THESIS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

Robust Learning with Limited Labels

Erik Wallin

Department of Electrical Engineering
Chalmers University of Technology
Gothenburg, Sweden, 2026

Robust Learning with Limited Labels

Erik Wallin
ISBN 978-91-8103-363-2

Acknowledgements, dedications, and similar personal statements in this thesis, reflect the author's own views.

Selected material from the author's licentiate thesis: Erik Wallin, "Semi-supervised learning with self-supervision for closed and open sets", *Chalmers University of Technology*, Gothenburg, Sweden, July 2023, is republished in this Ph.D. thesis.

Doktorsavhandlingar vid Chalmers tekniska högskola
Ny serie nr 5820
ISSN 0346-718X

Department of Electrical Engineering
Chalmers University of Technology
SE-412 96 Gothenburg, Sweden
Phone: +46 (0)31 772 1000

Cover:
Each circle represents a data point. Colored circles are labeled data, with each color corresponding to a distinct class. One of our challenges is to learn robust classification models from such partially labeled data.

Printed by Chalmers Digital Printing
Gothenburg, Sweden, January 2026

Robust Learning with Limited Labels

Erik Wallin
Department of Electrical Engineering
Chalmers University of Technology

Abstract

Deep learning-based classification systems commonly rely on conditions that are difficult to satisfy for real-world applications. One such requirement is the availability of large-scale, curated, and labeled training data. Another is the absence of unknown classes during training and deployment. Furthermore, many classification systems treat classes as independent, even when they form structured relationships that are important to account for. Overcoming these limitations is central to the practical deployment of these systems.

We address these challenges through five papers that study deep classification under limited supervision, the presence of unknown classes, hierarchical class structures, and combinations thereof. Paper A studies semi-supervised learning, where labeled and unlabeled training data are combined, and proposes a self-supervised component for better utilization of unlabeled data. Papers B and C address unknown classes within semi-supervised learning, enabling learning from realistic, uncurated, unlabeled data. In particular, Paper C proposes a probabilistic method that improves accuracy and uncertainty quantification when detecting unknown samples in this setting. Finally, Papers D and E study hierarchical open-set classification, *i.e.*, assigning unknown classes to appropriate high-level categories of a hierarchy, and propose a method that approximates the predictive distribution over both known classes and higher-level categories. This enables more expressive predictions of unknown samples than binary rejection.

In summary, the included papers propose methods that advance performance on benchmarks for their respective problem settings, while providing empirical analyses that improve understanding of the underlying challenges. Overall, this thesis contributes to more robust and accurate deep classification systems for real-world deployment.

Keywords: Deep learning, semi-supervised learning, open-set recognition, hierarchical classification.

List of Publications

This thesis is based on the following publications:

[A] **Erik Wallin**, Lennart Svensson, Fredrik Kahl, Lars Hammarstrand, "DoubleMatch: Improving Semi-Supervised Learning with Self-Supervision". *Proceedings of the International Conference on Pattern Recognition*, 2022.

[B] **Erik Wallin**, Lennart Svensson, Fredrik Kahl, Lars Hammarstrand, "Improving Open-Set Semi-Supervised Learning with Self-Supervision". *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2024.

[C] **Erik Wallin**, Lennart Svensson, Fredrik Kahl, Lars Hammarstrand, "ProSub: Probabilistic Open-Set Semi-supervised Learning with Subspace-Based Out-of-Distribution Detection". *Proceedings of the European Conference on Computer Vision*, 2024.

[D] **Erik Wallin**, Fredrik Kahl, Lars Hammarstrand, "ProHOC: Probabilistic Hierarchical Out-of-Distribution Classification via Multi-Depth Networks". *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

[E] **Erik Wallin**, Fredrik Kahl, Lars Hammarstrand, "Semi-Supervised Hierarchical Open-Set Classification". *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2026.

Other publications by the author, not included in this thesis, are:

[F] Adam Lilja, **Erik Wallin**, Junsheng Fu, Lars Hammarstrand, "Exploring Semi-Supervised Learning for Online Mapping". *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2025.

Contents

Acknowledgments

Pursuing a PhD is a solitary journey, and my case is no exception, but it would not have been possible without the people who supported and believed in me, both within and outside this project.

First, I want to thank my main supervisor and closest collaborator, Lars Hammarstrand, for supporting, encouraging, and challenging me throughout this journey. You made the process enriching and enjoyable, and helped me emerge on the other side with more knowledge, confidence, and experience than I could have hoped for. I also thank my co-supervisors, Fredrik Kahl and Lennart Svensson, for their mentorship and guidance.

I am grateful to everyone in the signal processing and computer vision groups for the coffee breaks, Linsen lunches, and for making the 7th floor a pleasant workplace. Special thanks go to Adam Lilja, with whom I had the pleasure to co-author Paper F, and to Xixi Liu for many engaging discussions about our research. I am also grateful to everyone I met during the much-needed departures from the office — be it through courses, WASP gatherings, or conferences around the world — for making these trips enjoyable.

I would also like to thank everyone at Saab for believing in me, giving me the opportunity to pursue this work, and for funding my Hawaii trip. Special thanks to my industry supervisors Håkan Warston, Patrik Dammert, and Albert Nummelin, as well as my managers Magnus Enger and Per Gustavsson, who all have played important roles in this project. This work traces back to Albert and Magnus trusting me to write my master's thesis in their group.

Finally, I want to thank my family for always being there: Mom, Dad, Lisa, Peter, and (more recently) Lars. You are always the most important. Above all, thank you to Lina, my soon-to-be wife, for standing by me from the start. You make coming home the highlight of my day.

Erik Wallin
Mölndal, January 2026

Funding

This work was supported by Saab AB and the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

Acronyms

CNN:	Convolutional neural network
DAG:	Directed acyclic graph
EMA:	Exponential moving average
ID:	In-distribution
KL:	Kullback–Leibler
LCA:	Lowest common ancestor
OOD:	Out-of-distribution
OSR:	Open-set recognition
OSSL:	Open-set semi-supervised learning
RNN:	Recurrent neural network
SSL:	Semi-supervised learning

Part I

Overview

CHAPTER 1

Introduction

In recent years, deep learning has become the dominant paradigm for automatic classification systems, achieving remarkable performance across a wide range of domains. Prominent examples include image recognition [1], [2], language understanding [3], [4], and audio classification [5], [6]. This progress continues, driven by the expansion of datasets, increased computational resources, and the development of increasingly refined methods.

For a long time, the achievements in deep learning-based classification relied on the framework of supervised learning. In supervised learning, a model is trained on data where each sample is associated with a ground-truth label, and the objective is to learn a mapping from inputs to labels by optimizing a task-specific loss. The goal of this optimization is for the learned model to generalize to new, unseen data, enabling its deployment in real-world scenarios.

Despite its success, the supervised learning paradigm relies on several assumptions and requirements that are often violated in practical deployments. One common requirement is access to large amounts of labeled data. In practice, labeled data are often scarce and depend upon substantial effort from human annotators to obtain, making it challenging to create training data that sufficiently cover all relevant deployment scenarios. Another widespread

practice is to operate under a closed-world assumption, *i.e.*, that the model encounters only data from known classes during both training and testing. However, in real-world settings, it is difficult to ensure that the deployed models are only exposed to familiar data. Moreover, many approaches assume a flat class structure, treating all classes as independent. In many applications, classes instead have known semantic relationships that can be leveraged during learning to improve performance and robustness.

This thesis investigates methods for robust classification under realistic deployment conditions in its five appended papers, where multiple idealized assumptions may be violated simultaneously. We study methods for semi-supervised learning to reduce the reliance on labeled data by incorporating large amounts of unlabeled data. Furthermore, we investigate various aspects of open-set recognition and open-set learning to effectively handle data from unknown classes, both during model training and deployment. Finally, we consider classification with hierarchical class structures, where semantic relationships between classes are exploited to improve prediction quality and uncertainty handling. In the following sections, we describe the three challenges that motivate the work of this thesis and summarize how our appended papers address these challenges.

1.1 Challenge I: Learning from limited labeled data

Many breakthroughs in supervised learning can be credited to the availability of large labeled datasets. Prominent examples include, *e.g.*, ImageNet [7], comprising over 14 million labeled images, or the text dataset SQuAD [8] that consists of over 100,000 question-answer pairs. Constructing such datasets requires extensive effort from human annotators and is both expensive and time-consuming. As a result, applying supervised learning in domains where large labeled datasets are unavailable remains challenging, motivating alternative approaches that reduce the reliance on labeled data.

A paradigm that addresses this challenge is *semi-supervised learning* (SSL) by using both labeled and unlabeled data during training. Compared to labeled data, unlabeled data are typically much cheaper to obtain and can often be collected through, *e.g.*, web scraping or from unsupervised sensor streams. In a standard semi-supervised learning setup, a small labeled dataset defines the target classes, while a much larger unlabeled dataset is used to

guide learning. Many modern semi-supervised learning methods are driven by two core techniques: pseudo-labeling and consistency regularization. Pseudo-labeling involves letting a model trained on labeled data generate labels for unlabeled samples, which are then incorporated into the training process in a supervised manner. Consistency regularization, on the other hand, encourages the model to produce consistent predictions under perturbations of the data or model parameters. Together, these techniques have enabled impressive performance gains, including state-of-the-art results on the image classification benchmark ImageNet through the use of auxiliary unlabeled data [9].

However, several questions remain open. In particular, it is still unclear how unlabeled data should be used most effectively. While pseudo-labeling and consistency regularization have emerged as key components in semi-supervised learning, the exact implementations of these techniques remain an ongoing area of investigation [10]–[14]. In parallel, recent work has explored alternative techniques from self-supervised learning (methods for learning entirely without labels) by, for example, incorporating pretext tasks and contrastive learning as complementary learning signals from unlabeled data [15]–[17]. In Paper A of this thesis, we contribute to this line of work by introducing a self-supervised auxiliary task as a form of consistency regularization, leading to improved utilization of unlabeled data.

A key limitation of many semi-supervised learning methods is their reliance on a closed-world assumption, where all unlabeled data are assumed to belong to the same set of classes as the labeled data. In practical deployments, however, unlabeled data are often uncurated and may contain samples from unknown classes. Such unknown data can degrade performance when applying standard methods for semi-supervised learning, naturally leading to the next challenge we address.

1.2 Challenge II: Recognizing unknown classes

In real-world deployments, classification models are frequently exposed to inputs that deviate from the training distribution. When encountering data from unknown classes, deep learning models are known to produce highly confident yet incorrect predictions unless explicitly designed otherwise [18]. Such failures are particularly problematic in safety-critical applications, where reliable uncertainty estimation and recognition of unfamiliar inputs are essential.

This problem is studied in the setting commonly referred to as *open-set recognition* (OSR), which equips classifiers with mechanisms to distinguish between in-distribution (ID) and out-of-distribution (OOD) data. A common strategy is to define a scalar scoring function based on the model's output or internal representations, such that ID samples receive higher scores than OOD samples. A long-standing baseline uses the confidence of the model's predicted probability distribution [18], *i.e.*, the maximum predicted class probability. More recent approaches have proposed more expressive scoring functions based on learned feature representations [19] or have modified the supervised training objective to explicitly facilitate OOD detection at inference time [20].

Open-set recognition is a central component in the practical deployment of semi-supervised learning. Many methods for semi-supervised learning operate under the naive assumption that the labeled set and the unlabeled set share the same underlying distribution, in particular that they contain the same classes. In practice, however, the appeal of unlabeled data lies precisely in their lack of human curation, which makes it difficult to guarantee the absence of unknown classes, corrupted samples, or other types of outliers. The presence of such out-of-distribution data can lead to degraded performance when using standard semi-supervised learning methods. This observation motivates the setting of *open-set semi-supervised learning* (OSSL) [21]–[23], where the goal is to leverage uncurated unlabeled data while simultaneously ensuring both reliable classification of known classes and detection of unknown classes during deployment. How to best approach open-set semi-supervised learning remains an open research question and is addressed in Papers B and C of this thesis, where we propose methods for OSSL.

Most open-set recognition methods approach the problem as a binary classification task, flagging a sample as either ID or OOD. While such decisions are useful for rejection, they provide limited information about the nature of the unknown inputs. In many applications, it is desirable not only to detect unfamiliar samples but also to understand how they relate to the known classes. This limitation leads us to our final challenge: incorporating class relationships into classification models.

1.3 Challenge III: Class relations

The standard classification paradigm in deep learning typically treats all classes as independent. In many real-world applications, however, classes have semantic relationships that should not be ignored. For example, confusing a *wolf* with a *dog* is typically a less severe error than confusing a *wolf* with a *car*. A flat classification model, however, treats both errors as equally incorrect despite their difference in severity. One common way to represent such relationships is through hierarchical structures. For example, the classes in the ImageNet dataset are organized within the WordNet graph [24], and biological species are structured according to biological taxonomies [25]. Even when such structures are available, they are often ignored in standard classification methods despite offering opportunities to improve both model performance and robustness.

Class hierarchies can be leveraged in several ways. They can be used to encourage predictions that respect semantic similarity, such that misclassifications remain close to the true class in the hierarchy (*i.e.*, to "make better mistakes") [26]–[30]. Another advantage of class hierarchies is that they allow models to share representations between semantically related classes, improving performance for classes that are poorly represented in the training data through transfer from similar well-populated classes [31].

Class hierarchies are particularly valuable in the context of open-set recognition, where they enable more informative predictions for unknown data. The standard paradigm of open-set recognition typically provides a binary decision indicating whether a sample is in-distribution or out-of-distribution [18], [19], [32]. With knowledge of the hierarchy, we can instead predict these unknown classes to a high-level category (*i.e.*, an internal node of the class hierarchy) [33]–[37]. For example, a classification model trained to classify animals, some of which are dog breeds, can predict an unknown dog breed as the coarse category *dog*, indicating that it belongs to the class of dogs but not to any of the known breeds. This setting, referred to as *hierarchical open-set classification*, is challenging, as it effectively constitutes a multi-class classification problem over unknown classes for which no labeled training data are available. Developing accurate and reliable methods for this setting is an open research problem and is the focus of Papers D and E of this thesis.

	Challenge I Few labels	**Challenge II** Unknown classes	**Challenge III** Class relations
Paper A: DoubleMatch	**SSL with self-supervision.**		
Paper B: SeFOSS	**OSSL with energy-based OSR and self-supervision.**		
Paper C: ProSub	**Probabilistic OSR in OSSL.**		
Paper D: ProHOC		**Class hierarchies for probabilistic hierarchical open-set classification.**	
Paper E: SemiHOC	**Hierarchical open-set classification with unlabeled training data.**		

Figure 1.1: Overview of how the included papers address the three introduced challenges.

1.4 Overview of the included papers

This thesis includes five appended papers that address the challenges introduced in Sections 1.1-1.3, both individually and in combination. Together, these works contribute toward more reliable and robust deployment of deep classification models in real-world scenarios.

Paper A addresses Challenge I. It introduces *DoubleMatch*, a method for semi-supervised learning under the closed-world assumption. DoubleMatch improves the utilization of unlabeled data by incorporating a self-supervised learning objective, enabling learning from more unlabeled samples compared to approaches relying solely on pseudo-labeling.

Papers B and C address the combination of Challenges I and II by studying open-set semi-supervised learning, the SSL setting where unknown classes can appear in the unlabeled training data and during deployment. Paper B proposes *SeFOSS*, which applies the self-supervision proposed in Paper A to OSSL and employs an energy-based scoring function to distinguish ID and OOD. Paper C introduces *ProSub*, a framework that extends and improves upon SeFOSS by introducing a probabilistic method that enables better accuracy and uncertainty quantification for ID/OOD predictions.

Paper D addresses Challenges II and III by considering the setting of hierarchical open-set classification. It introduces *ProHOC*, a framework that predicts unseen classes to internal nodes of the class hierarchy through depth-specific classifiers and probabilistic modeling. Finally, Paper E introduces *SemiHOC*, an extension of ProHOC to the semi-supervised setting, thereby addressing the combination of Challenges I, II, and III. SemiHOC leverages unlabeled data to improve performance in hierarchical open-set recognition by introducing *subtree pseudo-labels*, a form of pseudo-labeling tailored to this setting.

A brief description of each paper, along with a visualization of which challenges they address, is provided in Figure 1.1.

1.5 Thesis outline

This introductory chapter provides background and introduces the three challenges in deep classification that motivate this thesis. In the following chapter, we provide an overview of the field of semi-supervised learning for classification, focusing on methods applied in the domain of deep learning. In the third chapter, we cover the setting of open-set semi-supervised learning and provide a review of existing literature. The fourth chapter discusses the use of class hierarchies in deep learning. The fifth chapter summarizes the appended papers. Finally, the thesis concludes with summarizing remarks and an outlook for future work.

CHAPTER 2

Semi-supervised learning for classification

Semi-supervised learning is a paradigm that lies between supervised and unsupervised learning. In this setting, training data consist of both labeled data and unlabeled data. The idea is to leverage information from the unlabeled data, together with the typically much smaller labeled set, while training a model. Take, for example, the illustration in Figure 2.1. Given only information from labeled data, our best guess of the decision boundary might resemble the one shown in the left panel of the figure. However, when the unlabeled data are taken into account, it becomes clear that a more accurate decision boundary lies between the two half-moons. This improved boundary would be difficult to determine using only the labeled data.

This chapter provides an overview of semi-supervised learning. While there exist works on semi-supervised learning for regression, this chapter focuses on the realm of classification. To establish the foundations, we start by presenting a formal problem definition. Subsequently, we cover the necessary assumptions that underlie methods for semi-supervised learning. We proceed to examine the historical progression of methods in the field of semi-supervised learning. In the latter and largest part of this chapter, we turn our attention to the application of semi-supervised learning in deep learning: this part of the chapter explores

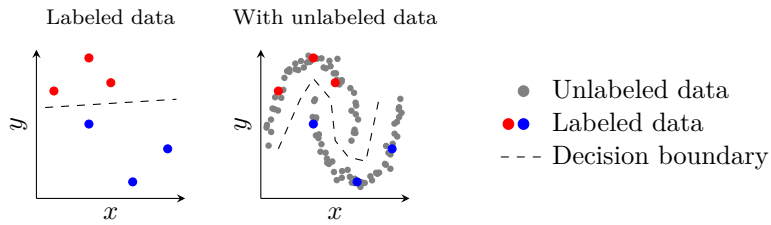

Figure 2.1: Illustration of semi-supervised classification with two classes (red and blue). With unlabeled data, we can better estimate the distributions of the two classes and thus improve our decision boundary.

various techniques that are used for semi-supervised learning in the domain of deep learning.

2.1 Problem definition

In semi-supervised learning, we are provided with a labeled training set of independent and identically distributed data,

$$\{(x_i, y_i)\}_{i=1}^{m}; \qquad (x_i, y_i) \in \mathcal{X} \times \mathcal{Y}, \tag{2.1}$$

where $\mathcal{X} \subseteq \mathbb{R}^D$ is the input space with D being the input dimension, and $\mathcal{Y} = \{1, \ldots, C\}$ is the label space with C being the number of classes. These data have an underlying distribution $p(x, y)$. In addition to the labeled training set, we have a set of independent and identically distributed unlabeled training data,

$$\{x_i^u\}_{i=1}^{n}; \qquad x_i^u \in \mathcal{X}, \tag{2.2}$$

where the underlying distribution $p(x)$ is the marginal distribution of $p(x, y)$.

The goal is to learn a mapping from the input space to the label space:

$$f_\theta : \mathcal{X} \to \mathbb{R}^C, \tag{2.3}$$

where f_θ is parameterized by θ. The C scalars, generally denoted *logits*, are often transformed into a distribution over $\mathcal{Y}$ through the softmax function as

$$p_\theta(y|x) = \frac{\exp(f_\theta^y(x))}{\sum_{y' \in \mathcal{Y}} \exp(f_\theta^{y'}(x))}, \quad y \in \mathcal{Y}, \tag{2.4}$$

where $f_\theta^{y'}(x)$ is the y'th element of $f_\theta(x)$. The final prediction can be obtained by selecting the class with the highest probability. The mapping, f_θ, is learned by minimizing the sum of two expected loss terms, one for labeled data and one for unlabeled data:

$$\underset{\theta}{\operatorname{argmin}} \left(\mathbb{E}_{x,y \sim p(x,y)} \left[l(f_\theta(x), y) \right] + \alpha \mathbb{E}_{x \sim p(x)} \left[\Omega_\theta(x) \right] \right), \tag{2.5}$$

where α is a scaling parameter to control the balance between the two terms. The expectations are typically evaluated with Monte Carlo approximations using batches of the training data. The term for fitting the labeled training data is $l : \mathbb{R}^C \times \mathcal{Y} \to \mathbb{R}$, which is generally implemented as a cross-entropy loss. The learning from unlabeled data occurs through the regularization term

$$\Omega_\theta : \mathcal{X} \to \mathbb{R}, \tag{2.6}$$

which also depends on the model parameters θ. The construction of this regularization term is one of the key challenges in semi-supervised learning, as it defines how we utilize the unlabeled data for improving our learned model.

2.2 Assumptions

In order to learn from unlabeled data, we need to make some assumptions regarding the underlying structure of the data. The book *Semi-Supervised Learning* by Chapelle *et al.* [38] suggests three main assumptions in the form of the *smoothness assumption*, the *cluster assumption*, and the *manifold assumption*.

The smoothness assumption

The smoothness assumption states that if two data points are close in the input space, then their corresponding outputs should also be close. Intuitively,

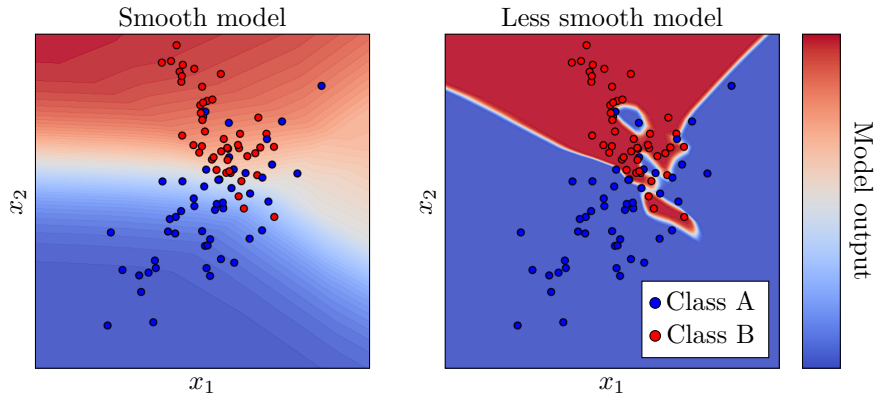

Figure 2.2: Under the smoothness assumption, the left-hand model may be preferred over the right-hand model, despite its poorer fit to the training data, as it better satisfies the belief that similar inputs should produce similar outputs across the entire data space.

this enables us to propagate information from labeled training data to nearby unlabeled samples. This assumption is not unique to semi-supervised learning and is also typically enforced in supervised learning by applying regularization techniques that help the model generalize to unseen test data.

An illustration of the smoothness assumption is given in Figure 2.2 in a fully supervised setting. The right panel shows a model that fits the labeled training data precisely but displays sharp shifts in its predictions across the data space. In contrast, the left panel depicts a model with a less accurate fit to the training data but smoother variations in its outputs. Under the smoothness assumption, we may prefer the left-hand model despite its poorer training accuracy, as it better aligns with the expectation that similar inputs should yield similar outputs.

A common strategy for encouraging smoothness is weight regularization [39]: applying penalties on the norm of the model's weight vector, which indirectly limits model complexity and discourages sharp changes in predictions.

A challenge for high-dimensional data is to define the *closeness* between data points, as standard Euclidean distances tend to become non-expressive in high dimensions. We will revisit the notion of closeness when discussing the manifold assumption.

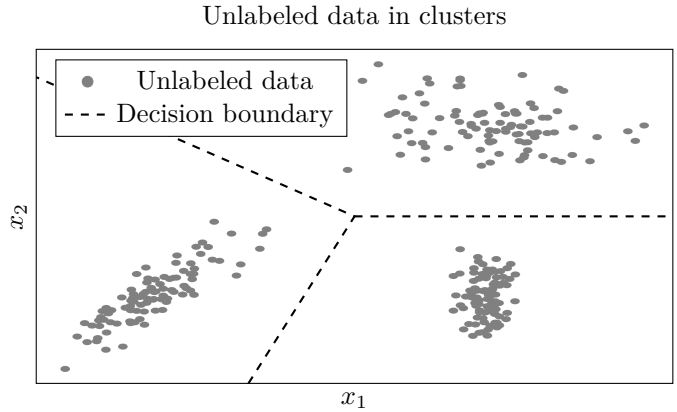

Figure 2.3: The cluster assumption tells us that decision boundaries should lie in low-density regions, as shown in the figure.

The cluster assumption

The cluster assumption states that data points that lie in the same cluster are likely to share class, or equivalently, that decision boundaries should lie in *low-density regions*. This implies that each cluster contains points from only a single class, but a class can be associated with many clusters. When combined with the smoothness assumption, this encourages models to produce few and smooth decision boundaries between clusters.

Figure 2.3 illustrates this concept for a two-dimensional unlabeled dataset. The data form three well-separated clusters. The cluster assumption suggests that decision boundaries should be placed between these clusters, as depicted in the figure.

Now we have the building blocks for a simple SSL strategy:

1. Identify clusters in the combined labeled and unlabeled data.

2. Assign clusters to the corresponding class of any labeled data within them.

3. Place smooth decision boundaries in low-density regions between the clusters.

Such an approach can be effective for low-dimensional problems with well-separated clusters. However, real-world data are often in a high-dimensional space where clusters are not easily separable. This motivates the manifold assumption, which enables the smoothness and cluster assumptions to be applied even in such complex settings.

The manifold assumption

The manifold assumption states that high-dimensional data lie roughly on a low-dimensional manifold. Distances measured along this manifold can provide a more meaningful notation of closeness and density than Euclidean distances in the raw input space, which often become uninformative in high dimensions. These manifold-based distances can then be used to apply the smoothness and cluster assumptions in settings where direct input-space distances fail.

Without delving into formal definitions, a manifold can be thought of as a low-dimensional surface embedded within a higher-dimensional space. While the manifold may be curved globally, it resembles Euclidean space locally. Figure 2.4 illustrates this with a set of two-dimensional points on a one-dimensional manifold in the form of a spiral. In the two-dimensional data space, the dark-red points on the right appear closer to the central blue points

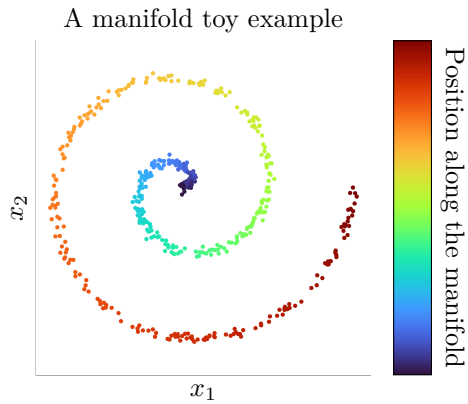

Figure 2.4: Two-dimensional points lying on a one-dimensional spiral manifold. Distances measured along the manifold differ from Euclidean distances in the two-dimensional data space.

than to the red points on the left. However, when measuring distances along the manifold, the blue and dark-red points are separated by the largest pairwise distances.

The difference between input-space distances and manifold distances is further illustrated with real-world image data in Figure 2.5. The query image, taken from the ImageNet-1k dataset [7], is compared to its four nearest neighbors according to two distance metrics: the Euclidean distance in pixel space and the distances in the lower-dimensional representation produced by the DinoV2 [40] image encoder. DinoV2 is an image encoding model trained without labels to produce semantically meaningful image representations. In pixel space, the nearest neighbors share similar background colors but are not semantically related to the query. In contrast, the nearest neighbors in DinoV2's representation space are images of the same bird species.

Note that DinoV2 does not explicitly learn the data manifold, but its learned representations can be interpreted as an approximation of it. This representation space serves as a practical illustration of the manifold assumption, as its distances capture semantic similarity more effectively than raw pixel-space distances.

Figure 2.5: Comparison of nearest neighbors in pixel space and a learned lower-dimensional representation. Pixel-space distances often reflect superficial similarities (*e.g.*, color or brightness), whereas distances in a learned representation space better capture semantic similarity.

2.3 The history of semi-supervised learning

The first instances of semi-supervised learning in the literature appeared in the 1960s and 1970s [41]–[43]. These methods employed a technique today called *self-training*, which involves an iterative process where the model is initially trained using only labeled data. In each subsequent step, model predictions on unlabeled data are used to expand the training set, and the model is retrained using the new training set. At this time, the methods were very general and were often referred to as *pattern recognition machines*.

In the 1990s, there was growing interest in more application-focused semi-supervised learning for text applications [44], [45]. Text is a typical domain where a lot of unlabeled data are available, but labeled data are expensive. For example, Yarowsky [44] used a form of self-training for semi-supervised sense classification of words.

2.4 Semi-supervised learning in deep learning

In the deep learning paradigm, input data are typically high-dimensional, and our learned models are neural networks with many hidden layers. Naturally, many new techniques for semi-supervised learning have emerged to cater to this setting. This section covers some of the most popular techniques for semi-supervised learning in deep learning. Note that some details of the covered methods in this section may differ from the original works. The main purpose of this section is to give an overview of the general ideas and approaches of this paradigm.

Pseudo-labeling

One of the dominant techniques for semi-supervised learning in deep learning is pseudo-labeling. This essentially means using model predictions on unlabeled training data as training labels. A simple early version of this technique was introduced by the pseudo-label method [46]. The pseudo-label method simply takes the class with the largest predicted probability for each unlabeled sample and uses this as the training label. Sticking to the notation from (2.6), letting Ω_θ be an element-wise loss for unlabeled data, pseudo-labeling can be written as

$$\Omega_\theta^{\text{pseudo-label}} = H(e_{\hat{y}},\, p_\theta(y|x))\,, \tag{2.7}$$

where $p_\theta(y|x)$ is the probability distribution predicted by the model and

$$\hat{y} = \underset{y \in \mathcal{Y}}{\operatorname{argmax}}\, p_\theta(y \mid x) \tag{2.8}$$

is the most probable class, *i.e.*, the pseudo-label, with $e_{\hat{y}} \in \{0,1\}^C$ denoting its one-hot encoding. The cross-entropy between two discrete probability distributions, p^a and p^b, defined over the label set $\mathcal{Y}$ is

$$H(p^a, p^b) = -\sum_{y \in \mathcal{Y}} p^a(y) \log p^b(y), \tag{2.9}$$

where $p^a(y)$ denotes the probability of class y given distribution p^a.

It has later been found that using only pseudo-labels for data with confident model predictions tends to yield better results. For example, FixMatch [11] and UDA [10] assign pseudo-labels to unlabeled data that satisfy

$$\max_y p_\theta(y|x) > \tau, \tag{2.10}$$

where τ is the confidence threshold. This results in a loss on unlabeled data that looks similar to

$$\Omega_\theta^{\text{FixMatch}} = \mathbb{1}\{\max_{y'} p_\theta(y'|x) > \tau\} H\left(e_{\hat{y}}, p_\theta(y|x)\right), \tag{2.11}$$

where $\mathbb{1}\{\cdot\}$ is the indicator function.

Relating the pseudo-labeling technique back to our assumptions of Section 2.2, we can interpret the pseudo-labeling technique as an application of the clustering assumption. When we train our model to produce confident predictions in high-density regions (regions where our training data are located), we are implicitly pushing the decision boundaries to low-density regions in accordance with the clustering assumption.

Adaptive and dynamic thresholds

The pseudo-labeling procedure of FixMatch and UDA, as described in (2.10), relies on a static threshold τ. Many recent works have focused on replacing this static threshold with dynamic and adaptive thresholds. This direction of research is motivated by two main factors. Firstly, the varying learning

difficulties associated with different classes incentivize using class-dependent thresholds. For example, the model may produce less confident predictions for a particular class, causing fewer pseudo-labels and hindering learning for that class. Secondly, neural networks tend to generate increasingly confident predictions as the training progresses, suggesting that thresholds can be modified based on the number of completed training steps.

One example of a method that proposes a dynamic confidence schedule as a function of the training time is Dash [12]. This work identifies that FixMatch tends to produce very few pseudo-labels early in training, but also increasingly many *incorrect* pseudo-labels in the later stages of training. To counteract this, Dash suggests a schedule for the threshold that decreases monotonically as training progresses. The dynamic threshold is computed as

$$\tau_t^{\text{Dash}} = C\gamma^{-(t-1)}\hat{p}, \tag{2.12}$$

where t is the current timestep in training, C and γ are constant hyperparameters, and $\hat{p}$ is the base threshold that is computed based on a pre-training phase.

A method that instead proposes adaptive thresholds per class, *e.g.*, is FlexMatch [13]. FlexMatch adjusts the class-dependent thresholds depending on how many pseudo-labels are assigned to each class: if a class is less frequent in the pseudo-labels, its threshold is lowered to assign more pseudo-labels corresponding to that particular class. A similar method that also uses class-dependent adaptive thresholds is FreeMatch [14]. In FreeMatch, the class-dependent thresholds are computed based on the average prediction confidence for each class: classes that are less confidently predicted are assigned lower thresholds.

Consistency regularization

Another major technique for semi-supervised learning in deep learning is consistency regularization. The idea of consistency regularization is to minimize the difference in predictions for similar data points. These similar data points are often generated by applying perturbations to the training data. Given an unlabeled training sample, x, and the corresponding perturbation, $\tilde{x}$, the

general structure for consistency regularization looks like

$$\Omega_\theta^{\text{Consistency regularization}} = d\left(f_\theta(x), f_\theta(\tilde{x})\right), \tag{2.13}$$

where $d(\cdot,\cdot)$ is some distance measure, *e.g.*, mean squared error or KL divergence. The distance may also be calculated between two different perturbations, instead of the original data and a single perturbation.

Consistency regularization relates to both the smoothness assumption and the cluster assumption of Section 2.2. The smoothness assumption is addressed by manually constructing close inputs through perturbations, to then encourage similar predictions for these nearby inputs. Additionally, the consistency regularization is applied mainly in high-density regions due to the concentration of training data in these regions. This implicitly enforces similar predictions within clusters, which moves decision boundaries toward low-density regions, in accordance with the cluster assumption.

How perturbations for consistency regularization are designed has been an active field of research. An early version of consistency regularization in semi-supervised learning was used in the Ladder network [47], where perturbed data are created by adding Gaussian noise to the activations at each layer of the network. The noisy activations are then denoised by a trainable decoder network. The resulting loss is the sum of squared errors between the denoised activations and the activations from a clean pass through the network for all layers:

$$\Omega_\theta^{\text{Ladder networks}} = \sum_{l=1}^{L} \lambda_l \|z_l - \hat{z}_l\|^2, \tag{2.14}$$

where L is the number of layers in the network, λ_l is a layer-dependent scaling factor, z_l are the clean activations for an unlabeled sample x at layer l, $\hat{z}_l$ are the corresponding denoised activations, $\|\cdot\|$ is the l^2 norm.

The subsequent Π-model [48] instead applies consistency regularization directly to the predicted probability distributions obtained from two perturbations of the input. These predictions are denoted $\hat{p}_\theta^a(y|x)$ and $\hat{p}_\theta^b(y|x)$. The perturbations are obtained by applying two instances of some stochastic data augmentation on x along with two different realizations of the stochastic *dropout* regularization [49] in the forward pass through the neural network.[1]

[1] Dropout is a common regularization technique for neural networks that involves stochastically masking neurons and their connections in each forward pass during training.

The obtained loss for unlabeled data is

$$\Omega_\theta^{\text{Π-model}} = \|\hat{p}_\theta^a(y|x) - \hat{p}_\theta^b(y|x)\|^2, \tag{2.15}$$

where the norm is taken over the class dimension.

Many SSL methods rely on the idea of a *teacher-student framework*. Within this terminology, a teacher prediction is typically treated as ground truth for a student prediction. A common strategy for producing stable teacher predictions is to use moving averages. The concept of using moving averages as teacher predictions was introduced in the Temporal ensembling method [48]. Temporal ensembling uses the same perturbation strategy as the Π-model. However, instead of using two different perturbations, the teacher prediction is an exponential moving average of the perturbed student prediction, updated each epoch[2], as

$$p_\theta^{\text{teacher}}(y|x) \leftarrow \beta p_\theta^{\text{teacher}}(y|x) + (1-\beta)p_\theta^{\text{student}}(y|x), \tag{2.16}$$

where β is the momentum parameter (typically close to, but smaller than 1). The loss is then given by

$$\Omega_\theta^{\text{Temporal ensembling}} = \|p_\theta^{\text{teacher}}(y|x) - p_\theta^{\text{student}}(y|x)\|^2. \tag{2.17}$$

The method Mean teacher [50] develops the idea of using moving averages as teacher predictions by taking an exponential moving average of the model parameters. This has the advantage that the exponential moving average can be updated every training step instead of once every epoch. The average of the model parameters is updated each training step as

$$\theta_{\text{EMA}} \leftarrow \beta\theta_{\text{EMA}} + (1-\beta)\theta. \tag{2.18}$$

Mean teacher uses a perturbation strategy that consists of a data augmentation, Gaussian noise on the input layer, and dropout. The perturbation is applied both to the teacher prediction and the student prediction (in two different realizations). The resulting loss is

$$\Omega_\theta^{\text{Mean teacher}} = \|\hat{p}_{\theta_{\text{EMA}}}(y|x) - \hat{p}_\theta(y|x)\|^2. \tag{2.19}$$

[2]An epoch in this context means the time it takes to cycle through the full training set in the training process.

The methods covered so far in this section rely on random perturbations for consistency regularization, *i.e.*, these methods smooth the prediction function in random directions around the input. The method Virtual adversarial training (VAT) [51] takes another approach. In VAT, the idea is to smooth the prediction function in the *least* smooth direction with respect to the input, *i.e.*, the *adversarial direction*. The adversarial direction is, in this context, defined as the direction of the point, within a small region of the input, that gives the largest change in prediction (relative to the unperturbed input). Formally, the loss is written as

$$\Omega^{\text{VAT}} = d_{\text{KL}}\left(p_\theta(y|x), p_\theta(y|x + r_{\text{adv}})\right), \tag{2.20}$$

where

$$r_{\text{adv}} = \underset{r;\|r\|<\epsilon}{\operatorname{argmax}}\, d_{\text{KL}}\left(p_\theta(y|x), p_\theta(y|x + r)\right). \tag{2.21}$$

Here, $d_{\text{KL}}(\cdot, \cdot)$ is the KL-divergence and ϵ is a small scalar that sets the size for the region in which we look for the adversarial direction. Unfortunately, there exists no closed-form expression for r_{adv}, so VAT uses a one-step power iteration to approximate r_{adv}.

Data augmentation

Early implementations of consistency regularization often relied on simple techniques for data augmentation, such as horizontal flips and translations in the context of images. However, notable achievements were made with the introduction of optimized domain-specific augmentations in ReMixMatch [52], FixMatch [11], and UDA [10]. These augmentations are, *e.g.*, RandAugment [53] for images, which comprises a set of operations, such as shearing, rotating, and adjusting colors or brightness. For a domain like language, these domain-specific augmentations can be, *e.g.*, back-translation [54] that involves translating a sentence from language A to language B, and then back to language A, to obtain a slightly perturbed version of the original sentence. Notably, ReMixMatch and FixMatch pioneered a setup of using weak augmentations for teacher predictions and strong augmentations for student predictions in the image domain. The weak augmentation consists of a horizontal flip and translation, and the strong augmentation consists of Cutout [55], followed by two randomly sampled operations from RandAugment.

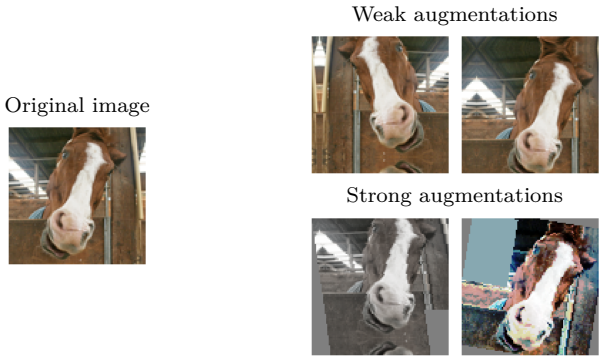

Figure 2.6: The currently widely used augmentation strategies for semi-supervised learning, consisting of weak and strong augmentations of images. Weak augmentations are horizontal flips and stochastic translations. Strong augmentations comprise operations such as Cutout, shearing, rotations, and color filters.

Examples of these weak and strong augmentations can be seen in Figure 2.6. The augmentation strategy of ReMixMatch and FixMatch has been widely adopted by many subsequent works [12]–[14], [16], [17], [56], [57].

Interpolation consistency

Another form of data augmentation is to use interpolations of training data. This strategy was introduced for supervised learning under the name *mixup* [58]. The idea is to create new training data by interpolating both input data and corresponding labels using the Mix-operation, defined as

$$\text{Mix}_\lambda(a, b) = \lambda a + (1 - \lambda)b, \tag{2.22}$$

where λ is a parameter between 0 and 1 that is sampled from a Beta distribution. The methods Interpolation consistency training (ICT) [59] and MixMatch [60] introduced the idea of using interpolations in semi-supervised learning. For unlabeled data, we cannot interpolate ground-truth labels to form optimization targets; instead, we can interpolate model predictions. For example, ICT uses the exponential moving average of the model parameters to form targets for

interpolations of unlabeled data according to

$$\Omega_\theta^{\mathrm{ICT}}(x_a, x_b) = \|p_\theta(y|\mathrm{Mix}_\lambda(x_a, x_b)) - \mathrm{Mix}_\lambda(p_{\theta_{\mathrm{EMA}}}(y|x_a), p_{\theta_{\mathrm{EMA}}}(y|x_b))\|^2, \tag{2.23}$$

where x_a and x_b are two different unlabeled samples. Training with interpolations can be argued to be well-aligned with the cluster assumption of Section 2.2. If we are considering a classification problem with more than a few classes, it is likely that that x_a and x_b belong to different classes, and thus different clusters. Assuming x_a and x_b are not incorrectly predicted as the same class, the interpolation loss will move the decision boundary toward the region between these data, which is a low-density region.

Self-supervision

A related field to semi-supervised learning is *self-supervised learning*. In self-supervised learning, we are training a model using training data fully without labels. The goal is not to learn a classifier, but to learn a useful low-dimensional representation of the often high-dimensional data. Note how this relates to the manifold assumption of Section 2.2. Various techniques are commonly used for self-supervised learning. One is to enforce prediction consistency across augmentations of data (much like consistency regularization for semi-supervised learning) [61]–[64]. Another technique involves training the model to reconstruct masked regions of input data [4], [65]. Additionally, a common approach is to train the model to perform a pretext task, such as predicting the angle of a stochastic rotation applied to training images [66]–[68].

Influential works for self-supervised learning in the image domain made use of contrastive learning [61], [62], which means not only enforcing similar predictions for different versions of the same data, but also increasing the disagreement of representations given different data. One argument for the contrastive loss is that without enforcing the disagreements, the model can converge to the collapsed solution: predicting the same representation for all data. However, subsequent works [63], [64], [69] found that collapse can be avoided without contrastive learning by instead using exponential moving averages as teacher models and by the use of stop-gradient operations.

Many works borrow techniques from self-supervised learning for semi-supervised learning. The motivation is that the self-supervision can improve the latent representations of data in the model, or that it can help methods

based on confidence-based pseudo-labeling (see (2.11)) to utilize all unlabeled data, not only data that fall above the confidence threshold.

One work that incorporates techniques from self-supervised learning for semi-supervised learning is S4L [15]. S4L employs a rotation loss to unlabeled data, formulated as follows:

$$\Omega_\theta^{\text{S4L}} = \frac{1}{4} \sum_{r \in \mathcal{R}} H(r_{\text{target}}, g(z_r)), \tag{2.24}$$

where $\mathcal{R} = \{0°, 90°, 180°, 270°\}$ and r_{target} is the one-hot vector that denotes the current rotation, *e.g.*, $r_{\text{target}} = (0, 1, 0, 0)^T$ for $r = 90°$. Here, z_r is the latent representation of the network (predicted by some backbone model f_θ) for an unlabeled image x that has undergone rotation r, while g is a trainable 4-way classifier that predicts the rotation based on the latent representation. The rotation prediction serves as a typical pretext task, since the main interest lies in improving the latent representations. By creating latent representations that can be used for predicting rotations, they are hopefully also more useful for the downstream classification task.

EnAET [70] similarly employs a self-supervised pretext task for semi-supervised learning. However, instead of predicting rotations, the model is trained to predict the continuous parameters of more general transformations, such as projective and affine transformations.

In Paper A of this thesis, we introduce DoubleMatch, a method that employs self-supervision for semi-supervised learning. The purpose of DoubleMatch is to improve the utilization of unlabeled data in methods that employ confidence-based pseudo-labeling. To enable learning from all unlabeled data, DoubleMatch proposes an auxiliary self-supervised loss to all unlabeled data to align the latent representations for weak and strong augmentations of a given unlabeled image, given by

$$\Omega_\theta^{\text{DoubleMatch}} = -\frac{z_w \cdot g(z_s)}{\|z_w\| \cdot \|g(z_s)\|}, \tag{2.25}$$

where z_w and z_s are the latent representations for weak and strong augmentations of an unlabeled image x, respectively. The trainable linear mapping $g(\cdot)$ is used to map the latent representations of strongly augmented data to the latent space of weak augmentations.

Recently, CCSSL [56], SimMatch [17], and ProtoCon [16] have used forms of contrastive learning for latent representations where pseudo-labels are used for determining which data to push together and pull apart.

Pure self-supervision followed by supervised adaptation

A straightforward way to incorporate self-supervision in the context of semi-supervised learning is to adopt a two-stage training pipeline. In the first stage, often referred to as the pretraining stage, the model is trained using a purely self-supervised objective on all available data, *i.e.*, fully without labels. In the second stage, commonly called fine-tuning or adaptation, the pretrained model is adapted to the target classification task using the available labeled data in a standard supervised loss [71]–[73].

This approach has been shown to yield competitive performance, particularly when the second stage is extended to also incorporate unlabeled data through a semi-supervised fine-tuning objective [71]. A key advantage of this two-stage paradigm is that it enables the downstream use of computationally expensive large-scale pretrained models [4], [40]. In the less expensive second stage, these models can be adapted to target tasks using less compute and data than required during pretraining. In practice, this adaptation can consist of training one or a few task-specific layers on top of a large frozen pretrained backbone.

CHAPTER 3

Open-set semi-supervised learning

In semi-supervised learning, it is commonly assumed that labeled and unlabeled training data follow the same distribution and that the sets of classes for the labeled and unlabeled training data are identical. For many practical applications, this assumption is probably not reasonable. On the contrary, it seems natural to assume that the unlabeled set may contain outliers, unseen classes, or corrupted data. In this case, we want to ensure that these out-of-distribution (OOD) data do not harm the model training, and that our model can learn to identify the OOD data at test time. Take Figure 3.1 as an example. Here, the unlabeled data give us information about the distributions of class A and class B, but they also indicate the existence of a third class that is not present in our labeled training data. A well-trained model on these data should ideally be able to classify class A and class B, but also to identify data that likely do not belong to class A or class B.

This chapter gives an overview of the field of open-set semi-supervised learning. We start by expanding the problem formulation from Chapter 2 to fit the open-set problem. Next, we cover existing methods and techniques that attempt to tackle this problem. Finally, we summarize a few related research problems.

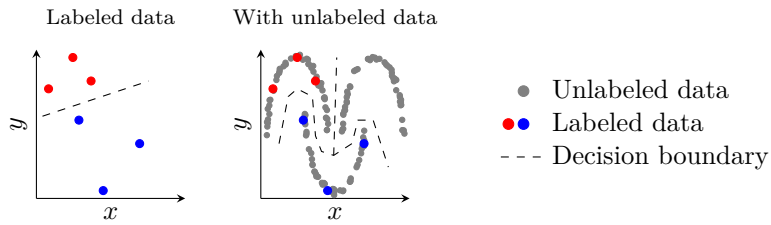

Figure 3.1: Illustration of open-set semi-supervised learning. The unlabeled data can improve our estimates of the class distributions, but they also indicate the presence of an unknown class. A preferable decision boundary is to classify red and blue based on the two leftmost half-moons, but also to reject samples from the unknown class corresponding to the rightmost half-moon.

3.1 Problem formulation

Similarly to the problem formulation presented for the closed-set setting in Chapter 2, we have a labeled training set

$$\{(x_i, y_i)\}_{i=1}^{m}; \qquad (x_i, y_i) \in \mathcal{X}_l \times \mathcal{Y}_l, \tag{3.1}$$

where again $\mathcal{X}_l \subseteq \mathbb{R}^D$ with D being the input dimension, and $\mathcal{Y}_l = \{1, \ldots, C\}$. We assume our labeled samples are independent and identically distributed from an underlying distribution $p_l(x, y)$. Additionally, we have the unlabeled training set

$$\{x_i^u\}_{i=1}^{n}; \qquad x_i^u \in \mathcal{X}_{ul}, \tag{3.2}$$

such that $\mathcal{X}_l \subseteq \mathcal{X}_{ul} \subseteq \mathbb{R}^D$, and the corresponding (unknown) labels associated with the unlabeled samples are in $\mathcal{Y}_{ul} = \{1, \ldots, C, C+1, \ldots, C+K\}$, meaning there are K novel classes in the unlabeled set that are not part of the labeled set. We assume that our unlabeled samples are independent and identically distributed with the underlying distribution $p_{ul}(x)$. Note that, in contrast to Chapter 2, we no longer assume that $p_{ul}(x)$ is the marginal distribution of $p_l(x, y)$.

In general, we are interested in learning the classification mapping corresponding to our labeled training set:

$$f_\theta : \mathcal{X}_l \to \mathbb{R}^C, \tag{3.3}$$

i.e., predicting logits over the set of known classes. However, we may also be interested in the binary classification of in-distribution and out-of-distribution data:

$$p_\theta(y \in \mathcal{Y}_l | x); \qquad x \in \mathcal{X}_{ul}, \tag{3.4}$$

meaning predicting the probability that a given sample belongs to the known classes, for an input drawn from the unlabeled data distribution. This probability is not directly modeled by the standard softmax distribution in (2.4), which only normalizes over known classes, and must therefore be estimated through alternative techniques. This is one of the key challenges in open-set semi-supervised learning.

Taking the classification of samples as ID or OOD one step further, we can also consider unknown classes that are fully unseen during training, *i.e.*, not part of the unlabeled set. These classes can be introduced by the test set, defined on the domain $\mathcal{X}_{\text{test}}$, such that $\mathcal{X}_{ul} \subseteq \mathcal{X}_{\text{test}} \subseteq \mathbb{R}^D$ where the corresponding classes belong to $\mathcal{Y}_{\text{test}} = \{1, \ldots, C + K + L\}$, meaning that we have a set of L new classes, not seen in the unlabeled data during training. We are in this case interested in modeling

$$p_\theta(y \in \mathcal{Y}_l | x); \qquad x \in \mathcal{X}_{\text{test}}. \tag{3.5}$$

Different works in open-set semi-supervised learning focus on different goals. Some works primarily aim to achieve high closed-set accuracy, *i.e.*, high accuracy on the known classes. This corresponds to having a well-performing closed-set classifier, f_θ, as defined in (3.3). These works argue that unknown classes in the unlabeled training set can harm the closed-set performance of traditional methods for semi-supervised learning. A common trend is that works focusing on this objective use terms such as *safe* semi-supervised learning or *robust* semi-supervised learning to describe their problem settings.

Other works place greater emphasis on open-set recognition, which involves the ability to distinguish between known and unknown classes. The motivation for these works is that if unknown classes appear at training, they are also likely to appear at deployment. Open-set recognition can be either in the form of distinguishing the known classes from the unknown classes in the unlabeled training set, as in (3.4), or in the form of identifying known classes in the presence of classes completely unseen during training, as described in (3.5). A key difference between these two settings is that in the former, exposure to

unknown classes through unlabeled training data can be exploited to learn a more effective open-set detector, which makes this setting rich in opportunities. In contrast, in the latter setting, no information about the unknown and unseen classes is available during training, making it more challenging to leverage the training data for improved open-set recognition.

The training objective of open-set semi-supervised learning can generally be written as

$$\underset{\theta}{\operatorname{argmin}} \left(\mathbb{E}_{x,y\sim p_l(x,y)} \left[l(f_\theta(x), y) \right] + \alpha \mathbb{E}_{x\sim p_{ul}(x)} \left[\Omega_\theta(x) \right] \right), \tag{3.6}$$

which is similar to the objective of the closed-set case (see (2.5)), with the difference that the unlabeled term now is an expectation over the distribution that may contain unknown classes, $p_{ul}(x)$.

3.2 Existing techniques for open-set semi-supervised learning

This section covers existing techniques for open-set semi-supervised learning. We try to categorize methods based on the kind of technique they employ, in an attempt to summarize existing approaches and research directions in this field.

Identifying in-distribution samples in unlabeled data

A recurring theme in methods for open-set semi-supervised learning is to curate the unlabeled data by identifying which samples belong to the known classes and which do not. When in-distribution data are identified, these can be used in the unsupervised loss of a traditional SSL method. How to best identify which data belong to the known or unknown classes is, however, still an open question.

A popular baseline method for open-set recognition, not limited to OSSL, is the softmax confidence score [18],

$$\max_y p_\theta(y|x), \tag{3.7}$$

which is based on the assumption that ID samples yield higher-confidence predictions than OOD data. Different forms of the softmax confidence are also widely used in OSSL [74]–[76]. For example, UASD [74] computes confidence using the average prediction over the most recent training epochs for an unlabeled sample,

$$c(x) = \max_y \frac{1}{t} \sum_{i=1}^{t} p_{\theta_i}(y|x), \tag{3.8}$$

where $p_{\theta_i}(y|x)$ for $i = 1, \ldots, t$ are the network predictions for sample x from the t most recent epochs during training. A sample is classified as ID if $c(x) > \tau$, where the threshold τ is set as the average confidence given a labeled ID validation set.

Beyond confidence-based criteria, several works extend their classification models with additional prediction heads dedicated to binary ID/OOD discrimination [21], [23], [77]–[80]. One prominent example is OpenMatch [21], which identifies ID samples using one-vs-all classifiers. In this setup, each known class is associated with a binary classifier that predicts whether a sample belongs to that class or not. A sample is classified as OOD if none of the one-vs-all classifiers produces a high-confidence prediction. An advantage of this approach is that each classifier has access to both positive and negative training data from the labeled training set. Both SSB [79] and IOMatch [80] build upon OpenMatch by employing one-vs-all classifiers for OSSL. Notably, IOMatch proposes a method for fusing the predictions of the one-vs-all classifiers and the closed-set classifier into a distribution over $C+1$ classes, *i.e.*, a distribution over the known classes and an "OOD class".

Energy-based scores [81] provide another alternative for identifying ID samples. In Paper B, we introduce SeFOSS, which uses the free-energy score to distinguish ID from OOD data. The free-energy score is obtained by

$$E(x) = -T \cdot \log \sum_{i=1}^{C} \exp\left(f_\theta^i(x)/T\right), \tag{3.9}$$

where T is a scalar hyperparameter and $f_\theta^i(x)$ is the predicted logit associated with class i. Safe-student [82] builds on the free-energy score by considering differences between the largest logits in its energy-discrepancy score. Energy-based methods are computationally inexpensive and have been shown to outperform softmax confidence in open-set recognition.

In Paper C, we introduce ProSub, which includes a new technique for separating ID from OOD samples by measuring distances to an ID subspace in the learned representation space. In addition, ProSub enables probabilistic ID/OOD predictions by modeling the distributions of these distance-based scores. This probabilistic formulation provides calibrated uncertainty estimates for ID/OOD decisions, which can be used to reliably select ID or OOD samples from unlabeled data.

Utilizing out-of-distribution data

In contrast to several works discussed in the previous section, which aim to exclude out-of-distribution data from the training objective completely, other works instead propose methods for explicitly leveraging OOD data, arguing that such data can be used to improve model performance.

A representative example is SSB [79], which allows the model to assign pseudo-labels (for inlier classes) to OOD samples. The motivation is that learning from visually similar OOD classes can increase data diversity and thereby improve the learned representations of the known classes. This aligns well with our findings in Paper B, which show that naive pseudo-labeling in OSSL does not necessarily harm closed-set classification performance. A drawback of assigning OOD data to known classes, however, is that the model's ability to discriminate between ID and OOD samples is weakened. SSB addresses this issue by learning a separate feature space, decoupled from the closed-set classification task, that is optimized specifically for open-set recognition. In this space, it applies the one-vs-all classifiers introduced in OpenMatch.

TOOR [76] aims to utilize OOD data by reducing the distribution gap between ID and OOD feature representations, a process referred to by the authors as "recycling of OOD data". This is achieved through adversarial training of a feature extractor $F(\cdot)$, parameterized by θ_F, together with a discriminator $D(\cdot)$, parameterized by θ_D. The feature extractor produces representations of input samples, while the discriminator is trained to classify these representations as originating from ID or OOD data. The adversarial objective is

$$\min_{\theta_F} \max_{\theta_D} \left[\mathbb{E}_{x \sim p_{\text{ood}}(x)} \log D(F(x)) + \mathbb{E}_{x \sim p_{\text{id}}(x)} \log(1 - D(F(x))) \right], \qquad (3.10)$$

where $p_{\text{id}}(x)$ and $p_{\text{ood}}(x)$ are the distributions of ID and OOD data, respectively. With this objective, the discriminator is trained to correctly distinguish ID from OOD samples, while the feature extractor is trained to fool the discriminator by making OOD features indistinguishable from ID features. The resulting alignment encourages OOD samples to lie closer to the ID feature distribution, allowing them to be incorporated into downstream objectives such as consistency regularization or pseudo-labeling to learn the ID classification task.

OSP [23] uses OOD samples to identify specific features that are present in OOD data. They then encourage the model to suppress these feature components when learning ID representations. Intuitively, this can be understood through examples where ID and OOD classes share spurious visual features. For instance, if the ID class is *butterfly* and the OOD class is *beetle*, both may appear on leafy backgrounds. By identifying such background features as prevalent in OOD data, the model can be encouraged to focus on class-relevant features rather than context.

Self-supervision

Another approach to utilizing OOD data in OSSL is to employ self-supervision over the entire unlabeled dataset, regardless of whether samples are ID or OOD. This enables the model to learn from all available data, without the need to confidently identify them as either ID or OOD.

Several works use auxiliary self-supervised tasks applied uniformly to all unlabeled data. Examples include T2T [78], OSP [23], and [83] that employ the rotation loss in (2.24). Other examples are SeFOSS and ProSub in Papers B and C of this thesis, which apply the self-supervision proposed for DoubleMatch in Paper A to open-set semi-supervised learning. Notably, Paper C demonstrates that this form of self-supervision can yield substantial improvements in open-set recognition when combined with a scoring function that synergizes with the learned representations.

A related but distinct strategy is proposed in OpenCOS [84], which performs self-supervised contrastive pretraining on all training data. The resulting model is used to separate ID and OOD samples within the unlabeled set. The labeled data, together with the detected ID samples, are subsequently used to fine-tune the model using a standard SSL method.

Finally, [83] and the ϒ-model [75] employ self-supervision in a clustering-based sense. In [83], the authors learn a feature space consisting of $C + 1$ clusters: one cluster per ID class and an additional cluster that captures all OOD data. By identifying these clusters, the original C-way classification problem is transformed into a $C + 1$-way problem that includes an OOD class. In contrast, the ϒ-model clusters the unlabeled data into $C + K$ clusters (one per ID class and one per unknown OOD class) and argues that ID classification can be improved by jointly learning to classify the OOD classes. This is achieved using deep clustering [85]. A practical limitation of this approach is that the number of unknown classes, K, must be specified as a hyperparameter.

Robust optimization

Another line of research for open-set semi-supervised learning attempts to adjust the optimization steps so that parameter updates never harm performance on ID data. Some of these methods resort to bi-level optimization. For example, DS3L [86] and WRSSL [87] use bi-level optimization to weight unlabeled data such that the resulting updates minimize a supervised loss on a labeled training set. For example, DS3L learns a weighting function $w_\alpha(\cdot)$, parameterized by α, that is used to weight each unlabeled sample in a traditional SSL loss. The bi-level optimization objective can be written as

$$\min_{\alpha} \mathbb{E}_{x,y\sim p_l(x,y)} \left[l(p_{\hat{\theta}}(y'|x), y) \right] \tag{3.11}$$

such that

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} \left(\mathbb{E}_{x,y\sim p_l(x,y)} \left[l(p_\theta(y'|x), y) \right] + \mathbb{E}_{x\sim p_{ul}(x)} \left[w_\alpha(x) \Omega_\theta(x) \right] \right), \tag{3.12}$$

where $l(\cdot,\cdot)$ and $\Omega_\theta(\cdot)$ are the loss functions for labeled and unlabeled data, respectively. The intuition behind this objective is that the model parameters are learned by the inner weighted SSL-objective, but the weighting function, learned through the outer objective, makes sure that the weighting of the unlabeled data causes the inner objective to be aligned with performance on ID data.

SPL [88] similarly uses bi-level optimization for robust optimization. However, instead of learning a weighting function for unlabeled data, it optimizes a mask for the model parameters. The idea is to find the parameters that are

associated with features corresponding to ID data to restrict negative effects from OOD data.

A different approach for robust optimization is proposed in Fix-a-Step [89]. The idea of Fix-a-Step is to ignore the gradient from unlabeled data if it does not point in a similar direction as the gradient for labeled data. Given the gradient associated with the labeled loss, $g^L = \nabla_\theta l$, and the gradients associated with the unlabeled loss, $g^U = \nabla_\theta \Omega$, the parameter update in each training step is

$$\theta \leftarrow \begin{cases} \theta - \epsilon(g^L + \alpha g^U), & \text{if } g^L \cdot g^U > 0 \\ \theta - \epsilon g^L & \text{otherwise,} \end{cases} \tag{3.13}$$

where ϵ is the learning rate and α is a scaling for the unlabeled loss. The intuition behind this procedure is that if the inner product between the gradients is positive, $g^L \cdot g^U > 0$, the angle between the gradients is less than 90°, and we thus assume that the unlabeled loss is somewhat aligned with the labeled loss. However, if the inner product is negative, the gradients are pointing in different directions, so the gradients from unlabeled data can potentially harm performance on labeled data. In this case, we ignore the gradient from unlabeled data.

3.3 Related research problems

Deep learning for classification is a highly active research area, with several subdomains that leverage unlabeled training data and address the detection of unknown classes. These settings share similarities with open-set semi-supervised learning but differ in their assumptions and objectives.

Open-world semi-supervised learning and novel class discovery

A closely related problem is *open-world semi-supervised learning* (OwSSL) [90]–[94]. Like OSSL, OwSSL assumes access to an unlabeled dataset that contains both known and unknown classes. However, rather than simply detecting unknown samples, the goal is to assign them to new, distinct classes. A common limitation of existing approaches is that they require the number of unknown classes to be specified in advance. Knowing the number of unknown classes can be difficult in practice.

Another domain with strong similarities to OwSSL is *novel class discovery* (NCD) [95]–[97]. NCD also assumes an unlabeled dataset composed of unknown classes, but with the stricter condition that the classes in the unlabeled data are entirely disjoint from those in the labeled set. Consequently, at the test time, the task is to classify the novel classes without the presence of the originally labeled classes.

An additional distinction is that OwSSL and NCD are typically formulated as *transductive* problems: evaluation is performed on the same unlabeled samples provided during training. In contrast, SSL and OSSL are generally defined in the *inductive* setting, where evaluation is carried out on separate, unseen test data.

Long-tailed semi-supervised learning

Another line of work that, like open-set semi-supervised learning, aims to relax the overly idealized assumptions of standard SSL is *long-tailed semi-supervised learning* (also referred to as class-imbalanced semi-supervised learning) [98]–[101]. Rather than assuming the presence of unknown classes, these methods address scenarios where the class distribution is imbalanced, *i.e.*, a few classes are dominant in the training data, whereas some are underrepresented. This imbalance can cause standard SSL approaches to neglect the underrepresented classes.

Unsupervised domain adaptation

Unsupervised domain adaptation (UDA[1]) [102], [103] closely resembles semi-supervised learning in that it relies on both a labeled and an unlabeled dataset. The key distinction is that the labeled data are drawn from a *source domain*, while the unlabeled data come from a *target domain*. Both domains share the same label space, but differ in their input distributions due to a covariate shift. In the image domain, *e.g.*, the source may consist of photographs while the target consists of sketches. The goal of UDA is to learn a classifier with high classification accuracy in the target domain. Methodologically, multiple UDA approaches employ techniques commonly used in SSL, such as pseudo-labeling and consistency regularization.

[1] Here, UDA refers to *unsupervised domain adaptation*, not to the SSL method UDA [10], where it denotes *unsupervised data augmentation*.

A variant of this setting, analogous to OSSL, is *open-set unsupervised domain adaptation* [104], [105], where the target domain additionally contains unknown classes not present in the source. The objective is to classify samples from known classes while also detecting those from unknown classes.

Finally, a practical difference from SSL is that UDA does not necessarily assume the labeled source set is small. Instead, the label scarcity lies in the target domain where only unlabeled data are available.

Open-set recognition

Because OSSL addresses the detection of unknown classes, there is a natural connection to the broader field of open-set recognition [18], [32], [81], [106]–[111]. In the standard formulation, a model is trained for classification on a fully labeled training set, and at test time the task is to classify samples from the known classes while detecting those from unknown classes. This problem is also referred to as OOD detection, novelty detection, or anomaly detection, although the boundaries between these terms are not fully established in the literature.

Many of the detection techniques used in OSSL are inspired by methods from the open-set recognition domain. For instance, both the confidence score baseline of [18] and the energy score of [81] are used in OSSL methods.

A subdomain with close similarity to OSSL is *semantically coherent OOD detection* (SC-OOD detection) [112], [113]. In SC-OOD detection, training involves both a labeled dataset and an unlabeled set. The unlabeled set contains a mix of 1) unknown classes that should be recognized as OOD and 2) samples from the known classes but with a domain shift. The objective is to classify the known classes, including the domain-shifted data from the unlabeled set, and to detect OOD classes.

In Table 3.1, we summarize the main differences between the subproblems discussed in this section. The assumptions and specifications for the datasets and tasks are indicated by the symbols defined in the table caption. Although the table is a simplification, it provides an overview of the differences and similarities within this family of research domains.

Table 3.1: Research domains related to open-set semi-supervised learning summarized by training data attributes and target task.
Legend:
✣ imbalanced
✪ contains unknown classes
❄ domain shift
♈ small set
♣ labeled classes not present
☽ source domain not present
present/required.

Domain	Labeled train	Unlabeled train	Task: Classify	Task: Detect	Key works
Standard SSL	♈				[11][60][14]
Robust SSL	♈	✪			[22][89]
Open-set SSL	♈	✪		✪	[77][21]
Open-world SSL	♈	✪	✪		[91][92]
Novel class discovery		✪♣	✪♣		[95][96]
Long-tailed SSL	♈✣	✣			[98][100]
UDA		❄☽	❄☽		[102][103]
Open-set UDA		✪❄☽	❄☽	✪☽	[104][114]
Open-set recognition				✪	[18][32]
SC-OOD detection		✪❄	❄	✪	[112][113]

CHAPTER 4

Hierarchical classification

In many real-world domains, classes are not independent but instead form structured relationships that can be organized into a hierarchy. Canonical examples include the Linnaean taxonomy in biology [25], the lexical database WordNet [24], the Enzyme Commission [115], and the Gene Ontology [116]. Beyond such foundational examples, hierarchical structures also arise in numerous application-specific settings, for example in traffic scene understanding, where classes can be organized from *vehicle type* to *vehicle model*, or in document classification for news articles, in which documents can be marked, *e.g.*, by *sport* at a coarse level and *league* at a finer level. A subset of Linnaean taxonomy is depicted in Figure 4.1.

These hierarchies can be used in many ways. They enable evaluation metrics that account for the semantic severity of classification errors [26], [27], allow the prediction of high-level categories for when the model is uncertain or encounters unknown classes [33], [117], and facilitate improved learning in long-tailed regimes by allowing sparsely represented classes to benefit from hierarchically close, well-sampled categories [31].

Hierarchical classification is commonly defined as the task of associating data samples with nodes in a class hierarchy. The hierarchy may take the

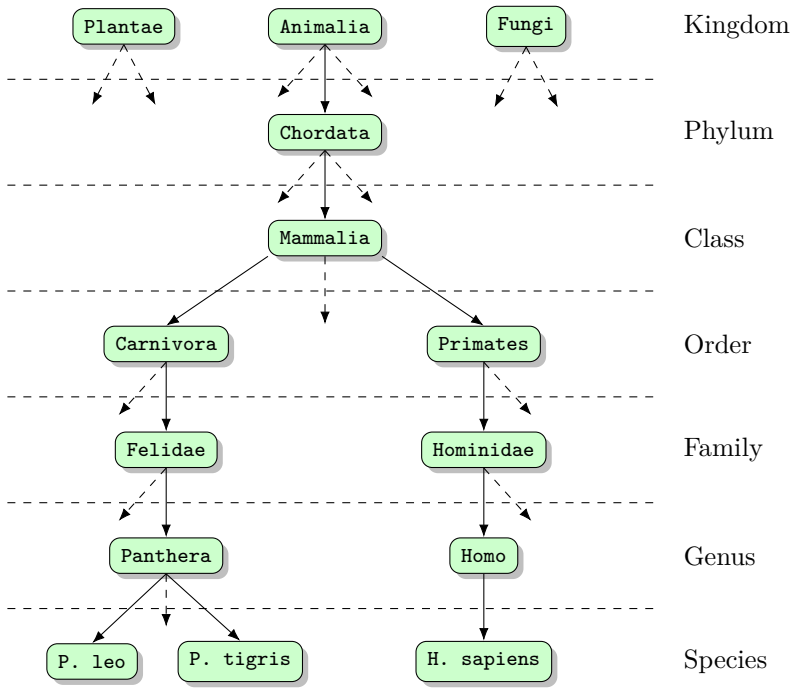

Figure 4.1: The figure shows a small subset of the Linnaean taxonomy, where entities such as lion, tiger, and human are organized from the finest semantic level (species) through intermediate taxonomic ranks to the coarsest level (kingdom). Dashed arrows indicate the presence of additional nodes and branches not shown.

form of a directed acyclic graph (DAG), as in WordNet, or a directed rooted tree, as in the Linnaean taxonomy. The key distinction is that DAGs permit categories to have multiple parents, whereas trees require a unique parent for each non-root node. In general, these hierarchies encode *is-a* relationships. Consequently, predicting a category also implies predicting all of its ancestor nodes. Figure 4.2 shows an example illustrating the distinction between trees and DAGs.

Hierarchical classification problems can be categorized along two dimensions [118]. The first concerns whether (the deepest) predictions are restricted to leaf nodes (*mandatory leaf-node prediction*) or may terminate at internal nodes of the hierarchy (*non-mandatory leaf-node prediction*). The second distinction is between *single-path* and *multi-path* prediction. In the single-path setting, the model predicts at most one node per depth level, forming a single path from the most specific predicted node to the root. By contrast, multi-path prediction permits multiple such paths, allowing several branches or leaves to be predicted simultaneously. This categorization is illustrated in Figure 4.3.

Papers D and E of this thesis fall within the domain of hierarchical classification and more specifically address *hierarchical open-set classification*. Hierarchical open-set classification is the hierarchical classification setting in which we predict classes not seen during training as the most appropriate internal nodes of the class hierarchy. Using the terminology introduced above, these papers consider tree-structured hierarchies, use non-mandatory leaf-node predictions, and adopt a single-path prediction setting. To place these contributions in a broader context, this chapter first reviews prior work on deep learning methods that leverage class hierarchies, and then moves on to describe the subdomain of hierarchical open-set classification in more detail.

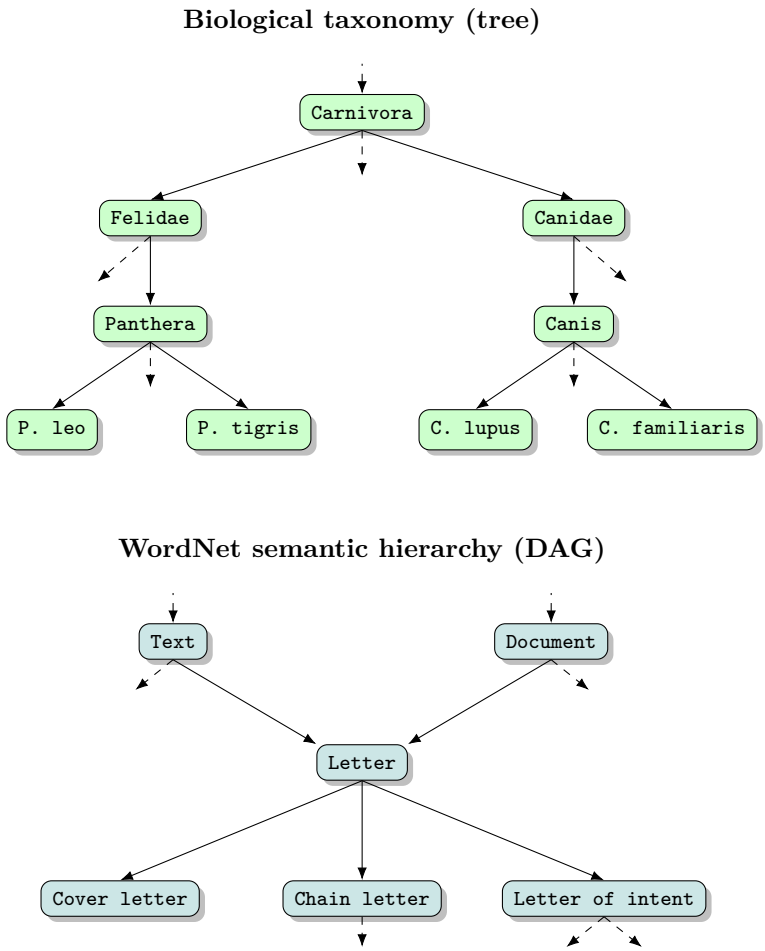

Figure 4.2: The biological taxonomy of the top panel forms a tree, where each category has exactly one parent. For example, a species belongs to exactly one genus. The bottom panel shows a part of the WordNet noun hierarchy, which categorizes *synsets* (sets of synonymous words that represent a single concept). The WordNet graph is formed by hypernym relations, where a synset A is a hypernym of B if every instance of B is a kind of A. In contrast to biological taxonomies, a synset may have multiple hypernyms. For example, *letter* is both a kind of *text* and a kind of *document*.

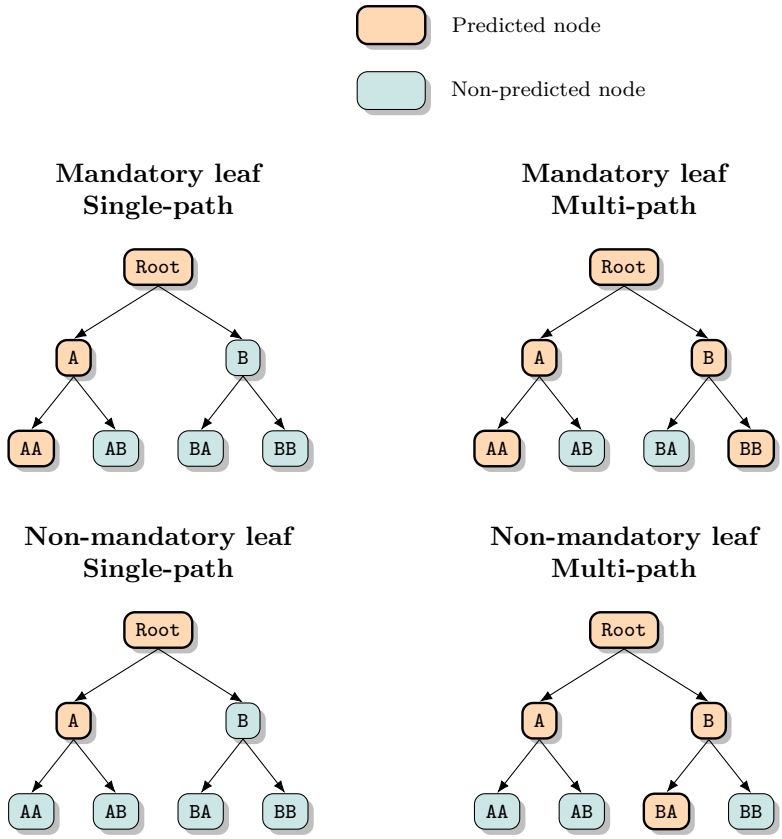

Figure 4.3: Examples of predictions under different hierarchical classification paradigms. In mandatory leaf-node prediction, the most specific node of each predicted path is a leaf, whereas in non-mandatory leaf-node prediction, a predicted path may terminate at an internal node of the hierarchy. In single-path prediction, the model predicts at most one node per depth level, forming a single path, while multi-path prediction allows multiple nodes to be predicted at the same depth.

4.1 Class hierarchies in deep learning

The idea of organizing classes into hierarchical structures predates deep learning [24], [25]. In this section, however, we restrict our attention to the use of hierarchies within deep learning. We further focus on works that treat the hierarchy as a given and fixed part of the problem formulation. There exist works that aim to learn class hierarchies from data [119]–[121], but such approaches are considered outside of the scope of this review.

We organize the literature by categorizing works based on the type of method they employ to utilize the hierarchy. For this purpose, we propose the categories hierarchical losses, hierarchical inference, hierarchical architectures, hierarchy-aware representation learning, and label granularity. Moreover, as mentioned above, these works can exploit the hierarchy for different objectives. We return to discuss tasks and evaluation at the end of this section.

Hierarchical losses

One line of work incorporates the class hierarchy into the loss function [26], [122]–[127]. An influential example of this is [26], which proposes two hierarchy-aware loss formulations: hierarchical cross entropy (HXE) and soft labels.

Hierarchical cross-entropy is based on factorizing the probability of a leaf class c into a product of conditional probabilities, corresponding to the path from the root to node c in the hierarchy:

$$p(c) = \prod_{c' \in \mathrm{Anc}^+(c) \setminus R} p(c' | \mathrm{Par}(c')), \tag{4.1}$$

where $\mathrm{Anc}^+(c)$ denotes the set of ancestors of c, including c itself, $\mathrm{Par}(c')$ is the parent node of c', and R is the root of the hierarchy. The root node is excluded from the product since $p(R) = 1$.

The factorization enables errors at higher levels of the hierarchy to be penalized more heavily than errors close to the leaves. Specifically, the hierarchical cross-entropy is defined as

$$\ell_{\mathrm{HXE}}(p, c) = -\sum_{c' \in \mathrm{Anc}^+(c) \setminus R} \lambda(c') \log(p(c' | \mathrm{Par}(c'))) \tag{4.2}$$

where p is the predicted distribution, c is the ground-truth class, and $\lambda(c')$ is a scaling function that decays with the depth of c', consequently assigning larger weights to loss terms corresponding to higher levels of the hierarchy than those near the leaves. In [26], they use $\lambda(c') = \exp(-\alpha D(c'))$, where $D(c')$ is the depth of node c' and α is a hyperparameter. When $\lambda(c') = 1$ for all c', the hierarchical cross-entropy reduces to the standard cross-entropy.

The second approach proposed in [26] replaces the one-hot target vector in the standard cross-entropy with a hierarchy-aware soft target distribution. The underlying idea is that the target representation for a class c should allocate non-zero probability mass to classes that are close in the hierarchy, thus encouraging hierarchy-aware predictions. This *soft labels* loss is defined as

$$\ell_{\text{soft}}(p, c) = -\sum_{c' \in \mathcal{C}_{\text{leaves}}} y_{c'}^{\text{soft}}(c) \log p(c'), \tag{4.3}$$

where p is the predicted distribution, c is the ground-truth class, and $\mathcal{C}_{\text{leaves}}$ is the set of leaf classes. The hierarchy-aware label embedding $y_{c'}^{\text{soft}}(c)$ is computed as

$$y_{c'}^{\text{soft}}(c) = \frac{\exp(-\beta d(c', c))}{\sum_{\tilde{c} \in \mathcal{C}_{\text{leaves}}} \exp(-\beta d(\tilde{c}, c))}, \tag{4.4}$$

where $d(\cdot, \cdot)$ denotes the distance between two nodes in the hierarchy. The parameter β controls the sharpness of the label embedding. In the limit $\beta \to \infty$, the soft labels loss converges to the standard one-hot cross-entropy.

Hierarchical inference

Another family of methods focuses on hierarchy-aware *inference* procedures rather than modifying the training objective [27], [30], [31], [117], [123], [128], [129]. These works replace the standard inference method, selecting the class with maximum probability as predicted by the model, with alternatives that account for the hierarchical structure.

One example is *conditional risk minimization* (CRM) [27], which operates on predictions from any off-the-shelf model that outputs a probability distribution over the leaf classes of the hierarchy. Instead of predicting the class with the highest probability, CRM selects the class that minimizes the expected

conditional risk based on the hierarchy:

$$\text{argmin}_{c \in \mathcal{C}_{\text{leaves}}} \sum_{c' \in \mathcal{C}_{\text{leaves}}} d(c', c) p(c'), \tag{4.5}$$

where $p(c')$ is the predicted probability for leaf class c', and $d(c', c)$ is a hierarchy-induced distance between classes. In [27], this distance is defined as the height of the lowest common ancestor (LCA) of c' and c.

Conceptually, CRM trades class accuracy for semantic accuracy by favoring predictions that are, on average, closer to the true class in the hierarchy. Importantly, this procedure is applied entirely at test time and does not require retraining the underlying model.

The CRM method proposed in [27] is closely related to the inference approach proposed in Paper D. However, while [27] restricts the decision space to leaf nodes, Paper D allows predictions at arbitrary hierarchy nodes. Moreover, while [27] measures distances via LCA height, Paper D uses the number of edges along the shortest path between two nodes as the distance $d(\cdot, \cdot)$. In addition to the decision rule, Paper D proposes a test-time method for computing a predictive distribution over the full hierarchy based on a set of per-level predictions, which can be categorized as a method for hierarchical inference.

Hierarchical architectures

A third approach incorporates the hierarchy by making architectural changes to the classification model [31], [130]–[138]. In contrast to hierarchy-aware losses or inference rules, which often use standard model architectures, these methods modify the network structure to reflect the hierarchy.

One example is proposed in [132], which suggests placing classification heads at multiple depths of a convolutional neural-network (CNN) architecture [139]. The shallow heads are trained to predict coarse categories, whereas deeper classification heads are trained to predict more fine-grained categories. At the end of training, all classification heads except the final one are discarded. The authors report that hierarchy-aware training improves the final flat classification performance.

Another architectural approach is introduced in [130], which combines a CNN and a recurrent neural network (RNN) [140] to model hierarchical prediction

as a top-down process. The CNN extracts image features, which are then used by an RNN to generate sequential, top-down predictions.

Hierarchy-aware representation learning

Other methods use class hierarchies to learn data representations that reflect the hierarchical structure [28], [29], [141]–[146]. For example, [143] constructs a set of class prototypes, one for each class, in a representation space, such that pairwise distances between prototypes in this space reflect distances in the class hierarchy. The model is then trained to map data samples close to their corresponding class prototypes. The main task in this work is image retrieval.

A notable subcategory of these works is those that employ hyperbolic representations [147]. Hyperbolic geometry has been shown to be particularly well-suited for representing hierarchical structures [148]. In hyperbolic space, the volume of a ball grows exponentially with its radius, mirroring the exponential growth in the number of nodes with tree depth. In contrast, the volume of a ball in Euclidean space grows only polynomially with its radius. Motivated by this property, several works use hyperbolic representations across various tasks with class hierarchies [141], [142], [144], [146].

Label granularity

An alternative approach is to exploit the fact that class labels can be represented at different granularity levels within a hierarchy [138], [149]–[152]. For example, the fine-grained label *dog* can be mapped to the coarser categories *mammal* or *animal*, depending on the chosen granularity level.

In [149], they find that varying the label granularity at which a model is trained causes the model to focus on different data features. For instance, models trained to classify fine-grained bird species emphasize discriminative details such as color patterns or beak shapes, whereas models trained to classify the coarser category *bird* focus on more generic features common to all birds, such as wings and feathers. They further show that training at a coarser label granularity can improve performance in weakly supervised object detection.

In semi-supervised learning, several works leverage label granularity by assigning pseudo-labels at coarser hierarchy levels when the model lacks sufficient confidence to assign pseudo-labels at the most fine-grained level [150]–[152].

This strategy can allow a larger set of unlabeled samples to contribute to the loss without the risk of incorrect fine-grained pseudo-labels.

In Papers D and E, we use approaches related to label granularity by training separate models at different depths of the hierarchy, with each model responsible for predicting classes at that depth. This design encourages depth-specific representations that capture complementary semantic information.

Tasks and evaluation

In the literature, class hierarchies are employed to facilitate a wide range of objectives. Some works leverage hierarchies for non-hierarchical tasks, for example, long-tailed recognition [31], weakly supervised object localization [138], object detection [127], and flat classification [132]. In these settings, the hierarchy is used as an inductive bias during training or inference, while evaluation is typically performed using non-hierarchical metrics.

A group of works that treats hierarchical consistency as an explicit objective focuses on *mistake severity* [26]–[30]. These works aim to minimize the average height of the lowest common ancestor (LCA) between the ground-truth and the prediction, motivated by the intuition that confusing semantically similar classes (*e.g.*, ribbon snake vs. whipsnake) is less severe than confusing semantically distant ones (*e.g.*, ribbon snake vs. steamroller). In fact, this type of metric was recommended by the creators of ImageNet for evaluations on that dataset [7], [153], but has since been largely overshadowed by the standard top-k accuracy.

The height of the LCA is not the only way to quantify hierarchical error. A closely related alternative is to measure the number of edges along the shortest path between the prediction and the ground-truth in the hierarchy. Another family of metrics consists of hierarchical precision, recall, and F1 scores [154], defined as

$$\mathcal{P}_H = \frac{\sum_i |P_i \cap T_i|}{\sum_i |P_i|} \tag{4.6}$$

$$\mathcal{R}_H = \frac{\sum_i |P_i \cap T_i|}{\sum_i |T_i|} \tag{4.7}$$

$$\mathcal{F}^1_H = \frac{2 \cdot \mathcal{P}_H \cdot \mathcal{R}_H}{\mathcal{P}_H + \mathcal{R}_H} \tag{4.8}$$

where P_i denotes the set of predicted nodes for sample i, T_i the corresponding ground-truth nodes, and $|\cdot|$ the set cardinality. Both P_i and T_i include the most specific nodes and all ancestors. These metrics are originally defined as micro-averages, as given by the equations above, but can alternatively be computed as macro-averages by computing class-wise scores and averaging these.

Each evaluation metric has its own advantages and limitations. The LCA height is well-suited for mandatory leaf-node prediction with single-path outputs and applies to both trees and DAGs. However, in non-mandatory leaf-node settings, it becomes less informative; for example, predicting the parent of the ground truth yields the same error as predicting a sibling. For that reason, in Papers D and E, where we consider a non-mandatory leaf-node setting, we instead use the number of edges in the shortest path between the prediction and the ground-truth. This provides an intuitive and interpretable notion of error severity for single-path predictions in non-mandatory leaf-node settings. A limitation of both LCA height and path-length metrics is that they are not well-defined for multi-path prediction settings.

Hierarchical precision, recall, and F1 scores are more general: they apply to mandatory and non-mandatory leaf-node settings, single- and multi-path predictions, and both trees and DAGs. However, these metrics are less directly interpretable than distance-based measures. Moreover, $\mathcal{P}_H$ and $\mathcal{R}_H$ are strongly influenced by the depth of the hierarchy. For example, predicting a sibling to the ground-truth at a large depth yields a smaller error than predicting a sibling to the ground-truth near the root. This effect can be problematic in unbalanced hierarchies, where the depths of leaves vary, making error comparisons across branches less straightforward.

4.2 Hierarchical open-set classification

Papers D and E study hierarchical open-set classification, a subdomain of hierarchical classification that considers the classification of unknown classes not present in the labeled training set. Unlike standard closed-set hierarchical classification, the goal is not only to predict among known classes, but also to produce semantically meaningful predictions for samples originating from unseen classes.

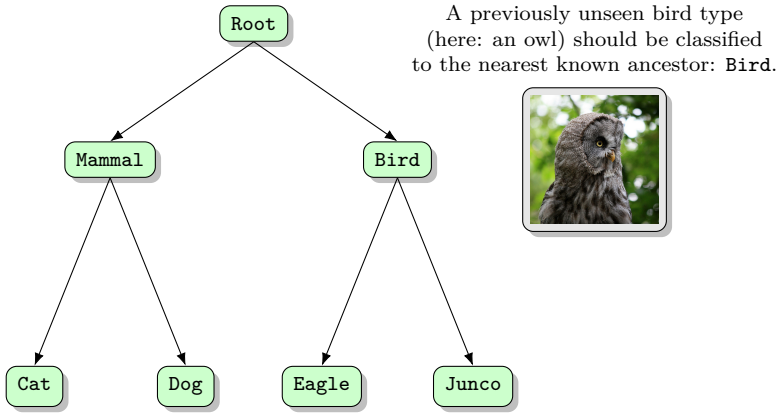

Figure 4.4: Illustration of hierarchical open-set classification. A model is trained using data from the leaf nodes of a known class hierarchy. At inference time, samples from previously unseen classes should be assigned to the most appropriate internal nodes of the hierarchy. In this example, an owl (unseen during training) should be classified as *bird*, without committing to any of the known bird subclasses.

The core idea is that samples from unseen classes should be assigned to an appropriate internal node in the class hierarchy, rather than receiving a binary OOD flag. For example, if the training data contains a set of animal classes, including multiple dog breeds, samples of unseen dogs should be classified as the coarser category *dog*. This contrasts with the large body of work on OOD detection, which focuses on the binary rejection of such samples, without producing a semantically informative prediction [18], [19], [32]. Figure 4.4 illustrates the idea of hierarchical open-set classification.

Beyond Papers D and E, hierarchical open-set classification has been studied in a limited number of works [33]–[37] and has appeared under various names, including *hierarchical novelty detection*, *hierarchical OOD classification*, *hierarchical OOD detection*, and *fine-grained OOD detection*. From a structural perspective, hierarchical open-set classification corresponds to a non-mandatory leaf-node setting: known classes should be predicted to leaf nodes, whereas unknown classes should be predicted to internal nodes. Existing approaches further consider the single-path prediction setting, where each sample is associated with a single ground-truth node. The problem could,

however, be extended to the multi-path setting. Moreover, while the problem naturally applies to DAGs, the current literature predominantly focuses on tree-structured hierarchies.

Top-down methods

The first work to study this setting [33] proposed the *top-down* method as one of two baseline approaches for hierarchical open-set classification. In this approach, conditional classifiers are associated with each internal node of the hierarchy. During inference, predictions proceed top-down from the root, stopping at the first internal node where the local classifier's confidence falls below a predefined threshold, or when a leaf node is reached. According to the method categorization introduced in Section 4.1, the top-down method is best described as a hierarchical inference approach.

Linderman *et al.* [37] follow a top-down strategy and assign node-specific thresholds by evaluating local *true negative rates* on in-distribution training data. Furthermore, they incorporate a hierarchical loss by training separate softmax classifiers at each internal node. To encourage uncertainty, the local classifiers are trained to produce uniform predictions for samples whose ground-truth labels do not lie within the descendant subtree of the corresponding node.

Flattening methods

The second baseline approach proposed in [33] is the *flattening* method. In this approach, a single model is trained to jointly predict both leaf classes and internal hierarchy nodes. Since labeled data for unknown classes corresponding to internal nodes are unavailable, training data for these nodes are typically constructed using samples from their known descendant leaf classes. This flattening method can be categorized as a label granularity approach. Figure 4.5 illustrates a comparison between the top-down and flattening approaches.

An example of a flattening-based method is proposed by Ruiz *et al.* [34], who learn a feature space in which prototypes represent the nodes of the hierarchy. The model and prototypes are optimized using a set of triplet losses [155]. Following the flattening strategy of [33], internal nodes are represented using data from their known descendant leaves. At inference time, predictions are

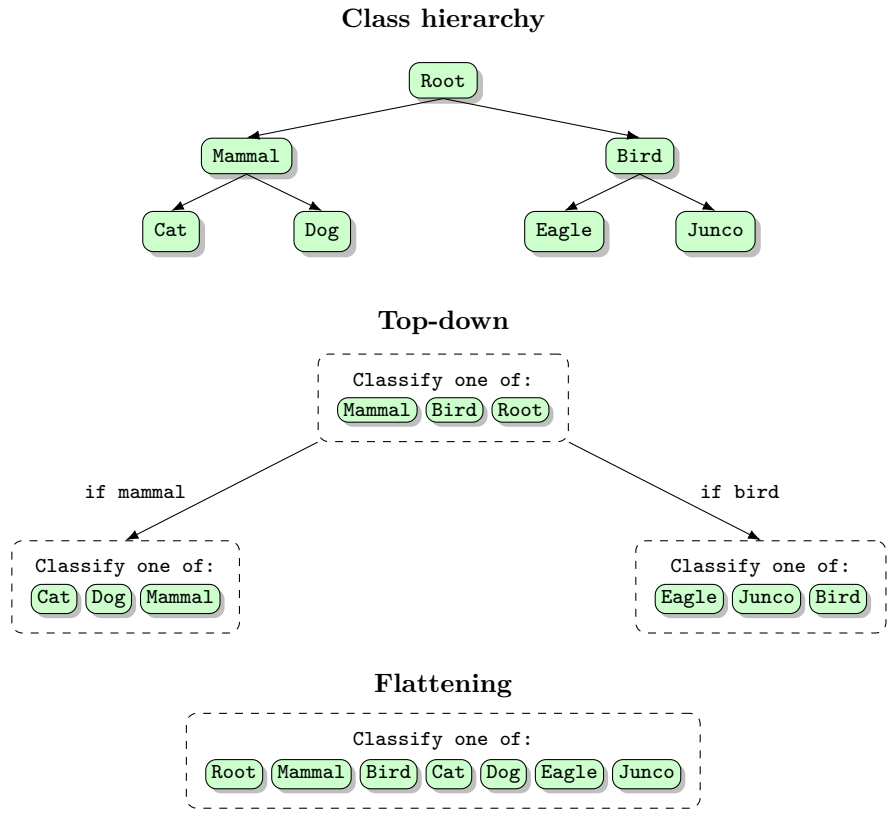

Figure 4.5: Comparison of top-down and flattening approaches for hierarchical open-set classification. In the top-down approach, we use local classifiers for internal nodes of the hierarchy, each predicting among its child nodes and a local OOD option. In contrast, the flattening approach uses a single classifier that predicts directly over all nodes in the hierarchy.

made by measuring distances between test samples and node prototypes in the learned feature space and selecting the closest node, whether leaf or internal.

Pyakurel *et al.* [35] use the flattening method but employ evidential deep learning [156] to capture model uncertainty. Evidential deep learning involves predicting the parameters of a Dirichlet distribution over the class probabilities, rather than directly producing a predictive distribution. This approach can be considered a more principled way of representing uncertainty in neural networks. The method [35] adapts evidential deep learning to hierarchical open-set classification in combination with the flattening approach.

Beyond baseline approaches

In [36], Pyakurel *et al.* move beyond the baseline approaches by extending the flattening paradigm. They train a neural network similarly to the other methods in the flattening category. However, instead of using these logits directly for predicting among the known and open-set categories, they introduce *state representations*. The logits corresponding to a valid state, *i.e.*, paths in the hierarchy, are multiplied to produce scores for each category. These are normalized to obtain the final predictive distribution.

In Paper D, we introduce ProHOC. ProHOC approaches hierarchical open-set classification from a label granularity perspective, similarly to flattening-based methods. Specifically, we exploit the training data by utilizing labels at all available granularity levels, *i.e.*, at all hierarchy depths. To this end, we train separate classification networks for each hierarchy depth, using all training data remapped to the corresponding granularity level, with each network responsible for classifying the classes within that depth. We further propose a method for approximating the local conditional distributions (*i.e.*, the probabilities over child nodes and the local OOD prediction) based on the predictions from the depth-specific networks. These conditional distributions are combined to evaluate the full predictive distribution over the hierarchy. Following our categorization in Section 4.1, ProHOC combines label granularity through the multi-depth formulation with hierarchical inference via probabilistic modeling.

Evaluation

As discussed in Section 4.1, there are multiple ways to evaluate hierarchical classification, and the same holds for hierarchical open-set classification. A

recurring observation in the works on hierarchical open-set classification is that closed-set performance (accuracy on ID data) and OOD performance (accuracy on OOD data) are often in a trade-off relationship. In particular, OOD performance can be improved by discouraging leaf-node predictions, whereas closed-set performance can be improved by encouraging leaf predictions.

Several works [33]–[35] explicitly explore this trade-off by controlling the bias between leaf and internal node predictions. This is commonly achieved by scaling the scores of internal nodes using a bias parameter. By evaluating performance across a range of bias values, a curve describing ID accuracy versus OOD accuracy can be obtained. The overall performance is then summarized by reporting the area under this curve. However, selecting an operating point (*i.e.*, a specific value of the bias parameter) is not straightforward in practice, since OOD data are not available during training, and therefore the operating point cannot be tuned according to OOD performance.

In contrast, the ProHOC method introduced in Paper D yields a default operating point by design. Consequently, we report performance only at this operating point. As discussed in Section 4.1, we primarily evaluate hierarchical distance in Papers D and E. We compute separate metrics for ID and OOD data, and define the overall performance as the mean of these two.

Semi-supervised hierarchical open-set classification

The works on hierarchical open-set classification discussed above treat open-set prediction as a purely test-time problem: models are trained on labeled ID data and are required to predict OOD data only at inference time. However, as shown in the literature on open-set semi-supervised learning (Chapter 3), exposure to unlabeled real-world data, containing both ID and OOD samples, can improve open-set performance.

Motivated by this observation, Paper E introduces the problem setting of *semi-supervised hierarchical open-set classification*. The overall objective remains the same as in the supervised case, but the training data now consists of a labeled ID set and an unlabeled set that may contain both ID and OOD samples. The unlabeled OOD samples belong somewhere in the hierarchy, but not at the leaf nodes.

In addition to introducing this problem setting, Paper E proposes SemiHOC, a method that builds upon ProHOC and introduces a pseudo-labeling strategy within a teacher-student framework, specifically adapted for the semi-supervised

hierarchical open-set setting. We show that incorporating the unlabeled data, containing both ID and OOD samples, leads to improved open-set performance compared to the supervised alternative.

CHAPTER 5

Summary of included papers

This chapter provides a summary of the included papers.

5.1 Paper A

Erik Wallin, Lennart Svensson, Fredrik Kahl, Lars Hammarstrand
DoubleMatch: Improving Semi-Supervised Learning with Self-Supervision
Published in the proceedings of the 2022 26th International Conference on Pattern Recognition
pp. 2871–2877
DOI: 10.1109/ICPR56361.2022.9956182

This paper proposes DoubleMatch, a method for (closed-set) semi-supervised learning. A common technique for existing methods for semi-supervised learning is to employ confidence-based pseudo-labeling on unlabeled data. This process assigns artificial labels to unlabeled data for which the model's predictions exceed a confidence threshold. Unlabeled data for which the model produces less confident predictions are disregarded from the training objective.

Consequently, these methods may ignore large parts of unlabeled data, in particular for more challenging classification problems. For better utilization of unlabeled data, this paper proposes the inclusion of a self-supervised component to enable learning from all unlabeled data. This additional self-supervision involves aligning feature predictions across weak and strong augmentations of each sample. More specifically, we implement this self-supervision as an extension of the widely adopted SSL baseline FixMatch. Our proposed method is evaluated on benchmark datasets CIFAR-10, CIFAR-100, SVHN, and STL-10. DoubleMatch demonstrates particularly strong results on CIFAR-100 and STL-10, with improved accuracies and training speed when compared to FixMatch. However, on the relatively simpler classification tasks of CIFAR-10 and SVHN, our proposed method is not equally effective. A possible explanation could be the model's ability to generate sufficiently many correct pseudo-labels when the classification problem is relatively straightforward, diminishing the benefits introduced by the additional self-supervision.

Contributions: I designed the method, implemented the code base, ran the experiments, and wrote all sections of the paper. Lennart Svensson, Fredrik Kahl, and Lars Hammarstrand contributed through discussion and feedback. Lars Hammarstrand created Figure 1 and Figure 2.

5.2 Paper B

Erik Wallin, Lennart Svensson, Fredrik Kahl, Lars Hammarstrand
Improving Open-Set Semi-Supervised Learning with Self-Supervision
Published in the proceedings of the 2024 IEEE/CVF Winter Conference on Applications of Computer Vision
pp. 2345–2354
DOI: 10.1109/WACV57701.2024.00235

This paper studies open-set semi-supervised learning (OSSL), a more realistic scenario where we assume that the unlabeled data may contain unknown classes not present in the labeled data. Many existing works for OSSL use methods that involve detecting ID data in unlabeled data for inclusion in a traditional SSL loss. The method proposed in this paper, SeFOSS, instead follows the philosophy of DoubleMatch from Paper A, aiming to learn from all unlabeled data, regardless of whether they are ID or OOD. To achieve this, SeFOSS

incorporates the self-supervision proposed by DoubleMatch on all unlabeled data. Additionally, SeFOSS applies a pseudo-labeling loss on unlabeled data that confidently belong to the known classes. To confidently identify ID samples among the unlabeled data, SeFOSS employs an energy-based score for ID/OOD discrimination, together with an adaptive thresholding procedure based on the energy distribution of labeled data. SeFOSS is evaluated and compared with existing methods for OSSL on open-set scenarios involving datasets CIFAR-10, CIFAR-100, SVHN, ImageNet, and noise. The experimental results show that SeFOSS exhibits an unmatched overall performance in terms of both closed-accuracy and OOD detection across the range of studied scenarios. While other methods perform well on a few scenarios, they fail to consistently and robustly perform on all scenarios. Moreover, this paper shows that methods for closed-set semi-supervised learning may perform better in terms of closed-set accuracy than previously reported by existing works. In fact, FixMatch outperforms all OSSL methods on closed-set accuracy in the experiments conducted in this paper. However, FixMatch performs poorly in terms of OOD detection, which is of significant importance for real-world applications.

Contributions: I designed the method, implemented the code base, ran the experiments, and wrote all sections of the paper. Lennart Svensson, Fredrik Kahl, and Lars Hammarstrand contributed through discussion and feedback.

5.3 Paper C

Erik Wallin, Lennart Svensson, Fredrik Kahl, Lars Hammarstrand
ProSub: Probabilistic Open-Set Semi-supervised Learning with Subspace-Based Out-of-Distribution Detection
Published in the proceedings of the 2024 European Conference on Computer Vision
Lecture Notes in Computer Science, vol. 15119, Springer, 2025
DOI: 10.1007/978-3-031-73030-6_8

In this paper, we continue the study of open-set semi-supervised learning and improve upon the SeFOSS framework. We introduce a new score for distinguishing in-distribution from out-of-distribution samples, and we propose a model for probabilistic predictions of whether samples are ID or OOD. The

score is based on the observation that ID features tend to lie close to a subspace in representation space; we approximate this subspace and compute the ID score using the cosine of the angle between features and this subspace. We find that the strong performance of this *subspace score* emerges as a consequence of using cosine-similarities for self-supervision. To obtain probabilistic ID/OOD predictions, we model the distributions of these scores for ID and OOD data using beta distributions, with parameters estimated via a variant of the expectation-maximization algorithm tailored to the OSSL setting. These components are incorporated into our ProSub framework, which achieves state-of-the-art results on several OSSL benchmarks and improves upon the results of SeFOSS.

Contributions: The ideas for the subspace score and the probabilistic modeling were developed primarily by Lars Hammarstrand and me. I implemented the code base, ran the experiments, and wrote all sections of the paper. Lennart Svensson, Fredrik Kahl, and Lars Hammarstrand contributed through discussion and feedback. Lars Hammarstrand did the main work in creating Figure 1 and Figure 2.

5.4 Paper D

Erik Wallin, Fredrik Kahl, Lars Hammarstrand
ProHOC: Probabilistic Hierarchical Out-of-Distribution Classification via Multi-Depth Networks
Published in the proceedings of the 2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition
pp. 20612–20621
DOI: 10.1109/CVPR52734.2025.01919

The two previous papers on open-set semi-supervised learning focus on binary OOD detection, where samples are predicted as either in-distribution or out-of-distribution, which is standard in the literature. In this paper, we move beyond the binary setting and address hierarchical open-set classification, where the goal is to predict OOD samples as the most appropriate internal category of a known class hierarchy. A sample may thus be assigned to one of the leaf categories (the ID classes) or to any internal node of the hierarchy, representing a fine-grained OOD category. In our framework, ProHOC, we

model the predictive distribution over the full hierarchy, corresponding to a distribution over all ID classes and fine-grained OOD categories. This distribution is approximated using classification networks trained at each depth of the hierarchy. The intuition is that fine-grained OOD samples are confidently predicted by higher-level models that generalize over broad categories, while deeper, more specialized models express higher uncertainty. We show that ProHOC achieves strong performance on hierarchical benchmarks and argue that it provides an extensible and scalable foundation for further work on hierarchical open-set classification.

Contributions: Lars Hammarstrand and I developed the idea of probabilistic modeling by estimating the conditionals. I developed the other parts of the method, implemented the code base, ran the experiments, and wrote all sections of the paper. Fredrik Kahl and Lars Hammarstrand contributed through discussion and feedback.

5.5 Paper E

Erik Wallin, Fredrik Kahl, Lars Hammarstrand
Semi-Supervised Hierarchical Open-Set Classification
To be published in the proceedings of the 2026 IEEE/CVF Winter Conference on Applications of Computer Vision.

In this final paper of the thesis, we combine the fields of open-set semi-supervised learning and hierarchical open-set classification by introducing semi-supervised hierarchical open-set classification. This setting follows the open-set semi-supervised learning paradigm, but instead of binary ID/OOD predictions, the goal is to assign OOD samples to the most appropriate internal node of a class hierarchy, as in the ProHOC paper. Building on the ProHOC framework, we extend hierarchical open-set classification to the semi-supervised setting. We identify two key challenges: 1) standard confidence-based pseudo-labeling is unreliable for OOD data, and 2) models tend to become overconfident during training, pushing OOD samples toward overly specific categories. We address these issues through two contributions, subtree pseudo-labels and age-gating, which we integrate into SemiHOC, a teacher–student framework that enables ProHOC to learn from unlabeled data. Our results show that SemiHOC outperforms self-supervised pretraining followed by supervised adaptation and even matches the fully supervised counterpart (using all available labels) with

only 20 labels per class on the iNaturalist19 benchmark, demonstrating that unlabeled open-set data can be effectively leveraged in this setting.

Contributions: I developed the method, implemented the code base, ran the experiments, and wrote all sections of the paper. Fredrik Kahl and Lars Hammarstrand contributed through discussion and feedback.

CHAPTER 6

Concluding remarks and future work

This thesis has studied deep learning-based classification under conditions common in real-world applications, which remain insufficiently addressed in much of the existing literature. In particular, we have focused on limited supervision through semi-supervised learning, the presence of unknown classes via open-set recognition, and structured semantic relationships between classes in the form of class hierarchies. The appended papers contribute with methods and empirical insights that improve robustness and reliability across these settings.

During the course of this work, the field of deep learning has undergone rapid development. In particular, there has been a shift towards large-scale foundation models, *i.e.*, models that are pretrained on large and diverse datasets, which can be adapted to downstream tasks. Examples of these include large language models such as Llama [157] and DeepSeek [158], vision models such as CLIP [159] and DINOv2 [40], and generative models such as Stable Diffusion [160]. As training such models is often infeasible for individual practitioners or academic groups, much of the community's focus has shifted toward adapting and applying these pretrained models rather than training models from scratch. The works in this thesis partly followed this shift by

using the vision foundation model DINOv2 as a backbone in Papers D and E. While these foundation models are useful in many tasks, we argue that there remains an important role for smaller-scale models trained from scratch, particularly in domains where foundation models do not exist or where data are not publicly available.

Future work

In Chapter 1, we motivated this thesis by introducing three challenges for deploying deep classification systems in real-world settings: limited labeled data, unknown classes, and class relations. This is not an exhaustive list. Other important issues that are not addressed in this thesis include, for example, class imbalances, *i.e.*, when the classes are unevenly represented in the data, and covariate shifts, *i.e.*, when the data distribution for a class differs between training and deployment. A natural continuation of this thesis is to extend the proposed methods to account for such scenarios.

One concrete example is the combination of open-set semi-supervised learning with long-tailed semi-supervised learning. While both settings aim to capture realistic aspects of semi-supervised learning, the first by accounting for unknown classes and the second by considering class-imbalanced scenarios, they have, to the best of our knowledge, so far only been studied separately. A valuable direction for future work is to consider both unknown classes and class imbalances simultaneously within a unified framework.

Another open research question relates to uncertainty in classification models. In particular, we have noticed in our work that it is challenging to distinguish between uncertainty arising from ambiguity among known classes and uncertainty caused by unseen classes. While these two types of uncertainty are often studied separately, they are related and frequently addressed using similar techniques (it is, *e.g.*, common to model both types of uncertainties using the maximum predicted probability). Methods for disentangling these sources of uncertainty could lead to more reliable and robust deep learning models.

Finally, research in deep learning-based classification is often conducted in the domain of computer vision, largely due to the availability of well-established benchmark datasets that enable evaluation and comparison. For this reason, the methods developed in this thesis are evaluated on image classification benchmarks, facilitating comparison with existing work. However, while the

experimental focus is on computer vision, the underlying ideas are not domain-specific. Extending these methods to other domains (*e.g.*, non-visual sensor data such as radar, sonar, lidar, and biomedical data) presents both opportunities and challenges. In particular, techniques such as data augmentation, which play a central role in semi-supervised learning, are domain-dependent and require adaptation to new domains. Moreover, the (in comparison) limited availability of public datasets and benchmarks outside computer vision remains an obstacle for method development and evaluation. Addressing these challenges is an important step toward the broader applicability of the methods studied in this thesis.

References

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2016.

[2] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention*, 2015.

[3] Y. Kim, "Convolutional neural networks for sentence classification," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2014.

[4] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019.

[5] S. Hershey, S. Chaudhuri, D. P. Ellis, *et al.*, "Cnn architectures for large-scale audio classification," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2017.

[6] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," in *Advances in Neural Information Processing Systems*, 2020.

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

[8] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "SQuAD: 100,000+ questions for machine comprehension of text," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2016.

[9] H. Pham, Z. Dai, Q. Xie, and Q. V. Le, "Meta pseudo labels," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

[10] Q. Xie, Z. Dai, E. Hovy, T. Luong, and Q. Le, "Unsupervised data augmentation for consistency training," in *Advances in Neural Information Processing Systems*, 2020.

[11] K. Sohn, D. Berthelot, N. Carlini, *et al.*, "FixMatch: Simplifying semi-supervised learning with consistency and confidence," in *Advances in Neural Information Processing Systems*, 2020.

[12] Y. Xu, L. Shang, J. Ye, *et al.*, "Dash: Semi-supervised learning with dynamic thresholding," in *Proceedings of the International Conference on Machine Learning*, 2021.

[13] B. Zhang, Y. Wang, W. Hou, *et al.*, "FlexMatch: Boosting semi-supervised learning with curriculum pseudo labeling," in *Advances in Neural Information Processing Systems*, 2021.

[14] Y. Wang, H. Chen, Q. Heng, *et al.*, "Freematch: Self-adaptive thresholding for semi-supervised learning," in *International Conference on Learning Representations*, 2023.

[15] X. Zhai, A. Oliver, A. Kolesnikov, and L. Beyer, "S4L: Self-supervised semi-supervised learning," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019.

[16] I. Nassar, M. Hayat, E. Abbasnejad, H. Rezatofighi, and G. Haffari, "Protocon: Pseudo-label refinement via online clustering and prototypical consistency for efficient semi-supervised learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

[17] M. Zheng, S. You, L. Huang, F. Wang, C. Qian, and C. Xu, "SimMatch: Semi-supervised learning with similarity matching," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

[18] D. Hendrycks and K. Gimpel, "A baseline for detecting misclassified and out-of-distribution examples in neural networks," in *International Conference on Learning Representations*, 2017.

[19] Y. Sun, Y. Ming, X. Zhu, and Y. Li, "Out-of-distribution detection with deep nearest neighbors," in *Proceedings of the International Conference on Machine Learning*, 2022.

[20] Y. Ming, Y. Sun, O. Dia, and Y. Li, "How to exploit hyperspherical embeddings for out-of-distribution detection?" In *International Conference on Learning Representations*, 2023.

[21] K. Saito, D. Kim, and K. Saenko, "Openmatch: Open-set semi-supervised learning with open-set consistency regularization," in *Advances in Neural Information Processing Systems*, 2021.

[22] S. Mo, J.-C. Su, C.-Y. Ma, *et al.*, "Ropaws: Robust semi-supervised representation learning from uncurated data," in *International Conference on Learning Representations*, 2023.

[23] Y. Wang, P. Qiao, C. Liu, G. Song, X. Zheng, and J. Chen, "Out-of-distributed semantic pruning for robust semi-supervised learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

[24] G. A. Miller, "WordNet: A lexical database for english," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[25] C. Linnaeus, *Systema Naturae per regna tria naturae, secundum classes, ordines, genera, species; cum characteribus, differentiis, synonymis, locis*. apud JB Delamolliere, 1789, vol. 1.

[26] L. Bertinetto, R. Mueller, K. Tertikas, S. Samangooei, and N. A. Lord, "Making better mistakes: Leveraging class hierarchies with deep networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[27] S. Karthik, A. Prabhu, P. K. Dokania, and V. Gandhi, "No cost likelihood manipulation at test time for making better mistakes in deep networks," in *International Conference on Learning Representations*, 2021.

[28] A. Garg, D. Sani, and S. Anand, "Learning hierarchy aware features for reducing mistake severity," in *Proceedings of the European Conference on Computer Vision*, 2022.

[29] T. Liang and J. Davis, "Inducing neural collapse to a fixed hierarchy-aware frame for reducing mistake severity," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

[30] K. Jain, S. Karthik, and V. Gandhi, "Test-time amendment with a coarse classifier for fine-grained classification," in *Advances in Neural Information Processing Systems*, 2023.

[31] T.-Y. Wu, P. Morgado, P. Wang, C.-H. Ho, and N. Vasconcelos, "Solving long-tailed recognition with deep realistic taxonomic classifier," in *Proceedings of the European Conference on Computer Vision*, 2020.

[32] S. Liang, Y. Li, and R. Srikant, "Enhancing the reliability of out-of-distribution image detection in neural networks," in *International Conference on Learning Representations*, 2018.

[33] K. Lee, K. Lee, K. Min, Y. Zhang, J. Shin, and H. Lee, "Hierarchical novelty detection for visual object recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018.

[34] I. Ruiz and J. Serrat, "Hierarchical novelty detection for traffic sign recognition," *Sensors*, vol. 22, no. 12, p. 4389, 2022.

[35] S. Pyakurel and Q. Yu, "Hierarchical novelty detection via fine-grained evidence allocation," in *Proceedings of the International Conference on Machine Learning*, 2024.

[36] S. Pyakurel and Q. Yu, "Learning state-based node representations from a class hierarchy for fine-grained open-set detection," in *Proceedings of the International Conference on Machine Learning*, 2025.

[37] R. Linderman, J. Zhang, N. Inkawhich, H. Li, and Y. Chen, "Fine-grain inference on out-of-distribution data with hierarchical classification," in *Proceedings of the Conference on Lifelong Learning Agents*, 2023.

[38] O. Chapelle, B. Schölkopf, and A. Zien, *Semi-supervised Learning*. The MIT Press, 2006.

[39] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*. Springer, 2006, vol. 4.

[40] M. Oquab, T. Darcet, T. Moutakanni, *et al.*, "DINOv2: Learning robust visual features without supervision," *Transactions on Machine Learning Research*, 2024.

[41] H. Scudder, "Probability of error of some adaptive pattern-recognition machines," *IEEE Transactions on Information Theory*, vol. 11, no. 3, pp. 363–371, 1965.

[42] S. Fralick, "Learning to recognize patterns without a teacher," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 57–64, 1967.

[43] A. Agrawala, "Learning with a probabilistic teacher," *IEEE Transactions on Information Theory*, vol. 16, no. 4, pp. 373–379, 1970.

[44] D. Yarowsky, "Unsupervised word sense disambiguation rivaling supervised methods," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 1995.

[45] K. Nigam, A. K. McCallum, S. Thrun, and T. Mitchell, "Text classification from labeled and unlabeled documents using em," *Machine Learning*, vol. 39, pp. 103–134, 2000.

[46] D.-H. Lee, "Pseudo-Label: The simple and efficient semi-supervised learning method for deep neural networks," in *Proceedings of the ICML Workshop on Challenges in Representation Learning*, 2013.

[47] A. Rasmus, M. Berglund, M. Honkala, H. Valpola, and T. Raiko, "Semi-supervised learning with ladder networks," in *Advances in Neural Information Processing Systems*, 2015.

[48] S. Laine and T. Aila, "Temporal ensembling for semi-supervised learning," in *International Conference on Learning Representations*, 2017.

[49] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[50] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," in *Advances in Neural Information Processing Systems*, 2017.

[51] T. Miyato, S.-i. Maeda, M. Koyama, and S. Ishii, "Virtual adversarial training: A regularization method for supervised and semi-supervised learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 8, pp. 1979–1993, 2018.

[52] D. Berthelot, N. Carlini, E. D. Cubuk, *et al.*, "ReMixMatch: Semi-supervised learning with distribution matching and augmentation anchoring," in *International Conference on Learning Representations*, 2020.

[53] E. D. Cubuk, B. Zoph, J. Shlens, and Q. Le, "RandAugment: Practical automated data augmentation with a reduced search space," in *Advances in Neural Information Processing Systems*, 2020.

[54] R. Sennrich, B. Haddow, and A. Birch, "Improving neural machine translation models with monolingual data," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2016.

[55] T. DeVries and G. W. Taylor, "Improved regularization of convolutional neural networks with cutout," *arXiv preprint arXiv:1708.04552*, 2017.

[56] F. Yang, K. Wu, S. Zhang, *et al.*, "Class-aware contrastive semi-supervised learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

[57] J. Wang, T. Lukasiewicz, D. Massiceti, X. Hu, V. Pavlovic, and A. Neophytou, "Np-match: When neural processes meet semi-supervised learning," in *Proceedings of the International Conference on Machine Learning*, 2022.

[58] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *International Conference on Learning Representations*, 2018.

[59] V. Verma, K. Kawaguchi, A. Lamb, *et al.*, "Interpolation consistency training for semi-supervised learning," *Neural Networks*, vol. 145, pp. 90–106, 2022.

[60] D. Berthelot, N. Carlini, I. Goodfellow, N. Papernot, A. Oliver, and C. A. Raffel, "MixMatch: A holistic approach to semi-supervised learning," in *Advances in Neural Information Processing Systems*, 2019.

[61] 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.

[62] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proceedings of the International Conference on Machine Learning*, 2020.

[63] M. Caron, H. Touvron, I. Misra, *et al.*, "Emerging properties in self-supervised vision transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

[64] J.-B. Grill, F. Strub, F. Altché, *et al.*, "Bootstrap your own latent - a new approach to self-supervised learning," in *Advances in Neural Information Processing Systems*, 2020.

[65] H. Bao, L. Dong, S. Piao, and F. Wei, "Beit: Bert pre-training of image transformers," *International Conference on Learning Representations*, 2022.

[66] C. Doersch, A. Gupta, and A. A. Efros, "Unsupervised visual representation learning by context prediction," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2015.

[67] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," in *International Conference on Learning Representations*, 2018.

[68] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *Proceedings of the European Conference on Computer Vision*, 2016.

[69] X. Chen and K. He, "Exploring simple siamese representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

[70] X. Wang, D. Kihara, J. Luo, and G.-J. Qi, "EnAET: A self-trained framework for semi-supervised and supervised learning with ensemble transformations," *IEEE Transactions on Image Processing*, vol. 30, pp. 1639–1647, 2020.

[71] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, and G. E. Hinton, "Big self-supervised models are strong semi-supervised learners," in *Advances in Neural Information Processing Systems*, 2020.

[72] J. Zbontar, L. Jing, I. Misra, Y. LeCun, and S. Deny, "Barlow twins: Self-supervised learning via redundancy reduction," in *Proceedings of the International Conference on Machine Learning*, 2021.

[73] P. Goyal, Q. Duval, I. Seessel, *et al.*, "Vision models are more robust and fair when pretrained on uncurated images without supervision," *arXiv preprint arXiv:2202.08360*, 2022.

[74] Y. Chen, X. Zhu, W. Li, and S. Gong, "Semi-supervised learning under class distribution mismatch," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[75] L. Han, H.-J. Ye, and D.-C. Zhan, "On pseudo-labeling for class-mismatch semi-supervised learning," *Transactions on Machine Learning Research*, 2022.

[76] Z. Huang, J. Yang, and C. Gong, "They are not completely useless: Towards recycling transferable unlabeled data for class-mismatched semi-supervised learning," *IEEE Transactions on Multimedia*, vol. 25, pp. 1844–1857, 2022.

[77] Q. Yu, D. Ikami, G. Irie, and K. Aizawa, "Multi-task curriculum framework for open-set semi-supervised learning," in *Proceedings of the European Conference on Computer Vision*, 2020.

[78] J. Huang, C. Fang, W. Chen, *et al.*, "Trash to treasure: Harvesting ood data with cross-modal matching for open-set semi-supervised learning," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

[79] Y. Fan, A. Kukleva, D. Dai, and B. Schiele, "SSB: Simple but strong baseline for boosting performance of open-set semi-supervised learning," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

[80] Z. Li, L. Qi, Y. Shi, and Y. Gao, "IOMatch: Simplifying open-set semi-supervised learning with joint inliers and outliers utilization," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

[81] W. Liu, X. Wang, J. Owens, and Y. Li, "Energy-based out-of-distribution detection," in *Advances in Neural Information Processing Systems*, 2020.

[82] R. He, Z. Han, X. Lu, and Y. Yin, "Safe-student for safe deep semi-supervised learning with unseen-class unlabeled data," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

[83] Q. Ma, J. Gao, B. Zhan, Y. Guo, J. Zhou, and Y. Wang, "Rethinking safe semi-supervised learning: Transferring the open-set problem to a close-set one," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

[84] J. Park, S. Yun, J. Jeong, and J. Shin, "Opencos: Contrastive semi-supervised learning for handling open-set unlabeled data," in *Proceedings of the European Conference on Computer Vision*, 2022.

[85] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, "Deep clustering for unsupervised learning of visual features," in *Proceedings of the European Conference on Computer Vision*, 2018.

[86] L.-Z. Guo, Z.-Y. Zhang, Y. Jiang, Y.-F. Li, and Z.-H. Zhou, "Safe deep semi-supervised learning for unseen-class unlabeled data," in *Proceedings of the International Conference on Machine Learning*, 2020.

[87] X. Zhao, K. Krishnateja, R. Iyer, and F. Chen, "How out-of-distribution data hurts semi-supervised learning," in *Proceedings of the IEEE International Conference on Data Mining*, 2022.

[88] R. He, Z. Han, Y. Yang, and Y. Yin, "Not all parameters should be treated equally: Deep safe semi-supervised learning under class distribution mismatch," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.

[89] Z. Huang, M.-J. Sidhom, B. Wessler, and M. C. Hughes, "Fix-a-step: Semi-supervised learning from uncurated unlabeled data," in *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2023.

[90] L.-Z. Guo, Y.-G. Zhang, Z.-F. Wu, J.-J. Shao, and Y.-F. Li, "Robust semi-supervised learning when not all classes have labels," in *Advances in Neural Information Processing Systems*, 2022.

[91] K. Cao, M. Brbic, and J. Leskovec, "Open-world semi-supervised learning," in *International Conference on Learning Representations*, 2022.

[92] M. N. Rizve, N. Kardan, and M. Shah, "Towards realistic semi-supervised learning," in *Proceedings of the European Conference on Computer Vision*, 2022.

[93] M. N. Rizve, N. Kardan, S. Khan, F. Shahbaz Khan, and M. Shah, "Openldn: Learning to discover novel classes for open-world semi-supervised learning," in *Proceedings of the European Conference on Computer Vision*, 2022.

[94] J. Liu, Y. Wang, T. Zhang, Y. Fan, Q. Yang, and J. Shao, "Open-world semi-supervised novel class discovery," in *Proceedings of the International Joint Conference on Artificial Intelligence*, 2023.

[95] K. Han, A. Vedaldi, and A. Zisserman, "Learning to discover novel visual categories via deep transfer clustering," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019.

[96] Z. Zhong, L. Zhu, Z. Luo, S. Li, Y. Yang, and N. Sebe, "Openmix: Reviving known knowledge for discovering novel visual categories in an open world," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

[97] W. Li, Z. Fan, J. Huo, and Y. Gao, "Modeling inter-class and intra-class constraints in novel class discovery," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

[98] X. Wang, Z. Wu, L. Lian, and S. X. Yu, "Debiased learning from naturally imbalanced pseudo-labels," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

[99] Y. Oh, D.-J. Kim, and I. S. Kweon, "Daso: Distribution-aware semantics-oriented pseudo-label for imbalanced semi-supervised learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

[100] C. Wei, K. Sohn, C. Mellina, A. Yuille, and F. Yang, "Crest: A class-rebalancing self-training framework for imbalanced semi-supervised learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

[101] J. Kim, Y. Hur, S. Park, E. Yang, S. J. Hwang, and J. Shin, "Distribution aligning refinery of pseudo-label for imbalanced semi-supervised learning," in *Advances in Neural Information Processing Systems*, 2020.

[102] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *Proceedings of the International Conference on Machine Learning*, 2015.

[103] G. Kang, L. Jiang, Y. Yang, and A. G. Hauptmann, "Contrastive adaptation network for unsupervised domain adaptation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

[104] K. Saito, S. Yamamoto, Y. Ushiku, and T. Harada, "Open set domain adaptation by backpropagation," in *Proceedings of the European Conference on Computer Vision*, 2018.

[105] P. Panareda Busto and J. Gall, "Open set domain adaptation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2017.

[106] W. J. Scheirer, A. de Rezende Rocha, A. Sapkota, and T. E. Boult, "Toward open set recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 7, pp. 1757–1772, 2012.

[107] A. Bendale and T. E. Boult, "Towards open set deep networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2016.

[108] D. Hendrycks, M. Mazeika, and T. Dietterich, "Deep anomaly detection with outlier exposure," in *International Conference on Learning Representations*, 2019.

[109] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *Advances in Neural Information Processing Systems*, 2017.

[110] K. Lee, K. Lee, H. Lee, and J. Shin, "A simple unified framework for detecting out-of-distribution samples and adversarial attacks," in *Advances in Neural Information Processing Systems*, 2018.

[111] S. Vaze, K. Han, A. Vedaldi, and A. Zisserman, "Open-set recognition: A good closed-set classifier is all you need," in *International Conference on Learning Representations*, 2022.

[112] J. Yang, H. Wang, L. Feng, *et al.*, "Semantically coherent out-of-distribution detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

[113] F. Lu, K. Zhu, W. Zhai, K. Zheng, and Y. Cao, "Uncertainty-aware optimal transport for semantically coherent out-of-distribution detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

[114] H. Liu, Z. Cao, M. Long, J. Wang, and Q. Yang, "Separate to adapt: Open set domain adaptation via progressive separation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

[115] A. G. McDonald, S. Boyce, and K. F. Tipton, "ExplorEnz: The primary source of the IUBMB enzyme list," *Nucleic Acids Research*, vol. 37, no. suppl_1, pp. D593–D597, 2009.

[116] M. Ashburner, C. A. Ball, J. A. Blake, *et al.*, "Gene ontology: Tool for the unification of biology," *Nature Genetics*, vol. 25, no. 1, pp. 25–29, 2000.

[117] J. Deng, J. Krause, A. C. Berg, and L. Fei-Fei, "Hedging your bets: Optimizing accuracy-specificity trade-offs in large scale visual recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2012.

[118] C. N. Silla Jr and A. A. Freitas, "A survey of hierarchical classification across different application domains," *Data Mining and Knowledge Discovery*, vol. 22, no. 1, pp. 31–72, 2011.

[119] Z. Yan, H. Zhang, R. Piramuthu, *et al.*, "HD-CNN: Hierarchical deep convolutional neural networks for large scale visual recognition," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2015.

[120] 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.

[121] K. Ahmed, M. H. Baig, and L. Torresani, "Network of experts for large-scale image categorization," in *Proceedings of the European Conference on Computer Vision*, 2016.

[122] A. Perotti, S. Bertolotto, E. Pastor, and A. Panisson, "Beyond one-hot-encoding: Injecting semantics to drive image classifiers," in *Proceedings of the World Conference on Explainable Artificial Intelligence*, 2023.

[123] J. Valmadre, "Hierarchical classification at multiple operating points," in *Advances in Neural Information Processing Systems*, 2022.

[124] C. Wu, M. Tygert, and Y. LeCun, "A hierarchical loss and its problems when classifying non-hierarchically," *PLoS ONE*, vol. 14, no. 12, e0226222, 2019.

[125] C.-A. Brust and J. Denzler, "Integrating domain knowledge: Using hierarchies to improve deep classifiers," in *Proceedings of the Asian Conference on Pattern Recognition*, 2019.

[126] A. Vaswani, Y. Samaga, G. Aggarwal, P. Netrapalli, and N. Hegde, "All mistakes are not equal: Comprehensive hierarchy aware multilabel predictions (champ)," in *Proceedings of the International Conference on Pattern Recognition*, 2024.

[127] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017.

[128] J. Davis, T. Liang, J. Enouen, and R. Ilin, "Hierarchical semantic labeling with adaptive confidence," in *Proceedings of the International Symposium on Visual Computing*, 2019.

[129] J. Davis, T. Liang, J. Enouen, and R. Ilin, "Hierarchical classification with confidence using generalized logits," in *Proceedings of the International Conference on Pattern Recognition*, 2021.

[130] Y. Guo, Y. Liu, E. M. Bakker, Y. Guo, and M. S. Lew, "CNN-RNN: A large-scale hierarchical image classification framework," *Multimedia Tools and Applications*, vol. 77, no. 8, pp. 10 251–10 271, 2018.

[131] X. Zhu and M. Bain, "B-CNN: Branch convolutional neural network for hierarchical classification," *arXiv preprint arXiv:1709.09890*, 2017.

[132] A. Bilal, A. Jourabloo, M. Ye, X. Liu, and L. Ren, "Do convolutional neural networks learn class hierarchy?" *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 1, pp. 152–162, 2017.

[133] M. Elhamod, K. M. Diamond, A. M. Maga, *et al.*, "Hierarchy-guided neural network for species classification," *Methods in Ecology and Evolution*, vol. 13, no. 3, pp. 642–652, 2022.

[134] J. Wehrmann, R. Cerri, and R. Barros, "Hierarchical multi-label classification networks," in *Proceedings of the International Conference on Machine Learning*, 2018.

[135] S. A. Memon, K. A. Khan, and H. Naveed, "HECNet: A hierarchical approach to enzyme function classification using a siamese triplet network," *Bioinformatics*, vol. 36, no. 17, pp. 4583–4589, 2020.

[136] M. Kulmanov, M. A. Khan, and R. Hoehndorf, "DeepGO: Predicting protein functions from sequence and interactions using a deep ontology-aware classifier," *Bioinformatics*, vol. 34, no. 4, pp. 660–668, 2018.

[137] D. Chang, K. Pang, Y. Zheng, Z. Ma, Y.-Z. Song, and J. Guo, "Your" flamingo" is my" bird": Fine-grained, or not," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

[138] F. Taherkhani, H. Kazemi, A. Dabouei, J. Dawson, and N. M. Nasrabadi, "A weakly supervised fine label classifier enhanced by coarse supervision," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019.

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

[140] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing*, MIT Press, 1986, pp. 318–362.

[141] A. Dhall, A. Makarova, O. Ganea, D. Pavllo, M. Greeff, and A. Krause, "Hierarchical image classification using entailment cone embeddings," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020.

[142] T. Long, P. Mettes, H. T. Shen, and C. G. Snoek, "Searching for actions on the hyperbole," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[143] B. Barz and J. Denzler, "Hierarchy-based image embeddings for semantic image retrieval," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2019.

[144] Z. Yu, T. Nguyen, Y. Gal, *et al.*, "Skin lesion recognition with class-hierarchy regularized hyperbolic embeddings," in *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention*, 2022.

[145] S. Zhang, R. Xu, C. Xiong, and C. Ramaiah, "Use all the labels: A hierarchical multi-label contrastive learning framework," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

[146] B. Chen, X. Huang, L. Xiao, Z. Cai, and L. Jing, "Hyperbolic interaction model for hierarchical multi-label classification," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[147] P. Mettes, M. Atigh, M. Keller-Ressel, J. Gu, and S. Yeung, "Hyperbolic deep learning in computer vision: A survey," *International Journal of Computer Vision*, vol. 132, pp. 1–25, 2024.

[148] M. Nickel and D. Kiela, "Poincaré embeddings for learning hierarchical representations," in *Advances in Neural Information Processing Systems*, 2017.

[149] E. Cole, K. Wilber, G. Van Horn, *et al.*, "On label granularity and object localization," in *Proceedings of the European Conference on Computer Vision*, 2022.

[150] A. Garg, S. Bagga, Y. Singh, and S. Anand, "Hiermatch: Leveraging label hierarchies for improving semi-supervised learning," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022.

[151] J. Wu, H. Yang, T. Gan, N. Ding, F. Jiang, and L. Nie, "Chmatch: Contrastive hierarchical matching and robust adaptive threshold boosted semi-supervised learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

[152] J.-C. Su and S. Maji, "Semi-supervised learning with taxonomic labels," in *Proceedings of the British Machine Vision Conference*, 2021.

[153] J. Deng, A. C. Berg, K. Li, and L. Fei-Fei, "What does classifying more than 10,000 image categories tell us?" In *Proceedings of the European Conference on Computer Vision*, 2010.

[154] S. Kiritchenko, S. Matwin, A. F. Famili, *et al.*, "Functional annotation of genes using hierarchical text categorization," in *Proceedings of the ACL Workshop on Linking Biological Literature, Ontologies and Databases: Mining Biological Semantics*, 2005.

[155] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2015.

[156] M. Sensoy, L. Kaplan, and M. Kandemir, "Evidential deep learning to quantify classification uncertainty," in *Advances in Neural Information Processing Systems*, 2018.

[157] H. Touvron, T. Lavril, G. Izacard, *et al.*, "Llama: Open and efficient foundation language models," *arXiv preprint arXiv:2302.13971*, 2023.

[158] A. Liu, B. Feng, B. Xue, *et al.*, "Deepseek-v3 technical report," *arXiv preprint arXiv:2412.19437*, 2024.

[159] A. Radford, J. W. Kim, C. Hallacy, *et al.*, "Learning transferable visual models from natural language supervision," in *Proceedings of the International Conference on Machine Learning*, 2021.

[160] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.