $\pm$ 0.003 | <b>0.831<math>\pm</math>0.004</b> | 0.737 $\pm$ 0.004 | 0.683 $\pm$ 0.003 | 0.679 $\pm$ 0.004 | <b>0.826<math>\pm</math>0.003</b> | 0.1      | 0.818 $\pm$ 0.003                 | 0.005 |
|               | 0.20   | 0.884 $\pm$ 0.002 | 0.879 $\pm$ 0.003                 | 0.850 $\pm$ 0.002 | 0.822 $\pm$ 0.002 | 0.819 $\pm$ 0.002 | <b>0.887<math>\pm</math>0.002</b> | 0.2      | <b>0.885<math>\pm</math>0.003</b> | 0.005 |
|               | 0.30   | 0.907 $\pm$ 0.002 | 0.899 $\pm$ 0.001                 | 0.900 $\pm$ 0.001 | 0.881 $\pm$ 0.002 | 0.879 $\pm$ 0.001 | <b>0.917<math>\pm</math>0.001</b> | 0.4      | <b>0.916<math>\pm</math>0.001</b> | 0.020 |
|               | 0.40   | 0.918 $\pm$ 0.002 | 0.896 $\pm$ 0.033                 | 0.927 $\pm$ 0.001 | 0.915 $\pm$ 0.001 | 0.914 $\pm$ 0.002 | <b>0.934<math>\pm</math>0.001</b> | 0.5      | <b>0.933<math>\pm</math>0.001</b> | 0.040 |
| CIFAR-10      | 0.10   | 0.740 $\pm$ 0.006 | 0.757 $\pm$ 0.006                 | 0.773 $\pm$ 0.006 | 0.698 $\pm$ 0.006 | 0.693 $\pm$ 0.007 | <b>0.787<math>\pm</math>0.005</b> | 0.5      | <b>0.794<math>\pm</math>0.008</b> | 0.040 |
|               | 0.20   | 0.773 $\pm$ 0.004 | 0.789 $\pm$ 0.004                 | 0.858 $\pm$ 0.004 | 0.821 $\pm$ 0.004 | 0.817 $\pm$ 0.005 | <b>0.860<math>\pm</math>0.004</b> | 0.6      | <b>0.863<math>\pm</math>0.004</b> | 0.060 |
|               | 0.30   | 0.771 $\pm$ 0.004 | 0.798 $\pm$ 0.003                 | 0.893 $\pm$ 0.003 | 0.873 $\pm$ 0.003 | 0.871 $\pm$ 0.003 | <b>0.894<math>\pm</math>0.003</b> | 0.7      | <b>0.894<math>\pm</math>0.003</b> | 0.080 |
|               | 0.40   | 0.769 $\pm$ 0.003 | 0.801 $\pm$ 0.003                 | 0.909 $\pm$ 0.003 | 0.901 $\pm$ 0.002 | 0.899 $\pm$ 0.003 | <b>0.912<math>\pm</math>0.003</b> | 0.8      | <b>0.912<math>\pm</math>0.003</b> | 0.100 |
| CIFAR-100     | 0.10   | 0.503 $\pm$ 0.009 | <b>0.520<math>\pm</math>0.009</b> | 0.485 $\pm$ 0.009 | 0.399 $\pm$ 0.007 | 0.397 $\pm$ 0.007 | 0.507 $\pm$ 0.009                 | 0.3      | <b>0.510<math>\pm</math>0.009</b> | 0.010 |
|               | 0.20   | 0.628 $\pm$ 0.006 | 0.638 $\pm$ 0.006                 | 0.652 $\pm$ 0.007 | 0.578 $\pm$ 0.006 | 0.575 $\pm$ 0.007 | <b>0.661<math>\pm</math>0.007</b> | 0.5      | <b>0.667<math>\pm</math>0.006</b> | 0.020 |
|               | 0.30   | 0.680 $\pm$ 0.006 | 0.688 $\pm$ 0.006                 | 0.742 $\pm$ 0.006 | 0.682 $\pm$ 0.006 | 0.680 $\pm$ 0.006 | <b>0.744<math>\pm</math>0.006</b> | 0.6      | <b>0.748<math>\pm</math>0.006</b> | 0.020 |
|               | 0.40   | 0.712 $\pm$ 0.004 | 0.718 $\pm$ 0.003                 | 0.800 $\pm$ 0.004 | 0.751 $\pm$ 0.004 | 0.746 $\pm$ 0.004 | <b>0.800<math>\pm</math>0.003</b> | 0.7      | <b>0.801<math>\pm</math>0.004</b> | 0.020 |

for brevity. In nearly all cases, the use of the proposed BL and PZ losses result in better performance than with the baseline loss functions.

At low corruption rates ( $\eta = 0.1$ ), FL occasionally performs well as consequence of it causing CL to detect more conservatively, favouring precision over recall compared to the proposed loss functions. At these low rates, the detection problem is highly imbalanced: high recall may be achieved while making few correct detections, but precision is dramatically reduced with relatively few false detections. In a context where detected examples are reviewed by experts, prioritizing recall over precision can be advantageous, as catching more errors generally outweighs the cost of incorrectly detecting a few more examples. Precision and recall (the two harmonic components of the F1 score) are explored for BL and PZ, and compared to FL in Figure 5.12, for the CIFAR-100 dataset at  $\eta = 0.1$ , a case selected since the F1 score of FL exceeds that of BL and PZ. Observe that by tuning the parameters of the proposed loss functions, recall is increased significantly, but at the cost of some precision. Because F1 is a harmonic mean, the small decrease in precision dominates the larger increase in recall, and the score is decreased. Again, this does not necessarily indicate less desirable behaviour at lower corruption rates, since having improved recall



(a) BL *vs.* FL, CIFAR-100,  $\eta = 0.1$ .



(b) PZ *vs.* FL, CIFAR-100,  $\eta = 0.1$ .

Figure 5.12: FL (dotted lines) results in better F1 scores than BL (a) or PZ (b) (solid lines) at  $\eta = 0.1$  on the CIFAR-100 dataset. This is a result of conservative detection, whereby fewer samples are detected and precision is maintained. Through tuning the parameters of the proposed loss functions, performance can be made fairly similar to that of FL, or recall can be increased dramatically while only marginally reducing precision, which may be considered worthwhile at low corruption rates.

may be worthwhile at the expense of having relatively few false-positives. To capture a more reasonable objective at lower corruption rates, Balanced Accuracy (mean of sensitivity and specificity) is reported in Table 5.3, in which the proposed BL and PZ perform strongly throughout.

In the main experiment of Tables 5.2 and 5.3, the corrupted datasets include pre-existing, unaccounted for label errors. While the usage of several datasets demonstrated the broad applicability and efficacy of the proposed loss functions, an additional experiment on a dataset without pre-existing label errors would enable a more precise measure of label error detection performance. To achieve this, a version of CIFAR-100 with test samples having human-verified label errors removed [163] is used. The cleaned test set is artificially corrupted in the same manner as in the main experiment, and the Confident Learning error detection framework is once again applied; however, rather than using a k-fold approach over the entire dataset, model training is performed strictly on the non-cleaned training set, and evaluation is performed on the cleaned (and artificially corrupted) test set. Results averaged over three seeds are shown in Table 5.4 and form a direct comparison to the

Table 5.3: As in Table 5.2, but here reporting the Balanced Accuracy metric. The proposed BL and PZ perform strongly throughout.

| Dataset       | $\eta$ | CE                | FL                | GCE                               | ANL-CE            | ANL-FL            | BL                                | $\gamma$ | PZ                                | $c$   |
|---------------|--------|-------------------|-------------------|-----------------------------------|-------------------|-------------------|-----------------------------------|----------|-----------------------------------|-------|
| MNIST         | 0.10   | 0.980 $\pm$ 0.002 | 0.971 $\pm$ 0.004 | 0.992 $\pm$ 0.001                 | 0.992 $\pm$ 0.000 | 0.992 $\pm$ 0.000 | <b>0.993<math>\pm</math>0.001</b> | 0.4      | <b>0.992<math>\pm</math>0.001</b> | 0.040 |
|               | 0.20   | 0.981 $\pm$ 0.001 | 0.974 $\pm$ 0.002 | 0.992 $\pm$ 0.000                 | 0.991 $\pm$ 0.000 | 0.991 $\pm$ 0.000 | <b>0.993<math>\pm</math>0.000</b> | 0.5      | <b>0.993<math>\pm</math>0.000</b> | 0.020 |
|               | 0.30   | 0.980 $\pm$ 0.001 | 0.971 $\pm$ 0.001 | 0.992 $\pm$ 0.000                 | 0.991 $\pm$ 0.000 | 0.990 $\pm$ 0.000 | <b>0.993<math>\pm</math>0.000</b> | 0.5      | <b>0.993<math>\pm</math>0.000</b> | 0.020 |
|               | 0.40   | 0.976 $\pm$ 0.001 | 0.966 $\pm$ 0.002 | 0.991 $\pm$ 0.000                 | 0.990 $\pm$ 0.000 | 0.989 $\pm$ 0.000 | <b>0.992<math>\pm</math>0.000</b> | 0.5      | <b>0.991<math>\pm</math>0.000</b> | 0.040 |
| Fashion-MNIST | 0.10   | 0.942 $\pm$ 0.002 | 0.941 $\pm$ 0.002 | <b>0.949<math>\pm</math>0.002</b> | 0.943 $\pm$ 0.001 | 0.942 $\pm$ 0.001 | <b>0.951<math>\pm</math>0.001</b> | 0.5      | 0.948 $\pm$ 0.002                 | 0.080 |
|               | 0.20   | 0.943 $\pm$ 0.002 | 0.935 $\pm$ 0.002 | 0.949 $\pm$ 0.001                 | 0.942 $\pm$ 0.001 | 0.940 $\pm$ 0.001 | <b>0.953<math>\pm</math>0.001</b> | 0.4      | <b>0.950<math>\pm</math>0.001</b> | 0.040 |
|               | 0.30   | 0.939 $\pm$ 0.001 | 0.929 $\pm$ 0.001 | 0.948 $\pm$ 0.001                 | 0.939 $\pm$ 0.001 | 0.938 $\pm$ 0.001 | <b>0.953<math>\pm</math>0.001</b> | 0.5      | <b>0.952<math>\pm</math>0.001</b> | 0.040 |
|               | 0.40   | 0.932 $\pm$ 0.002 | 0.912 $\pm$ 0.027 | 0.946 $\pm$ 0.001                 | 0.937 $\pm$ 0.001 | 0.936 $\pm$ 0.001 | <b>0.950<math>\pm</math>0.001</b> | 0.5      | <b>0.949<math>\pm</math>0.001</b> | 0.040 |
| CIFAR-10      | 0.10   | 0.934 $\pm$ 0.002 | 0.931 $\pm$ 0.002 | 0.955 $\pm$ 0.002                 | 0.946 $\pm$ 0.002 | 0.945 $\pm$ 0.002 | <b>0.956<math>\pm</math>0.001</b> | 0.7      | <b>0.956<math>\pm</math>0.002</b> | 0.060 |
|               | 0.20   | 0.908 $\pm$ 0.002 | 0.912 $\pm$ 0.002 | 0.952 $\pm$ 0.003                 | 0.941 $\pm$ 0.002 | 0.940 $\pm$ 0.002 | <b>0.952<math>\pm</math>0.002</b> | 0.7      | <b>0.953<math>\pm</math>0.002</b> | 0.080 |
|               | 0.30   | 0.862 $\pm$ 0.003 | 0.879 $\pm$ 0.002 | 0.944 $\pm$ 0.002                 | 0.934 $\pm$ 0.002 | 0.933 $\pm$ 0.002 | <b>0.945<math>\pm</math>0.002</b> | 0.7      | <b>0.945<math>\pm</math>0.002</b> | 0.080 |
|               | 0.40   | 0.802 $\pm$ 0.003 | 0.834 $\pm$ 0.003 | 0.931 $\pm$ 0.002                 | 0.925 $\pm$ 0.002 | 0.923 $\pm$ 0.002 | <b>0.933<math>\pm</math>0.002</b> | 0.8      | <b>0.934<math>\pm</math>0.002</b> | 0.100 |
| CIFAR-100     | 0.10   | 0.833 $\pm$ 0.004 | 0.829 $\pm$ 0.004 | 0.853 $\pm$ 0.004                 | 0.827 $\pm$ 0.005 | 0.825 $\pm$ 0.005 | <b>0.855<math>\pm</math>0.004</b> | 0.5      | <b>0.856<math>\pm</math>0.004</b> | 0.020 |
|               | 0.20   | 0.815 $\pm$ 0.003 | 0.815 $\pm$ 0.003 | 0.850 $\pm$ 0.004                 | 0.815 $\pm$ 0.005 | 0.812 $\pm$ 0.005 | <b>0.852<math>\pm</math>0.004</b> | 0.6      | <b>0.854<math>\pm</math>0.003</b> | 0.020 |
|               | 0.30   | 0.785 $\pm$ 0.004 | 0.789 $\pm$ 0.005 | 0.844 $\pm$ 0.005                 | 0.800 $\pm$ 0.005 | 0.797 $\pm$ 0.006 | <b>0.844<math>\pm</math>0.004</b> | 0.7      | <b>0.846<math>\pm</math>0.005</b> | 0.020 |
|               | 0.40   | 0.746 $\pm$ 0.004 | 0.755 $\pm$ 0.003 | 0.833 $\pm$ 0.004                 | 0.779 $\pm$ 0.005 | 0.774 $\pm$ 0.005 | <b>0.834<math>\pm</math>0.003</b> | 0.7      | <b>0.834<math>\pm</math>0.004</b> | 0.020 |

Table 5.4: Label error detections on the CIFAR-100 cleaned test set, averaged over three random trials, measured using the F1 score ( $\pm\sigma$ ). Performance is improved relative to Table 5.2, especially at lower  $\eta$  and for PZ loss.

| $\eta$ | CE                | FL                                | BL                                                   | PZ                                                |
|--------|-------------------|-----------------------------------|------------------------------------------------------|---------------------------------------------------|
| 0.1    | 0.576 $\pm$ 0.027 | <b>0.587<math>\pm</math>0.030</b> | 0.580 $\pm$ 0.024 ( $\gamma = 0.3$ )                 | <b>0.590<math>\pm</math>0.020</b> ( $c = 0.020$ ) |
| 0.2    | 0.670 $\pm$ 0.013 | 0.671 $\pm$ 0.016                 | <b>0.693<math>\pm</math>0.013</b> ( $\gamma = 0.3$ ) | <b>0.717<math>\pm</math>0.017</b> ( $c = 0.015$ ) |
| 0.3    | 0.697 $\pm$ 0.011 | 0.709 $\pm$ 0.006                 | <b>0.775<math>\pm</math>0.011</b> ( $\gamma = 0.5$ ) | <b>0.780<math>\pm</math>0.013</b> ( $c = 0.020$ ) |
| 0.4    | 0.728 $\pm$ 0.005 | 0.729 $\pm$ 0.002                 | <b>0.824<math>\pm</math>0.003</b> ( $\gamma = 0.7$ ) | <b>0.830<math>\pm</math>0.007</b> ( $c = 0.025$ ) |

CIFAR-100 experiments shown in Table 5.2. The performance of all loss functions is improved, especially at lower corruption rates (improvement of nearly 0.1 in F1 score at  $\eta = 0.1$ ). This improvement at  $\eta = 0.1$  matches expectations given that the proportion of pre-existing label errors is more significant at lower artificial corruption rates. The most dramatic improvement is seen for PZ loss for all  $\eta$ , which performs best in all cases.

## Ablation Studies

**Training Dynamics** To verify that the proposed loss functions operate as intended, the training dynamics, observed through gradients (shown in Figure 5.13) and predicted probability distributions (shown in Figure 5.14) for corrupt and clean data are investigated.

These findings, confirming the method through which the proposed loss function operate, are important to this chapter and thesis as a whole.

To verify that the proposed loss functions affect gradients during training as they are designed to do, the gradient of loss,  $\partial\mathcal{L}/\partial p_{\hat{y}}$ , is monitored during training for corrupted and clean samples. Box and whisker plots in Figure 5.13 indicate the distribution of gradients for corrupt (red) and clean (green) data at each epoch during training. A detailed (reduced vertical axis) view is shown in the top row, while the full view is shown in the bottom row. CIFAR-100 with  $\eta = 0.4$  is used for this study. With Cross Entropy loss, the gradients of corrupt data are large and negative, which provides a strong signal to the optimizer to fit to these corrupt data (undesired, not robust), whereas with Blurry Loss, gradients of corrupt data are large and *positive*, steering away from these corrupt data, and with Piecewise-zero Loss, gradients of corrupt data are nearly all at zero, imparting minimal impact on training. Therefore, the proposed Blurry and Piecewise-zero Losses operate as intended by minimizing fitting to corrupt data.

Notice that with BL, the gradients for clean data during the first epochs are mostly *positive*, but quickly settle to a moderate *negative* value, associated with predicted probabilities close to 1.0 (as can be seen in Figure 5.14).

Building on Figure 5.13, additional box and whisker plots in Figure 5.14 indicate the distribution of predicted probabilities during training. With all loss functions, the predicted probabilities of corrupt data are less than those of clean data; however, the difference is far greater with the proposed losses. With Cross Entropy loss, the gradients of corrupt data are large and negative, providing a strong signal to the optimizer to fit to these corrupt data, whereas with Blurry Loss, gradients of corrupt data are mainly large but *positive*, steering the training away from these corrupt data, and with Piecewise-zero Loss, gradients of corrupt data are nearly all at zero, imparting minimal impact on training.

**Non-uniform Corruption** The main experiment included only *uniform* artificial corruption. To better understand the performance of the proposed loss functions under the presence of *non-uniform* label errors, a study involving the CIFAR-10N and CIFAR-

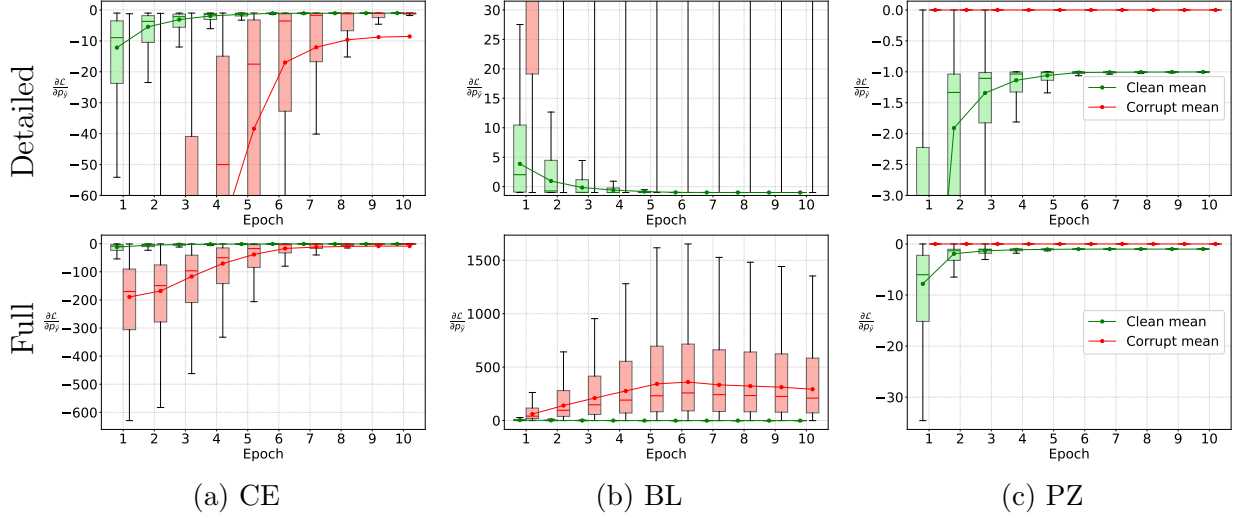


Figure 5.13: Gradient distributions per epoch, an important finding for this chapter and thesis, confirm the proposed BL and FZ loss operate as intended and minimize model fitting to corrupt data. Results are shown for CIFAR-100 at  $\eta = 0.4$ , with a detailed view in the top row and full vertical axes in the bottom row. Plots for corrupt data are shown in red and for clean data in green. Trimmed means (15th/85th percentiles) are used to avoid the influence of outliers, and 1.5 IQR whiskers are used. With CE, the gradients of corrupt data are large and negative, which provides a strong signal to the optimizer to fit to these corrupt data, whereas with BL, gradients of corrupt data are large and *positive*, steering away from these corrupt data, and with PZ, gradients of corrupt data are nearly all at zero, imparting minimal impact on training.

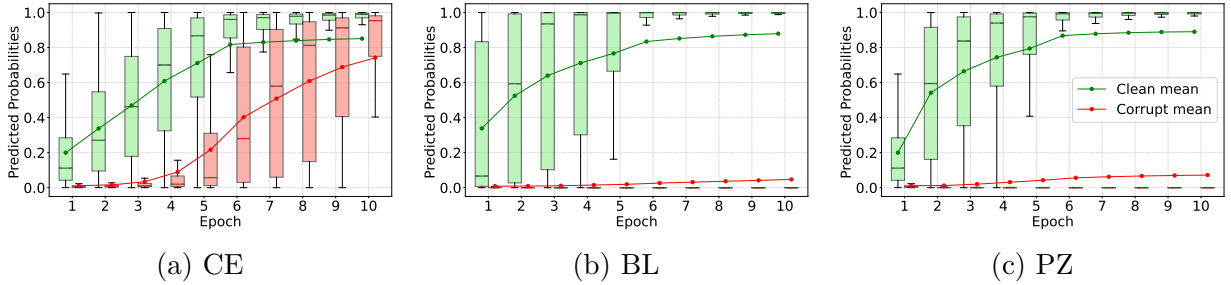


Figure 5.14: Predicted probability distributions per epoch, an important finding for this chapter and thesis, confirm the proposed BL and FZ loss operate as intended and better separate clean and corrupt samples. Results are shown for CIFAR-100 at  $\eta = 0.4$ . 1.5 IQR whiskers are used. With all loss functions, the predicted probabilities of corrupt data (red) are less than those of clean data (green); however, these differences are far greater with BL and PZ loss, leading to improved downstream detection of label errors.

Table 5.5: Label error detections on the CIFAR-10N and CIFAR-100N datasets with non-uniform corruption, averaged over five random trials, measured using the F1 score ( $\pm\sigma$ ). The proposed BL and PZ loss functions continue to exceed baseline performance, with non-uniform corruption.

| Dataset                        | CE                | Focal             | GCE               | ANL-CE            | ANL-FL            | BL                                  | $\gamma$ | PZ                                  | $c$  |
|--------------------------------|-------------------|-------------------|-------------------|-------------------|-------------------|-------------------------------------|----------|-------------------------------------|------|
| CIFAR-10N, $\eta \approx 0.1$  | $0.662 \pm 0.004$ | $0.634 \pm 0.002$ | $0.698 \pm 0.002$ | $0.652 \pm 0.006$ | $0.647 \pm 0.003$ | <b><math>0.698 \pm 0.004</math></b> | 0.7      | <b><math>0.704 \pm 0.006</math></b> | 0.08 |
| CIFAR-100N, $\eta \approx 0.4$ | $0.700 \pm 0.002$ | $0.684 \pm 0.002$ | $0.741 \pm 0.002$ | $0.721 \pm 0.002$ | $0.721 \pm 0.003$ | <b><math>0.742 \pm 0.003</math></b> | 0.8      | <b><math>0.745 \pm 0.003</math></b> | 0.10 |
| CIFAR-10N, $\eta \approx 0.4$  | $0.768 \pm 0.001$ | $0.769 \pm 0.002$ | $0.837 \pm 0.002$ | $0.865 \pm 0.001$ | $0.865 \pm 0.002$ | <b><math>0.867 \pm 0.001</math></b> | 0.9      | <b><math>0.866 \pm 0.001</math></b> | 0.15 |

100N [238] datasets with real-world human annotation errors is performed. This study follows the same format as the main experiment. Results averaged over five random seeds are shown in Table 5.5 against baseline loss functions, and for three variations of the CIFAR-N datasets, as described in Section 5.3.3. Rows are ordered with increasing corruption rate. Notice that the proposed loss functions are best (and second best) performing in all cases, and the optimal parameter setting tends to increase with corruption rate, congruent with the results of ablation Figure 5.15.

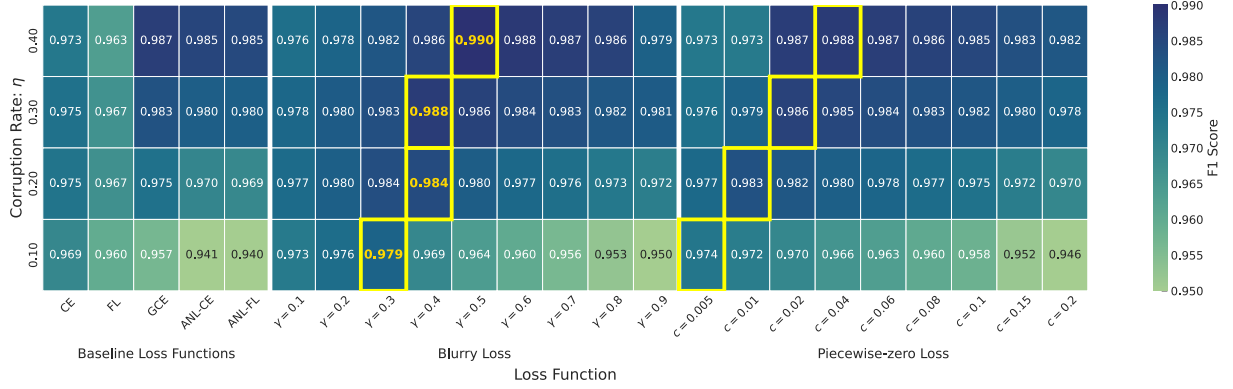
**Alternative Label Error Detection Framework** The main experiment used Confident Learning as the label error detection framework. To demonstrate the effectiveness and versatility of the proposed loss functions when used with other frameworks, an ablation

Table 5.6: Label error detections using the AUM framework and with the CIFAR-100 dataset, averaged over three random trials, measured using the F1 score ( $\pm\sigma$ ). The proposed losses continue to perform well using AUM, in addition to with CL (Tables 5.2 and 5.3).

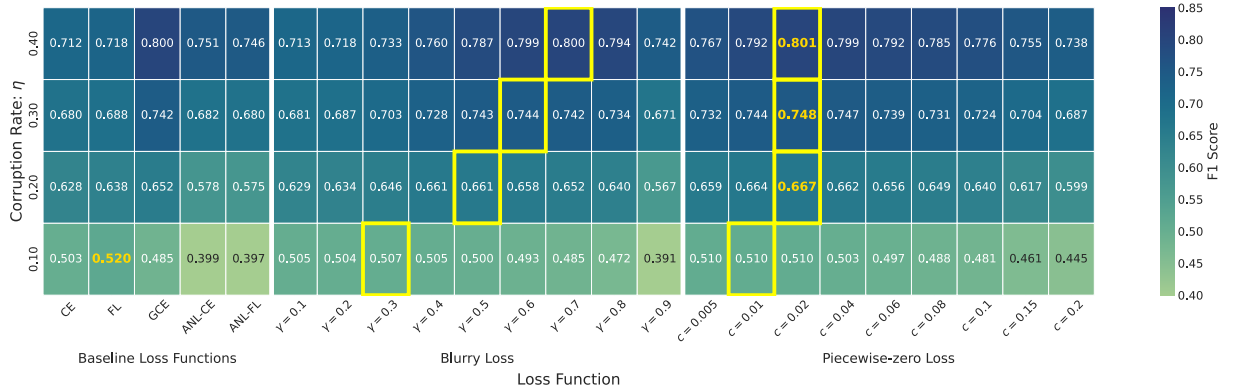
| $\eta$ | Loss Function                         |                                       |                                       |
|--------|---------------------------------------|---------------------------------------|---------------------------------------|
|        | CE                                    | BL ( $\gamma = 0.05$ )                | PZ ( $c = 0.025$ )                    |
| 0.1    | $0.7452 \pm 0.0071$                   | <b><math>0.7591 \pm 0.0180</math></b> | <b><math>0.8817 \pm 0.0074</math></b> |
| 0.2    | $0.8638 \pm 0.0042$                   | <b><math>0.8692 \pm 0.0032</math></b> | <b><math>0.8903 \pm 0.0085</math></b> |
| 0.3    | <b><math>0.9019 \pm 0.0021</math></b> | <b><math>0.9076 \pm 0.0031</math></b> | $0.8914 \pm 0.0044$                   |
| 0.4    | <b><math>0.9197 \pm 0.0024</math></b> | <b><math>0.9261 \pm 0.0005</math></b> | $0.8856 \pm 0.0125$                   |

study using the AUM [164] framework is performed on the CIFAR-100 dataset. Resulting label error detection F1 scores averaged over three random seeds are summarized in Table 5.6 and are directly comparable to the CIFAR-100 results of Table 5.2. The label error detection performance exceeds that of Confident Learning, and the proposed loss functions improve upon the baseline.

**Loss Function Parameters** To better understand the parameters of the proposed loss functions, an ablation study on these parameters is performed using a similar experimental setup as the main experiment. For each dataset and artificial corruption rate, the optimal parameter setting is identified. For these experiments, the delay,  $d$ , is set to 0 for BL and 1 for PZ loss. Results for MNIST and CIFAR-100 datasets are shown in Figure 5.15 as a heatmap of F1 score, along with a comparison to the baseline loss functions. For each corruption rate (row), the best performing configuration is indicated with bolded yellow text. To illustrate the trend in optimal parameter settings for the proposed loss functions, yellow boxes mark the best performing settings. Observe that optimal settings of both  $\gamma$  and cutoff,  $c$ , increase with increased corruption rate. This trend is consistent across datasets; however, more complicated / difficult datasets, such as CIFAR-100 relative to MNIST, seem also to demand greater settings of  $\gamma$  (but not  $c$ ). Note that larger settings for  $\gamma$  and  $c$  correspond to the proposed loss functions ‘ignoring’ a greater range of predicted probabilities.



(a) MNIST



(b) CIFAR-100

Figure 5.15: Heatmap showing variation in F1 score as loss function parameters are varied, for MNIST (a) and CIFAR-100 (b). In the left blocks, separated by a white line, results for baseline loss functions are shown. Two separate blocks are shown for the proposed loss functions, plotted over a range in their parameters. Best results are indicated with bolded yellow text. To illustrate the trend in optimal parameter settings for the proposed loss functions, yellow boxes mark the best performing parameter setting for each row. Note that delay,  $d$ , is set to 0 for BL and 1 for PZ loss.

**Delay** Delay,  $d$ , is introduced to allow the model to generalize prior to training with the proposed loss functions. This is especially important for PZ loss, which, at the initial stages of training, outright ‘ignores’ a large proportion of the dataset. To better understand how best to set the delay, an ablation study on this parameter is performed. Again, label errors are detected and performance is measured with F1 score, while  $d$  is varied. This study was performed across all the main experiment datasets and corruption rates, with fairly consistent results.

Results for the most complex dataset, CIFAR-100, are given in [Figure 5.16](#), for corruption rates ranging from  $\eta = 0.1$  to 0.4. Trends for various cutoff,  $c$ , settings are plotted. Observe that there is a large jump in F1 from  $d = 0$  to  $d = 1$  (for all  $\eta$ ), indicating that some delay is highly beneficial and allowing the model to somewhat generalize early in training is necessary for good application of PZ loss. Additionally, the best performance is achieved for  $d = 1$  (for an appropriate setting of  $c$ ), and F1 scores tend to towards the CE baseline (poorer performance) as delay is further increased. This indicates that having a greater number of epochs trained with PZ loss improves label error detection once the model is sufficiently generalized (one epoch with CE), and perhaps even that detrimental fitting to the erroneous data occurs with further training with CE. Furthermore, notice that the degree of downwards trending relates to the corruption rate, indicating that high corruption rates necessitate a greater need for training with PZ loss to achieve high label error detection performance.

Although not shown here, a similar preliminary study was performed for BL, which indicated that having no delay was always best, and thus delay is suggested only for PZ loss. This is likely a result of PZ loss being far more harsh than BL (outright ignoring of data rather than simply having positive gradients for samples with low predicted probabilities).

**Training Dynamics through AUM Statistic** As was explained in [Section 2.5.2](#), the AUM statistic [164] ([Equation \(2.4\)](#)) measures the difference between the logit values for a training example’s as-labelled class and its highest other class, averaged over the course of training. Correctly labelled samples tend to exhibit positive AUM values, while mislabelled samples tend to have significantly lower AUM values due to conflicting gradient updates

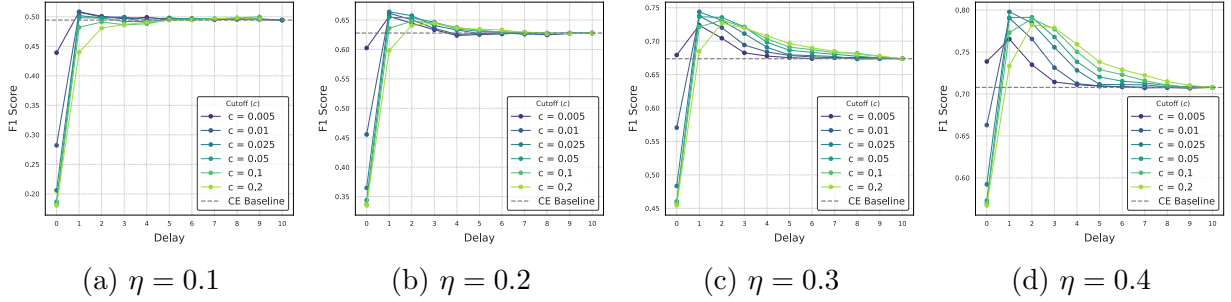


Figure 5.16: Detection F1 score *vs.* Delay,  $d$ , for PZ loss on CIFAR-100 at  $\eta$  ranging from 0.1 to 0.4 in panels (a) to (d). Observe that the best performance comes at  $d = 1$  in all cases, for an appropriate setting of the cutoff,  $c$ .

for samples of the same class. To better understand how the proposed loss functions affect the training dynamics on corrupt *vs.* clean data, the AUM is measured for all samples, and the AUM distributions for corrupt and clean data are compared. Figure 5.17 illustrates the AUM distributions for training with CE, BL, and PZ loss on the CIFAR-100 dataset with artificial corruption at  $\eta = 0.1$  (following the same method used throughout this section). The mean of each distribution is indicated using a dashed vertical line. To quantitatively understand the impacts of training using the proposed loss functions, two measures of dissimilarity in distributions are computed: Cohen’s  $d$  [242] (Effect size) and Wasserstein distance [243, 244], both of which are sensitive to differences in distribution position and width. Note that an ideal robust training scheme would result in a large difference in the AUM distributions of corrupt and clean samples.

Under CE training (Figure 5.17a), clean and corrupted samples exhibit distinct but moderately overlapping distributions. Although separation exists, the overlap reflects CE’s tendency to fit both clean and noisy labels, leading to partial memorization of mislabelled samples.

Training with Blurry Loss (Figure 5.17b), produced a markedly improved separation. Quantitatively, a Cohen’s  $d$  of 3.43 (*vs.* 3.20 for CE), and Wasserstein distance of 8.40 (*vs.* 6.11 for CE) were measured, indicating improved separation of corrupt and clean data. This increased separation of the distributions supports the intention and hypothesis that

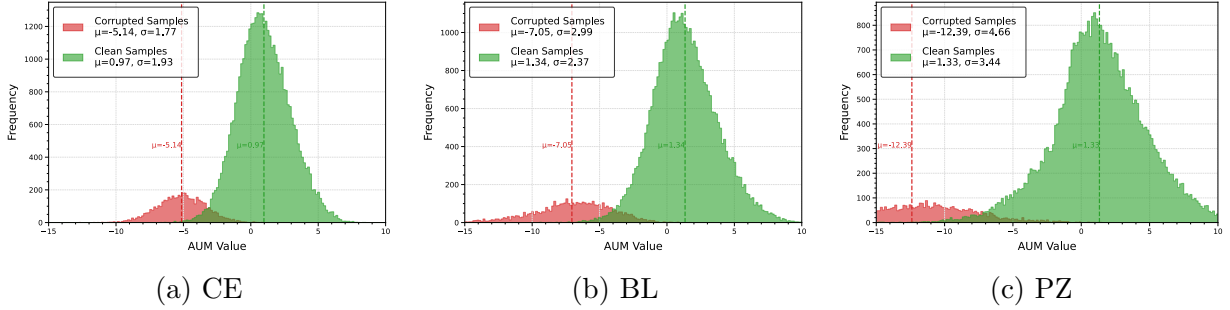


Figure 5.17: AUM distributions for artificially corrupted and clean samples in CIFAR-100 with corruption at  $\eta = 0.1$  for training with CE (a), BL (b), and PZ loss (c). Greater separation in distributions indicates a greater robustness of the loss function to mislabelled samples during training. Separation is measured using Cohen’s  $d$   $\{3.20 \text{ (CE)} < 3.43 \text{ (BL)} < 3.82 \text{ (PZ)}\}$  and Wasserstein distance  $\{6.11 \text{ (CE)} < 8.40 \text{ (BL)} < 13.72 \text{ (PZ)}\}$ .

BL de-weights hard-to-classify (likely mislabelled) samples, reducing the extent to which the model fits to label errors. Consequently, the discrimination between clean and noisy samples becomes more pronounced, enhancing the effectiveness of AUM-based label error detection.

The most dramatic effect was observed when using Piecewise-zero Loss (Figure 5.17c). Here, Cohen’s  $d$  increased to 3.82 (*vs.* 3.20 for CE), and the Wasserstein distance to 13.72 (*vs.* 6.11 for CE), indicating greater separation than both CE and BL. The sharp cutoff mechanism of PZ effectively prevents low-confidence samples from contributing to weight updates during training, almost eliminating their influence. As a result, mislabelled examples tend consistently to have highly negative AUM values, leading to the clearest distributional separation among all loss functions tested. Indeed, the F1 score notably increased by more than 13% relative to CE, as was shown in Table 5.6.

Both BL and PZ loss functions substantially improve the margin-based separability of clean versus mislabelled samples compared to CE. These results validate the design objectives: the proposed loss functions mitigate the harmful fitting to noisy labels by selective de-weighting or exclusion. This enhanced separability directly translates into improved performance for label error detection frameworks like AUM and Confident Learning.

### 5.3.5 Conclusion of Study of Robust Loss Functions

This section proposes and provides an in-depth empirical evaluation of two loss functions, Blurry Loss and Piecewise-zero Loss, for improving model robustness and the downstream detection of label errors. By de-weighting or ignoring samples likely to be mislabelled, these losses enhance the performance of label error detection frameworks across a variety of artificial corruption settings. Extensive experiments and ablation studies demonstrate their broad applicability and competitive performance relative to existing robust loss functions. These findings support the use of such loss functions as practical tools for improving data quality and model reliability in supervised learning scenarios fraught with label errors.

## 5.4 Chapter Conclusion

This chapter addressed the challenges of label errors in training data by establishing a framework through which to study model robustness, comparing the robustness of different deep neural network architectures, and through the development of specialized loss functions robust to the presence of label errors.

From the sensitivity analysis of [Section 5.1](#), it was discovered that, for the CNN model tested, remarkably, the model continued to perform with high accuracy (over 95%) even when *most* training data was mislabelled! This lead to further studies evaluating other architectures in [Section 5.2](#). Five types of neural network architectures were investigated, including a MLP, two convolutional, and two transformer-based architectures. Results indicate that the convolutional architectures tend to be more robust than the MLP and transformer-based architectures. An additional experiment using KNN classifiers explored the relationship between a classifier’s global-*vs.*-local nature and its robustness to label errors, and indicate that more global classifiers tend to be more robust, but also underfit in cases of low corruption. Finally, in [Section 5.3](#), two robust loss functions, Blurry Loss and Piecewise-zero Loss, were proposed and thoroughly evaluated. By de-weighting or ignoring samples likely to be mislabelled, these losses enhance the performance of label error detection frameworks across a variety of artificial corruption settings. These robust

loss functions may be used as practical tools for improving data quality and model reliability in supervised learning scenarios fraught with label errors.

While the approaches of this chapter were presented separately, they may also be applied synergistically to improve overall model robustness and generalizability, and the detection (and even possible cleaning) of label errors in datasets. Future work is described in [Section 8.1.1](#).

## Chapter 6

# Improved Hierarchical Classification with Hyperbolic Neural Networks<sup>1</sup>

Hierarchical data, characterized by inherent tree-like relationships between samples, appears across domains from language and social networks to biological taxonomy [202, 245]. In such data, similarity between samples reflects their proximity in the underlying hierarchy — organisms from the same taxonomic branch, for instance, are expected to be more alike than those from different branches. A key challenge in modelling hierarchical data lies in its fine-grained nature [246, 247]: as hierarchical depth increases, the number of categories grows exponentially, while inter-class differences diminish.

Traditional neural networks represent features in Euclidean space, which struggles to efficiently embed such exponentially expanding structures [188, 248]. In contrast, hyperbolic spaces — geometries of constant negative curvature — naturally accommodate hierarchical relationships, as their exponential growth properties align with those of tree-like data [148, 245]. Hyperbolic Neural Networks (HNNs) exploit this property by embedding features in curved hyperbolic space, enabling more compact and expressive representations of hierarchically structured data. Background and related Hyperbolic Neural Network work is

---

<sup>1</sup>Components of this chapter were originally presented at the *10<sup>th</sup> Annual Conference on Vision and Intelligent Systems (CVIS)*. See the [Statement of Contributions](#) for details.

provided in [Section 2.6](#), giving an overview of hyperbolic representations, the mechanisms of hyperbolic neural networks, and their usage.

This chapter investigates how classification performance differs between hyperbolic and Euclidean neural networks, and uncovers an intriguing “Dying ReLU” phenomenon [\[196, 197\]](#) that appears to be alleviated by the application of Hyperbolic Neural Networks. Although alternative activation functions such as Leaky ReLU [\[249\]](#) and Exponential Linear Units (ELU) [\[250\]](#) have been demonstrated to avoid dead ReLUs [\[251\]](#) in conventional Euclidean neural networks, the Dying ReLU problem has not previously been studied in the context of Hyperbolic Neural Networks. Results of this chapter indicate that the use of Hyperbolic Neural Networks imparts a regularization effect such that the Dying ReLU problem is avoided and alternative activations are not required.

## 6.1 Introduction & Problem Definition

Hierarchies are present across many domains [\[202, 245\]](#), from natural language to social networks to the taxonomy of life. To be considered hierarchical, data must possess an inherent tree-like structure that governs the relationships between samples [\[132, 245\]](#). For example, in the taxonomy of life, any pair of organisms from the same branch of life (such as two animals), ought to be more similar (i.e., nearer in the taxonomy tree) than any pair from different branches of life (such as one animal and one plant). Of course, in applications where machine learning is used, data would not be actual real-world entities; rather, it would be images, measurements, or other descriptions of these entities possessing hierarchical relationships or structure.

The main challenge associated with hierarchical data is contending with fine-grainedness [\[246, 247\]](#), whereby there are many leaf-node categories, and instances of neighbouring categories become exceedingly similar as a function of hierarchical depth. For example, some datasets with an inherent hierarchical structure are the BIOSCAN-1M and BIOSCAN-5M datasets, developed in [Chapter 4](#), which include over 5 million images (1 M for BIOSCAN-1M) of insects collected from across the globe along with associated taxonomic labels.

These data are categorized according to Linnaean taxonomy [19–21]. Note that within any taxonomic group, there are multiple sub-groups, e.g., many species within the same genus. This effect leads to an *exponentially* increasing number of categories as hierarchical depth is increased. This presents an interesting machine learning challenge:

*How might one best classify samples of arbitrarily numerous and similar (low inter-class distance) categories?*

Conventional artificial neural networks operate with their hidden features in *Euclidean* space, where conventional measures such as Euclidean distance or Cosine similarity can be reliably used [245]. However, to keep-up with exceedingly high numbers of categories, conventional deep neural networks require similarly great numbers of parameters to adequately embed tree-like hierarchical data, leading to the costly training, usage, and storage of such models. Indeed, Bourgain’s theorem [188, 248] shows that low-distortion (i.e., distance-preserving) embeddings of trees is impossible in Euclidean space, even with an unbounded number of dimensions. Feature spaces represented in various hyperbolic geometries (i.e., non-Euclidean spaces of constant negative curvature) such as the Poincaré Disk model [252, 253] or Lorentz Model [245, 254] are hypothesized to better suit hierarchical data than Euclidean spaces [148, 188, 245], with the prevailing intuitive argument being that the scaling of objects in those spaces results in exponential growth of their characteristic dimensions (whereas in Euclidean space, only polynomial growth occurs)<sup>2</sup> which closely mimics the exponential growth in the number of leaf-nodes in a tree as depth is increased. While this justification may not be scientifically satisfying, experiments have demonstrated an improvement in classification performance of neural networks using hyperbolic feature spaces [245]. Hyperbolic Neural Networks [148, 245] cast their features into a hyperbolic embedding space where space is curved, resulting in interesting properties such as the existence of curved margin hyperplanes (when projected back to Euclidean space) and low distortion embeddings of tree-like data [148, 245].

---

<sup>2</sup>In little detail, in Euclidean space, when, for example, a sphere is scaled by a factor of 2, its radius increases by a factor of 2, its area increases by a factor of  $2^2$ , and its volume increases by a factor of  $2^3$ . These are all categorized as polynomial growth. In contrast, in Hyperbolic spaces, these characteristic dimensions of the object increase *exponentially*, exceeding any polynomial growth in the limit.

This chapter seeks to study the improvement in classification performance of hyperbolic neural networks (HNNs) compared to conventional Euclidean neural networks (ENNs) as a function of the depth of hierarchy of the data being classified. To the best of the author’s knowledge, this effect has not been specifically studied nor quantized in any prior works.

## 6.2 Method

The research of this chapter seeks to study the specific performance benefits of using HNNs *vs.* ENNs for classifying hierarchical data. In particular, an understanding of how the two network types compare in relation to the *depth* of hierarchy would be of value. Rather than approaching complex, large-scale, real-world problems such as taxonomic classification of insects from the BIOSCAN dataset (a domain where only Euclidean methods have been used [1, 2, 255]), first understanding to what extent HNNs may be able to help is pursued. Based on this, recommendations for future work are made.

Details of the experiments are given in [Section 6.3](#); however, briefly, HNNs and ENNs will be constructed with similar structure, and trained and tested on hierarchical data for a classification task. Code for experiments performed in this chapter are based upon the work of [191] and use the Poincaré model, whereby embeddings are transformed via [Equation \(2.9\)](#). The depth of hierarchy in the data will be adjusted over a range of experiments. In addition to this, the dimensionality of the data will also be varied to avoid the possible pitfalls of overly simple or difficult classification problems. For each experimental configuration, multiple trials will be performed (with more trials run for the less computationally expensive cases), thereby mitigating the effects of random differences in training.

## 6.3 Experiments

To study the impact of using hyperbolic neural networks *vs.* conventional Euclidean neural networks, a problem must be set up whereby the data are drawn from, and labels cor-

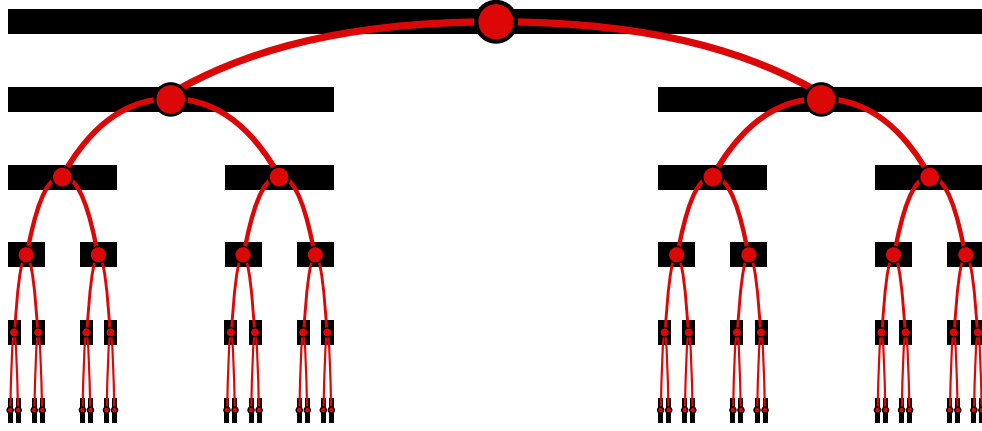


Figure 6.1: A view of Cantor’s Middle Third Set, with the implicit hierarchy made explicit via the tree-like connecting lines. Graphic produced by Sam Derbyshire [8] (CCSA).

respond to, various depths of a hierarchy. Furthermore, both hyperbolic and Euclidean neural networks must be constructed in a manner by which they share as many of the same characteristics as possible, thereby isolating the effect of the use of either hyperbolic or Euclidean feature representation spaces.

### 6.3.1 Experiment Design

#### Synthetic Data

To perform this study in a maximally controlled environment, generated synthetic hierarchical data will be used. A method of synthetic data generation is developed, motivated by the generation algorithm of the Cantor’s Middle Third Set [256, 257], shown graphically in Figure 6.1, whereby an implicit hierarchy is formed.

For Cantor’s Middle Third Set, at each iteration, the entire set is ‘copied’, scaled to one third the size, and shifted, either left or right, thereby creating a middle third gap. At any iteration of generation, any subset can be labelled as having been generated by a sequence of left and right shifts, thereby producing a hierarchical label, where earlier shifts are more significant (higher in hierarchy) than later shifts (lower in hierarchy) which

are more specific and correspond to fine-grained differences. Note that the data itself are hierarchical, not merely the labels, as the distance between any sister subsets (i.e., those having shared parents) is smaller than the distance between any cousin subsets (i.e., those of the same depth but having different parents).

While Cantor’s Middle Third Set can be iteratively generated to have arbitrarily many subsets (categories), the set has gaps and is arbitrarily sparse, a characteristic that conflicts with the concept of low inter-class distance. Therefore, to enforce having low inter-class distance, an approach involving the simple bisection of a continuum is used. This data generation method is illustrated in [Figure 6.2](#). An arbitrary (exponentially growing) number of subsets can be formed and sister subsets have arbitrarily (exponentially decaying) small distance. This concept can be extended to higher dimensions,  $n$ , by considering a closed higher dimensional interval

$$I = \overbrace{[0, 1] \times [0, 1] \times \cdots}^{n \text{ times}} = [0, 1]^n \subset \mathbb{R}^n$$

that can be bisected independently in each dimension. Therefore, for any depth,  $d$ , the number of categories would be  $(2^n)^d$ . Data samples may then be drawn uniformly from the closed interval,  $I$ , and labelled categorically according to their location with respect to the repeated orthogonal bisections. The purpose behind exploring higher dimensions is to avoid potential pitfalls of certain dimensionality problems being either too simple or too difficult.

## Model Architecture

For fair comparison between ENNs and HNNs, a simple three-layer architecture is used and, as much as possible, shared between the two network types. A diagram of the architecture is shown in [Figure 6.3](#), whereby the only difference between the ENN and HNN arise from the conversion to hyperbolic space after the penultimate layer. Operations in the hyperbolic space, while mathematically different, are directly analogous. Note that the input size is determined by the data dimension,  $n$ , and the output size is determined by the total number

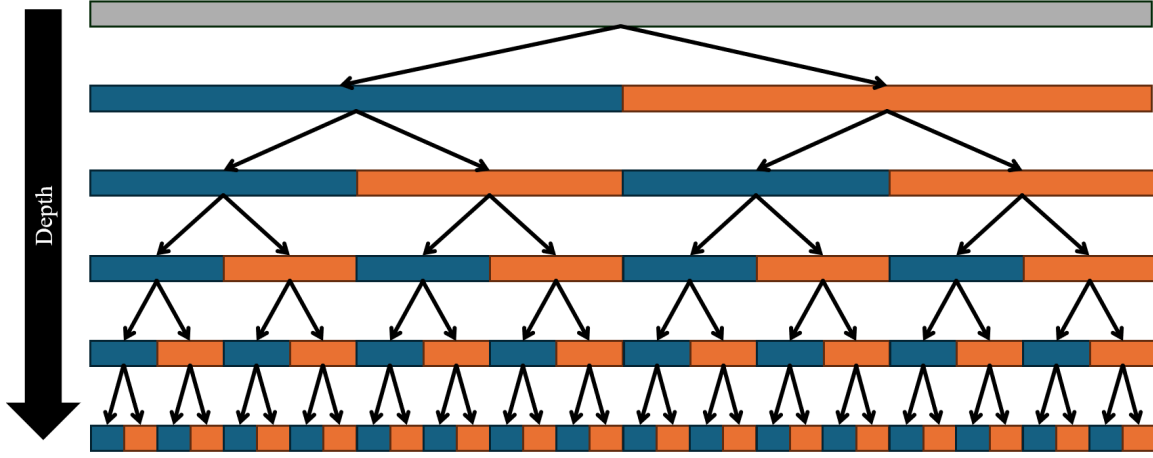


Figure 6.2: A segment of the real line,  $\mathbb{R}$ , is repeatedly bisected (with depth, downwards), giving rise to a dense set with implicit hierarchical structure, indicated by arrows for each subsequent bisection.

of categories,  $(2^n)^d$ . For simplicity, the primary hidden feature dimension is left constant at `hidden_size` = 64. The final embedding dimension is set to `embedding_size` = 2 for the 1D and 2D data experiments, but set to `embedding_size` = 3 for the 3D data experiments, as preliminary experiments indicated that this modest increase in the quantity of network parameters allowed for a substantial increase in performance.

## Overall Experiment Plan

Experiments comparing the classification accuracy of HNNs and ENNs trained and tested on the same data are run for data dimensions  $n = 1, 2, 3$ , and depths of hierarchy of  $d = 1, \dots, 10$ . Note that in the higher-dimensional cases, fewer depths are used, as the number of categories grows exponentially, leading to greatly increased training time while revealing no further trends in classification performance. A maximal depth of  $d = 5$  is used in the 2D case (yielding 1024 categories), and  $d = 3$  in the 3D case (yielding 512 categories). For each combinations of parameters, 20 trials are performed, such that median and quartile test accuracies may be reported as quantitative results. In the case of the first six depths ( $d = 1, \dots, 6$ ) for the 1D case, 200 trials were performed.

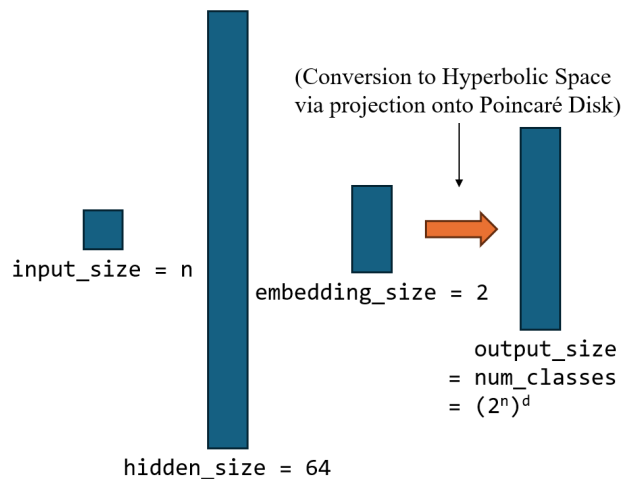


Figure 6.3: The layer structure for both the ENNs and HNNs to be used in this experiment. Note that the only difference is the conversion to hyperbolic space at the penultimate layer. Both networks types (ENN and HNN) share the same number of parameters.

For qualitative results, views of latent embeddings will be explored, allowing some insights into the differences in feature extraction between the two model types to be made.

As is discussed later in [Section 6.3.2](#), the Dying ReLU phenomenon [196, 197] is observed during these experiments, prompting two small studies on the occurrence and prevalence of this phenomenon in the two network types, once with the synthetic data, and once with real data, the MNIST [167] image dataset (to test whether the Dying ReLU phenomenon is particular to the synthetic dataset used in the main experiments, although MNIST is not hierarchical). The dying ReLU phenomenon is a special version of a vanishing gradient problem in which ReLU-activated neurons become inactive, only outputting 0 for any training input and having 0 gradient, thereby stopping the backpropagation of gradients and imposing significant difficulty during training. This was detected by checking whether any weights in the embedding layer have zero gradient for all training inputs. In some cases, training can be effectively halted, as gradients cannot be backpropagated, or the model can become effectively useless, as the dead ReLU prevents the forward propagation of information within the network.

### 6.3.2 Experiment Results

Results of the comparison experiments are aggregated in [Figure 6.4](#), whereby median test accuracy is plotted as function of depth of hierarchy,  $d$ , with the inter-quartile range shown as a shaded region. Separate plots are made for each data dimensionality,  $n = 1, 2, 3$ , shown top to bottom.

Notice that as either hierarchical depth or dimensionality are increased, classification accuracy falls for both the ENN and HNN models. This corroborates the expectation that classification is indeed more difficult for high dimensional data and at deeper levels of hierarchy. It is clear that the HNN models consistently outperform the ENN models, on average and in terms of consistency, as indicated by their relatively lesser interquartile range. While this is true, notice that the best performing ENN ([blue star](#) ★) approximately matches the performance of the HNN in some configurations! This indicates that while HNNs do seem to lead to better performance on average, it is impossible for ENNs to match that performance. Therefore, the difference seems to lie in the *training* of these models. The conversion to hyperbolic space in HNNs seems to act as a regularizer, guiding the optimizer towards consistently better solutions — a feature lacking in the ENNs.

To better understand the feature extraction mechanisms of the two network types, feature embeddings are plotted. Two examples are shown in [Figure 6.5](#): one for an ENN and one for an HNN. In both cases, the data dimensionality  $n = 2$  and depth of hierarchy  $d = 2$ . Notice that for the HNN, the embeddings tend to be scattered towards the perimeter of the Poincaré disk, whereas for the ENN, the embeddings are more evenly distributed. This effect for the HNN embeddings is a result of all categories being *leaf-nodes* of the inherent hierarchy of the data. This matches what has been shown in prior work [\[198, 245, 258\]](#), whereby data from the root of the tree-like hierarchy tend to be embedded near the center of the Poincaré disk and data from branches emanate outwards in all directions, ending in leaf-nodes at the periphery. Due to the curvature of the hyperbolic space, points along the perimeter of the Poincaré disk appear to be packed close together in the Euclidean projection, but in fact are not necessarily close at all in the hyperbolic space. This apparent scaling effect is demonstrated well in [Figure 6.6](#), where the Poincaré

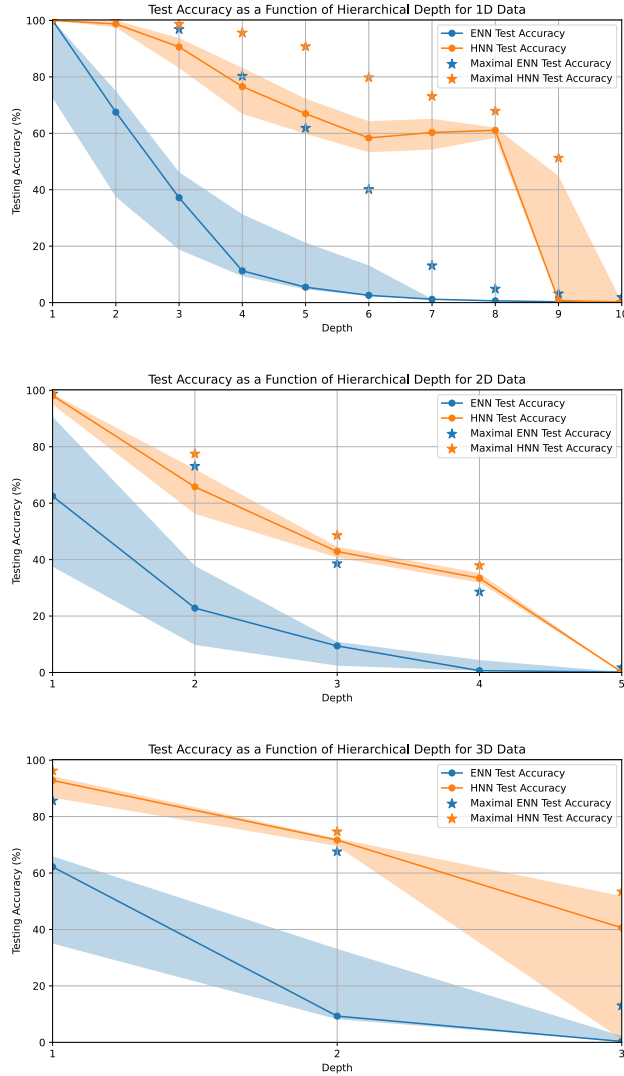


Figure 6.4: Experimental results for the 1D, 2D, and 3D data experiments are shown for ENNs (blue) and HNNs (orange), in that order, from top to bottom. Median test accuracy is plotted as function of depth of hierarchy, with the inter-quartile range shown as a shaded region. 20 trials were performed (except in the 1D case for  $d \leq 6$ , where 200 trials were performed), with the best model performance for each depth and network type shown as a star. Observe that in all cases, accuracy tends to drop as depth is increased, but the HNN's accuracy remains greater than that of the ENN. While this is true, notice that the best performing ENN (blue star  $\star$ ) approximately matches the performance of the HNN in some configurations.

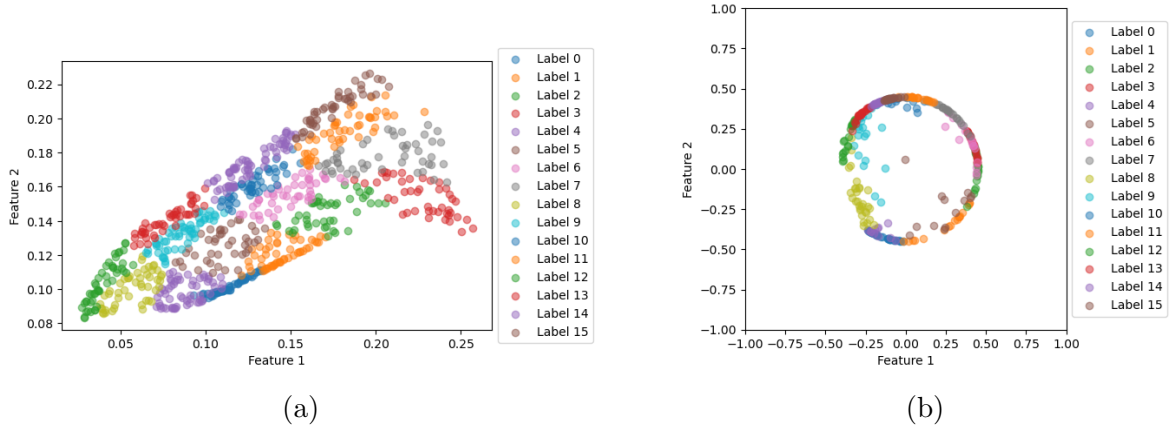


Figure 6.5: (a) Embeddings of the ENN, and (b) Embeddings of the HNN. In both cases, the data dimensionality  $n = 2$  and depth of hierarchy  $d = 2$ . Observe a remarkable difference in the structure of the embeddings.

disk is tiled with equal-sized triangles. Near the edges, the triangular tiles appear to shrink to infinitesimal sizes, but in fact are constant sized when viewed in the original hyperbolic space.

In some cases, the embeddings for ENNs were identically zero in one or more dimensions! Cases of this are shown in Figure 6.7. This prompted a small deeper investigation, leading to the discovery that the Dying ReLU [196, 197] phenomenon is highly prevalent in the ENNs used for these experimental classification problems. Again, dead ReLUs were detected by checking whether any weights in the embedding layer have zero gradient for all training inputs. Similar to the experiments of Figure 6.4, this investigation explored cases for dimensions  $n \in \{1, 2, 3\}$  and depths of hierarchy  $d \in \{1, 2, 3, 4, 5, 6\}$ , with five trials performed for each model type in each case. Table 6.1 summarizes the instances of dead RuLUs in each experimental configuration. Indeed, in more trials than not, one or more ReLUs died during the training of the ENNs. Remarkably, there were no experimental configurations (pairings of  $n$  and  $d$ ) in which the ENNs *never* had a ReLU die (i.e., having zero dead ReLUs in at least some trials of a given experimental configuration), and simultaneously, there were no cases where the HNNs *ever* had a ReLU die! This

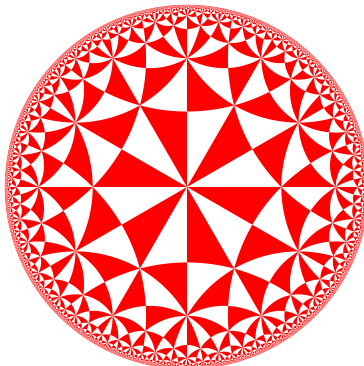


Figure 6.6: Triangular tiling of the Poincaré disk. All triangular tiles are of equal size in the hyperbolic space. Image sourced from [7] (CC0).

finding further emphasizes the apparent regularization effect induced by the conversion to hyperbolic space.

To ensure that the prevalence of the Dying ReLU phenomenon is not simply artifact of the synthetic data used in the experiments, a secondary study on real data, the MNIST image dataset, was performed, effectively repeating the experiments of Table 6.1. This study used the same ENN and HNN architecture as the main experiments (Figure 6.3) by “flattening” the input MNIST images. Ten trials were performed, with the rate of occurrence of dead ReLUs, median test accuracy, and maximal test accuracy reported in Table 6.2. In 60% of trials, both ReLUs in the embedding layer died for the ENN, whereas no dead ReLUs were found for the HNN, congruent with the previous results on synthetic data. In cases without dead ReLUs, the ENN performed similarly to the HNN ( $\sim 92\%$  accuracy). Examples of typical embeddings (without dead ReLUs) are shown in Figure 6.8, where, once again, the HNN embeddings are scattered towards the perimeter of the Poincaré disk and are fairly well separated, whereas the ENN embeddings tend to be more scattered and are less visually separated.

It is hypothesized that the presence of dead ReLUs in “bottle neck” layers (those with few neurons) are more detrimental during training than those in larger layers. Consider, in the extreme case, a linear layer with only one neuron. If this ReLU dies, then the propagated gradient to upstream layers is necessarily zero, which then means their weights

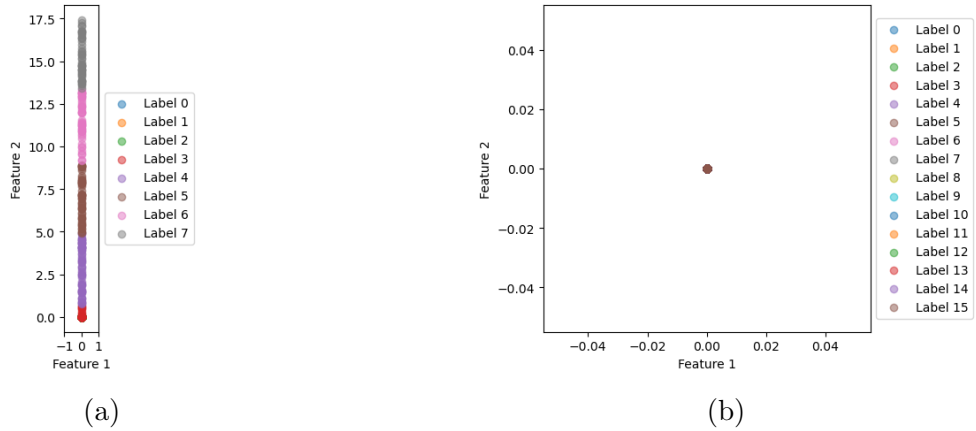


Figure 6.7: Embeddings for ENNs with dead ReLUs. (a) A single ReLU at the embedding layer is dead, leading to the collapse of the first feature. Model trained on data of dimensionality  $n = 1$  and depth of hierarchy  $d = 3$  (b) Both ReLUs at the embedding layer are dead, leading to the complete collapse of both features. Model trained on data of dimensionality  $n = 2$  and depth of hierarchy  $d = 2$ .

will not update, and that this ReLU will necessarily stay dead — not activate again in future epochs — and effectively halt training. Furthermore, this dead ReLU would block any propagation of information (predicted probabilities) to the output of the network, effectively making the model useless. On the other hand, in layers with more outputs, if one ReLU dies, back-propagated gradients from its neighbouring neurons in the same layer will continue to update the upstream weights, which will allow the overall network to continue training and may allow the dead ReLU to become active again in future epoch. Indeed, the presence of some dead ReLUs in large layers may not at all be detrimental, and is in fact reminiscent of Dropout Regularization [259, 260], whereby a small proportion of layer inputs are zeroed. In these experiments, the embedding size was only 2 or 3, therefore creating a bottleneck and causing dead ReLUs in this layer to severely hamper model training and final classification performance.

In the experiments, the embedding dimension was always kept very low (at either 2 or 3). To understand to what extent the small embedding dimension limited the performance of the ENNs, a small additional experiment was performed whereby the embedding

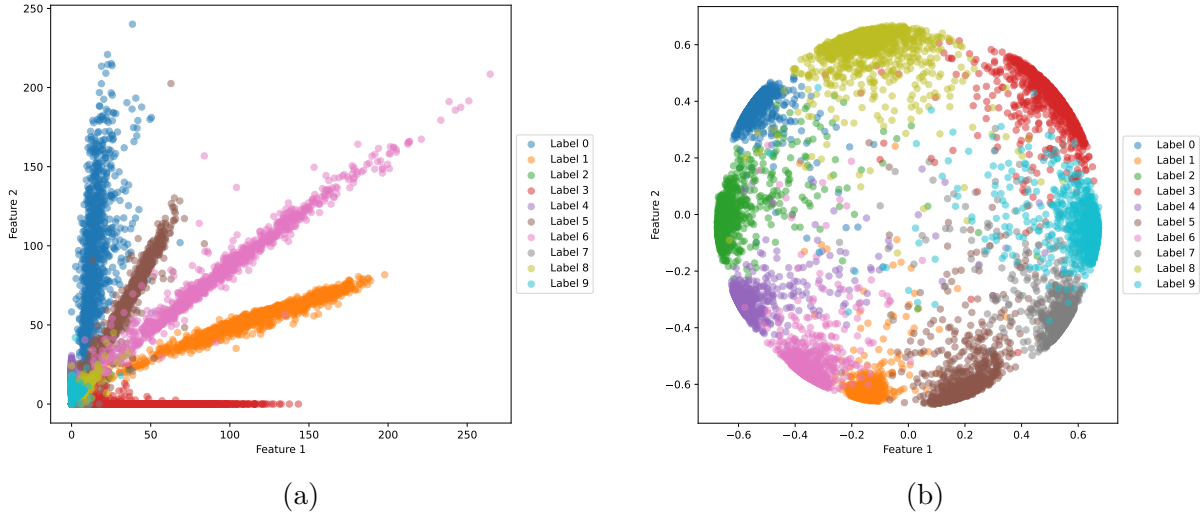


Figure 6.8: Embeddings are shown for the MNIST data for the ENN (a) and HNN (b). Observe the stark difference in the structure of the embeddings between the ENN and HNN, similar to what was seen in the synthetic data embeddings of [Figure 6.5](#).

Table 6.1: Rate of occurrence of Dead ReLUs by Dimensionality,  $n$ , Depth of Hierarchy,  $d$ , and Model Type for Synthetic data experiments.

|     |            | Rate of Occurrence (%) |     |     |    |     |    |
|-----|------------|------------------------|-----|-----|----|-----|----|
| $n$ | Model Type | $d$                    |     |     |    |     |    |
|     |            | 1                      | 2   | 3   | 4  | 5   | 6  |
| 1   | ENN        | 60                     | 100 | 100 | 80 | 100 | 80 |
|     | HNN        | 0                      | 0   | 0   | 0  | 0   | 0  |
| 2   | ENN        | 60                     | 100 | 80  | 80 |     |    |
|     | HNN        | 0                      | 0   | 0   | 0  |     |    |
| 3   | ENN        | 80                     | 100 | 100 |    |     |    |
|     | HNN        | 0                      | 0   | 0   |    |     |    |

Table 6.2: Building on [Table 6.1](#), for the MNIST experiments, the Rate of Occurrence of Dead ReLUs, Median, and Maximal Test Accuracy are reported by Model Type.

| Model Type | Rate of Occurrence (%) | Median Acc. (%) | Maximal Acc. (%) |
|------------|------------------------|-----------------|------------------|
| ENN        | 60                     | 11.35           | 92.28            |
| HNN        | 0                      | 87.30           | 92.10            |

dimension was increased until the ENN consistently matched the performance of the HNN. For simplicity of this exploration, the data of dimensionality  $n = 1$  and depth of hierarchy  $d = 4$  was used. As was seen in the top panel of [Figure 6.4](#), accuracy for the HNN model tended to be roughly 80% for this configuration. Starting from an embedding dimension of 2, it was found that a minimum embedding dimension of 16 was required to reliably achieve similar performance. Even in this case, dead ReLUs were detected.

## 6.4 Chapter Conclusion

The findings of this chapter firmly indicate that hyperbolic neural networks consistently match or outperform Euclidean neural networks on classification tasks involving hierarchical data. In particular, as the depth of hierarchy is increased, the disparity between HNN and ENN accuracy is increased with ENN accuracy even falling to near-zero within the parameter ranges explored.

While high classification accuracy is theoretically possible for ENNs, even for hierarchical data (after all, deep neural networks are touted to be universal function approximators and could perform similar transformations as are done by the HNNs), it seems that the use of hyperbolic representation spaces acts as a form of *regularization*, allowing for far better network parameter optimization during training. This regularization effect is especially obvious when considering the prevalence of dying ReLUs in the ENNs in this study. The dying ReLU problem never occurred for HNNs, whereas for ENNs it was highly common and problematic. Lastly, this regularization effect results in the ability to use far fewer network parameters while maintaining accuracy. It was seen that as many as eight times

more parameters were required (increasing the embedding dimension from 2 to 16) for the ENN to reliably achieve similar classification accuracy as the HNN in one tested configuration. These results together demonstrate that hyperbolic neural networks are both more parameter-efficient and robustly-trainable than conventional Euclidean networks, especially for tasks involving deeper hierarchies. Future work is discussed in [Section 8.1.2](#), and Hyperbolic Neural Networks are applied again in [Chapter 7](#) in a study exploring whether embeddings in hyperbolic space can provide more accurate fine-grained classification in a multimodal setting.

# Chapter 7

## Hierarchical Embeddings and Multi-modal Retrieval<sup>1</sup>

In [Chapter 6](#), Hyperbolic Neural Networks (HNNs) were studied in comparison to conventional Euclidean neural networks (ENNs). In particular, the study empirically tested the improvement in classification performance of HNNs compared to conventional ENNs as a function of the depth of hierarchy of the data being classified. Here, Hyperbolic Neural Networks are applied again in a study exploring whether embeddings in hyperbolic space can provide more accurate fine-grained classification in a multimodal setting. Relevant background on Hyperbolic Neural Networks is provided in [Section 2.6](#), giving an overview of hyperbolic representations, the mechanisms of hyperbolic neural networks, and their usage. Furthermore, an overview of related Multimodal and Hyperbolic representation work is given in [Section 2.6.1](#).

---

<sup>1</sup>Note that the work presented in this chapter was carried out in collaboration with the broader BIOSCAN ML Team. Much of the writing of this chapter was originally prepared for a submission to the 2025 ICCV Beyond Euclidean Workshop [\[5\]](#). See the [Statement of Contributions](#) for details.

## 7.1 Introduction

As was discussed earlier in [Section 2.1](#) and [chapter 4](#), specimen identification is an essential step for monitoring and mitigating biodiversity loss, requiring accurate classification of organisms within the taxonomic hierarchy across diverse ecosystems. DNA barcodes [\[10, 11\]](#) provide a way to classify specimens to known taxa or identify them as novel to science, but classification to the species level remains challenging, especially when barcodes are unavailable. To tackle this, CLIBD [\[209\]](#) showed that using contrastive learning to align DNA barcode embeddings to image embeddings can improve classification at the species level even when restricted to only using images for inference.

However, a key limitation of CLIBD [\[209\]](#) and other recent biodiversity-focused multimodal methods [\[207\]](#) is that they do not utilize the known taxonomic hierarchy of the input data. Motivated by the effectiveness of hyperbolic embeddings for capturing hierarchical relationships [\[211\]](#), similar to the conclusions of [Chapter 6](#), an exploration is conducted examining whether embeddings in hyperbolic space can provide more accurate fine-grained classification. The model takes inputs from multiple modalities — DNA barcodes, specimen images, and hierarchical taxonomic labels — and is trained to co-align their embeddings into a shared hyperbolic space to promote taxonomic alignment across modalities. Furthermore, to capture the inherent hierarchical structure of the taxonomic labels, a novel loss, termed *stacked entailment loss*, is developed to enforce the geometric relationships between taxonomic ranks.

The method differs from prior work, in three key ways. First, rather than focusing on vision-language, biologically grounded modalities — DNA barcodes and taxonomic labels — that are more salient for species-level classification are incorporated. Second, *explicit* taxonomic hierarchies are leveraged to guide representation learning rather than relying on *implicit* hierarchical signals such as caption specificity or object part composition. Third, stacked entailment loss enforces consistency across hierarchical ranks.

The experimental results show that the hyperbolic multimodal learning framework achieves strong performance in taxonomic classification and retrieval, especially at higher taxonomic ranks. The approach consistently matches or outperforms Euclidean baselines.

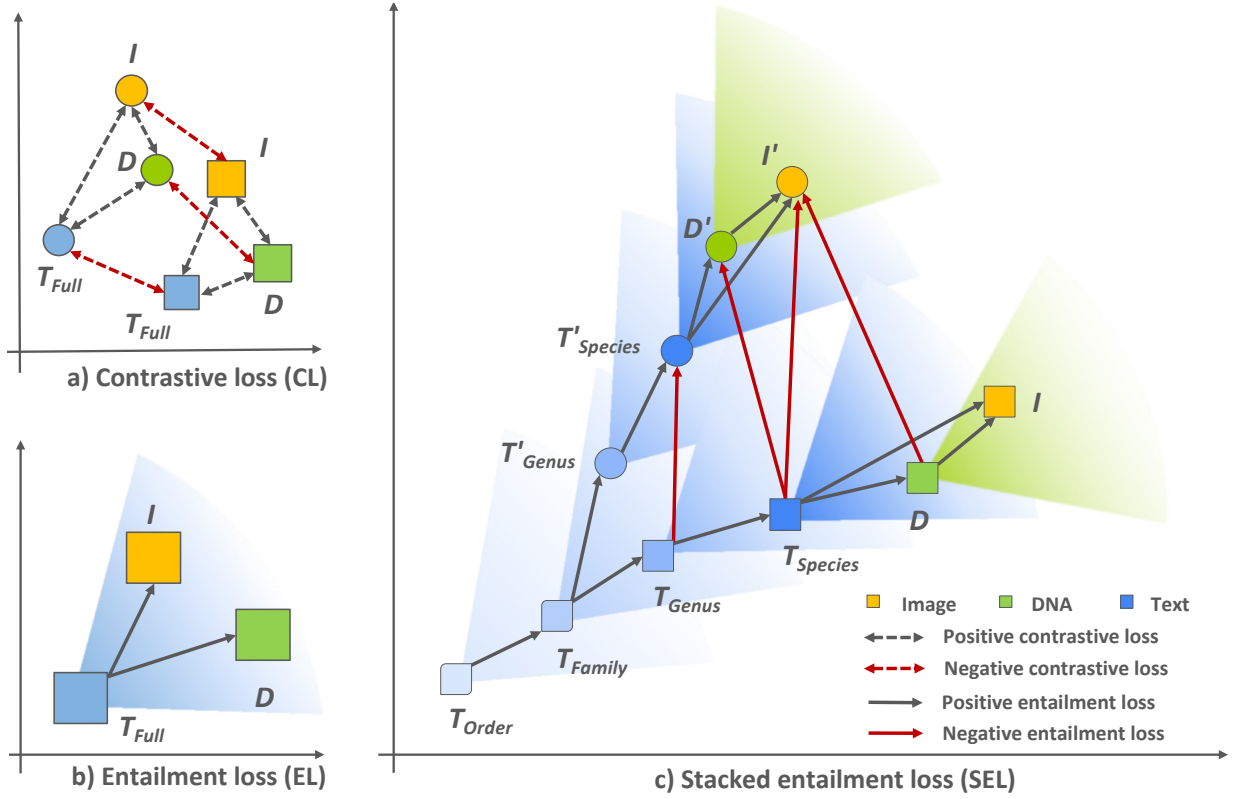


Figure 7.1: (a) **Contrastive loss**: instance-level alignment between modalities. (b) **Entailment loss**: enforces hierarchy within the text modality using entailment cones. (c) **Stacked entailment loss**: combines EL and cross-modal constraints by aligning image and DNA embeddings to multiple levels of the text hierarchy. Figure produced for and taken from [5].

However, all methods — including including the proposed — face challenges in fine-grained species classification, particularly for previously unseen taxa. These results highlight both the potential of hyperbolic learning for hierarchical biological data, and the ongoing difficulty of open-world classification for biodiversity.

## 7.2 Approach

A multimodal representation learning [118, 205] framework is proposed that unifies specimen DNA barcodes, images, and taxonomic labels into a shared *hyperbolic* embedding space. With the aim of preserving hierarchical taxonomic relationships, hyperbolic geometry is leveraged to enhance classification accuracy and improve representation quality across the hierarchy.

The hyperbolic multimodal learning framework employs three specialized encoders to process each of the data modalities: an image encoder extracts visual features, a DNA encoder encodes genetic sequences, and a text encoder captures semantic information from taxonomic labels of varying depth. These encoders independently map their inputs into a common embedding space, in which contrastive learning aligns multimodal representations for downstream tasks. An extension of CLIBD [209] is presented by lifting embeddings into hyperbolic space, followed by evaluation on the BIOSCAN-1M dataset described in [Section 4.1](#).

### 7.2.1 Input and Output Specification

During training, the encoders are jointly optimized using triplets of aligned data—specimen image, DNA barcode, and hierarchical taxonomic labels (e.g., “Order: Diptera; Family: Syrphidae; Genus: Episyrphus; Species: Episyrphus balteatus”) — so that their embeddings are both cross-modally aligned and geometrically consistent with the taxonomic hierarchy. This objective supports flexible inference with any subset of modalities while preserving multi-level taxonomic relationships in the learned space. At inference time,

the model supports both uni- and cross-modal retrieval, allowing it to taxonomically classify specimens using any available combination of images, DNA barcodes and taxonomic labels. This enables robust downstream use in biodiversity monitoring and taxonomic classification, even with missing or noisy modalities.

## 7.2.2 Encoders

The experimental setup from [209] is adapted, using pretrained ViT-B/16, BERT-Small, and BarcodeBERT encoders for image, text, and DNA barcode modalities.

Each encoder produces Euclidean embeddings, which are then projected into a Lorentzian hyperbolic space with curvature  $c$ , via an exponential mapping centred at the origin. Refer to [211] for further details. The shared space enables contrastive alignment across modalities while preserving the hierarchical taxonomic structure.

## 7.2.3 Model Details

### Overall Architecture

The proposed framework comprises three modality-specific encoders, following the setup used in CLIBD [209]:

- **Image encoder:** A ViT-B<sup>2</sup> backbone is employed and is initialized with ImageNet-21k pretraining and further tuned on ImageNet-1k [232].
- **DNA encoder:** BarcodeBERT [261] with 5-mer tokenization, pretrained via masked language modeling on 893k DNA barcode sequences [262]. This corpus is related to but does not overlap with BIOSCAN-1M, making it suitable for unbiased evaluation.
- **Text encoder:** A pretrained BERT-Small model [263] is used to embed taxonomic labels at different ranks.

---

<sup>2</sup>Implemented as `vit_base_patch16_224` from the `timm` library.

Each encoder produces Euclidean embeddings of size  $d = 768$ , which are mapped to a Lorentzian hyperbolic space  $\mathbb{H}_L^n$  (with curvature  $c > 0$ ) via an exponential map described next in this section.

## Hyperbolic Projection and Distances

Following MERU [211], encoder outputs (Euclidean vectors) are projected onto the Lorentzian hyperboloid using the exponential map. Note that this exponential mapping is different than the exponential mapping of Equation (2.9) in Section 2.6 that maps to Poincaré disk — rather than to the Lorentzian hyperboloid, as is done here. Similar to Equation (2.9),  $c > 0$  denotes the curvature of the hyperbolic space  $\mathbb{H}_c^d$ ; smaller values of  $c$  correspond to a “flatter” geometry, while larger values lead to more strongly curved spaces.

The general exponential map from a tangent vector  $v \in T_p \mathbb{H}_c^d$ , where  $T_p \mathbb{H}_c^d$  denotes the tangent space at point  $p$  in the Lorentz model of  $\mathbb{H}_c^d$ , to the manifold is given by:

$$\text{Exp}_p(v) = \cosh(\sqrt{c} \|v\|_{\mathbb{L}}) p + \frac{\sinh(\sqrt{c} \|v\|_{\mathbb{L}})}{\sqrt{c} \|v\|_{\mathbb{L}}} v.$$

Hyperbolic distances are computed via:

$$d_{\mathbb{H}}(x, y) = \frac{1}{\sqrt{c}} \cosh^{-1}(-\langle x, y \rangle_{\mathbb{L}}).$$

## Entailment Cones

The half-aperture angle of the cone centred at  $u$  is:

$$\alpha(u) = \sin^{-1} \left( \frac{K}{\|u\|_{\mathbb{H}}} \right),$$

where  $K = 2r_{\min}/\sqrt{c}$ . Here  $r_{\min}$  is a small constant 0.1, which is used to set boundary conditions near the origin and prevent  $\alpha(u)$  from diverging when  $\|u\|_{\mathbb{E}}$  is small.

Table 7.1: Example of taxonomic labels and their full-text concatenation.

| Rank      | Label                                             |
|-----------|---------------------------------------------------|
| Order     | Hymenoptera                                       |
| Family    | Formicidae                                        |
| Genus     | Myrmica                                           |
| Species   | Myrmica specioides                                |
| Full-text | Hymenoptera Formicidae Myrmica Myrmica specioides |

### Input Text Construction

Taxonomic labels are encoded per rank using the BERT-Small tokenizer [263]. Full-text inputs concatenate all ranks with spaces as separators (see Table 7.1).

### Training Details

Training is performed on the `train_seen` split (36k samples) of BIOSCAN-1M. The batch size is 2000 for CL-only runs and 1520 for SEL runs across  $4 \times A100$  (80GB). Optimization is with Adam ( $\beta_1 = 0.9$ ,  $\beta_2 = 0.98$ , weight decay  $1e-4$ ), with a one-cycle LR schedule ( $1e-6$  to  $5e-5$ ). Mixed precision is used. All negatives come from in-batch sampling; for entailment loss, negatives are taxonomy-aware.

#### 7.2.4 Stacked Entailment Loss

To better leverage the inherent structure of the biological taxonomy, a hierarchical learning objective is proposed, termed *stacked entailment loss* (SEL). This mechanism is designed to explicitly enforce geometric relationships between taxonomic ranks — order, family, genus, and species — within hyperbolic space (see Figure 7.1). The design is inspired by compositional entailment mechanisms introduced in prior work [212], but adapted to reflect the nested and non-overlapping nature of biological hierarchies.

The core idea is to constrain the embeddings of lower-level taxa (e.g., genus) to lie within an entailment cone of their parent nodes (e.g., family). This entailment constraint

is applied between each consecutive pair of levels in the hierarchy to ensure each child node is within the space “above” its parent, using a margin-based loss. Additionally, a *negative entailment* loss term is introduced which ensures each child node is *not* within the space “above” nodes from the preceding layer that are *not* its parent.

Given a batch  $\mathcal{B} = \{(x_i, y_i, k_i)\}_{i=1}^B$ , where  $x_i$  and  $y_i$  are embeddings and  $k_i$  the class, positive pairs are defined as  $\mathcal{P} = \{(i, j) : k_i = k_j\}$  and negative pairs as  $\mathcal{N} = \{(i, j) : k_i \neq k_j\}$ . The corresponding entailment losses are:

$$\mathcal{L}_{\text{ent}}^+ = \frac{1}{|\mathcal{P}|} \sum_{(i,j) \in \mathcal{P}} \max(0, \text{ext}(x_i, y_j) - \text{aper}(x_i)) \quad (7.1)$$

$$\mathcal{L}_{\text{ent}}^- = \frac{1}{|\mathcal{N}|} \sum_{(i,j) \in \mathcal{N}} \max(0, \text{aper}(x_i) - \text{ext}(x_i, y_j) + m) \quad (7.2)$$

where  $\text{ext}(x, y)$  denotes the exterior angle between  $x$  and  $y$  in hyperbolic space,  $\text{aper}(x)$  is the cone aperture of  $x$ , and  $m$  is the margin for negative pairs. The positive and negative entailment loss are then combined into:

$$\mathcal{L}_{\text{ent}} = 1/2 (\mathcal{L}_{\text{ent}}^+ + \mathcal{L}_{\text{ent}}^-). \quad (7.3)$$

Unlike flat contrastive objectives, which treat all positive pairs equally, the stacked entailment loss introduces a directional notion of containment in the taxonomic hierarchy (from parent to child), ensuring that more specific taxa (fine-grained nodes) are properly nested under their broader ancestors in the hyperbolic hierarchy. The overall stacked-entailment loss consists of two parts:

$$\mathcal{L}_{\text{SEL}} = \mathcal{L}_{\text{SEL-intra}} + \mathcal{L}_{\text{SEL-inter}}. \quad (7.4)$$

The first component, **intra-modal entailment loss**, enforces hierarchy among taxonomic labels. Let the taxonomy have  $R$  levels (e.g., order, family, genus, species), indexed  $r = 1, 2, \dots, R$  from root to leaf.  $T_r$  is the embedding at rank  $r$ , and  $\mathbb{1}_r$  an indicator function for the availability of the label at rank  $r$ . The intra-modal stacked entailment loss is then,

$$\mathcal{L}_{\text{SEL-intra}} = \frac{1}{\sum_{r=2}^R \mathbb{1}_r} \sum_{r=2}^R \mathbb{1}_r \times \mathcal{L}_{\text{ent}}(T_r, T_{r-1}). \quad (7.5)$$

Secondly, an **inter-modal entailment loss** is introduced that bridges the taxonomic labels with other modalities:

$$\mathcal{L}_{\text{SEL-inter}} = \frac{1}{3} \left( \mathcal{L}_{\text{ent}}(I, T_{R'}) + \mathcal{L}_{\text{ent}}(D, T_{R'}) + \mathcal{L}_{\text{ent}}(I, D) \right) \quad (7.6)$$

where  $I$  and  $D$  are the embeddings of images and DNA barcodes respectively, and  $T_{R'}$  refers to the deepest available taxonomic label (i.e.,  $T_{\text{Species}}$  if species is known,  $T_{\text{Genus}}$  if species isn't known but genus is, *etc.*). This term ensures that modality-specific inputs are not only aligned with the correct label, but also geometrically nested within the same hierarchical space. Since there can be multiple specimens with the same DNA barcode, and the same specimen can have different images, the barcode is considered to be more abstract than the image and also include an entailment loss term from barcode to image in the inter-modality objective.

In summary, the proposed stacked entailment loss unifies taxonomic ordering and modality alignment, and embeds hierarchical structure into model training. This loss enables better generalization, especially with incomplete labels or unseen species. By explicitly modelling the hierarchical containment of taxonomic levels, this approach enables independent retrieval and prediction at any rank (e.g., order, family, genus, or species), facilitating multi-level querying and evaluation directly within the learned representation. This stands in contrast to CLIBD, which produces predictions at all levels jointly. The stacked entailment loss of Equation (7.4) is extended with two variants.

- **Image-DNA contrastive loss:** By adding a contrastive loss term,  $\mathcal{L}_{\text{CL}}(I, D)$ , based on the negative Lorentz distance between image,  $I$ , and DNA embeddings,  $D$ , stronger cross-modal alignment is encouraged and can improve the accuracy of image-to-DNA retrieval.
- **Full-text supervision:** An extra language input is introduced by concatenating taxonomic labels from all four ranks (order, family, genus, species), as is used in CLIBD, with Full Text concatenated embedding  $T_{\text{Full Text}}$ . The full text is also used for contrastive alignment to the image and DNA embeddings.

Table 7.2: **Macro** top-1 accuracy (%) comparison of different training objectives across taxonomic levels on BIOSCAN-1M. CL: contrastive loss. EL: entailment loss; SEL: stacked entailment loss; **Best** results are shown in bold; second-best are underlined.

| Rank    | Method | EL config. | Full Text | Space            | DNA-to-DNA  |             |             | Image-to-Image |             |             | Image-to-DNA |             |             |
|---------|--------|------------|-----------|------------------|-------------|-------------|-------------|----------------|-------------|-------------|--------------|-------------|-------------|
|         |        |            |           |                  | Seen        | Unseen      | H.M.        | Seen           | Unseen      | H.M.        | Seen         | Unseen      | H.M.        |
| Order   | CLIBD  | –          | ✓         | $\mathbb{R}^n$   | <b>89.1</b> | 87.8        | 88.4        | <b>99.5</b>    | <b>66.4</b> | <b>79.6</b> | <b>98.7</b>  | <b>49.5</b> | <b>65.9</b> |
|         | CL     | –          | ✓         | $\mathbb{H}_L^n$ | <b>89.1</b> | 85.6        | 87.3        | 98.5           | 61.2        | 75.5        | 89.1         | 47.8        | 62.2        |
|         | EL+CL  | Pos.       | ✓         | $\mathbb{H}_L^n$ | 88.6        | 86.5        | 87.5        | 98.6           | 56.9        | 72.1        | 77.8         | 48.4        | 59.7        |
|         | SEL    | Pos.+Neg.  | ✗         | $\mathbb{H}_L^n$ | 88.4        | <b>90.8</b> | <b>89.6</b> | 79.3           | 62.3        | 69.8        | <b>98.7</b>  | <u>48.9</u> | <u>65.4</u> |
|         | SEL+CL | Pos.+Neg.  | ✗         | $\mathbb{H}_L^n$ | 88.7        | 86.3        | 87.5        | <u>99.4</u>    | <u>65.9</u> | 79.3        | 78.6         | 48.2        | 59.7        |
|         | SEL+CL | Pos.+Neg.  | ✓         | $\mathbb{H}_L^n$ | 88.9        | <u>88.2</u> | <u>88.5</u> | 99.0           | 60.9        | 75.4        | 78.6         | <u>48.9</u> | 60.3        |
| Family  | CLIBD  | –          | ✓         | $\mathbb{R}^n$   | <u>90.8</u> | 75.8        | 82.6        | <b>89.2</b>    | <b>52.2</b> | <b>65.9</b> | <b>83.6</b>  | <b>19.3</b> | <b>31.4</b> |
|         | CL     | –          | ✓         | $\mathbb{H}_L^n$ | 90.3        | 76.6        | <u>82.9</u> | <u>83.9</u>    | <u>48.5</u> | <u>61.4</u> | <u>79.6</u>  | <u>18.8</u> | <u>30.4</u> |
|         | EL+CL  | Pos.       | ✓         | $\mathbb{H}_L^n$ | 89.3        | 74.9        | 81.4        | 81.9           | 37.6        | 51.5        | 76.7         | 16.8        | 27.6        |
|         | SEL    | Pos.+Neg.  | ✗         | $\mathbb{H}_L^n$ | 86.8        | <b>78.8</b> | 82.6        | 79.0           | 41.8        | 54.7        | 78.9         | 18.4        | 29.9        |
|         | SEL+CL | Pos.+Neg.  | ✗         | $\mathbb{H}_L^n$ | 89.0        | 76.9        | 82.5        | 79.6           | 46.6        | 58.8        | 78.7         | 17.3        | 28.4        |
|         | SEL+CL | Pos.+Neg.  | ✓         | $\mathbb{H}_L^n$ | <b>91.2</b> | <u>77.0</u> | <b>83.6</b> | 82.4           | 41.5        | 55.2        | 78.1         | 17.4        | 28.4        |
| Genus   | CLIBD  | –          | ✓         | $\mathbb{R}^n$   | 85.2        | 64.3        | 73.3        | <b>71.3</b>    | <b>35.0</b> | <b>47.0</b> | <b>70.8</b>  | <b>7.1</b>  | <b>12.9</b> |
|         | CL     | –          | ✓         | $\mathbb{H}_L^n$ | <b>86.4</b> | 64.9        | <u>74.1</u> | <u>65.6</u>    | 32.4        | 43.4        | 66.9         | 6.5         | 11.8        |
|         | EL+CL  | Pos.       | ✓         | $\mathbb{H}_L^n$ | 84.7        | 63.1        | 72.3        | 63.0           | 22.8        | 33.5        | 64.2         | <u>6.6</u>  | 11.9        |
|         | SEL    | Pos.+Neg.  | ✗         | $\mathbb{H}_L^n$ | 82.7        | <u>65.9</u> | 73.4        | 62.1           | 29.2        | 39.7        | 63.1         | <u>6.6</u>  | <u>12.0</u> |
|         | SEL+CL | Pos.+Neg.  | ✗         | $\mathbb{H}_L^n$ | 83.6        | <b>66.9</b> | <b>74.3</b> | 63.3           | <u>33.1</u> | <u>43.5</u> | <u>67.6</u>  | 6.4         | 11.7        |
|         | SEL+CL | Pos.+Neg.  | ✓         | $\mathbb{H}_L^n$ | <u>85.8</u> | 64.8        | 73.9        | 64.8           | 27.5        | 38.6        | 64.8         | 6.2         | 11.4        |
| Species | CLIBD  | –          | ✓         | $\mathbb{R}^n$   | 81.8        | 60.6        | 69.7        | <b>55.1</b>    | <b>24.3</b> | <b>33.7</b> | <b>55.8</b>  | 0.7         | 1.4         |
|         | CL     | –          | ✓         | $\mathbb{H}_L^n$ | <b>84.4</b> | 61.8        | <b>71.4</b> | <u>48.2</u>    | 22.6        | <u>30.8</u> | 53.7         | 0.9         | 1.7         |
|         | EL+CL  | Pos.       | ✓         | $\mathbb{H}_L^n$ | 82.5        | 60.1        | 69.6        | 45.4           | 14.3        | 21.8        | 50.5         | 0.9         | 1.8         |
|         | SEL    | Pos.+Neg.  | ✗         | $\mathbb{H}_L^n$ | 79.5        | <u>62.3</u> | 69.9        | 45.5           | 20.0        | 27.8        | 52.0         | <b>1.1</b>  | <b>2.1</b>  |
|         | SEL+CL | Pos.+Neg.  | ✗         | $\mathbb{H}_L^n$ | 80.5        | <b>63.2</b> | <u>70.8</u> | 46.8           | <u>22.8</u> | 30.7        | <u>54.2</u>  | 0.7         | 1.4         |
|         | SEL+CL | Pos.+Neg.  | ✓         | $\mathbb{H}_L^n$ | <u>82.6</u> | 62.0        | <u>70.8</u> | 47.8           | 19.0        | 27.2        | 51.4         | <u>1.0</u>  | <b>2.1</b>  |

## 7.3 Experiments and Results

The Euclidean-space CLIBD model [209] is used as a baseline, and adapt the CLIBD training pipeline to use hyperbolic-space based on the MERU framework [211]. Various combinations of loss functions were explored, including entailment loss, stacked entailment loss, and contrastive loss. Experiments were conducted on four NVIDIA A100 GPUs (80GB VRAM each). A batch size of 2000 ( $4 \times 500$ ) was used, except for experiments using stacked entailment, which could only fit a batch size of 1520 ( $4 \times 380$ ). All models were trained for 50 epochs with Adam [77]. The learning rate was scheduled using a one-cycle policy [264], ranging from  $1 \times 10^{-6}$  to  $5 \times 10^{-5}$ . For the contrastive loss, a trainable temperature was used and was initialized to 0.07.

Table 7.3: **Micro** top-1 accuracy (%) comparison of different training objectives across taxonomic levels on BIOSCAN-1M. CL: contrastive loss. EL: entailment loss; SEL: stacked entailment loss; **Best** results are shown in bold; second-best are underlined.

| Rank    | Method | EL Settings | Full Text | Space            | DNA-to-DNA  |             |             | Image-to-Image |             |             | Image-to-DNA |             |             |
|---------|--------|-------------|-----------|------------------|-------------|-------------|-------------|----------------|-------------|-------------|--------------|-------------|-------------|
|         |        |             |           |                  | Seen        | Unseen      | H.M.        | Seen           | Unseen      | H.M.        | Seen         | Unseen      | H.M.        |
| Order   | CLIBD  | -           | ✓         | $\mathbb{R}^n$   | <b>99.2</b> | <u>98.2</u> | <b>98.7</b> | <b>99.6</b>    | <b>98.3</b> | <b>98.9</b> | <u>99.4</u>  | <b>96.4</b> | <b>97.9</b> |
|         | CL     | -           | ✓         | $\mathbb{H}_L^n$ | 99.1        | 98.0        | 98.6        | <u>99.4</u>    | <u>98.0</u> | <u>98.7</u> | <b>99.5</b>  | 95.5        | <u>97.5</u> |
|         | EL+CL  | Pos.        | ✓         | $\mathbb{H}_L^n$ | <b>99.2</b> | 97.9        | 98.6        | 99.3           | <u>97.1</u> | <u>98.2</u> | 99.2         | 95.8        | 97.4        |
|         | SEL    | Pos.+Neg.   | ✗         | $\mathbb{H}_L^n$ | 99.1        | <u>98.2</u> | 98.6        | <u>99.4</u>    | 97.7        | 98.6        | 99.1         | 95.0        | 97.0        |
|         | SEL+CL | Pos.+Neg.   | ✗         | $\mathbb{H}_L^n$ | 99.1        | <b>98.3</b> | <b>98.7</b> | <u>99.4</u>    | 97.7        | 98.5        | 98.9         | 95.5        | 97.2        |
|         | SEL+CL | Pos.+Neg.   | ✓         | $\mathbb{H}_L^n$ | <b>99.2</b> | 98.1        | 98.6        | <u>99.4</u>    | 97.9        | 98.6        | 99.1         | <u>96.0</u> | <u>97.5</u> |
| Family  | CLIBD  | -           | ✓         | $\mathbb{R}^n$   | <b>97.5</b> | 91.8        | <u>94.6</u> | <b>95.4</b>    | <b>85.7</b> | <b>90.3</b> | <b>94.8</b>  | <b>69.7</b> | <b>80.4</b> |
|         | CL     | -           | ✓         | $\mathbb{H}_L^n$ | 97.1        | 91.8        | 94.4        | <u>94.3</u>    | <u>84.7</u> | <u>89.2</u> | <u>93.9</u>  | 68.1        | 79.0        |
|         | EL+CL  | Pos.        | ✓         | $\mathbb{H}_L^n$ | <u>97.2</u> | 90.6        | 93.8        | 93.5           | 80.3        | 86.4        | 93.2         | 66.4        | 77.6        |
|         | SEL    | Pos.+Neg.   | ✗         | $\mathbb{H}_L^n$ | 97.0        | <b>92.5</b> | <b>94.7</b> | 93.4           | 83.0        | 87.9        | 92.5         | 67.2        | 77.8        |
|         | SEL+CL | Pos.+Neg.   | ✗         | $\mathbb{H}_L^n$ | 96.7        | <u>92.4</u> | 94.5        | 93.6           | 83.9        | 88.5        | 93.0         | 67.5        | 78.2        |
|         | SEL+CL | Pos.+Neg.   | ✓         | $\mathbb{H}_L^n$ | 97.1        | 91.3        | 94.1        | <u>94.3</u>    | 83.3        | 88.5        | 93.8         | <u>68.6</u> | <u>79.2</u> |
| Genus   | CLIBD  | -           | ✓         | $\mathbb{R}^n$   | 94.8        | 85.1        | 89.7        | <b>88.2</b>    | <b>69.0</b> | <b>77.4</b> | <b>87.1</b>  | <b>37.1</b> | <b>52.1</b> |
|         | CL     | -           | ✓         | $\mathbb{H}_L^n$ | <b>95.3</b> | 85.6        | <u>90.2</u> | <u>85.6</u>    | <u>68.0</u> | <u>75.8</u> | <u>86.0</u>  | <u>36.1</u> | <u>50.8</u> |
|         | EL+CL  | Pos.        | ✓         | $\mathbb{H}_L^n$ | <u>95.1</u> | 84.6        | 89.5        | 83.2           | 60.4        | 70.0        | 84.5         | 34.6        | 49.1        |
|         | SEL    | Pos.+Neg.   | ✗         | $\mathbb{H}_L^n$ | 94.0        | <u>86.1</u> | 89.9        | 83.7           | 65.7        | 73.6        | 83.2         | 35.3        | 49.5        |
|         | SEL+CL | Pos.+Neg.   | ✗         | $\mathbb{H}_L^n$ | 94.2        | <b>86.7</b> | <b>90.3</b> | 83.8           | 67.0        | 74.5        | 84.4         | 34.1        | 48.6        |
|         | SEL+CL | Pos.+Neg.   | ✓         | $\mathbb{H}_L^n$ | 95.0        | 85.5        | 90.0        | 84.8           | 65.3        | 73.8        | 84.2         | 35.0        | 49.4        |
| Species | CLIBD  | -           | ✓         | $\mathbb{R}^n$   | 93.0        | 82.0        | 87.2        | <b>77.4</b>    | <b>53.4</b> | <b>63.2</b> | <b>78.3</b>  | 1.9         | 3.6         |
|         | CL     | -           | ✓         | $\mathbb{H}_L^n$ | <b>93.6</b> | 82.7        | <b>87.8</b> | <u>73.3</u>    | <u>52.1</u> | <u>60.9</u> | <u>77.6</u>  | <b>2.4</b>  | <b>4.6</b>  |
|         | EL+CL  | Pos.        | ✓         | $\mathbb{H}_L^n$ | <u>93.5</u> | 81.6        | 87.2        | 69.8           | 44.5        | 54.4        | 75.9         | 1.5         | 2.9         |
|         | SEL    | Pos.+Neg.   | ✗         | $\mathbb{H}_L^n$ | 91.8        | <u>83.2</u> | 87.3        | 71.8           | 50.2        | 59.1        | 75.2         | 1.4         | 2.8         |
|         | SEL+CL | Pos.+Neg.   | ✗         | $\mathbb{H}_L^n$ | 92.1        | <b>83.5</b> | 87.6        | 71.8           | 51.2        | 59.8        | 75.7         | 1.6         | 3.1         |
|         | SEL+CL | Pos.+Neg.   | ✓         | $\mathbb{H}_L^n$ | 93.3        | 82.7        | <u>87.7</u> | 72.7           | 49.4        | 58.8        | 75.6         | <u>2.0</u>  | <u>3.8</u>  |

### 7.3.1 Metrics and Datasets

Experiments are conducted on the BIOSCAN-1M portion of the BIOSCAN-5M (making use of its available partitions and splits). For simplicity, all models are trained on CLIBD’s `train_seen` split of BIOSCAN-1M (36 k samples), which ensures all samples have complete species-level labels. The CLIBD results reported in the experimental results are likewise obtained by training on this same `train_seen` split, rather than using a pretrained CLIBD model. Expanding the experiments to the full BIOSCAN-1M training dataset is left to future work.

Similar to CLIBD, classification performance is evaluated across taxonomic ranks and for both seen and unseen classes, using class-averaged (macro) top-1 accuracy.

### 7.3.2 Results

The hyperbolic SEL strategy is compared against baselines on the BIOSCAN-1M dataset across three retrieval tasks (DNA-to-DNA, Image-to-Image, and Image-to-DNA) evaluated at four taxonomic levels (order, family, genus, and species). The performance of training with contrastive loss (CL) in hyperbolic space is examined in comparison to training in Euclidean space (CLIBD [209]). Subsequently, various training strategies in hyperbolic space are evaluated, including an approach similar to MERU [211] with entailment loss and contrastive losses (EL + CL), contrasted to different variants of SEL. Table 7.2 reports *macro* Top-1 accuracy for seen and unseen taxa, as well as their harmonic mean. Table 7.3 reports *micro* Top-1 accuracy is presented, averaging over individual instances. Compared to the macro accuracy, which treat all classes evenly, the micro accuracy gives more weight to classes with more instances.

Across all retrieval tasks, models achieve high accuracy at the coarsest levels, but this falls off substantially as ranks become more fine-grained, especially for image-based retrieval. Overall, similar trends are seen in the comparative performance of the different methods for both macro and micro averaged results, with the micro averaged accuracy being substantially higher (as the macro averaged accuracy is pulled down by rare

classes). Hyperbolic models consistently achieve results that are comparable to the Euclidean CLIBD baseline across all ranks and retrieval settings. This result is consistent with those of [Chapter 6](#), where the hyperbolic models tended to be comparable or, in many cases, exceed the performance of their Euclidean counterparts. SEL methods consistently perform best at unseen DNA retrieval, whereas the Euclidean model performs best at image retrieval. Comparing EL+CL to SEL+CL (both with full text), SEL+CL always dominates the EL+CL, showing the utility of the stacked entailment over single-layer entailment. Comparing SEL+CL with and without full text, full text supervision improves unimodal seen taxa retrieval, but decreases unseen taxa and cross-modal performance.

## 7.4 Chapter Conclusion

The experiments demonstrate that hyperbolic learning can effectively capture hierarchical structure in biological data and provides performance competitive with established Euclidean methods. These findings align well with those of [Chapter 6](#), wherein the hyperbolic embeddings improved the structure of the latent space. However, neither framework explored in this chapter fully overcomes the persistent challenge of fine-grained, open-world species identification.

Improving classification at fine-grained taxonomic ranks and for novel, unseen taxa remains a key direction for future work. Potential strategies include addressing class imbalance, enhancing data augmentation, or leveraging more advanced hierarchical or uncertainty-aware methods.

## Chapter 8

# Conclusion and Future Work

The work presented here in this thesis represents several contributions to the field of machine learning and steps towards improving insect classification as a contribution to the BIOSCAN project (and the broader fields of conservation biology and ecology). The key contributions include:

- Constructing two large-scale machine-learning-ready insect datasets — BIOSCAN-1M and BIOSCAN-5M — and establishing baseline classification performance ([Chapter 4](#)),
- Investigating the robustness of various neural network architectures to mislabelled data and developing two novel loss functions, designed to improve model robustness to label errors and improve the downstream detection of label errors ([Chapter 5](#)), for which [Figures 5.13](#) and [5.14](#) demonstrate that the proposed robust loss functions operate as intended by avoiding incorrectly steering the optimizer towards fitting to erroneous samples,
- A comparative study between Hyperbolic and conventional Euclidean neural networks, wherein a dying ReLU problem was found to be mitigated by the use of Hyperbolic neural networks, and across all experiments, the Hyperbolic neural networks consistently matched or outperformed their Euclidean counterpart on classification tasks involving hierarchical data ([Chapter 6](#)), and finally

- Developing a multimodal representation learning framework in which Hyperbolic and conventional Euclidean neural networks are compared, and found to be competitive, in their ability to effectively capture hierarchical structures in biological data ([Chapter 7](#)).

The developments of this thesis, though only a single component of the BIOSCAN project and somewhat abstract compared to the more concrete work of insect collection and processing, represent a significant step towards the crucial tracking of changes in insect abundance, biomass, and diversity (possibly even to the species-level) over time and across a multitude of geographical locations. This information is critical to understanding the impacts of ecological change and for formulating strategies for mitigating further catastrophic damage to the global ecosystem, the underlying motivation for this research.

The contributions of this thesis lead to substantial future developments, as outlined below in the Future Work section.

## 8.1 Future Work

Although much technical ground is covered in this thesis, there remain many experiments and research areas that should be more deeply explored in order to gain a better understanding and to properly make technical use of the research presented in this thesis. A brief summary is listed here:

- [Section 8.1.1](#) outlines next steps for *quantitatively* assessing model robustness, exploring other model attributes and hyperparameters in terms of their impacts on robustness, and detecting label errors in the BIOSCAN dataset.
- [Section 8.1.2](#) discusses extensions of the study of Hyperbolic neural networks and the dying ReLU phenomenon found in [Chapter 6](#).
- [Section 8.1.3](#) discusses the use of semi-supervised learning for deeper taxonomic classification.

- [Section 8.1.4](#) discusses a possible direction towards estimating insect biomass from images.

### 8.1.1 Future Work Contending with Label Errors

In [Section 5.2](#), model robustness was evaluated comparatively between architecture types on a qualitative basis, based on how well test accuracy was maintained as dataset corruption rate,  $\eta$ , was increased. In general, accuracy-*vs.*-corruption curves that were as flat as possible (accuracy not impacted as  $\eta$  is increased) and with a sharp cut-off at  $\eta' = 1 - 1/K$  (for balanced  $K$ -class datasets) were viewed as being highly robust. In contrast, if accuracy rolled-off at earlier corruption rates,  $\eta < \eta'$ , the architecture was deemed less robust.

One way in which this study could be extended is by *quantifying* robustness through a set of measures. Motivated by metrics such as 95% rise time [265], ubiquitous in fields of Electrical Engineering and Control Systems, one proposed method for measuring robustness would be to assess the minimal corruption rate required for test accuracy to fall by a specified percentage, e.g., 5%, denoted  $\eta_{-5\%}$ , relative to the zero-corruption baseline accuracy. The larger this metric is, the more robust the model. While this would indicate robustness (i.e., lack of sensitivity to corruption), a shortcoming is that this metric does not indicate at all whether the model performs well relative to other (possibly less robust) models. Indeed, a simple “random guessing” model would be entirely robust, but would be of no practical use. Instead, one would rather have a model with less robustness (rounded accuracy curve) but greater accuracy over the range of corruption rates. Therefore, a second metric could evaluate the *area* under the accuracy curve, thereby combining the effects of flatness and high accuracy. These measures of robustness are illustrated in [Figure 8.1](#) for a sketched accuracy-*vs.*-corruption curve.

Additionally, for the study of Architectural Robustness in [Section 5.2](#), a broader exploration of the impacts on model attributes and hyperparameters would be of interest. Example attributes to be studied include: the depth of the network, the width (size) of network layers, convolutional kernel size, the use of drop-out regularization, max-pooling,

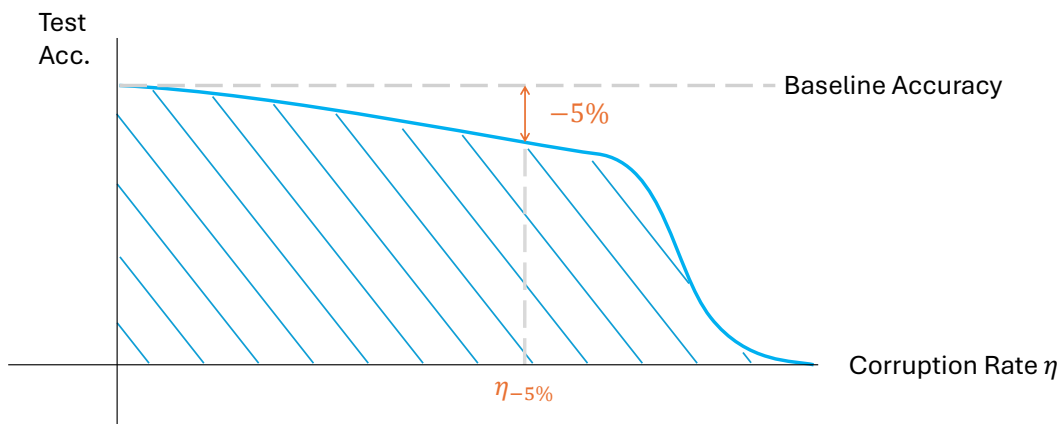


Figure 8.1: Measures of model robustness. Method #1: Measure corruption rate at which the model’s performance (Test Accuracy) drops by 5% relative to the zero-corruption baseline,  $\eta_{-5\%}$ . Method #2: The area under curve combines performance with robustness (penalizing underfit models which tend to be robust but perform poorly).

*etc.* Understanding how these attributes affect robustness could help with the *design* of robust architectures, rather than simply selecting from existing ones.

Lastly, in [Section 5.3](#), Blurry and Piecewise-zero Loss functions were developed to be robust to label errors and to improve the downstream detection of mislabelled data. The experiments within that section tested the detection of label errors in data with *correct* labels available for comparison. Results were promising, and so it would be excellent to attempt detecting mislabelled data within the BIOSCAN datasets. Proposals of mislabelled samples could be generated by existing detection methods, such as CL or AUM, and the use of Blurry or Piecewise-zero loss, and would need to be verified by experts, possibly research collaborators from CBG. Uncovering and correcting these errors would be a highly meaningful step towards improving the dataset for both machine learning use and the overall BIOSCAN project.

### 8.1.2 Future Work on the Study of Hyperbolic Neural Networks

Future work could extend the study of Hyperbolic *vs.* Euclidean Neural Networks, presented in [Chapter 6](#), in multiple ways. The first goal would be to determine the extent to which models trained at greater depths of hierarchy (i.e., representing leaf-nodes) capture the hierarchy of the data, as measured by the ability to later fine-tune on samples at lesser depths (i.e., parent-nodes). Next, explorations on the impacts of network size / depth on performance could be pursued, since only a naïve network architecture was used in the experiments in this thesis. Further study of the Dying ReLU phenomenon would also be of interest. Investigations of where these appear within the network (characterizing the distribution), beyond at the penultimate (embedding) layer, and whether different network architectures may be more or less susceptible could be pursued. Lastly, a logical extension of this analysis would be to move from synthetic data to real-world hierarchical datasets. This proposed future analysis would extend the research to the space of image data, a domain far more complex (in terms of dimensionality, correlations, and structure) than the simplistic points in  $\mathbb{R}^n$  used in the experiments of [Chapter 6](#).

### 8.1.3 Semi-Supervised Learning for Deeper Taxonomic Classification with Few Labelled Data

In the BIOSCAN insect datasets, fewer and fewer labels are present as one goes down the hierarchy into more specific taxonomic ranks (see [Table 4.5](#)). This makes sense, as insects become increasingly similar to one another, making discerning separate groups more and more difficult. Supervised learning, as was used in [Section 4.2](#) to establish a baseline for order- family-level classification, at the most specific ranks (e.g., species) is impossible without labels. To address this limitation in the data, developing a semi-supervised framework would be promising.

To begin, rather than developing and experimenting on the entire BIOSCAN dataset at once — which is too large for rapid iteration — early development should instead be conducted on smaller datasets. However, existing benchmark datasets commonly used for

semi- and self-supervised learning, such as STL-10 [266], are often too simple to serve as effective proxies. Therefore, something in between the difficulty of the BIOSCAN dataset and STL-10 is desired: representative subsets of the BIOSCAN data. One can imagine adjusting the difficulty of the dataset / problem by tuning the phylogenetic diversity or tree-balance of the subset, and therefore creating subsets ranging in difficulty from the level of STL-10 to the full BIOSCAN-5M dataset. One must consider the sub-sampling strategies that will provide training data in a balanced and diverse manner from the taxonomy of the BIOSCAN data. Related to sub-sampling is the measurement of balance and diversity. An option to measure diversity is the Shannon Diversity Index [267], and to measure tree-balance is the Colless Index [268]. This approach would allow one to evaluate how well existing semi-supervised methods perform as the level of data difficulty (i.e., diversity of taxonomy, tree-balance, depth of hierarchy, etc.) is increased, and observe where existing methods break down, thereby leading to an improved understanding of the challenges presented by the BIOSCAN dataset from which targeted developments can be made.

#### 8.1.4 Insect Biomass Estimation<sup>1</sup>

Insect body mass can be an indicator of several life-history traits which can be useful for understanding the relationships between these traits and insect decline [269, 270]. Existing insect decline studies only measure total biomass of bulk samples. This could be made far more specific through the image-based estimation of insect biomass. Therefore, one goal of the BIOSCAN team is to develop a machine learning model capable of estimating insect biomass from the images collected for the overall BIOSCAN project.

Current data available to the BIOSCAN team is a small collection of insect weights (masses) associated with taxonomic classifications and BINs. The BIOSCAN-5M dataset has images and associated with taxonomic classifications and BINs. Therefore, images of the same BIN can be associated to the insect masses in order to train a mass estimation model. Because the dataset of weights has multiple masses per BIN, and the BIOSCAN-

---

<sup>1</sup>Note that preliminary work towards Biomass Estimation was carried out in collaboration with Marcus Chong and Vikram Chandramohan, two Undergraduate Research Assistants.

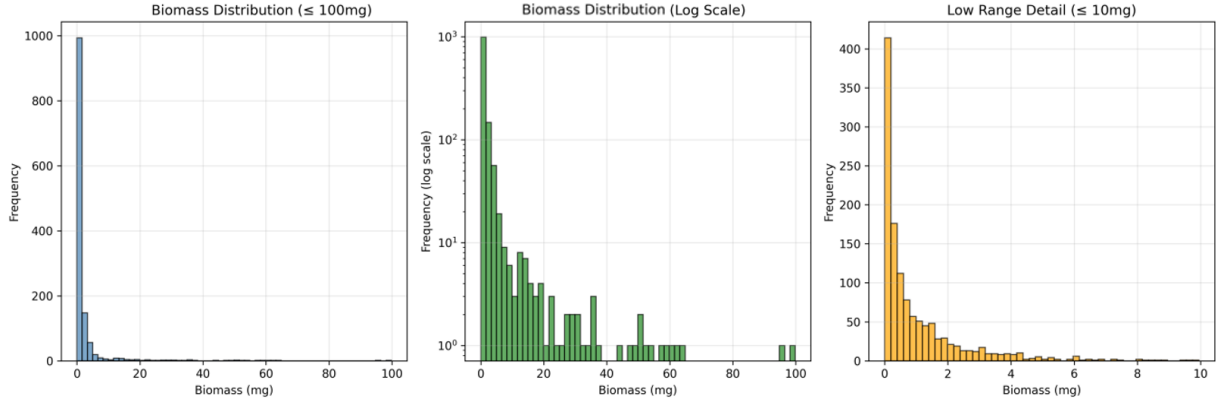


Figure 8.2: The distribution in insect masses. The left panel shows all masses  $\leq 100$  mg. The center panel shows the same range, but plots frequency in log scale. The right panel shows detail at the low end of the range, all masses  $\leq 10$  mg. Observe the long-tailed appearance of this data, whereby most masses are very small, and only a few are large.

5M dataset has multiple images per BIN, this creates a many-to-many association, which may be problematic, as insect sizes (in images) and mass (weight) vary between records. A naïve approach is simply to use the mean mass per BIN, and use all images of that BIN for training. Then, development can proceed with any choice of image-based architecture (e.g., ResNet, ideally with pre-trained weights). The choice of loss function is important, as the standard, Mean Squared Error (MSE), is unfairly biased towards errors on large masses, and the distribution of insect masses appears to be long-tailed, as seen in [Figure 8.2](#). Therefore, a loss function such as Mean Absolute Percent Error [\[271\]](#) (MAPE), would be more appropriate, since error terms are relative to the ground-truth mass (thereby treating small and large insects the same).

# References

- [1] Z. Gharaee *et al.*, “A step towards worldwide biodiversity assessment: The BIOSCAN-1M insect dataset,” in *Advances in Neural Information Processing Systems*, vol. 36, Curran Associates, Inc., 2023, pp. 43 593–43 619. DOI: [10.48550/arXiv.2307.10455](https://arxiv.org/abs/10.48550/arXiv.2307.10455). [Online]. Available: [https://proceedings.neurips.cc/paper\\_files/paper/2023/file/87dbbdc3a685a97ad28489a1d57c45c1-Paper-Datasets\\_and\\_Benchmarks.pdf](https://proceedings.neurips.cc/paper_files/paper/2023/file/87dbbdc3a685a97ad28489a1d57c45c1-Paper-Datasets_and_Benchmarks.pdf). Cited on pages [iv](#), [v](#), [1](#), [5](#), [8](#), [20](#), [30](#), [39](#), [40](#), [44](#), [99](#).
- [2] Z. Gharaee *et al.*, “BIOSCAN-5M: A multimodal dataset for insect biodiversity,” in *Advances in Neural Information Processing Systems*, A. Globerson *et al.*, Eds., vol. 37, Curran Associates, Inc., 2024, pp. 36 285–36 313. [Online]. Available: [https://proceedings.neurips.cc/paper\\_files/paper/2024/file/3fdbb472813041c9ecef04c20c2b1e5a-Paper-Datasets\\_and\\_Benchmarks\\_Track.pdf](https://proceedings.neurips.cc/paper_files/paper/2024/file/3fdbb472813041c9ecef04c20c2b1e5a-Paper-Datasets_and_Benchmarks_Track.pdf). Cited on pages [iv](#), [v](#), [5](#), [8](#), [20](#), [30](#), [41](#), [43](#), [45](#), [47](#), [48](#), [99](#).
- [3] N. Pellegrino, N. Zhao, and P. Fieguth, “The effects of label errors in training data on model performance and overfitting,” *Journal of Computational Vision and Imaging Systems*, vol. 9, no. 1, pp. 26–29, 2023. Cited on pages [iv](#), [v](#), [52](#), [79](#).
- [4] N. Pellegrino, D. Szczecina, and P. Fieguth, “Loss functions robust to the presence of label errors,” *Presented at the 10<sup>th</sup> Annual Conference on Vision and Intelligent Systems*, 2024, arXiv preprint [arXiv:2511.16512](https://arxiv.org/abs/2511.16512). Cited on pages [iv](#), [v](#), [52](#).
- [5] Z. Gong *et al.*, “Hyperbolic multimodal representation learning for biological taxonomies,” in *2nd Beyond Euclidean Workshop: Hyperbolic and Hyperspherical Learning for Computer Vision*. Cited on pages [iv](#), [v](#), [112](#), [114](#).

- [6] P. Halasz, *File: Biological classification l pengo.svg*, [https://en.wikipedia.org/wiki/File:Biological\\_classification\\_L\\_Pengo.svg](https://en.wikipedia.org/wiki/File:Biological_classification_L_Pengo.svg), Mar. 2007. Cited on pages iv, 6.
- [7] T. Ruen and R. Nelson. “Hyperbolic domains.” (Mar. 2020), [Online]. Available: [https://en.wikipedia.org/wiki/File:Hyperbolic\\_domains\\_642.png](https://en.wikipedia.org/wiki/File:Hyperbolic_domains_642.png). Cited on pages iv, 107.
- [8] S. Derbyshire. “Cantor set binary tree.” (Apr. 2016), [Online]. Available: [https://commons.wikimedia.org/wiki/File:Cantor\\_set\\_binary\\_tree.svg](https://commons.wikimedia.org/wiki/File:Cantor_set_binary_tree.svg). Cited on pages iv, 100.
- [9] C. Mora, D. P. Tittensor, S. Adl, A. G. Simpson, and B. Worm, “How many species are there on earth and in the ocean?” *PLoS biology*, vol. 9, no. 8, e1001127, 2011. Cited on page 1.
- [10] P. D. Hebert, A. Cywinska, S. L. Ball, and J. R. DeWaard, “Biological identifications through DNA barcodes,” *Proceedings of the Royal Society of London. Series B: Biological Sciences*, vol. 270, no. 1512, pp. 313–321, 2003. Cited on pages 1, 2, 6, 7, 113.
- [11] T. W. Braukmann *et al.*, “Metabarcoding a diverse arthropod mock community,” *Molecular ecology resources*, vol. 19, no. 3, pp. 711–727, 2019. Cited on pages 1, 6, 113.
- [12] G. Ceballos, P. R. Ehrlich, A. D. Barnosky, A. García, R. M. Pringle, and T. M. Palmer, “Accelerated modern human-induced species losses: Entering the sixth mass extinction,” *Science advances*, vol. 1, no. 5, e1400253, 2015. Cited on page 2.
- [13] G. Ceballos, P. R. Ehrlich, and P. H. Raven, “Vertebrates on the brink as indicators of biological annihilation and the sixth mass extinction,” *Proceedings of the National Academy of Sciences*, vol. 117, no. 24, pp. 13 596–13 602, 2020. Cited on page 2.
- [14] BIOSCAN, Jun. 2022. [Online]. Available: <https://ibol.org/programs/bioscan>. Cited on pages 2, 5.

- [15] N. E. Stork *et al.*, “How many species of insects and other terrestrial arthropods are there on earth,” *Annual review of entomology*, vol. 63, no. 1, pp. 31–45, 2018. Cited on page 2.
- [16] P. D. Hebert *et al.*, “Counting animal species with DNA barcodes: Canadian insects,” *Philosophical Transactions of the Royal Society B: Biological Sciences*, vol. 371, no. 1702, p. 20150333, 2016. Cited on pages 2, 6.
- [17] T. T. Høye *et al.*, “Deep learning and computer vision will transform entomology,” *Proceedings of the National Academy of Sciences*, vol. 118, no. 2, e2002545117, 2021. Cited on page 2.
- [18] C. Linnaeus, *Systema Naturae per regna tria naturae, secundum classes, ordines, genera, species; cum characteribus, differentiis, synonymis, locis*, 10th. Holmiae, Impensis Direct. Laurentii Salvii, 1758-1759, 1758, vol. 1. Cited on page 6.
- [19] A. V. Brower and R. T. Schuh, *Biological systematics: principles and applications*. Cornell University Press, 2021. Cited on pages 6, 98.
- [20] G. C. Griffiths, “On the foundations of biological systematics,” *Acta biotheoretica*, vol. 23, no. 3-4, pp. 85–131, 1974. Cited on pages 6, 98.
- [21] G. Lecointre and H. L. Guyader, *The Tree of Life: A Phylogenetic Classification*. Society of Systematic Zoology, 2007. Cited on pages 6, 98.
- [22] C. Moritz and C. Cicero, “DNA barcoding: Promise and pitfalls,” *PLoS biology*, vol. 2, no. 10, e354, 2004. Cited on pages 6, 7.
- [23] K. M. Ruppert, R. J. Kline, and M. S. Rahman, “Past, present, and future perspectives of environmental DNA (eDNA) metabarcoding: A systematic review in methods, monitoring, and applications of global eDNA,” *Global Ecology and Conservation*, vol. 17, e00547, 2019. Cited on pages 6, 7.
- [24] T. Stoeck, L. Frühe, D. Forster, T. Cordier, C. I. Martins, and J. Pawlowski, “Environmental DNA metabarcoding of benthic bacterial communities indicates the benthic footprint of salmon aquaculture,” *Marine Pollution Bulletin*, vol. 127, pp. 139–149, 2018. Cited on pages 6, 7.

- [25] J. Pawlowski *et al.*, “The future of biotic indices in the ecogenomic era: Integrating (e) DNA metabarcoding in biological assessment of aquatic ecosystems,” *Science of the Total Environment*, vol. 637, pp. 1295–1310, 2018. Cited on pages 6, 7.
- [26] R. R. Sokal, P. H. A. Sneath, *et al.*, “Principles of numerical taxonomy,” *Principles of numerical taxonomy.*, 1963. Cited on page 7.
- [27] M. Blaxter *et al.*, “Defining operational taxonomic units using DNA barcode data,” *Philosophical Transactions of the Royal Society B: Biological Sciences*, vol. 360, no. 1462, pp. 1935–1943, 2005. Cited on page 7.
- [28] S. Ratnasingham and P. D. Hebert, “A DNA-based registry for all animal species: The barcode index number (BIN) system,” *PloS one*, vol. 8, no. 7, e66213, 2013. Cited on pages 7, 31.
- [29] *Barcode of life data system*. [Online]. Available: <https://boldsystems.org/>. Cited on pages 7, 44.
- [30] C. G. A. R. Network *et al.*, “The cancer genome atlas pan-cancer analysis project,” *Nature Genetics*, vol. 45, pp. 1113–20, 2013. DOI: [10.1038/ng.2764](https://doi.org/10.1038/ng.2764). Cited on page 7.
- [31] M.-S. Kim *et al.*, “A draft map of the human proteome,” *Nature*, vol. 509, pp. 575–581, 2014. DOI: [10.1038/nature13302](https://doi.org/10.1038/nature13302). Cited on page 7.
- [32] J. Kattge *et al.*, “TRY – a global database of plant traits,” *Global Change Biology*, vol. 17, no. 9, pp. 2905–2935, 2011. DOI: [10.1111/j.1365-2486.2011.02451.x](https://doi.org/10.1111/j.1365-2486.2011.02451.x). Cited on page 7.
- [33] P. Flicek *et al.*, “Ensembl 2014,” *Nucleic Acids Res.*, vol. 42, pp. D749–55, 2014. DOI: [10.1093/nar/gkt1196](https://doi.org/10.1093/nar/gkt1196). Cited on page 7.
- [34] P. B. Jensen, L. J. Jensen, and S. Brunak, “Mining electronic health records: Towards better research applications and clinical care,” *Nature Reviews Genetics*, vol. 13, pp. 395–405, 2012. DOI: [10.1038/nrg3208](https://doi.org/10.1038/nrg3208). Cited on page 7.

- [35] Y. Lu and S. Young, “A survey of public datasets for computer vision tasks in precision agriculture,” *Computers and Electronics in Agriculture*, vol. 178, p. 105760, 2020. DOI: [10.1016/j.compag.2020.105760](https://doi.org/10.1016/j.compag.2020.105760). Cited on page 7.
- [36] D. Xu, Y. Zhao, X. Hao, and X. Meng, “Pink-Eggs Dataset V1: A step toward invasive species management using deep learning embedded solutions,” 2023. DOI: [10.48550/arxiv.2305.09302](https://doi.org/10.48550/arxiv.2305.09302). arXiv: [2305.09302](https://arxiv.org/abs/2305.09302). Cited on page 7.
- [37] A. Galloway, G. W. Taylor, A. Ramsay, and M. Moussa, “The Ciona17 dataset for semantic segmentation of invasive species in a marine aquaculture environment,” in *Proceedings of the Conference on Computer and Robot Vision (CRV)*, IEEE, 2017, pp. 361–366. DOI: [10.1109/CRV.2017.46](https://doi.org/10.1109/CRV.2017.46). Cited on page 7.
- [38] W. He, K. Han, Y. Nie, C. Wang, and Y. Wang, “Species196: A one-million semi-supervised dataset for fine-grained species recognition,” *Advances in Neural Information Processing Systems*, vol. 36, 2024. Cited on pages 7, 8, 20.
- [39] T. Ilyas *et al.*, “CWD30: A comprehensive and holistic dataset for crop weed recognition in precision agriculture,” 2023. DOI: [10.48550/arxiv.2305.10084](https://doi.org/10.48550/arxiv.2305.10084). arXiv: [2305.10084](https://arxiv.org/abs/2305.10084). Cited on page 8.
- [40] X. Liu, W. Min, S. Mei, L. Wang, and S. Jiang, “Plant disease recognition: A large-scale benchmark dataset and a visual region and loss reweighting approach,” *IEEE Transactions on Image Processing*, vol. 30, pp. 2003–2015, 2021. DOI: [10.1109/TIP.2021.3049334](https://doi.org/10.1109/TIP.2021.3049334). Cited on page 8.
- [41] X. Wu, C. Zhan, Y.-K. Lai, M.-M. Cheng, and J. Yang, “IP102: A large-scale benchmark dataset for insect pest recognition,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 8787–8796. Cited on pages 8, 20, 36.
- [42] N. Kumar *et al.*, “Leafsnap: A computer vision system for automatic plant species identification,” in *Computer Vision—ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7–13, 2012, Proceedings, Part II 12*, Springer, 2012, pp. 502–516. Cited on pages 8, 36.

- [43] G. Van Horn *et al.*, “Building a bird recognition app and large scale dataset with citizen scientists: The fine print in fine-grained dataset collection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 595–604. Cited on pages 8, 36.
- [44] J. D. Wegner, S. Branson, D. Hall, K. Schindler, and P. Perona, “Cataloging public objects using aerial and street-level images-urban trees,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 6014–6023. Cited on pages 8, 36.
- [45] A. Olsen *et al.*, “DeepWeeds: A multiclass weed species image dataset for deep learning,” *Scientific reports*, vol. 9, no. 1, p. 2058, 2019. DOI: [10.1038/s41598-018-38343-3](https://doi.org/10.1038/s41598-018-38343-3). Cited on page 8.
- [46] Q.-J. Wang *et al.*, “Pest24: A large-scale very small object data set of agricultural pests for multi-target detection,” *Computers and Electronics in Agriculture*, vol. 175, p. 105 585, 2020. Cited on pages 8, 36.
- [47] C. Garcin *et al.*, “Pl@ntNet-300K: A plant image dataset with high label ambiguity and a long-tailed distribution,” in *NeurIPS 2021-35th Conference on Neural Information Processing Systems*, 2021. Cited on pages 8, 20, 34, 36, 37.
- [48] G. Van Horn, E. Cole, S. Beery, K. Wilber, S. Belongie, and O. MacAodha, “Benchmarking representation learning for natural world image collections,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 12 879–12 888. DOI: [10.1109/CVPR46437.2021.01269](https://doi.org/10.1109/CVPR46437.2021.01269). Cited on pages 8, 20.
- [49] S. C. Lowe *et al.*, “BenthicNet: A global compilation of seafloor images for deep learning applications,” 2024. DOI: [10.48550/arxiv.2405.05241](https://doi.org/10.48550/arxiv.2405.05241). arXiv: [2405.05241 \[cs.CV\]](https://arxiv.org/abs/2405.05241). Cited on page 8.
- [50] H. Nguyen, T. Truong, X. Nguyen, A. Dowling, X. Li, and K. Luu, “Insect-Foundation: A foundation model and large-scale 1M dataset for visual insect understanding,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Los Alamitos, CA, USA: IEEE Computer Society, Jun. 2024, pp. 21 945–21 955. DOI: [10.1109/CVPR52733.2024.02072](https://doi.org/10.1109/CVPR52733.2024.02072). Cited on pages 8, 20.

- [51] K. Wang *et al.*, “New image dataset and new negative sample judgment method for crop pest recognition based on deep learning models,” *Ecological Informatics*, vol. 69, p. 101620, 2022. Cited on page 8.
- [52] S. Dong *et al.*, “Automatic crop pest detection oriented multiscale feature fusion approach,” *Insects*, vol. 13, no. 6, p. 554, 2022. Cited on page 8.
- [53] S. Xing and H. J. Lee, “Crop pests and diseases recognition using DANet with TLDP,” *Computers and Electronics in Agriculture*, vol. 199, p. 107144, 2022. Cited on page 8.
- [54] J. C. Gomes and D. L. Borges, “Insect pest image recognition: A few-shot machine learning approach including maturity stages classification,” *Agronomy*, vol. 12, no. 8, p. 1733, 2022. Cited on pages 8, 36.
- [55] L. Liu, R. Wang, C. Xie, R. Li, F. Wang, and L. Qi, “A global activated feature pyramid network for tiny pest detection in the wild,” *Machine Vision and Applications*, vol. 33, no. 5, p. 76, 2022. Cited on page 8.
- [56] Z. Li, X. Jiang, X. Jia, X. Duan, Y. Wang, and J. Mu, “Classification method of significant rice pests based on deep learning,” *Agronomy*, vol. 12, no. 9, p. 2096, 2022. Cited on page 8.
- [57] A. Affouard, H. Goëau, P. Bonnet, J.-C. Lombardo, and A. Joly, “Pl@ntNet app in the era of deep learning,” in *ICLR: International Conference on Learning Representations*, 2017. Cited on page 8.
- [58] M. Martineau, D. Conte, R. Raveaux, I. Arnault, D. Munier, and G. Venturini, “A survey on image-based insect classification,” *Pattern Recognition*, vol. 65, pp. 273–284, 2017. Cited on page 9.
- [59] W. Li, T. Zheng, Z. Yang, M. Li, C. Sun, and X. Yang, “Classification and detection of insects from field images using deep learning for smart pest management: A systematic review,” *Ecological Informatics*, vol. 66, p. 101460, 2021. Cited on page 9.

- [60] F. A. G. d. Silveira, E. C. Tetila, G. Astolfi, A. B. d. Costa, and W. P. Amorim, “Performance analysis of yolov3 for real-time detection of pests in soybeans,” in *Intelligent Systems: 10th Brazilian Conference, BRACIS 2021, Virtual Event, November 29–December 3, 2021, Proceedings, Part II*, Springer, 2021, pp. 265–279. Cited on page 9.
- [61] T. Domingues, T. Brandão, and J. C. Ferreira, “Machine learning for detection and prediction of crop diseases and pests: A comprehensive survey,” *Agriculture*, vol. 12, no. 9, p. 1350, 2022. Cited on page 9.
- [62] S. Couliably, B. Kamsu-Foguem, D. Kamissoko, and D. Traore, “Deep learning for precision agriculture: A bibliometric analysis,” *Intelligent Systems with Applications*, p. 200 102, 2022. Cited on page 9.
- [63] Z. Miao, G. Huang, N. Li, T. Sun, and Y. Wei, “A review of plant disease and insect pest detection based on deep learning,” in *Proceedings of 2022 Chinese Intelligent Systems Conference: Volume II*, Springer, 2022, pp. 103–118. Cited on page 9.
- [64] Y. Ji, Z. Zhou, H. Liu, and R. V. Davuluri, “DNABERT: Pre-trained bidirectional encoder representations from transformers model for DNA-language in genome,” *Bioinformatics*, vol. 37, no. 15, pp. 2112–2120, 2021. Cited on page 9.
- [65] P. M. Nadkarni, L. Ohno-Machado, and W. W. Chapman, “Natural language processing: An introduction,” *Journal of the American Medical Informatics Association*, vol. 18, no. 5, pp. 544–551, 2011. Cited on page 9.
- [66] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” 2018. arXiv: [1810.04805](https://arxiv.org/abs/1810.04805). Cited on page 9.
- [67] M. A. Helaly, S. Rady, and M. M. Aref, “BERT contextual embeddings for taxonomic classification of bacterial DNA sequences,” *Expert Systems with Applications*, vol. 208, p. 117972, 2022. Cited on page 9.

- [68] F. Mock, F. Kretschmer, A. Krieser, S. Böcker, and M. Marz, “Taxonomic classification of DNA sequences beyond sequence similarity using deep neural networks,” *Proceedings of the National Academy of Sciences*, vol. 119, no. 35, e2122636119, 2022. Cited on page 9.
- [69] S. Badirli, Z. Akata, G. Mohler, C. Picard, and M. Dundar, “Fine-Grained Zero-Shot learning with DNA as side information,” in *Advances in Neural Information Processing Systems 34 (NeurIPS 2021)*, Dec. 2021. Cited on page 9.
- [70] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016. [Online]. Available: <http://www.deeplearningbook.org>. Cited on pages 9, 12, 13, 17, 19, 52.
- [71] W. Yan, *Computational methods for deep learning*. Springer, 2021. Cited on pages 9, 12, 13, 19.
- [72] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *nature*, vol. 521, no. 7553, pp. 436–444, 2015. Cited on pages 9, 10, 13, 52.
- [73] K. O’Shea and R. Nash, “An introduction to convolutional neural networks,” 2015. arXiv: [1511.08458](https://arxiv.org/abs/1511.08458). Cited on page 9.
- [74] J. Gu *et al.*, “Recent advances in convolutional neural networks,” *Pattern recognition*, vol. 77, pp. 354–377, 2018. Cited on page 9.
- [75] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, “A survey of convolutional neural networks: Analysis, applications, and prospects,” *IEEE transactions on neural networks and learning systems*, 2021. Cited on page 9.
- [76] S. Ruder, “An overview of gradient descent optimization algorithms,” 2016. arXiv: [1609.04747](https://arxiv.org/abs/1609.04747). Cited on page 10.
- [77] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” 2014. arXiv: [1412.6980](https://arxiv.org/abs/1412.6980). Cited on pages 10, 80, 121.
- [78] T. M. Cover, *Elements of information theory*. John Wiley & Sons, 1999. Cited on pages 10, 13.

- [79] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*. Springer, 2006, vol. 4. Cited on pages [10](#), [12](#), [13](#), [19](#).
- [80] R. Szeliski, *Computer vision: algorithms and applications*. Springer Nature, 2022. Cited on page [10](#).
- [81] E. R. Davies, *Computer and machine vision: theory, algorithms, practicalities*. Academic Press, 2012. Cited on page [10](#).
- [82] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” *Advances in neural information processing systems*, vol. 25, 2012. Cited on pages [11](#), [52](#), [65](#).
- [83] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” 2014. arXiv: [1409.1556](#). Cited on page [11](#).
- [84] C. Szegedy *et al.*, “Going deeper with convolutions,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9. Cited on page [11](#).
- [85] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778. Cited on pages [11](#), [33](#), [62](#), [65](#), [70](#), [80](#), [81](#).
- [86] W. Zhiqiang and L. Jun, “A review of object detection based on convolutional neural network,” in *2017 36th Chinese control conference (CCC)*, IEEE, 2017, pp. 11 104–11 109. Cited on page [11](#).
- [87] A. Dhillon and G. K. Verma, “Convolutional neural network: A review of models, methodologies and applications to object detection,” *Progress in Artificial Intelligence*, vol. 9, no. 2, pp. 85–112, 2020. Cited on page [11](#).
- [88] R. Kaur and S. Singh, “A comprehensive review of object detection with deep learning,” *Digital Signal Processing*, p. 103 812, 2022. Cited on page [11](#).

- [89] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587. Cited on page 11.
- [90] R. Girshick, “Fast r-cnn,” in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448. Cited on page 11.
- [91] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” *Advances in neural information processing systems*, vol. 28, 2015. Cited on page 11.
- [92] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788. Cited on page 11.
- [93] J. Redmon and A. Farhadi, “Yolo9000: Better, faster, stronger,” *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-January, pp. 6517–6525, Dec. 2016. DOI: [10.1109/CVPR.2017.690](https://doi.org/10.1109/CVPR.2017.690). [Online]. Available: <https://arxiv.org/abs/1612.08242v1>. Cited on page 11.
- [94] J. Redmon and A. Farhadi, *Yolov3: An incremental improvement*, Apr. 2018. [Online]. Available: <https://arxiv.org/abs/1804.02767v1>. Cited on page 11.
- [95] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, *Yolov4: Optimal speed and accuracy of object detection*, Apr. 2020. [Online]. Available: <https://arxiv.org/abs/2004.10934v1>. Cited on page 11.
- [96] C. Li *et al.*, *Yolov6: A single-stage object detection framework for industrial applications*, Sep. 2022. [Online]. Available: <https://arxiv.org/abs/2209.02976v1>. Cited on page 11.
- [97] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, *Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors*, Jul. 2022. [Online]. Available: <https://arxiv.org/abs/2207.02696v1>. Cited on page 11.

- [98] G. Jocher, A. Chaurasia, and J. Qiu, *YOLO by Ultralytics*, version 8.0.0, Jan. 2023. [Online]. Available: <https://github.com/ultralytics/ultralytics>. Cited on page 11.
- [99] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, “Focal loss for dense object detection,” in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988. Cited on pages 11, 22, 75–77, 80.
- [100] V. Nasteski, “An overview of the supervised machine learning methods,” *Horizons*, vol. 4, pp. 51–62, 2017. Cited on pages 11, 52.
- [101] A. Mathew, P. Amudha, and S. Sivakumari, “Deep learning techniques: An overview,” *Advanced Machine Learning Technologies and Applications: Proceedings of AMLTA 2020*, pp. 599–608, 2021. Cited on pages 11, 52.
- [102] H. B. Barlow, “Unsupervised learning,” *Neural computation*, vol. 1, no. 3, pp. 295–311, 1989. Cited on page 11.
- [103] K. Sindhu Meena and S. Suriya, “A survey on supervised and unsupervised learning techniques,” in *Proceedings of international conference on artificial intelligence, smart grid and smart city applications: AISGSC 2019*, Springer, 2020, pp. 627–644. Cited on page 11.
- [104] A. Saxena *et al.*, “A review of clustering techniques and developments,” *Neurocomputing*, vol. 267, pp. 664–681, 2017. Cited on page 12.
- [105] X. Huang, L. Wu, and Y. Ye, “A review on dimensionality reduction techniques,” *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 33, no. 10, p. 1950017, 2019. Cited on page 12.
- [106] R. Zebari, A. Abdulazeez, D. Zeebaree, D. Zebari, and J. Saeed, “A comprehensive review of dimensionality reduction techniques for feature selection and feature extraction,” *Journal of Applied Science and Technology Trends*, vol. 1, no. 2, pp. 56–70, 2020. Cited on page 12.
- [107] X. J. Zhu, “Semi-supervised learning literature survey,” *University of Wisconsin-Madison Department of Computer Sciences*, 2005. Cited on page 12.

- [108] O. Chapelle, B. Scholkopf, and A. Zien, “Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews],” *IEEE Transactions on Neural Networks*, vol. 20, no. 3, pp. 542–542, 2009. Cited on page [12](#).
- [109] Y. Ouali, C. Hudelot, and M. Tami, “An overview of deep semi-supervised learning,” 2020. arXiv: [2006.05278](#). Cited on page [12](#).
- [110] J. E. Van Engelen and H. H. Hoos, “A survey on semi-supervised learning,” *Machine learning*, vol. 109, no. 2, pp. 373–440, 2020. Cited on page [12](#).
- [111] R. S. Michalski, “A theory and methodology of inductive learning,” in *Machine learning*, Elsevier, 1983, pp. 83–134. Cited on page [12](#).
- [112] M. J. Prince and R. M. Felder, “Inductive teaching and learning methods: Definitions, comparisons, and research bases,” *Journal of engineering education*, vol. 95, no. 2, pp. 123–138, 2006. Cited on page [12](#).
- [113] T. Joachims *et al.*, “Transductive inference for text classification using support vector machines,” in *Icml*, vol. 99, 1999, pp. 200–209. Cited on page [12](#).
- [114] A. Arnold, R. Nallapati, and W. W. Cohen, “A comparative study of methods for transductive transfer learning,” in *Seventh IEEE international conference on data mining workshops (ICDMW 2007)*, IEEE, 2007, pp. 77–82. Cited on page [12](#).
- [115] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” in *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, 2019, pp. 4171–4186. Cited on page [12](#).
- [116] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, “A simple framework for contrastive learning of visual representations,” in *International conference on machine learning*, PmLR, 2020, pp. 1597–1607. Cited on page [12](#).
- [117] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, “Momentum contrast for unsupervised visual representation learning,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 9729–9738. Cited on page [12](#).

- [118] A. Radford *et al.*, “Learning transferable visual models from natural language supervision,” in *International Conference on Machine Learning*, PMLR, 2021, pp. 8748–8763. DOI: [10.48550/arXiv.2103.00020](https://doi.org/10.48550/arXiv.2103.00020). Cited on pages [12](#), [25](#), [115](#).
- [119] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, “Masked autoencoders are scalable vision learners,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 16 000–16 009. Cited on page [12](#).
- [120] J. Zhang, L. Yang, S. M. S. Mohammadabadi, and F. Yan, “A survey on self-supervised learning: Recent advances and open problems,” *Neurocomputing*, p. 131 409, 2025. Cited on page [12](#).
- [121] F. Nazarieh, Z. Feng, M. Awais, W. Wang, and J. Kittler, “A survey of cross-modal visual content generation,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 34, no. 8, pp. 6814–6832, 2024. Cited on page [12](#).
- [122] J. S. Bridle, “Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition,” in *Neurocomputing: Algorithms, architectures and applications*, Springer, 1990, pp. 227–236. Cited on page [13](#).
- [123] K. Banerjee, R. R. Gupta, K. Vyas, B. Mishra, *et al.*, “Exploring alternatives to softmax function,” 2020. arXiv: [2011.11538](https://arxiv.org/abs/2011.11538). Cited on page [13](#).
- [124] N. Japkowicz and S. Stephen, “The class imbalance problem: A systematic study,” *Intelligent data analysis*, vol. 6, no. 5, pp. 429–449, 2002. Cited on page [14](#).
- [125] B. Krawczyk, “Learning from imbalanced data: Open challenges and future directions,” *Progress in Artificial Intelligence*, vol. 5, no. 4, pp. 221–232, 2016. Cited on page [14](#).
- [126] L. Yang, H. Jiang, Q. Song, and J. Guo, “A survey on long-tailed visual recognition,” *International Journal of Computer Vision*, vol. 130, no. 7, pp. 1837–1872, 2022. Cited on page [14](#).

- [127] Y. Cui, M. Jia, T.-Y. Lin, Y. Song, and S. Belongie, “Class-balanced loss based on effective number of samples,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 9268–9277. Cited on page 14.
- [128] Z. Liu, Z. Miao, X. Zhan, J. Wang, B. Gong, and S. X. Yu, “Large-scale long-tailed recognition in an open world,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 2537–2546. Cited on page 14.
- [129] H. Castrup, “Distributions for uncertainty analysis,” in *Proc. Int. Dimensional Workshop*, 2001, pp. 1–12. Cited on page 14.
- [130] C. Gini, “Variabilitae mutabilita. reprinted in memorie di metodologica statistica,” *Rome: Libreria Eredi Virgilio Veschi*, vol. 1, 1912. Cited on pages 14, 36.
- [131] M. O. Lorenz, “Methods of measuring the concentration of wealth,” *Publications of the American statistical association*, vol. 9, no. 70, pp. 209–219, 1905. Cited on page 14.
- [132] C. N. Silla and A. A. Freitas, “A survey of hierarchical classification across different application domains,” *Data Mining and Knowledge Discovery*, vol. 22, pp. 31–72, 2011. Cited on pages 16, 17, 97.
- [133] A. Naik and H. Rangwala, *Large scale hierarchical classification: state of the art*. Springer, 2018. Cited on pages 16, 17.
- [134] P. M. Rezende, J. S. Xavier, D. B. Ascher, G. R. Fernandes, and D. E. Pires, “Evaluating hierarchical machine learning approaches to classify biological databases,” *Briefings in Bioinformatics*, vol. 23, no. 4, bbac216, 2022. Cited on page 16.
- [135] X.-S. Wei *et al.*, “Fine-grained image analysis with deep learning: A survey,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 44, no. 12, pp. 8927–8948, 2021. Cited on page 16.
- [136] B. Zhao, J. Feng, X. Wu, and S. Yan, “A survey on deep learning-based fine-grained object classification and semantic segmentation,” *International Journal of Automation and Computing*, vol. 14, no. 2, pp. 119–135, 2017. Cited on page 16.

- [137] C. Qiu and W. Zhou, “A survey of recent advances in cnn-based fine-grained visual categorization,” in *2020 IEEE 20th International Conference on Communication Technology (ICCT)*, IEEE, 2020, pp. 1377–1384. Cited on page 16.
- [138] B. Arslan, S. Memiş, E. B. Sönmez, and O. Z. Batur, “Fine-grained food classification methods on the uec food-100 database,” *IEEE Transactions on Artificial Intelligence*, vol. 3, no. 2, pp. 238–243, 2021. Cited on page 16.
- [139] Y. Cui, Z. Gu, D. Mahajan, L. Van Der Maaten, S. Belongie, and S.-N. Lim, “Measuring dataset granularity,” 2019. arXiv: [1912.10154](#). Cited on page 16.
- [140] D. Koller and M. Sahami, “Hierarchically classifying documents using very few words,” Stanford InfoLab, Tech. Rep., 1997. Cited on page 16.
- [141] S. Kiritchenko, *Hierarchical text categorization and its application to bioinformatics*. University of Ottawa, 2006. Cited on page 16.
- [142] N. Cesa-Bianchi, C. Gentile, A. Tironi, and L. Zaniboni, “Incremental algorithms for hierarchical classification,” *Advances in neural information processing systems*, vol. 17, 2004. Cited on pages 16, 17.
- [143] D. Gao, W. Yang, H. Zhou, Y. Wei, Y. Hu, and H. Wang, “Deep hierarchical classification for category prediction in e-commerce system,” 2020. arXiv: [2005.06692](#). Cited on page 17.
- [144] R. La Grassa, I. Gallo, and N. Landro, “Learn class hierarchy using convolutional neural networks,” *Applied Intelligence*, vol. 51, no. 10, pp. 6622–6632, 2021. Cited on page 17.
- [145] K. Bjerge *et al.*, “Hierarchical classification of insects with multitask learning and anomaly detection,” *Ecological Informatics*, vol. 77, p. 102278, 2023. Cited on page 17.
- [146] A. Naik and H. Rangwala, “Improving large-scale hierarchical classification by rewiring: A data-driven filter based approach,” *Journal of Intelligent Information Systems*, vol. 52, no. 1, pp. 141–164, 2019. Cited on page 17.

- [147] M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks,” in *European conference on computer vision*, Springer, 2014, pp. 818–833. Cited on page [17](#).
- [148] O. Ganea, G. Bécigneul, and T. Hofmann, “Hyperbolic neural networks,” *Advances in neural information processing systems*, vol. 31, 2018. Cited on pages [17](#), [24](#), [25](#), [96](#), [98](#).
- [149] T. Yu and H. Zhu, “Hyper-parameter optimization: A review of algorithms and applications,” 2020. arXiv: [2003.05689](#). Cited on page [18](#).
- [150] L. Yang and A. Shami, “On hyperparameter optimization of machine learning algorithms: Theory and practice,” *Neurocomputing*, vol. 415, pp. 295–316, 2020. Cited on page [18](#).
- [151] P. Ren *et al.*, “A comprehensive survey of neural architecture search: Challenges and solutions,” *ACM Computing Surveys (CSUR)*, vol. 54, no. 4, pp. 1–34, 2021. Cited on page [18](#).
- [152] D. Baymurzina, E. Golikov, and M. Burtsev, “A review of neural architecture search,” *Neurocomputing*, vol. 474, pp. 82–93, 2022. Cited on page [18](#).
- [153] K. T. Chitty-Venkata and A. K. Somani, “Neural architecture search survey: A hardware perspective,” *ACM Computing Surveys*, vol. 55, no. 4, pp. 1–36, 2022. Cited on page [18](#).
- [154] K. Weiss, T. M. Khoshgoftaar, and D. Wang, “A survey of transfer learning,” *Journal of Big data*, vol. 3, no. 1, pp. 1–40, 2016. Cited on page [18](#).
- [155] G. Csurka, “A comprehensive survey on domain adaptation for visual applications,” *Domain adaptation in computer vision applications*, pp. 1–35, 2017. Cited on page [18](#).
- [156] S. Niu, Y. Liu, J. Wang, and H. Song, “A decade survey of transfer learning (2010–2020),” *IEEE Transactions on Artificial Intelligence*, vol. 1, no. 2, pp. 151–166, 2020. Cited on page [18](#).

- [157] F. Zhuang *et al.*, “A comprehensive survey on transfer learning,” *Proceedings of the IEEE*, vol. 109, no. 1, pp. 43–76, 2020. Cited on page [18](#).
- [158] C. Shorten and T. M. Khoshgoftaar, “A survey on image data augmentation for deep learning,” *Journal of big data*, vol. 6, no. 1, pp. 1–48, 2019. Cited on page [19](#).
- [159] A. Mikołajczyk and M. Grochowski, “Data augmentation for improving deep learning in image classification problem,” in *2018 international interdisciplinary PhD workshop (IIPhDW)*, IEEE, 2018, pp. 117–122. Cited on page [19](#).
- [160] N. E. Khalifa, M. Loey, and S. Mirjalili, “A comprehensive survey of recent trends in deep learning for digital images augmentation,” *Artificial Intelligence Review*, pp. 1–27, 2022. Cited on page [19](#).
- [161] J. Wang, L. Perez, *et al.*, “The effectiveness of data augmentation in image classification using deep learning,” *Convolutional Neural Networks Vis. Recognit*, vol. 11, no. 2017, pp. 1–8, 2017. Cited on page [19](#).
- [162] C. Northcutt, L. Jiang, and I. Chuang, “Confident learning: Estimating uncertainty in dataset labels,” *Journal of Artificial Intelligence Research*, vol. 70, pp. 1373–1411, 2021. Cited on pages [19](#), [21](#), [52](#), [75](#), [81](#), [82](#).
- [163] C. G. Northcutt, A. Athalye, and J. Mueller, “Pervasive label errors in test sets destabilize machine learning benchmarks,” *Advances in Neural Information Processing Systems*, 2021. Cited on pages [19](#), [20](#), [52](#), [53](#), [56](#), [84](#).
- [164] G. Pleiss, T. Zhang, E. Elenberg, and K. Q. Weinberger, “Identifying mislabeled data using the area under the margin ranking,” *Advances in Neural Information Processing Systems*, vol. 33, pp. 17 044–17 056, 2020. Cited on pages [19–21](#), [75](#), [81](#), [82](#), [89](#), [91](#).
- [165] Z. Zhang and M. Sabuncu, “Generalized cross entropy loss for training deep neural networks with noisy labels,” *Advances in neural information processing systems*, vol. 31, 2018. Cited on pages [19](#), [23](#), [53](#), [76](#), [79](#), [80](#).

- [166] X. Ye, X. Li, T. Liu, Y. Sun, W. Tong, *et al.*, “Active negative loss functions for learning with noisy labels,” *Advances in Neural Information Processing Systems*, vol. 36, pp. 6917–6940, 2023. Cited on pages 19, 23, 53, 76, 79, 80.
- [167] L. Deng, “The mnist database of handwritten digit images for machine learning research,” *IEEE signal processing magazine*, vol. 29, no. 6, pp. 141–142, 2012. Cited on pages 20, 52, 53, 55, 66, 79, 103.
- [168] A. Krizhevsky and G. Hinton, “Learning multiple layers of features from tiny images,” University of Toronto, Toronto, Ontario, Tech. Rep. 0, 2009. [Online]. Available: <https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf>. Cited on pages 20, 52, 53, 66, 79.
- [169] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *2009 IEEE conference on computer vision and pattern recognition*, IEEE, 2009, pp. 248–255. Cited on pages 20, 52, 53.
- [170] H. Song, M. Kim, D. Park, Y. Shin, and J.-G. Lee, “Learning from noisy labels with deep neural networks: A survey,” *IEEE transactions on neural networks and learning systems*, vol. 34, no. 11, pp. 8135–8153, 2022. Cited on pages 20, 23.
- [171] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, “Understanding deep learning requires rethinking generalization,” 2016. arXiv: 1611.03530. Cited on page 20.
- [172] L. Jiang, Z. Zhou, T. Leung, L.-J. Li, and L. Fei-Fei, “MentorNet: Learning data-driven curriculum for very deep neural networks on corrupted labels,” in *International conference on machine learning*, PMLR, 2018, pp. 2304–2313. Cited on pages 21, 75.
- [173] H. Song, M. Kim, and J.-G. Lee, “Selfie: Refurbishing unclear samples for robust deep learning,” in *International conference on machine learning*, PMLR, 2019, pp. 5907–5915. Cited on pages 21, 75.
- [174] Y. Kim, J. Yim, J. Yun, and J. Kim, “Nlnl: Negative learning for noisy labels,” in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 101–110. Cited on pages 21, 75.

- [175] I. J. Good, “Rational decisions,” *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 14, no. 1, pp. 107–114, 1952. Cited on pages [22](#), [24](#), [75–77](#), [80](#), [81](#).
- [176] A. Ghosh, H. Kumar, and P. S. Sastry, “Robust loss functions under label noise for deep neural networks,” in *Proceedings of the AAAI conference on artificial intelligence*, vol. 31, 2017. Cited on page [23](#).
- [177] G. Patrini, A. Rozza, A. Krishna Menon, R. Nock, and L. Qu, “Making deep neural networks robust to label noise: A loss correction approach,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1944–1952. Cited on page [23](#).
- [178] A. Ghosh, N. Manwani, and P. Sastry, “Making risk minimization tolerant to label noise,” *Neurocomputing*, vol. 160, pp. 93–107, 2015. Cited on page [23](#).
- [179] G. E. Box and D. R. Cox, “An analysis of transformations,” *Journal of the Royal Statistical Society Series B: Statistical Methodology*, vol. 26, no. 2, pp. 211–243, 1964. Cited on page [23](#).
- [180] X. Ma, H. Huang, Y. Wang, S. Romano, S. Erfani, and J. Bailey, “Normalized loss functions for deep learning with noisy labels,” in *International conference on machine learning*, PMLR, 2020, pp. 6543–6553. Cited on pages [23](#), [24](#).
- [181] Y. Wang, X. Ma, Z. Chen, Y. Luo, J. Yi, and J. Bailey, “Symmetric cross entropy for robust learning with noisy labels,” in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 322–330. Cited on page [24](#).
- [182] T. Pang, C. Du, Y. Dong, and J. Zhu, “Towards robust detection of adversarial examples,” *Advances in neural information processing systems*, vol. 31, 2018. Cited on page [24](#).
- [183] A. K. Menon, A. S. Rawat, S. J. Reddi, and S. Kumar, “Can gradient clipping mitigate label noise?” In *International conference on learning representations*, 2020. Cited on page [24](#).

- [184] X. Zhou, X. Liu, J. Jiang, X. Gao, and X. Ji, “Asymmetric loss functions for learning with noisy labels,” in *International conference on machine learning*, PMLR, 2021, pp. 12 846–12 856. Cited on page 24.
- [185] Y. Lyu and I. W. Tsang, “Curriculum loss: Robust learning and generalization against label corruption,” in *International Conference on Learning Representations*, 2019. Cited on page 24.
- [186] J. Lamping, R. Rao, and P. Pirolli, “A focus+ context technique based on hyperbolic geometry for visualizing large hierarchies,” in *Proceedings of the SIGCHI conference on Human factors in computing systems*, 1995, pp. 401–408. Cited on page 24.
- [187] M. Nickel and D. Kiela, “Poincaré embeddings for learning hierarchical representations,” *Advances in neural information processing systems*, vol. 30, 2017. Cited on page 24.
- [188] F. Sala, C. De Sa, A. Gu, and C. Ré, “Representation tradeoffs for hyperbolic embeddings,” in *International conference on machine learning*, PMLR, 2018, pp. 4460–4469. Cited on pages 24, 96, 98.
- [189] O. Ganea, G. Bécigneul, and T. Hofmann, “Hyperbolic entailment cones for learning hierarchical embeddings,” in *International Conference on Machine Learning*, PMLR, 2018, pp. 1646–1655. Cited on page 24.
- [190] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, “Geometric deep learning: Going beyond euclidean data,” *IEEE Signal Processing Magazine*, vol. 34, no. 4, pp. 18–42, 2017. Cited on page 24.
- [191] V. Khrulkov, L. Mirvakhabova, E. Ustinova, I. Oseledets, and V. Lempitsky, “Hyperbolic image embeddings,” in *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2020. Cited on pages 24, 99.
- [192] A. Paszke *et al.*, “Automatic differentiation in pytorch,” in *NIPS-W*, 2017. Cited on page 24.

- [193] A. Ermolov, L. Mirvakhabova, V. Khrulkov, N. Sebe, and I. Oseledets, “Hyperbolic vision transformers: Combining improvements in metric learning,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 7409–7419. Cited on page [25](#).
- [194] Y. Bengio, P. Simard, and P. Frasconi, “Learning long-term dependencies with gradient descent is difficult,” *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994. Cited on page [25](#).
- [195] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997. Cited on page [25](#).
- [196] A. F. Agarap, “Deep learning using rectified linear units (relu),” 2018. arXiv: [1803.08375](#). Cited on pages [25](#), [97](#), [103](#), [106](#).
- [197] L. Lu, Y. Shin, Y. Su, and G. E. Karniadakis, “Dying relu and initialization: Theory and numerical examples,” 2019. arXiv: [1903.06733](#). Cited on pages [25](#), [97](#), [103](#), [106](#).
- [198] Y. Guo, X. Wang, Y. Chen, and S. X. Yu, “Clipped hyperbolic classifiers are super-hyperbolic classifiers,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 11–20. Cited on pages [25](#), [104](#).
- [199] Q. Liu, M. Nickel, and D. Kiela, “Hyperbolic graph neural networks,” *Advances in neural information processing systems*, vol. 32, 2019. Cited on page [25](#).
- [200] I. Chami, Z. Ying, C. Ré, and J. Leskovec, “Hyperbolic graph convolutional neural networks,” *Advances in neural information processing systems*, vol. 32, 2019. Cited on page [25](#).
- [201] W. Chen *et al.*, “Fully hyperbolic neural networks,” 2021. arXiv: [2105.14686](#). Cited on page [25](#).
- [202] R. Shimizu, Y. Mukuta, and T. Harada, “Hyperbolic neural networks++,” 2020. arXiv: [2006.08210](#). Cited on pages [25](#), [96](#), [97](#).
- [203] P. Tabaghi and I. Dokmanić, “Hyperbolic distance matrices,” in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 1728–1738. Cited on page [25](#).

- [204] I. Chami, A. Gu, D. P. Nguyen, and C. Ré, “Horopca: Hyperbolic dimensionality reduction via horospherical projections,” in *International Conference on Machine Learning*, PMLR, 2021, pp. 1419–1429. Cited on page 25.
- [205] C. Jia *et al.*, “Scaling up visual and vision-language representation learning with noisy text supervision,” in *International conference on machine learning*, PMLR, 2021, pp. 4904–4916. Cited on pages 25, 115.
- [206] X. Zhai, B. Mustafa, A. Kolesnikov, and L. Beyer, “Sigmoid loss for language image pre-training,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct. 2023, pp. 11 975–11 986. DOI: [10.48550/arXiv.2303.15343](https://doi.org/10.48550/arXiv.2303.15343). Cited on page 25.
- [207] S. Stevens *et al.*, “BioCLIP: A vision foundation model for the tree of life,” in *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Los Alamitos, CA, USA: IEEE Computer Society, Jun. 2024, pp. 19 412–19 424. DOI: [10.1109/CVPR52733.2024.01836](https://doi.org/10.1109/CVPR52733.2024.01836). Cited on pages 25, 113.
- [208] J. Gu *et al.*, “BioCLIP 2: Emergent properties from scaling hierarchical contrastive learning,” 2025. DOI: [10.48550/arxiv.2505.23883](https://doi.org/10.48550/arxiv.2505.23883). arXiv: [2505.23883 \[cs.CV\]](https://arxiv.org/abs/2505.23883). Cited on page 25.
- [209] Z. Gong *et al.*, “CLIBD: Bridging vision and genomics for biodiversity monitoring at scale,” in *International Conference on Learning Representations*, 2025. DOI: [10.48550/arxiv.2405.17537](https://doi.org/10.48550/arxiv.2405.17537). arXiv: [2405.17537 \[cs.AI\]](https://arxiv.org/abs/2405.17537). Cited on pages 25, 26, 42, 113, 115, 116, 121, 123.
- [210] S. Liu, J. Chen, L. Pan, C.-W. Ngo, T.-S. Chua, and Y.-G. Jiang, “Hyperbolic visual embedding learning for zero-shot recognition,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2020. DOI: [10.1109/CVPR42600.2020.00929](https://doi.org/10.1109/CVPR42600.2020.00929). Cited on page 26.
- [211] K. Desai, M. Nickel, T. Rajpurohit, J. Johnson, and S. R. Vedantam, “Hyperbolic image-text representations,” in *International Conference on Machine Learning*, PMLR, 2023, pp. 7694–7731. DOI: [10.48550/arXiv.2304.09172](https://doi.org/10.48550/arXiv.2304.09172). Cited on pages 26, 113, 116, 117, 121, 123.

- [212] A. Pal, M. van Spengler, G. M. D. di Melendugno, A. Flaborea, F. Galasso, and P. Mettes, “Compositional entailment learning for hyperbolic vision-language models,” in *International Conference on Learning Representations*, 2025. DOI: [10.48550/arxiv.2410.06912](https://doi.org/10.48550/arxiv.2410.06912). Cited on pages 26, 118.
- [213] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, “End-to-end object detection with transformers,” in *Proc. of the European Conference on Computer Vision*, Springer, 2020, pp. 213–229. Cited on pages 33, 42.
- [214] T.-Y. Lin *et al.*, “Microsoft COCO: Common objects in context,” in *Proc. of the European Conference on Computer Vision*, Springer, 2014, pp. 740–755. Cited on page 33.
- [215] A. Kar *et al.*, *Toronto annotation suite*, <https://aidemos.cs.toronto.edu/toronto-annotation-suite/>, 2021. Cited on page 33.
- [216] G. Van Horn *et al.*, “The iNaturalist species classification and detection dataset,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8769–8778. Cited on page 36.
- [217] M.-E. Nilsback and A. Zisserman, “Automated flower classification over a large number of classes,” in *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, IEEE, 2008, pp. 722–729. Cited on page 36.
- [218] P. Millan Arias *et al.*, “BarcodeBERT: Transformers for biodiversity analysis,” *arXiv preprint arXiv:2405.17537v1*, 2023. DOI: [10.48550/arxiv.2405.17537](https://doi.org/10.48550/arxiv.2405.17537). eprint: [2405.17537](https://arxiv.org/abs/2405.17537). Cited on page 41.
- [219] S. C. Lowe, J. B. Haurum, S. Oore, T. B. Moeslund, and G. W. Taylor, “An empirical study into clustering of unseen datasets with self-supervised encoders,” 2024. DOI: [10.48550/arxiv.2406.02465](https://doi.org/10.48550/arxiv.2406.02465). arXiv: [2406.02465](https://arxiv.org/abs/2406.02465) [cs.LG]. Cited on page 41.
- [220] D. Steinke *et al.*, “Towards a taxonomy machine: A training set of 5.6 million arthropod images,” *Data*, vol. 9, no. 11, 2024, ISSN: 2306-5729. DOI: [10.3390/data9110122](https://doi.org/10.3390/data9110122). Cited on page 42.

- [221] R. Hickling, D. B. Roy, J. K. Hill, R. Fox, and C. D. Thomas, “The distributions of a wide range of taxonomic groups are expanding polewards,” *Global change biology*, vol. 12, no. 3, pp. 450–455, 2006. DOI: [10.1111/j.1365-2486.2006.01116.x](https://doi.org/10.1111/j.1365-2486.2006.01116.x). Cited on page 46.
- [222] J. A. Sheridan and D. Bickford, “Shrinking body size as an ecological response to climate change,” *Nature climate change*, vol. 1, no. 8, pp. 401–406, 2011. DOI: [10.1038/nclimate1259](https://doi.org/10.1038/nclimate1259). Cited on page 48.
- [223] R. H. Steckel, “Stature and the standard of living,” *Journal of economic literature*, vol. 33, no. 4, pp. 1903–1940, 1995. Cited on page 48.
- [224] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 2002. Cited on pages 52, 65.
- [225] N. Natarajan, I. S. Dhillon, P. K. Ravikumar, and A. Tewari, “Learning with noisy labels,” *Advances in neural information processing systems*, vol. 26, 2013. Cited on pages 54, 79.
- [226] PyTorch, *Basic mnist example*, Sep. 2022. [Online]. Available: <https://github.com/pytorch/examples/tree/main/mnist>. Cited on page 56.
- [227] T. Cover and P. Hart, “Nearest neighbor pattern classification,” *IEEE transactions on information theory*, vol. 13, no. 1, pp. 21–27, 1967. Cited on pages 62, 65.
- [228] C. Cortes and V. Vapnik, “Support-vector networks,” *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995. Cited on page 62.
- [229] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors,” *nature*, vol. 323, no. 6088, pp. 533–536, 1986. Cited on pages 62, 65.
- [230] V. N. Vapnik, “An overview of statistical learning theory,” *IEEE transactions on neural networks*, vol. 10, no. 5, pp. 988–999, 1999. Cited on page 64.
- [231] S. Geman, E. Bienenstock, and R. Doursat, “Neural networks and the bias/variance dilemma,” *Neural computation*, vol. 4, no. 1, pp. 1–58, 1992. Cited on page 64.

- [232] A. Dosovitskiy *et al.*, “An image is worth 16x16 words: Transformers for image recognition at scale,” 2020. arXiv: [2010.11929](#). Cited on pages [65](#), [116](#).
- [233] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, “Training data-efficient image transformers & distillation through attention,” in *International conference on machine learning*, PMLR, 2021, pp. 10 347–10 357. Cited on page [65](#).
- [234] S. Khan, M. Naseer, M. Hayat, S. W. Zamir, F. S. Khan, and M. Shah, “Transformers in vision: A survey,” *ACM computing surveys (CSUR)*, vol. 54, no. 10s, pp. 1–41, 2022. Cited on pages [65](#), [67](#).
- [235] G. Bachmann, S. Anagnostidis, and T. Hofmann, “Scaling mlps: A tale of inductive bias,” *Advances in Neural Information Processing Systems*, vol. 36, pp. 60 821–60 840, 2023. Cited on pages [65](#), [67](#).
- [236] H. Xiao, K. Rasul, and R. Vollgraf, “Fashion-mnist: A novel image dataset for benchmarking machine learning algorithms,” 2017. arXiv: [1708.07747](#). Cited on pages [66](#), [79](#).
- [237] T. Clanuwat, M. Bober-Irizar, A. Kitamoto, A. Lamb, K. Yamamoto, and D. Ha, “Deep learning for classical japanese literature,” 2018. arXiv: [1812.01718](#). Cited on page [66](#).
- [238] J. Wei, Z. Zhu, H. Cheng, T. Liu, G. Niu, and Y. Liu, “Learning with noisy labels revisited: A study using real-world human annotations,” in *International Conference on Learning Representations*, 2022. [Online]. Available: <https://openreview.net/forum?id=TBWA6PLJZQm>. Cited on pages [79](#), [88](#).
- [239] S. S. Kadam, A. C. Adamuthe, and A. B. Patil, “Cnn model for image classification on mnist and fashion-mnist dataset,” *Journal of scientific research*, vol. 64, no. 2, pp. 374–384, 2020. Cited on page [80](#).
- [240] C. J. Van Rijsbergen, *Information retrieval. 2nd. newton, ma*, 1979. Cited on page [81](#).

- [241] K. H. Brodersen, C. S. Ong, K. E. Stephan, and J. M. Buhmann, “The balanced accuracy and its posterior distribution,” in *2010 20th International Conference on Pattern Recognition*, 2010, pp. 3121–3124. DOI: [10.1109/ICPR.2010.764](https://doi.org/10.1109/ICPR.2010.764). Cited on page [81](#).
- [242] J. Cohen, *Statistical power analysis for the behavioral sciences*. routledge, 2013. Cited on page [92](#).
- [243] L. V. Kantorovich, “Mathematical methods of organizing and planning production,” *Management science*, vol. 6, no. 4, pp. 366–422, 1960. Cited on page [92](#).
- [244] L. N. Vaserstein, “Markov processes over denumerable products of spaces, describing large systems of automata,” *Problemy Peredachi Informatsii*, vol. 5, no. 3, pp. 64–72, 1969. Cited on page [92](#).
- [245] W. Peng, T. Varanka, A. Mostafa, H. Shi, and G. Zhao, “Hyperbolic deep neural networks: A survey,” *IEEE Transactions on pattern analysis and machine intelligence*, vol. 44, no. 12, pp. 10 023–10 044, 2021. Cited on pages [96–98](#), [104](#).
- [246] Z. Akata, S. Reed, D. Walter, H. Lee, and B. Schiele, “Evaluation of output embeddings for fine-grained image classification,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 2927–2936. Cited on pages [96](#), [97](#).
- [247] G. Van Horn and P. Perona, “The devil is in the tails: Fine-grained classification in the wild,” 2017. arXiv: [1709.01450](https://arxiv.org/abs/1709.01450). Cited on pages [96](#), [97](#).
- [248] J. Bourgain, “On lipschitz embedding of finite metric spaces in hilbert space,” *Israel Journal of Mathematics*, vol. 52, pp. 46–52, 1985. Cited on pages [96](#), [98](#).
- [249] A. L. Maas, A. Y. Hannun, A. Y. Ng, *et al.*, “Rectifier nonlinearities improve neural network acoustic models,” in *Proc. icml*, Atlanta, GA, vol. 30, 2013, p. 3. Cited on page [97](#).
- [250] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, “Fast and accurate deep network learning by exponential linear units (elus),” p. 11, 2015. arXiv: [1511.07289](https://arxiv.org/abs/1511.07289). Cited on page [97](#).

- [251] R. Mahima, M. Maheswari, S. Roshana, E. Priyanka, N. Mohanan, and N. Nandhini, “A comparative analysis of the most commonly used activation functions in deep neural network,” in *2023 4th International Conference on Electronics and Sustainable Communication Systems (ICESC)*, IEEE, 2023, pp. 1334–1339. Cited on page [97](#).
- [252] E. Beltrami, *Teoria fondamentale degli spazii di curvatura costante memoria*. F. Zanetti, 1868. Cited on page [98](#).
- [253] J. W. Milnor, “Hyperbolic geometry: The first 150 years,” *Bulletin of the American Mathematical Society*, vol. 6, no. 1, pp. 9–24, 1982. Cited on page [98](#).
- [254] M. Nickel and D. Kiela, “Learning continuous hierarchies in the lorentz model of hyperbolic geometry,” in *International conference on machine learning*, PMLR, 2018, pp. 3779–3788. Cited on page [98](#).
- [255] J. Harvie, “Computer vision based taxonomic classification of insecta using deep learning models,” Ph.D. dissertation, University of Guelph, 2022. Cited on page [99](#).
- [256] H. J. S. Smith, “On the integration of discontinuous functions,” *Proceedings of the London Mathematical Society*, vol. 1, no. 1, pp. 140–153, 1874. Cited on page [100](#).
- [257] D. R. Nelson, “The cantor set-a brief introduction,” *uwaterloo. ca*, 2007. Cited on page [100](#).
- [258] E. Mathieu, C. Le Lan, C. J. Maddison, R. Tomioka, and Y. W. Teh, “Continuous hierarchical representations with poincaré variational auto-encoders,” *Advances in neural information processing systems*, vol. 32, 2019. Cited on page [104](#).
- [259] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, “Improving neural networks by preventing co-adaptation of feature detectors,” 2012. arXiv: [1207.0580](#). Cited on page [108](#).
- [260] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler, “Efficient object localization using convolutional networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 648–656. Cited on page [108](#).

- [261] P. M. Arias *et al.*, “BarcodeBERT: Transformers for biodiversity analysis,” 2023. DOI: [10.48550/arxiv.2311.02401](https://doi.org/10.48550/arxiv.2311.02401). arXiv: [2311.02401](https://arxiv.org/abs/2311.02401). Cited on page [116](#).
- [262] J. R. DeWaard *et al.*, “A reference library for Canadian invertebrates with 1.5 million barcodes, voucher specimens, and DNA samples,” *Scientific data*, vol. 6, no. 1, p. 308, 2019. DOI: [10.1038/s41597-019-0320-2](https://doi.org/10.1038/s41597-019-0320-2). Cited on page [116](#).
- [263] I. Turc, M.-W. Chang, K. Lee, and K. Toutanova, “Well-read students learn better: On the importance of pre-training compact models,” 2019. DOI: [10.48550/arxiv.1908.08962](https://doi.org/10.48550/arxiv.1908.08962). arXiv: [1908.08962](https://arxiv.org/abs/1908.08962). Cited on pages [116](#), [118](#).
- [264] L. N. Smith, “A disciplined approach to neural network hyper-parameters: Part 1 - learning rate, batch size, momentum, and weight decay,” 2018. DOI: [10.48550/arXiv.1803.09820](https://doi.org/10.48550/arXiv.1803.09820). arXiv: [1803.09820](https://arxiv.org/abs/1803.09820). Cited on page [121](#).
- [265] W. S. Levine, *The Control Systems Handbook: Control System Advanced Methods*. CRC press, 2018. Cited on page [127](#).
- [266] A. Coates, A. Ng, and H. Lee, “An analysis of single-layer networks in unsupervised feature learning,” in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, JMLR Workshop and Conference Proceedings, 2011, pp. 215–223. Cited on page [130](#).
- [267] C. E. Shannon, “A mathematical theory of communication,” *The Bell system technical journal*, vol. 27, no. 3, pp. 379–423, 1948. Cited on page [130](#).
- [268] D. H. Colless, *Phylogenetics: The theory and practice of phylogenetic systematics*, 1982. Cited on page [130](#).
- [269] M. Yazdanian *et al.*, “Ecological and life-history traits predict temporal trends in biomass of boreal moths,” *Insect Conservation and Diversity*, vol. 16, no. 5, pp. 600–615, 2023. Cited on page [130](#).
- [270] D. Sukkar, J. Falla-Angel, and P. Laval-Gilly, “Insect decline in the anthropocene: Historical parallels and emerging monitoring tools,” *Insects*, vol. 16, no. 8, p. 841, 2025. Cited on page [130](#).

- [271] A. Botchkarev, “Performance metrics (error measures) in machine learning regression, forecasting and prognostics: Properties and typology,” 2018. arXiv: [1809.03006](#). Cited on page [131](#).