MSc Artificial Intelligence
Master Thesis

On the Calibration of Learning to Defer Systems

by
Rajeev Verma
13250507

July 27, 2022

48 ECTS
November 2021 - July 2022

Supervisor:
Dr. Eric Nalisnick

Examiner:
Dr. Eric Nalisnick

Second reader:
Dharmesh Tailor

Universiteit van Amsterdam

To my 'super' parents

Abstract

With Artificial Intelligence (AI) systems getting deployed in consequential decision-making applications, it's safer to ask for human advice in these situations when the AI system is at the risk of providing the wrong decision. The *learning to defer* (L2D) framework has the potential to make AI systems safer. For a given input, the system can defer the decision to a human if the human is more likely than the model to take the correct action. We study the confidence calibration of L2D systems, investigating if the probabilities they output are sound. We find that Mozannar and Sontag's (2020) multiclass framework is not confidence calibrated with respect to expert correctness. Moreover, it is not even guaranteed to produce valid probabilities due to its parameterization being degenerate for this purpose. We propose an L2D system based on one-vs-all classifiers that is able to produce calibrated probabilities of expert correctness. Furthermore, our loss function is also a consistent surrogate for multiclass L2D, like Mozannar and Sontag's (2020). Our experiments verify that not only is our system calibrated, but this benefit comes at no cost to accuracy. Our model's accuracy is always comparable (and often superior) to Mozannar and Sontag's (2020) model's in tasks ranging from hate speech detection to galaxy classification to diagnosis of skin lesions. We further provide consistent learning algorithms for learning to defer in the case of multiple experts and study their confidence-calibration properties. Our experiments demonstrate that our proposed system provides well-calibrated confidence even in the case of multiple experts.

Keywords: AI-Safety, Human-AI Collaboration, Uncertainty Quantification, Confidence Calibration, Decision-making, Probabilistic Machine Learning, Learning Theory.

Acknowledgements

This thesis was a long haul, and I won't defer my responsibility to thank the people behind this. Firstly, I would like to thank my supervisor, Eric Nalisnick, for introducing me to this exciting problem and always encouraging my ideas. I would also like to thank Dharmesh Tailor for agreeing to be the second reader for my thesis. I am also thankful to Hussein Mozannar for taking the time to meet and discuss this work with me. Also, thanks to Daniel Barrejón for helpful comments on the preliminary draft for this thesis and for the code to plot nice-looking figures.

Additionally, I would like to thank my mom, who had a much better and more realistic plan for me to finish this thesis. To my disadvantage, I ignored it. But she kept on rallying, doing maths on how many more days and hours I'd take.

I also take the opportunity to thank my friend Shuai, who turned out to be my personal cheerleader all this time. And thanks to Paras for being a great 'ranting' buddy.

Lastly, if you are reading this thesis, thanks a lot!

Contents

List of Figures

List of Tables

1 Introduction

—I have a story for you, he said. A story about my own encounter with the king of the winds.

— Amor Towles, The Lincoln Highway

In recent times, machine learning systems are being deployed in ever more consequential and high-stakes tasks such as healthcare (Zoabi et al., 2021; Kadampur and Al Riyaee, 2020), criminal justice (Zhong et al., 2018; Chalkidis et al., 2019), and autonomous driving (Grigorescu et al., 2020). In these high-stake applications, the cost associated with misclassification is very high. Thus, the trust and safety of these systems are paramount (Hendrycks and Dietterich, 2019; Nguyen et al., 2015). One near-term solution is ensuring a human is involved in decision-making. Sensitive and safety-critical applications like healthcare and self-driving cars would help if these systems could abstain from predicting so that expert humans can intervene. For example, *learning with a rejection option* (Chow, 1957) allows the model to abstain from making a decision, instead passing the burden to a human. The decision to abstain or not is usually derived from the model's confidence. A winding stretch of road for a self-driving car could make the system unconfident in its abilities. The system would then refuse to drive and forces the human to take control. When the system becomes confident again (e.g., on a straight road), it can take back control from the human. This prevents the system from making the wrong predictions and results in fatal situations.

Learning to defer (L2D) (Madras et al., 2018) is another framework that supports machine-human collaboration. In L2D, the human's confidence is modeled and the machine's. This allows the system to compare the human's and model's expected performances. Thus, L2D systems defer when *the human is more likely than the model to take the correct action*. Returning to the example of a self-driving car, and L2D system would pass control to the human only when it expects the human to drive better than itself. In addition to safety, such behavior allows for an efficient *division of labor* between the human and machine. By knowing what the human knows, the model is free to adapt itself to complement the human. The model can concentrate on performing easy tasks well if it knows a human can be relied upon for harder tasks.

Most previous work has attempted to improve the overall accuracy of L2D systems. However, if these systems are to be used in safety-critical scenarios, other factors such as trust, transparency, and fairness are important as well (Madras et al., 2018). Furthermore, to enable successful intervention from human experts in critical cases, these models must predict *responsibly*. This means they should be faithful to the expert humans. A machine learning model should accurately reflect its uncertainty and the risk associated with decision-making based on its prediction. Tschandl et al. (2020) found that AI systems can mislead physicians into incorrect diagnoses, even when the doctor is initially confident. To help prevent such scenarios, we want our systems to be well *calibrated*. The output probabilities should reflect the true uncertainties of the model and human. In other words, the L2D system should be a good forecaster.

If the system says the expert has a 70% chance of being correct, then the expert should be correct in about 70 out of 100 cases. If an L2D system is being used for medical diagnosis, a doctor will want to inspect the system's probabilities, at least for sanity checking. In the event of a misdiagnosis, these probabilities will also be useful for finding the source of bugs or other problems. Madras et al. (2018) emphasize the importance of responsibility and fairness for L2D frameworks.

In this thesis, we study the calibration of L2D systems. We focus on Mozannar and Sontag's (2020) formulation since it is the only *consistent* surrogate loss for multiclass L2D. Consistency is an important criterion in machine learning theory for the success of a classification algorithm optimizing the surrogate loss instead of the target loss (which may be difficult to computationally optimize). It means that the minimizer of the surrogate loss would correspond to the Bayes optimal solution to the learning problem. Mozannar and Sontag's (2020) proposed the first surrogate loss for L2D by using an augmented label space and novel reduction to cost-sensitive learning that resembles the cross-entropy loss for a softmax parameterization. They prove that the minimizers of their surrogate loss function over all measurable functions agree with the Bayes optimal rule for L2D.

We find that the Mozannar and Sontag (2020) loss results in models that are not well-calibrated with respect to expert correctness. The problem is intrinsic: the softmax parameterization allows the estimator to be *greater than one*. This results due to the *fragility* of the Bayes rule for L2D, where the calibrated confidence estimates are a sufficient condition but not the necessary one. Thus, the problem suffers from *identifiability* issues and allows for many solutions that do not correspond to valid confidence estimates. We propose an alternative loss based on one-vs-all classifiers that do not have this issue. We use the method of *error correcting output codes* (Ramaswamy et al., 2018) to show the multiclass L2D problem reduces to multiple binary classification problems. In turn, our one-vs-all surrogate is a consistent loss function, thus making it a superior alternative to Mozannar and Sontag's (2020) loss. In experiments ranging from hate speech detection to galaxy classification to diagnosis of skin lesions, our model always performs comparably, if not better than, the Mozannar and Sontag (2020) formulation in addition to other L2D frameworks (e.g., Okati et al. (2021)) and common baselines (e.g., confidence thresholds).

We also extend the consistent surrogate methods for L2D to consider multiple experts, designing consistent surrogates for learning to defer with multiple experts for the first time. We further examine the confidence calibration properties for both the surrogate methods finding that Mozannar and Sontag's (2020) formulation exhibits mis-calibration error, which further increases egregiously with increasing the number of experts. Furthermore, their method allows mis-calibration to propagate between the estimates of expert correctness confidence. Our proposed loss function is stable for multi-experts settings and doesn't show similar properties.

1.1 Organization

The thesis further consists of seven chapters and an appendix after that. In Chapter 2 (Preliminaries), we introduce the general theory of solving

classification problems with surrogate losses along with necessary results on their calibration and consistency with respect to (w.r.t) the target loss. In Chapter 3 (Learning to Defer), we theoretically formalize the problem of learning to defer and derive its Bayes optimal rule. We further study the only consistent surrogate loss proposed in the literature for learning to defer and investigate the confidence calibration property of the confidence estimates returned by a classification algorithm minimizing this surrogate loss. This chapter states the main result of this thesis w.r.t the confidence calibration property. In Chapter 4 (One-vs-All Surrogate Loss for Learning to Defer), we state the main contribution of this thesis and prove our main result. We empirically verify the properties of our proposed method in Chapter 5 (Experiments). In Chapter 6 (Calibration of Learning to Defer to Multiple Experts), we extend the learning to defer framework to allow multiple experts and study the confidence calibration property in this setting. We survey related works in Chapter 7 (Related Work) and conclude this thesis with discussion, reflections, and directions for future work in Chapter 8 (Discussion). We provide proof of all the necessary statements of this thesis in Chapter A (Appendix).

Preliminaries 2

He began at the beginning—the very beginning—by opening to the endpapers. And it was a good thing he had.

— Amor Towles, The Lincoln Highway

In this chapter, we formalize a classification problem in machine learning in its general setting and state its goals. We assume that the reader is comfortable with probability theory and slightly familiar with measure theory (the notion of measurable functions).

2.1 A General Classification Problem and Surrogate Losses

Let $\mathcal{X} \subseteq \mathbb{R}^d$ denotes an input space, and $\mathcal{Y}$ denotes an output label space which we will always assume to be a categorical encoding of multiple K classes, i.e. $\mathcal{Y} = \{1, 2, \ldots, \mathrm{K}\} := [\mathrm{K}]$. We also assume an unknown distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$, and let $\mathbf{x}$ and y be random variables over $\mathcal{D}$. And let $\boldsymbol{x}$ and y denote the realizations of random variables $\mathbf{x}$ and y respectively. Furthermore, let $\hat{\mathcal{Y}} = [\hat{\mathrm{K}}]$ denotes an output prediction label space, which again is assumed to be categorical encoding of $\hat{K}$ classes[1]. A classification problem in machine learning aims to learn a mapping $h : \mathcal{X} \to \hat{\mathcal{Y}}$. We call h a prediction function, and we evaluate the performance of such a prediction function via a loss function $\ell : \mathcal{Y} \times \hat{\mathcal{Y}} \to \mathbb{R}_+$.[2] The goal of classification problem is then to find a prediction function h that minimizes the ℓ-*risk* which is defined as

$$\mathcal{R}^{\ell}_{\mathcal{D}}[h] = \mathbb{E}_{\boldsymbol{x},y\sim\mathcal{D}}\left[\ell\left(y, h\left(\boldsymbol{x}\right)\right)\right]. \tag{2.1}$$

We also define a minimal ℓ-risk, referred to as the *Bayes* ℓ-*risk* [3], denoted as $\mathcal{R}^{\ell,*}_{\mathcal{D}}$, as below

$$\mathcal{R}^{\ell,*}_{\mathcal{D}} = \inf_{h:\mathcal{X}\to\hat{\mathcal{Y}}} \mathcal{R}^{\ell}_{\mathcal{D}}[h].$$

In practical settings, we assume access to the finite sample $S = \{\boldsymbol{x}_i, y_i\}_{i=1}^{\mathrm{N}}$ drawn independently and identically distributed from $\mathcal{D}$, denoted as $\mathcal{D}^{\mathrm{N}}$. We also fix some hypothesis class $\mathcal{H}$, and the classification algorithm aims to find $h_S \in \mathcal{H}$ by minimizing an empirical version of ℓ-*risk* $\hat{\mathcal{R}}^{\ell}_{\mathcal{D}}[h]$, defined formally as

$$\hat{\mathcal{R}}^{\ell}_{\mathcal{D}}[h] = \frac{1}{\mathrm{N}}\sum_{i=1}^{\mathrm{N}} \ell\left(y_i, h\left(\boldsymbol{x}_i\right)\right), \quad (\boldsymbol{x}_i, y_i) \sim \mathcal{D}^{\mathrm{N}}. \tag{2.2}$$

An important notion of success for such a classification learning algorithm is the convergence of $\mathcal{R}^{\ell}_{\mathcal{D}}[h_S] \to \mathcal{R}^{\ell,*}_{\mathcal{D}}$, i.e. when the learning algorithm receives increasingly large sample $S \sim \mathcal{D}^{\mathrm{N}}$, the ℓ-*risk* of the function h_S returned by the learning algorithm converges in probability to the *Bayes*

1: Note that K and $\hat{\mathrm{K}}$ can be different.

2: Example of one such loss function is $\ell_{0-1} : (y, \hat{y}) \mapsto \mathbb{I}[y \neq \hat{y}]$, where $\mathbb{I}$ is an indicator function. ℓ_{0-1} is called $0-1$ loss and will be particularly interesting in this thesis.

3: *Bayes* ℓ-*risk* is the minimal ℓ-*risk* one can hope to achieve, and the classification problem aims to find h with *Bayes* ℓ-*risk*.

ℓ-*risk*, written formally as

$$\forall \epsilon > 0 \; \mathbb{P}_{S \sim \mathcal{D}^{\mathrm{N}}} \left(\mathcal{R}^{\ell}_{\mathcal{D}}[h_S] > \mathcal{R}^{\ell,*}_{\mathcal{D}} + \epsilon \right) \to 0 \text{ as } \mathrm{N} \to \infty. \tag{2.3}$$

However, minimizing the ℓ-*risk* (similarly, empirical ℓ-*risk*) is computationally difficult for some classes of loss functions. For instance, for ℓ_{0-1}, computationally minimizing $\ell - risk$ is NP-hard. Thus, as an alternative to ℓ, we employ a surrogate loss function $\psi : \mathcal{Y} \times \mathcal{V} \to \mathbb{R}_+$ over a surrogate prediction space $\mathcal{V} \subseteq \mathbb{R}^{\hat{\mathrm{K}}}$ and minimize a ψ-*risk* instead.

More formally, for a surrogate prediction space $\mathcal{V} \subseteq \mathbb{R}^{\hat{\mathrm{K}}}$, a surrogate loss $\psi : \mathcal{Y} \times \mathcal{V} \to \mathbb{R}_+$, the goal is to learn a function $f : \mathcal{X} \to \mathcal{V}$ over some class of functions $\mathcal{F}$ and a suitable decoding function $g : \mathcal{V} \to \hat{\mathcal{Y}}$. [4] We then have the usual notions of $\mathcal{R}^{\psi}_{\mathcal{D}}[f]$ and $\mathcal{R}^{\psi,*}_{\mathcal{D}}$.

4: Note that $g \circ f : \mathcal{X} \to \hat{\mathcal{Y}}$. And a common example of such a decoding function in machine learning is the arg max function.

An important question in such a setting is whether the convergence $\mathcal{R}^{\psi}_{\mathcal{D}}[f_S] \to \mathcal{R}^{\psi,*}_{\mathcal{D}}$ implies the convergence $\mathcal{R}^{\ell}_{\mathcal{D}}[g \circ f_S] \to \mathcal{R}^{\ell,*}_{\mathcal{D}}$?

A positive answer to this question is necessary for the success of the classification problem learned by minimizing a surrogate loss ψ, and it is formally known as the *consistency* of the surrogate loss ψ with respect to (w.r.t) the target loss ℓ. We formalize it in Definition 2.1.1.

Definition 2.1.1 *($\mathcal{F}$-Consistency). A surrogate loss function ψ is said to be $\mathcal{F}$-consistent with respect to the loss function ℓ if for any sequence of functions $f_n \in \mathcal{F}$,*

$$\mathcal{R}^{\psi}_{\mathcal{D}}[f_n] \to \mathcal{R}^{\psi,*}_{\mathcal{D}} \implies \mathcal{R}^{\ell}_{\mathcal{D}}[g \circ f_n] \to \mathcal{R}^{\ell,*}_{\mathcal{D}} \tag{2.4}$$

for all distributions $\mathcal{D}$.

Intuitively, if a surrogate loss ψ is consistent w.r.t. the target loss ℓ, we can solve the classification problem by minimizing $\mathcal{R}^{\psi}_{\mathcal{D}}$ instead of $\mathcal{R}^{\ell}_{\mathcal{D}}$. However, in practice, verifying if some surrogate loss is consistent w.r.t. some target loss is difficult. To make this analysis easier, we use the regular notion of conditional probability, and denote $\eta_y = \mathbb{P}(\mathrm{y} = y | \mathbf{x} = \boldsymbol{x})$. We can now calculate the ℓ-*risk* for some loss function ℓ using an iterated integral, as below

$$\mathcal{R}^{\ell}_{\mathcal{D}}[h] = \mathbb{E}_{\boldsymbol{x} \sim \mathbf{x}} \left[\sum_{y=1}^{K} \eta_y(\boldsymbol{x}) \, \ell(y, h(\boldsymbol{x})) \right] = \mathbb{E}_{\boldsymbol{x} \sim \mathbf{x}} \left[\boldsymbol{\eta}(\boldsymbol{x})^{\mathrm{T}} \cdot \boldsymbol{\ell}(h(\boldsymbol{x})) \right], \tag{2.5}$$

where

$$\boldsymbol{\eta}(\boldsymbol{x}) = [\eta_1(\boldsymbol{x}), \eta_2(\boldsymbol{x}), \ldots, \eta_K(\boldsymbol{x})]^{\mathrm{T}}, \text{ and}$$
$$\boldsymbol{\ell}(h(\boldsymbol{x})) = [\ell(\mathrm{y} = 1, h(\boldsymbol{x})), \ell(\mathrm{y} = 2, h(\boldsymbol{x})), \ldots, \ell(\mathrm{y} = \mathrm{K}, h(\boldsymbol{x}))]^{\mathrm{T}}.$$

The quantity $\boldsymbol{\eta}(\boldsymbol{x})^{\mathrm{T}} \cdot \boldsymbol{\ell}(h(\boldsymbol{x}))$ is of crucial importance to us that it has a special name. We call it *inner* ℓ-*risk*, and denote it as $\mathcal{C}^{\ell}_{\boldsymbol{\eta}(\boldsymbol{x}), \boldsymbol{x}}[h]$. We formalize it in Definition 2.1.2.

Definition 2.1.2 *(Inner ℓ-risk and Bayes inner ℓ-risk). For a given loss function ℓ and a prediction function $h \in \mathscr{H}$, and $\forall \boldsymbol{\eta} \in [0,1]^K$, and $\forall \boldsymbol{x} \in \mathscr{X}$, the quantity*

$$\mathscr{C}^{\ell}_{\boldsymbol{\eta}(x),\boldsymbol{x}}[h] := \boldsymbol{\eta}(\boldsymbol{x})^T \cdot \boldsymbol{\ell}(h(\boldsymbol{x}))$$

is known as the inner ℓ-risk. Similarly, we have minimal inner ℓ-risk,

$$\mathscr{C}^{\ell,*}_{\boldsymbol{\eta}(x),\boldsymbol{x}} := \inf_{h \in \mathscr{H}} \mathscr{C}^{\ell}_{\boldsymbol{\eta}(x),\boldsymbol{x}}[h].$$

It's worth reiterating that $\mathscr{R}^{\ell}_{\mathscr{D}}[h] = \mathbb{E}_{x \in \mathbf{x}}[\mathscr{C}^{\ell}_{\boldsymbol{\eta}(x),\boldsymbol{x}}[h]]$. The next question that will make our analysis of studying if some surrogate loss is consistent w.r.t. to some target loss is under what conditions it also holds true that $\mathscr{R}^{\ell,*}_{\mathscr{D}} = \mathbb{E}_{x \in \mathbf{x}}[\mathscr{C}^{\ell,*}_{\boldsymbol{\eta}(x),\boldsymbol{x}}]$? [5] An additional assumption we need for it to hold is that of *minimizability*, as defined below in Definition 2.1.3.

5: Note that it's not necessarily true in all cases.

Definition 2.1.3 *(Minimizable loss function). Given a distribution $\mathscr{D}$ over $\mathscr{X} \times \mathscr{Y}$, a loss function ℓ, and a hypothesis class $\mathscr{H}$, we call ℓ $\mathscr{D}$-minimizable if for all $\epsilon > 0$, $\exists h_\epsilon \in \mathscr{H}$ such that for all $\boldsymbol{x} \in \mathscr{X}$, we have*

$$\mathscr{C}^{\ell}_{\boldsymbol{\eta}(x),\boldsymbol{x}}[h_\epsilon] < \mathscr{C}^{\ell,*}_{\boldsymbol{\eta}(x),\boldsymbol{x}} + \epsilon,$$

where $\boldsymbol{\eta}(\boldsymbol{x})$ is defined as before.

For a minimizable loss function ℓ, one can prove that $\boldsymbol{x} \mapsto \mathscr{C}^{\ell,*}_{\boldsymbol{\eta}(x),\boldsymbol{x}}$ is a *measurable* function, and it would hold true that $\mathscr{R}^{\ell,*}_{\mathscr{D}} = \mathbb{E}_{x \in \mathbf{x}}[\mathscr{C}^{\ell,*}_{\boldsymbol{\eta}(x),\boldsymbol{x}}]$. [6] When the loss function ℓ is $\mathscr{D} - minimizable$ for some hypothesis class $\mathscr{H}$, we call it *minimizability*. It is easy to see that *minimizability* trivially holds for all the hypothesis class $\mathscr{H}$ that are *realizable* (i.e. the Bayes optimal function h^* belongs to $\mathscr{H}$). [7] Minimizable loss functions are convenient for our classification problem as they enable *pointwise* minimization of $\mathscr{C}^{\ell}_{\boldsymbol{\eta}(x),\boldsymbol{x}}$ for each $\boldsymbol{x} \in \mathscr{X}$, in order to achieve $\mathscr{R}^{\ell,*}_{\mathscr{D}}$. They also give us a powerful tool to establish the consistency of a surrogate loss w.r.t. the target loss. This tool is called the *calibration* of a surrogate loss w.r.t. the target loss, and it is formally defined in Definition 2.1.4.

6: For a complete proof, we refer the reader to Steinwart (2007).

7: Note that *realizability* trivially holds when $\mathscr{H}$ is the hypothesis class of all *measurable* functions.

Definition 2.1.4 *($\mathscr{F}$-Calibration). For some surrogate prediction space $\mathscr{V} \subseteq \mathbb{R}^{\hat{K}}$ and a decoding function $g : \mathscr{V} \to \hat{\mathscr{Y}}$, a surrogate loss function $\psi : \mathscr{Y} \times \mathscr{V} \to \mathbb{R}_+$ is said to be $\mathscr{F}$-calibrated with respect to the loss function $\ell : \mathscr{Y} \times \hat{\mathscr{Y}} \to \mathbb{R}_+$ if, $\forall \epsilon > 0$, $\forall \boldsymbol{\eta} \in [0,1]^K$, $\forall \boldsymbol{x} \in \mathscr{X}$, $\exists \delta > 0$ such that for any function $f \in \mathscr{F}$*

$$\mathscr{C}^{\psi}_{\boldsymbol{\eta},\boldsymbol{x}}[f] < \mathscr{C}^{\psi,*}_{\boldsymbol{\eta},\boldsymbol{x}} + \delta \implies \mathscr{C}^{\ell}_{\boldsymbol{\eta},\boldsymbol{x}}[g \circ f] < \mathscr{C}^{\ell,*}_{\boldsymbol{\eta},\boldsymbol{x}} + \epsilon.$$

Intuitively, ψ is $\mathscr{F}$-calibrated w.r.t. ℓ if for any given threshold $\epsilon > 0$, there exists $\delta > 0$ such that every $f \in \mathscr{F}$ that approximates $\mathscr{C}^{\psi,*}_{\boldsymbol{\eta},\boldsymbol{x}}$ upto δ, also approximates $\mathscr{C}^{\ell,*}_{\boldsymbol{\eta},\boldsymbol{x}}$ upto the target level ϵ. In other words, $\mathscr{F}$-calibration guarantees that the convergence $\mathscr{C}^{\psi}_{\boldsymbol{\eta},\boldsymbol{x}}[f] \to \mathscr{C}^{\psi,*}_{\boldsymbol{\eta},\boldsymbol{x}}$ implies the desired convergence $\mathscr{C}^{\ell}_{\boldsymbol{\eta},\boldsymbol{x}}[g \circ f] \to \mathscr{C}^{\ell,*}_{\boldsymbol{\eta},\boldsymbol{x}}$ for each $\boldsymbol{x} \in \mathscr{X}$. Thus, it is analogous to the *consistency* property of a surrogate loss except for each $\boldsymbol{x} \in \mathscr{X}$. [8]

8: A surrogate loss ψ is said to be calibrated with respect to the target loss ℓ if *successfully* minimizing ψ results in a classifier f with suitable decoding function g whose *inner ℓ-risk* is close to the *Bayes inner ℓ-risk* for each $\boldsymbol{x} \in \mathscr{X}$.

$\mathcal{F}$-*Calibration* is a *necessary* condition for $\mathcal{F}$-*Consistency*. However, when we are working with *minimizable* loss functions, one can see that $\mathcal{F}$-*Calibration* also implies $\mathcal{F}$-*Consistency*. Thus, besides *pointwise* minimization of $\mathcal{C}^{\ell}_{\eta(x),x}$ to achieve $\mathcal{R}^{\ell,*}_{\mathcal{D}}$, *minimizable* loss functions also enable one to *pointwisely* verify the convergence $\mathcal{C}^{\ell}_{\eta,x}[g \circ f] \to \mathcal{C}^{\ell,*}_{\eta,x}$ in order to guarantee the desired convergence $\mathcal{R}^{\ell}_{\mathcal{D}}[g \circ f] \to \mathcal{R}^{\ell,*}_{\mathcal{D}}$. And the notion of $\mathcal{F}$-*Calibration* allows us to *pointwisely* establish the convergence $\mathcal{C}^{\ell}_{\eta,x}[g \circ f] \to \mathcal{C}^{\ell,*}_{\eta,x}$ when the classification problem is solved by employing some surrogate loss ψ. These ideas will be central to this thesis in the later chapters. In this next section, we look at Binary classification — a specific case of a general classification problem.

2.2 Binary Classification

We continue the notation and the setting from the last section. For binary classification, without loss of generality, we take $\mathcal{Y} = \{-1, 1\}$, and here we consider $\hat{\mathcal{Y}} = \{-1, 1\}$ as well (i.e. the output label space and the prediction label space are same). We aim to find a prediction function $h : \mathcal{X} \to \mathcal{Y}$, and we evaluate its performance via $\ell_{0-1} : \mathcal{Y} \times \hat{\mathcal{Y}} \to \{0, 1\}$. In this setting, one can write down the prediction function $h^*_{\mathcal{D}}$ that achieves the *Bayes* ℓ-*risk*, and we state it formally in Lemma 2.2.1.

Lemma 2.2.1 *For a probability distribution $\mathcal{D}$ over $\mathcal{X} \times \{-1, 1\}$, and a function $h^*_{\mathcal{D}} : \mathcal{X} \to \{-1, 1\}$ defined as*

$$h^*_{\mathcal{D}}(\boldsymbol{x}) = \begin{cases} 1, & \text{if } \mathbb{P}(\mathrm{y} = 1 | \mathbf{x} = \boldsymbol{x}) \geq \frac{1}{2} \\ -1, & \text{otherwise}, \end{cases}$$

then $h^*_{\mathcal{D}}$ attains the Bayes ℓ-risk, i.e. for any other prediction function $h : \mathcal{X} \to \{-1, 1\}$, it holds that $\mathcal{R}^{\ell_{0-1}}_{\mathcal{D}}[h^*_{\mathcal{D}}] \leq \mathcal{R}^{\ell_{0-1}}_{\mathcal{D}}[h]$.

For the proof of lemma 2.2.1, see Appendix A.1.1. We call $h^*_{\mathcal{D}}$ as the *Bayes optimal binary classifier*. Unfortunately, we assume $\mathcal{D}$ is unknown (and thus $\mathbb{P}(\mathrm{y} = 1 | \mathbf{x} = \boldsymbol{x})$ is also unknown). Thus, we cannot use this to solve the binary classification problem. So, we take a machine learning perspective and aim to learn a prediction function h that closely approximates $h^*_{\mathcal{D}}$. As for the reasons mentioned in the last section, we employ a surrogate loss $\psi : \{-1, 1\} \times \mathcal{V} \to \mathbb{R}_+$ on some surrogate prediction space $\mathcal{V} \subseteq \mathbb{R}$. The calibration of binary surrogate losses w.r.t. ℓ_{0-1} is well-studied in the literature, and we state one important result in Definition 2.2.1.

Definition 2.2.1 *(Bartlett et al. (2006)). For a surrogate prediction space $\mathcal{V} \subseteq \mathbb{R}$ and a prediction function $f : \mathcal{X} \to \mathcal{V}$, we say a binary classification surrogate loss $\psi : \{-1, 1\} \times \mathcal{V} \to \mathbb{R}_+$ is classification-calibrated if, for any $\eta \neq \frac{1}{2}$, we have*

$$\inf_{f(x) \cdot \left(\eta(x) - \frac{1}{2}\right) \leq 0} \mathcal{C}^{\psi}_{\eta(x),x}[f] > \inf_{f(x)} \mathcal{C}^{\psi}_{\eta(x),x}[f], \tag{2.6}$$

where $\eta(x) := \mathbb{P}(\mathrm{y} = 1|\mathbf{x} = \boldsymbol{x})$.

This definition simply states that for a function f^* returned by a classification algorithm that minimizes $\mathscr{C}^{\psi}_{\eta(x),\boldsymbol{x}}$ for a surrogate loss function ψ *calibrated* w.r.t. ℓ_{0-1}, $\operatorname{sign}(f^*(\boldsymbol{x}))$ corresponds to $\operatorname{sign}\left(\eta(\boldsymbol{x}) - \frac{1}{2}\right)$ where $\operatorname{sign} : \mathscr{V} \to \{-1, 1\}$, [9] i.e. the decoding function g in this case is the sign function. In other words, minimizing $\mathscr{C}^{\psi}_{\eta(\boldsymbol{x}),\boldsymbol{x}}$ over the surrogate space $\mathscr{C}$ gives the Bayes optimal binary classifier.

9: The sign function is defined as:

$$\operatorname{sign}(\boldsymbol{x}) = \begin{cases} 1, & \text{if } \boldsymbol{x} \geq 0 \\ -1, & \text{if } \boldsymbol{x} < 0 \end{cases}$$

The above result should be unsurprising given how we defined the *calibration* of a surrogate loss w.r.t. some target loss in the previous section. However, we stated it explicitly as this result will be central to the main contribution of this thesis. In the next section, we discuss a special class of binary surrogate losses, called class probability estimation losses.

2.3 Binary Proper Losses and Proper Composite Surrogate Losses

In last section, we considered the surrogate prediction space $\mathscr{V} \subseteq \mathbb{R}$. In this section, we first consider a special class of loss functions, called binary class probability estimation (CPE) losses, where $\mathscr{V} = [0, 1]$. We continue to follow the notation from previous sections. However, to stress upon this specific $\mathscr{V}$, we work directly in the surrogate space and drop dependencies on $\boldsymbol{x}$, and the prediction function f in our notation. Instead of using $\mathscr{C}^{\ell}_{\eta(\boldsymbol{x}),\boldsymbol{x}}[f] = \eta(\boldsymbol{x})\,\ell(\mathrm{y} = 1, f(\boldsymbol{x})) + (1 - \eta(\boldsymbol{x}))\,\ell(\mathrm{y} = -1, f(\boldsymbol{x}))$, we simplify the notation and rewrite it as $\mathscr{C}^{\ell}_{\eta(\boldsymbol{x}),\boldsymbol{x}}[f] = \mathscr{C}^{\ell}_{\eta(\boldsymbol{x})}(\hat{p})$, where $\hat{p} = f(\boldsymbol{x})$. We further simplify it and replace $\eta(\boldsymbol{x})$ by $p \in [0, 1]$ and denote the *inner* ℓ*-risk* as $\mathscr{C}^{\ell}_{p}(v)$ to make it more general. With this notation, we define Proper loss in Definition 2.3.1.

Definition 2.3.1 *(Binary Proper Loss function). A binary class probability estimation (CPE) loss function* $\ell : \{-1, 1\} \times [0, 1] \to \mathbb{R}_+$*, is called a proper loss if*

$$\mathscr{C}^{\ell}_{p}(\hat{p}) \geq \mathscr{C}^{\ell}_{p}(p), \ \forall p, \hat{p} \in [0, 1], \hat{p} \neq p. \tag{2.7}$$

We call ℓ *a strictly proper loss function when the above inequality is strict.*

Intuitively, the above definition means that a proper loss function enables class probability estimation. This means that the minimizer of $\mathscr{C}^{\ell}_{p}(\hat{p})$ recovers true probability. More formally,

$$\forall p, \hat{p} \in [0, 1] \ \ p \in \operatorname*{arg\,min}_{\hat{p} \in [0,1]} \mathscr{C}^{\ell}_{p}(\hat{p}).$$

And when the above minimizer is unique $\forall p \in [0, 1]$, we call ℓ a strictly proper loss function. Thus, for binary classification, if we employ a strictly proper loss function in the surrogate prediction space $\mathscr{V} = [0, 1]$, and minimize $\mathscr{C}^{\ell}_{\eta(\boldsymbol{x})}(f(\boldsymbol{x}))$, the prediction function f^* that we get estimates class probability $\eta(\boldsymbol{x})$ for each $\boldsymbol{x} \in \mathscr{X}$, i.e. $\eta(\boldsymbol{x}) = f^*(\boldsymbol{x})$.

Can we extend this idea of class probability estimation to a general surrogate space $\mathcal{V} \subseteq \mathbb{R}$?

A positive answer to the above question is given by *binary proper composite loss functions*. This is done by employing a function called the *link function* $\gamma : [0,1] \to \mathcal{V}$, and composing a prediction function operating on the probability estimation surrogate space $\mathcal{V}_1 = [0,1]$ with this link function γ to get to the general surrogate space $\mathcal{V} \subseteq \mathbb{R}$. We define binary proper composite loss function more formally in Definition 2.3.2.

Definition 2.3.2 *(Binary proper composite loss function). Given a surrogate space $\mathcal{V} \subseteq \mathbb{R}$, a binary surrogate loss function $\psi : \{-1,1\} \times \mathcal{V} \to \mathbb{R}_+$, a proper loss function $\ell : \{-1,1\} \times [0,1] \to \mathbb{R}_+$, and a strictly increasing link function $\gamma : [0,1] \to \mathcal{V}$, we call ψ a binary proper composite surrogate loss function if*

$$\psi(y,v) = \ell\left(y, \gamma^{-1}(v)\right), \; v \in \mathcal{V}. \tag{2.8}$$

And we call ψ a strictly proper composite loss function when ℓ is a strictly proper loss function.

This now allows us to extend the ideas of *properness* to any surrogate prediction space. Specifically, if we learn a prediction function f^* by minimizing $\mathcal{C}_p^{\psi}(f(\boldsymbol{x}))$ where ψ is a binary proper composite surrogate loss function with an increasing *link* function γ, then by definition we have that

$$\forall p \in [0,1] \;\; \gamma(p) \in \arg\min_{f(\boldsymbol{x}) \in \mathcal{V}} \mathcal{C}_p^{\psi}(f(\boldsymbol{x})).$$

Similar to before, when the above minimizer is unique $\forall p \in [0,1]$, we call ψ a strictly proper composite loss function. Binary proper composite losses are widely studied in the literature (Reid and Williamson, 2009; Reid and Williamson, 2010; Buja et al., 2005). The focus of this thesis is strictly proper composite loss functions, as they are also *classification-calibrated* (Definition 2.2.1) surrogate functions for binary classification (Reid and Williamson, 2010). We give examples of some strictly proper composite surrogates in Table Section 2.3.

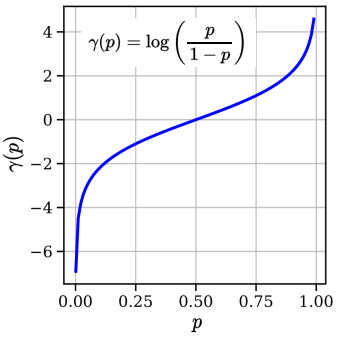


Figure 2.1: *Link function for the logistic loss.* We can see that it is a strictly increasing function of $p \in [0,1]$.

Name	ψ	$\mathcal{V}$	γ	γ^{-1}
Logistic	$\log(1+e^{-yv})$	$\mathbb{R}$	$\log\left(\frac{p}{1-p}\right)$	$\frac{1}{1+e^{-v}}$
Exponential	e^{-yv}	$\mathbb{R}$	$\frac{1}{2}\log\left(\frac{p}{1-p}\right)$	$\frac{1}{1+e^{-2v}}$
Squared	$(1-yv)^2$	$[-1,1]$	$2p-1$	$\frac{v+1}{2}$

Table 2.1: *Examples of few binary strictly proper composite loss functions.* $\mathcal{V}$ denotes surrogate prediction space, $\psi : \{-1,1\} \times \mathcal{V} \to \mathbb{R}_+$ is the surrogate loss function, $\gamma : [0,1] \to \mathbb{R}$ is the *link* function, and $\gamma^{-1} : \mathcal{V} \to [0,1]$ is the inverse link function.

We conclude this section by illustrating the binary classification problem using Logistic Loss in Example 2.3.1.

Example 2.3.1 For an input space $\mathcal{X}$, the output space $\mathcal{Y} = \{-1,1\}$ and the distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$, the goal in binary classification

is to learn a mapping $h : \mathcal{X} \to \mathcal{Y}$ with minimum ℓ_{0-1}-*risk* $\mathcal{R}_{\mathcal{D}}^{\ell_{0-1}}[h]$, where $\ell_{0-1} = \mathbb{I}[y = h(\boldsymbol{x})], (\boldsymbol{x}, y) \sim \mathcal{D}^{\mathrm{N}}$.

Obviously, minimizing $\mathcal{R}_{\mathcal{D}}^{\ell_{0-1}}[h]$ is computationally infeasible. So, we employ a logistic loss function as the surrogate loss function $\psi : \mathcal{Y} \times \mathcal{V} \to \mathbb{R}_+$, where $\psi : (y, v) \mapsto \log\left(1 + \mathrm{e}^{-yv}\right)$ over the surrogate prediction space $v \in \mathcal{V}$, $\mathcal{V} = \mathbb{R}$. We now minimize $\mathcal{R}_{\mathcal{D}}^{\psi}[f]$ instead to achieve $\mathcal{R}_{\mathcal{D}}^{\psi,*}$, and find a surrogate prediction function $f^* : \mathcal{X} \to \mathcal{V}$. Assuming *minimizability*, we can use the *pointwise* approach to minimize $\mathcal{C}_{\eta(x),\boldsymbol{x}}^{\psi}[f], \forall \boldsymbol{x} \in \mathcal{X}$ to this end.

Now, we know that logistic loss is a *classification-calibrated* loss function for ℓ_{0-1}, so we can deduce that f^* returned by such a classification algorithm together with a suitable decoding function $g : \mathcal{V} \to \{-1, 1\}$ (in this case, the sign function) correspond to the *Bayes optimal binary classifier*, i.e. we have

$$h(\boldsymbol{x}) = \operatorname{sign}\left(f^*(\boldsymbol{x})\right) = \operatorname{sign}\left(\eta(\boldsymbol{x}) - \frac{1}{2}\right), \quad \forall \boldsymbol{x} \in \mathcal{X}. \tag{2.9}$$

Furthermore, we know that logistic loss is also a strictly proper composite loss function with the inverse *link* function $\gamma^{-1} : \mathcal{V} \to [0, 1]$, $\gamma^{-1} : v \mapsto \frac{1}{1+\mathrm{e}^{-v}}$. Thus, we know that $\gamma^{-1}\left(f^*(\boldsymbol{x})\right)$ estimates the class probability $\eta(\boldsymbol{x}), \forall \boldsymbol{x} \in \mathcal{X}$. Combining everything, we can write down the final predictor as follows:

$$h(\boldsymbol{x}) = \operatorname{sign}\left(f^*(\boldsymbol{x})\right) = \operatorname{sign}\left(\gamma^{-1}\left(f^*(\boldsymbol{x})\right) - \frac{1}{2}\right), \quad \forall \boldsymbol{x} \in \mathcal{X}. \tag{2.10}$$

2.4 Code Based Surrogates for Multiclass Classification

This section reviews a general strategy to design surrogate loss functions. In particular, we study the output code-based methods where a coding matrix is used to decompose the multi-class classification problem into multiple binary classification problems. Mention could be made of error-correcting coding mechanism (Dietterich and Bakiri, 1995; Langford et al., 2005; Allwein et al., 2001). We briefly describe the setup here and refer the reader to Ramaswamy et al. (2014) for full details. The goal of such a code based mechanism is to use a code matrix $\mathbf{M} = \{-1, 1, 0\}^{\mathrm{K} \times \hat{\mathrm{K}}}$ to decompose a $\hat{\mathrm{K}}$-class classification problem into $\hat{\mathrm{K}}$ binary classification problems. Following the notation from Section 2.1, we use $\mathbf{M}$ to split the training sample $S = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{\mathrm{N}}$ into $\hat{\mathrm{K}}$-training samples $\tilde{S}_j$ for each $j \in [\hat{\mathrm{K}}]$ such that $\tilde{S}_j = \{(\boldsymbol{x}_i, \mathbf{M}_{y_i,j}) ; i \in [1, \mathrm{N}], \mathbf{M}_{y_i,j} \neq 0\}$. Thus, each $\tilde{S}_j$ is a subset from the original S with output labels replaced by the binary labels provided by $\mathbf{M}$. For $\mathcal{V} \subseteq \mathbb{R}$, we use these $\tilde{S}_j$ to learn a $\hat{\mathrm{K}}$-binary classifiers $f_j : \mathcal{X} \to \mathcal{V}$. Thus, for each $\boldsymbol{x} \in \mathcal{X}$, we get a prediction $f(\boldsymbol{x}) = \left[f_1(\boldsymbol{x}), \ldots, f_{\hat{\mathrm{K}}}(\boldsymbol{x})\right] \in \mathbb{R}^{\hat{\mathrm{K}}}$. We then a use a suitable decoding function to map $f(\boldsymbol{x})$ to the original prediction space $\hat{\mathcal{Y}}$. If we use some suitable surrogate loss $\ell : \{-1, 1\} \times \mathcal{V} \to \mathbb{R}_+$, then intuitively, the whole code matrix based mechanism can be viewed

as learning a function $f : \mathcal{X} \to \mathcal{V}^{\hat{K}}$ by minimizing a surrogate multiclass classification loss $\psi : \mathcal{Y} \times \mathcal{V}^{\hat{k}} \to \mathbb{R}_+$ given as

$$\psi(y, \mathbf{v}) = \sum_{j=1}^{\hat{K}} \left(\mathbb{I}\left[\mathbf{M}_{yj} = 1\right] \ell\left(1, v_j\right) + \mathbb{I}\left[\mathbf{M}_{yj} = -1\right] \ell\left(-1, v_j\right) \right). \quad (2.11)$$

Obviously, we care about the consistency of such a surrogate loss ψ for a successful classification algorithm. Ramaswamy et al. (2014) analyze the conditions related to consistency and calibration of such a surrogate loss for general losses. Next, we give a simple example to illustrate the procedure described above.

Example 2.4.1 Consider $\mathcal{Y} = \hat{\mathcal{Y}} = [K]$. Define $\mathbf{M} = \{-1, 1\}^{K \times K}$ such that $\mathbf{M}_{yj} = 1$ if $y = j$, otherwise $\mathbf{M}_{yj} = -1$. In words, $\mathbf{M}$ has 1 on diagonal entries and -1 everywhere else. For this $\mathbf{M}$, it's easy to see that

$$\phi(y, v) = \ell(1, v_y) + \sum_{j=1, j \neq y}^{K} \ell(-1, v_j).$$

Taking ℓ to be a binary logistic loss, we have:

$$\phi(y, v) = \log(1 + e^{-v_y}) + \sum_{j=1, j \neq y}^{K} \log(1 + e^{v_j}).$$

or,

$$\phi(y, v) = -\log\left(\frac{1}{1 + e^{-v_y}}\right) - \sum_{j=1, j \neq y}^{K} \log\left(\frac{1}{1 + e^{v_j}}\right).$$

Thus, we recovered a common one-vs-all formulation of the multi-class classification problem.

Summary of the Chapter

In this chapter, we study the general classification problem and the use of surrogate loss functions to solve it. We define the notions *consistency* and calibration of the surrogate loss with respect to the target loss. We then study a class of surrogate losses called proper composite losses, along with a general procedure to design surrogate losses.

Learning to Defer | 3

— Homer began his story in medias res, which means in the middle of the thing. To be in medias res, thought Billy, there should be just as many important things that have happened as important things that haven't happened yet.

— Amor Towles, The Lincoln Highway

In this chapter, we formalize Learning to Defer (L2D) and study it in light of the ideas we discussed in the last chapter. Our focus will be on the *classifier-rejector* (Cortes et al., 2016a; Cortes et al., 2016b) approaches to L2D. We will also state the *softmax* surrogate loss function (Mozannar and Sontag, 2020) for L2D, which is also the only *consistent* surrogate loss function for L2D. We continue using the notation from the last chapter.

Note on Contributions: This chapter is a retrospective formalization of learning to defer that draws ideas from the excellent works of Mozannar and Sontag (2020) and Chow (1957). However, there are some crucial differences as well. For example, Mozannar and Sontag (2020) argue the rationale for deferring only if the risk of the expert making the prediction is less than the risk of the classifier. Based on this motivation, they derive the Bayes optimal rejector and classifier. However, we consider the problem in a more general setting and theoretically prove that comparing the risk of the classifier and the expert provides an optimal rule for learning to defer (refer to Theorem 3.1.1 and Corollary 3.1.2).

3.1 Learning to Defer as a General Classification Problem

In L2D, besides the usual input space $\mathcal{X}$, output label space $\mathcal{Y}$, output prediction label space $\hat{\mathcal{Y}}$, and the distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$, we also have expert's label space $\mathcal{M}$. We always assume that $\mathcal{M} = \mathcal{Y}$. Let m denote the random variable for the expert's predictions, and we denote its realizations by m. We assume the expert's predictions are sampled from some expert distribution, and we make a simplifying assumption that $m \sim \mathbb{P}(\mathrm{m}|\mathbf{x}, \mathrm{y})$. And we assume access to the dataset $S = \{\boldsymbol{x}_i, y_i, m_i\}_{i=1}^{\mathrm{N}}$. In a general *classifier-rejector* approach to L2D, the goal is to learn two mappings: the *classifier*, $h : \mathcal{X} \to \hat{\mathcal{Y}}$, and the *rejector*, $r : \mathcal{X} \to \{0, 1\}$. When $r(\boldsymbol{x}) = 0$, the classifier makes the prediction in the typical way, i.e. $h(\boldsymbol{x})$. When $r(\boldsymbol{x}) = 1$, the classifier abstains and allows the expert to provide the prediction $m \sim \mathbb{P}(\mathrm{m}|\mathbf{x} = \boldsymbol{x}, \mathrm{y} = y)$. [1] The goal of the L2D classification problem is to find these functions, h, and r. We obviously employ loss functions $\ell_{\text{clf}} : \mathcal{Y} \times \hat{\mathcal{Y}} \to \mathbb{R}_+$ and $\ell_{\text{exp}} : \mathcal{Y} \times \mathcal{M} \to \mathbb{R}_+$ to evaluate the quality of the functions h and r respectively. We use the rejector $\hat{r} := r(\boldsymbol{x})$ to define the full loss function $L : \mathcal{Y} \times \hat{\mathcal{Y}} \times \{0, 1\} \times \mathcal{M} \to \mathbb{R}_+$ for L2D as follows,

$$L(y, \hat{y}, \hat{r}, m) = (1 - \hat{r}) \cdot \ell_{\text{clf}}(y, \hat{y}) + \hat{r} \cdot \ell_{\text{exp}}(y, m). \tag{3.1}$$

1: Thus, the rejector can be seen as a *meta-classifier*, determining which inputs are appropriate to pass to the expert.

As with any other classification problem, the goal in L2D is to find h^* and r^* with minimum *L-risk*, which we write again for completion,

$$\mathscr{R}^{L}_{\mathscr{D},\mathrm{m}}[r,h] = \mathbb{E}_{\boldsymbol{x}\sim\mathbf{x},y\sim\mathrm{y},m\sim\mathrm{m}}\left[\ell\left(y,h\left(\boldsymbol{x}\right),r\left(\boldsymbol{x}\right),m\right)\right].$$

And we aim to attain the *Bayes L-risk*, written for completion as below:

$$\mathscr{R}^{L,*}_{\mathscr{D},\mathrm{m}} = \inf_{\substack{r:\mathscr{X}\to\{0,1\}\\ h:\mathscr{X}\to\hat{\mathscr{Y}}}} \mathscr{R}^{L}_{\mathscr{D},\mathrm{m}}[r,h].$$

What classifier h^* and rejector r^* attain the *Bayes L-risk* $\mathscr{R}^{L,*}_{\mathscr{D},\mathrm{m}}$ for L2D?

To answer the above question, we frame L2D as a general classification problem with output label space $\mathscr{Y} = [K]$, and the output prediction label space $\mathscr{Y}^{\perp} = \mathscr{Y} \cup \{\perp\}$ where the prediction label $\perp$ means the decision of deferral. We use the loss function L defined in Equation 3.1. Thus, the prediction function $h : \mathscr{X} \to \mathscr{Y}^{\perp}$ returned by an algorithm solving this general classification problem is mapping an input $\boldsymbol{x} \in \mathscr{X}$ to either one of the output labels or to the deferral prediction, in which case, the final prediction will be provided by the expert. In this setting, we can write the closed-form expressions for h^* and r^*, as Theorem 3.1.1 establishes. We call h^* and r^* as the *Bayes optimal* classifier and rejector respectively.

Theorem 3.1.1 *For the classifier loss function $\ell_{\mathrm{clf}} : \mathscr{Y} \times \hat{\mathscr{Y}} \to \mathbb{R}_+$, the expert loss function $\ell_{\mathrm{exp}} : \mathscr{Y} \times \mathscr{M} \to \mathbb{R}_+$, for each $\hat{y} \in \mathscr{Y}$ define the quantity*

$$Z_{\hat{y}}(\boldsymbol{x}) = \sum_{y\in\mathscr{Y}} \left(w_{y,\hat{y}} - w_{y,\perp}\right) \mathbb{P}\left(\mathrm{y} = y|\mathbf{x} = \boldsymbol{x}\right) \mathbb{P}(\mathbf{x} = \boldsymbol{x}), \tag{3.2}$$

where

$$\begin{aligned} w_{y,\hat{y}} &:= \ell_{\mathrm{clf}}\left(y,\hat{y}\right), \text{ and} \\ w_{y,\perp} &:= \sum_{m\in\mathscr{M}} \ell_{\mathrm{exp}}(y,m) \cdot \mathbb{P}\left(\mathrm{m} = m|\mathbf{x} = \boldsymbol{x}, \mathrm{y} = y\right), \end{aligned}$$

then the Bayes optimal classifier h^* and rejector r^* are given as

$$\begin{aligned} h^*\left(\boldsymbol{x}\right) &= \arg\min_{\hat{y}\in\mathscr{Y}} Z_{\hat{y}}(\boldsymbol{x}) \\ r^*\left(\boldsymbol{x}\right) &= \mathbb{I}\left[Z_{\hat{y}}\left(\boldsymbol{x}\right) \geq 0; \forall \hat{y} \in \mathscr{Y}\right]. \end{aligned}$$

We provide the proof of Theorem 3.1.1 in Appendix A.1.2. We further simplify the *Bayes* optimal rejector for L2D in Corollary 3.1.2.

Corollary 3.1.2 *The Bayes optimal rejector r^* for Learning to Defer (L2D) is given as:*

$$r^*\left(\boldsymbol{x}\right) = \begin{cases} 1 \text{ if } \mathbb{E}_{\mathrm{y}|\mathbf{x}}\left[\ell_{\mathrm{clf}}\left(\hat{y},y\right)\right] \geq \mathbb{E}_{\mathrm{y}|\mathbf{x}}\mathbb{E}_{\mathrm{m}|\mathbf{x},\mathrm{y}}\left[\ell_{\mathrm{exp}}\left(m,y\right)\right] \ \forall \hat{y} \in \mathscr{Y} \\ 0 \text{ otherwise} \end{cases}$$

The above statement gives us a simple operationalization of learning to defer framework. It states that the optimal rejection rule is to compare the classifier's risk and the expert's risk for each sample $\boldsymbol{x} \in \mathcal{X}$, and then defer to the expert if the former is greater than the latter for each prediction $\hat{y} \in \mathcal{Y}$. [2]

The above results hold for general losses, ℓ_{clf} and ℓ_{exp}. However, we pay special attention to the canonical $0-1$ loss function as defined in the last chapter. For completion, we rewrite L2D loss function from Equation 3.1 for $0-1$ loss function below,

$$L_{0-1}(y, \hat{y}, \hat{r}, m) = (1 - \hat{r}) \cdot \mathbb{I}[y \neq \hat{y}] + \hat{r} \cdot \mathbb{I}[y \neq m]. \tag{3.3}$$

In this special case, getting a further simplified form of h^* and r^* is an easy exercise. Since this form will be the main focus of this thesis, we state it as a definition in Definition 3.1.1.

Definition 3.1.1 *(Bayes Optimal rejector and classifier for $0-1$ L2D loss). For the L_{0-1} L2D loss defined in Equation 3.3, the Bayes optimal classifier h^* and rejector r^* are given as*

$$h^*(\boldsymbol{x}) = \arg\max_{y \in \mathcal{Y}} \mathbb{P}(\mathrm{y} = y | \mathbf{x} = \boldsymbol{x}),$$

$$r^*(\boldsymbol{x}) = \mathbb{I}\left[\mathbb{P}(\mathrm{m} = \mathrm{y} | \mathbf{x} = \boldsymbol{x}) \geq \max_{y \in \mathcal{Y}} \mathbb{P}(\mathrm{y} = y | \mathbf{x} = \boldsymbol{x})\right],$$

where $\mathbb{P}(\mathrm{m} = \mathrm{y} | \mathbf{x} = \boldsymbol{x})$ is the probability that the expert is correct, and $\mathbb{P}(\mathrm{y} = y | \mathbf{x} = \boldsymbol{x})$ is the regular class probability.

Thus, for L_{0-1} L2D loss, the optimal rejection rule says to compare the confidences of the classifier and the expert. And if the expert has higher confidence in providing the correct prediction, we defer the sample to the expert. Otherwise, the classifier makes predictions in a typical way.

Similar to the binary classification problem in the last chapter, we don't know $\mathbb{P}(\mathrm{m} = \mathrm{y} | \mathbf{x} = \boldsymbol{x})$ and $\mathbb{P}(\mathrm{y} = y | \mathbf{x} = \boldsymbol{x})$. Thus, we again take a machine learning perspective and aim to learn a classifier and a rejector pair (h, r) that approximates (h^*, r^*). And what's now appearing to be a recurring theme, it's computationally infeasible to minimize $\mathcal{R}^{L_{0-1}}_{\mathcal{D},\mathrm{m}}[r, h]$. So, we need to employ a surrogate loss function over some surrogate prediction space and a suitable decoding function. Quite obviously now, we want such a surrogate loss function to be *consistent* w.r.t. L_{0-1}. Fortunately, such a surrogate loss function is proposed in the literature, and we describe it in the next section.

2: Intuitively, it makes sense to ask for the expert's opinion when the classifier's prediction is comparatively riskier. This is the motivation for the learning to defer framework, especially in safety-critical applications like healthcare, autonomous driving, etc.

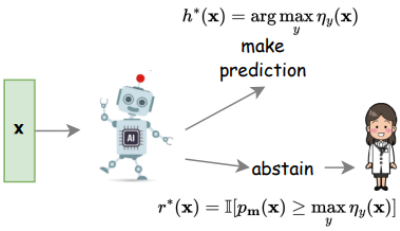


Figure 3.1: *Operationalization of the L2D framework.* The optimal rejection rule is to compare the expert's correctness probability $p_{\mathrm{m}}(\boldsymbol{x}) := \mathbb{P}(\mathrm{m} = \mathrm{y} | \mathbf{x})$ with the classifier's confidences $\eta_y(\boldsymbol{x})$, and defer if former is greater than $\eta_y(\boldsymbol{x}) \, \forall y \in \mathcal{Y}$.

3.2 Softmax Surrogate Loss for Learning to Defer

Mozannar and Sontag (2020) proposed the first consistent surrogate loss for L_{0-1}. They accomplish this by considering learning to defer as a general classification problem with the output prediction label space $\mathcal{Y}^{\perp}$ as

described in the last section. Secondly, Mozannar and Sontag (2020) use a reduction to cost sensitive learning (Ling and Sheng, 2010) that ultimately resembles the cross-entropy loss for a softmax parameterization [3]. The goal is to learn a surrogate prediction function $f : \mathcal{X} \to \mathbb{R}^{K+1}$ with a suitable decoding function $g : \mathbb{R} \to \mathcal{Y}^{\perp}$. The surrogate prediction function f can be considered *component-wise*, and specify $f_k : \mathcal{X} \mapsto \mathbb{R}$ for $k \in [1, K]$ where k denotes the class index, and let $f_{\perp} : \mathcal{X} \mapsto \mathbb{R}$ denotes the deferral ($\perp$) option. Thus, each $f_i,\ i \in [K] \cup \{\perp\}$ can be considered as a prediction function operating in the surrogate space $\mathbb{R}$. Mozannar and Sontag (2020) proposed the surrogate loss function $\phi_{SM} : \mathbb{R}^{K+1} \times \mathcal{Y} \times \mathcal{M} \to \mathbb{R}_+$. The proposed softmax-parameterized surrogate loss is given as:

$$\begin{aligned}\phi_{SM}(f_1, \ldots, f_K, f_{\perp}; y, m) = \\ &- \log\left(\frac{\exp\{f_y(\boldsymbol{x})\}}{\sum_{y' \in \mathcal{Y}^{\perp}} \exp\{f_{y'}(\boldsymbol{x})\}}\right) \\ &- \mathbb{I}[m = y]\ \log\left(\frac{\exp\{f_{\perp}(\boldsymbol{x})\}}{\sum_{y' \in \mathcal{Y}^{\perp}} \exp\{f_{y'}(\boldsymbol{x})\}}\right).\end{aligned} \tag{3.4}$$

The intuition is that the first term maximizes the function f_k associated with the true label. The second term then maximizes the rejection function $f_{\perp}$ but only if the expert's prediction is correct. The decoding function g is the simple arg max function, i.e. $g \circ f : \boldsymbol{x} \mapsto \arg\max_{i \in \mathcal{Y}^{\perp}} f_i(\boldsymbol{x})$. Thus, the rejector $r(\boldsymbol{x}) = \mathbb{I}\left[g \circ f\left(\boldsymbol{x}\right) = K + 1\right]$, and the classifier $h(\boldsymbol{x}) = g \circ f\left(\boldsymbol{x}\right)$ if $r(\boldsymbol{x}) = 0$. [4]

The function ϕ_{SM} is the first convex (in f) consistent surrogate loss proposed for L2D (Mozannar and Sontag, 2020). Convexity eases the optimization problem, and the consistency implies that minimizers of $\mathcal{R}^{\phi_{SM}}_{\mathcal{D},m}[f]$ over all measurable functions f agree with the Bayes optimal solution. Specifically, this means that the minimizer $f^* = \left(f_1^*, \ldots, f_K^*, f_{\perp}^*\right)$ of $\mathcal{R}^{\phi_{SM}}_{\mathcal{D},m}[f]$ corresponds to the Bayes optimal rejector r^* and classifier h^* through the composition of the decoding function g as defined above.

How good of a forecaster the function returned by an L2D classification algorithm working with ϕ_{SM} is?

As stated above, ϕ_{SM} gives Bayes optimal rejector and classifier. From Definition 3.1.1, we know that that the Bayes optimal rejector and classifier require estimates of $\mathbb{P}(y = y|\mathbf{x} = \boldsymbol{x})$ and $\mathbb{P}(m = m|\mathbf{x} = \boldsymbol{x}, y = y)$. However, we would like to assert that while these estimates are sufficient to recover the Bayes optimal rejector and classifier, they are not necessary. Due to the comparative nature of the Bayes rule, $g \circ f^*$ learned by ϕ_{SM} could agree with the Bayes rule but won't necessarily result in correct estimates of the above probabilities. In the next section, we will check the forecasting abilities of this L2D system, i.e., given a function f^* learned by the L2D classification algorithm minimizing the surrogate loss ϕ_{SM}, how *good* are these confidence estimates. Having *good* estimates would not only add to the Bayes optimality of f^*, but is also a critical requirement for the reliability, trustworthiness, and transparency of such an L2D system, especially in safety-critical applications.

Continuing forward, when it is clear from the context, we will use the

3: Softmax function, $\boldsymbol{s} : \mathbb{R}^K \to \Delta^{K-1}$, where Δ^{K-1} denotes the $K-1$ dimensional probability simplex, is given as:

$$\boldsymbol{s}\left(\boldsymbol{x}\right) = \left[s\left(\boldsymbol{x}_1\right), \ldots, s\left(\boldsymbol{x}_K\right)\right]$$
$$s\left(\boldsymbol{x}_i\right) = \frac{\exp\{\boldsymbol{x}_i\}}{\sum_{j \in [K]} \exp\{\boldsymbol{x}_j\}}.$$

4: In practice, Mozannar and Sontag (2020) introduce a hyperparameter $\alpha \in \mathbb{R}^+$ that re-weights the classifier loss when the expert is correct. Using $\alpha < 1$ encourages a higher degree of division of labor between classifier and expert. Yet, for all $\alpha \neq 1$, the surrogate is no longer consistent.

shorthand softmax surrogate loss to mean the function f^* returned by an L2D classification algorithm minimizing the softmax surrogate loss function.

3.3 Problem with the Softmax Surrogate Loss

Before investigating *goodness* of the confidence estimates of the softmax surrogate loss function, we need to settle two questions: a) How to get the confidence estimates? b) How to define *goodness*? Fortunately, we have a unique way to answer the question a. And a natural answer to the question b is estimating the true confidence estimates. However, we will use *calibration* of the confidence estimates[5] as a measure of the said *goodness*. Why we do so should be self-explanatory, simply because we don't know the true confidence estimates. Additionally, *calibration* of the confidence estimates is a very popular metric for the reliability and trustworthiness of a machine learning system (Vaicenavicius et al., 2019). Next, we look at these two questions separately.

5: This should not be confused with the *calibration* of some surrogate loss w.r.t some target loss. To make the distinction, we will use calibration of the confidence estimates vs calibration of the surrogate loss w.r.t. the target loss.

How to get the confidence estimates for the softmax surrogate loss function ϕ_{SM}?

We saw before that ϕ_{SM} works in the surrogate prediction space $\mathbb{R}^{\text{K}+1}$, and f^* can be considered *component-wise*, i.e. $f^*(\boldsymbol{x}) = \left[f_1^*(\boldsymbol{x}), \ldots, f_{\text{K}}^*(\boldsymbol{x}), f_\perp^*(\boldsymbol{x})\right]$, $f_i^* : \mathcal{X} \to \mathbb{R}$. To get the confidence estimates $\mathbb{P}(\text{y} = y|\mathbf{x} = \boldsymbol{x})$ for some $y \in \mathcal{Y}$, we employ an inverse *link* function $\gamma_y : \mathbb{R}^{\text{K}+1} \to [0,1]$. Similarly for $\mathbb{P}(\text{m} = \text{y}|\mathbf{x} = \boldsymbol{x})$, we need $\gamma_{\text{m}} : \mathbb{R}^{\text{K}+1} \to [0,1]$. We derive these inverse *link* functions in Appendix A.2.1, and write them below:

$$\begin{aligned} \gamma_y\left(f^*(\boldsymbol{x})\right) &= \frac{\exp\{f_y^*(\boldsymbol{x})\}}{\sum_{y' \in \mathcal{Y}} \exp\{f_{y'}^*(\boldsymbol{x})\}} \\ \gamma_{\text{m}}\left(f^*(\boldsymbol{x})\right) &= \frac{\exp\{f_\perp^*(\boldsymbol{x})\}}{\sum_{y' \in \mathcal{Y}} \exp\{f_{y'}^*(\boldsymbol{x})\}} \end{aligned} \tag{3.5}$$

We can quickly verify that γ_y gives *valid* confidences. [6] Furthermore, denoting $p_\perp\left(f_\perp^*(\boldsymbol{x})\right) = \frac{\exp\{f_\perp^*(\boldsymbol{x})\}}{\sum_{y' \in \mathcal{Y}_\perp} \exp\{f_{y'}^*(\boldsymbol{x})\}}$, it's quite obvious that $\gamma_{\text{m}}\left(f^*(\boldsymbol{x})\right) = \frac{p_\perp\left(f_\perp^*(\boldsymbol{x})\right)}{1 - p_\perp\left(f_\perp^*(\boldsymbol{x})\right)}$. Since $p_\perp \in [0,1]$, it's clear that the fu of γ_{m} is $[0,\infty)$. Thus, for γ_{m} to be a *valid* inverse *link* function, it must hold that $p_\perp \in [0, 0.5]$. This fact will be of grave importance for our answer to question b above, so we state it explicitly in Proposition 3.3.1.

6: Check: $\sum_{y \in \mathcal{Y}} \gamma_y = 1$. Also, $\gamma_y \in [0,1]$.

Proposition 3.3.1 *For a prediction function f^* of ϕ_{SM}, if $\exists \boldsymbol{x} \in \mathcal{X}$ such that $p_\perp\left(f_\perp^*(\boldsymbol{x})\right) > 0.5$, then $\gamma_{\text{m}}\left(f_\perp^*(\boldsymbol{x})\right) > 1$. Thus, γ_{m} cannot estimate $\mathbb{P}(\text{m} = \text{y}|\mathbf{x} = \boldsymbol{x})$.*

It's worth noting again that γ_{m} is a *unique* inverse *link* function to estimate $\mathbb{P}(\text{m} = \text{y}|\mathbf{x} = \boldsymbol{x})$. So, in case we get these *invalid* confidence estimates for $\mathbb{P}(\text{m} = \text{y}|\mathbf{x} = \boldsymbol{x})$, it would be due to the estimation errors in $f^*(\boldsymbol{x})$. However, this does not affect the *consistency* property of ϕ_{SM} w.r.t L_{0-1}.

As already mentioned in the last section, true confidence estimates are sufficient to get the Bayes optimal rejector and classifier (and hence for *consistency*), but they are not necessary. Rather, it would mean that the softmax parameterization admits many solutions that do not correspond to valid estimators for $\mathbb{P}(\mathrm{y} = \mathrm{m}|\boldsymbol{x})$, and yet behave according to Bayes optimal rule. In other words, the Bayes solutions seem to be 'fragile' in the sense that they require $p_{\perp}\left(f^{*}(\boldsymbol{x})\right) \leq 1/2$ while its true range is $[0, 1]$.

How to define the *goodness* of the confidence estimates?

We use the notion of *calibration* of the confidence estimates as a measure of the *goodness* of the confidence estimates. More specifically, we use *confidence calibration* (Dawid, 1982) as a measure. The definition applies to any general confidence estimate, however, we will define it in the context of expert's correctness probability $p_{\mathrm{m}}(\boldsymbol{x}) := \mathbb{P}(\mathrm{m} = \mathrm{y}|\mathbf{x} = \boldsymbol{x})$. Given $p_{\mathrm{m}}(\boldsymbol{x})$, we call p_{m} *confidence-calibrated* if, for any confidence level $c \in [0, 1]$, the actual proportion of times the expert is correct is equal to c:

$$\mathbb{P}(\mathrm{m} = \mathrm{y} \mid p_{\mathrm{m}}(\boldsymbol{x}) = c) = c. \tag{3.6}$$

This statement should hold for all possible instances $\boldsymbol{x}$ with confidence c. Confidence calibration is a desirable quality for a forecasting system to be a *good* forecaster. Intuitively, it means that if the system says the expert has 75% chances of being correct, then the expert should actually be correct in 75 out of those 100 cases. There are other notions of calibration of the confidence estimates as well, for example, distribution calibration, and classwise calibration. Since expert correctness is a binary classification problem, distribution calibration, confidence calibration, and classwise calibration all coincide (Vaicenavicius et al., 2019).

To quantify *confidence calibration*, we use a metric called expected calibration error (ECE). It is formally defined as

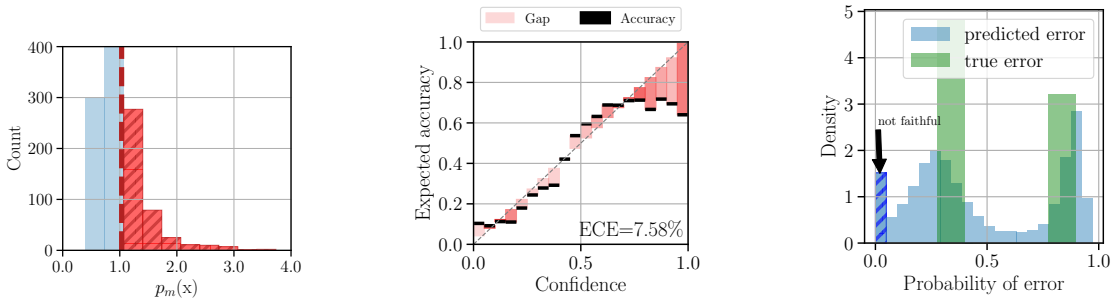
$$\mathrm{ECE}(p_{\mathrm{m}}) = \mathbb{E}_{\mathbf{x}}|\mathbb{P}\left(\mathrm{m} = \mathrm{y} \mid p_{\mathrm{m}}(\mathbf{x}) = c\right) - c|. \tag{3.7}$$

In practice, ECE is usually computed by equal-width binning of the predictions according to the confidence level. This metric is also visualized as plots called reliability diagrams, where for each such bin, we verify if the calibration equation (Equation 3.7) is true or not. The difference between the bin's confidence and the empirical accuracy is plotted, and this average difference is known as ECE.

Now that we have established answers to both the questions `a` and `b`, we verify if $p_{\perp}\left(f_{\perp}^{*}(\boldsymbol{x})\right) > 0.5$, holds true in practice or not.

3.3.1 Empirical Confirmation

Our goal here is to show the existence of $\boldsymbol{x} \in \mathcal{X}$, such that $p_{\perp}\left(f_{\perp}^{*}(\boldsymbol{x})\right) > 0.5$, and consequently, $p_{\mathrm{m}}(\boldsymbol{x}) > 1$. We use a CIFAR-10 (Krizhevsky, 2009) simulation that is similar to Mozannar and Sontag's (2020) CIFAR-10 experiment. We simulate expert who is assumed to have non-uniform expertise: 75% chance of being correct on the first five classes, and 20% (i.e. random) chance on the last five classes. We train learning to defer



(a) Empirical Distribution of p_{m} Values **(b)** Reliability Diagram and ECE **(c)** Empirical Distribution of $1 - p_{\mathrm{m}}(\mathbf{x})$

Figure 3.2: *Calibration of Softmax Parameterization on CIFAR-10*: Subfigure (a) reports the observed values of $p_{\mathrm{m}}(\mathbf{x})$ on the CIFAR-10 simulation study. We find that 39.4% of test samples have $p_{\mathrm{m}}(\mathbf{x}) > 1$ (denoted in red). Subfigure (b) reports a reliability diagram and the expected calibration error (ECE) when $p_{\mathrm{m}}(\mathbf{x})$ is restricted to $(0, 1]$. The shade of the bin color represents the proportion of samples in the bin (darker shade, more samples). Subfigure (c) shows the distribution of risk estimates. Note the clear bias towards zero error.

system with this expert using ϕ_{SM}. Subfigure 3.2a shows a histogram of the values of $p_{\mathrm{m}}(\boldsymbol{x})$ as observed on the CIFAR-10 test set. The blue bars represent the values less than or equal to one. The red bars show the pathological cases greater than one. 39.4% of the test samples (3940 instances) resulted in $p_{\mathrm{m}}(\boldsymbol{x}) > 1$.

We also consider modifying $p_{\mathrm{m}}(\boldsymbol{x})$ so that all values greater than one are rounded down to one. In this case, since now $p_{\mathrm{m}}(\boldsymbol{x})$ is forcibly restricted to $(0, 1]$, we can perform standard evaluations of confidence calibration, i.e. plotting a reliability diagram and computing *expected calibration error* (ECE) as defined in Equation 3.7.

Subfigure 3.2b shows the reliability diagram and reports the ECE for confidence calibration when $p_{\mathrm{m}}(\boldsymbol{x})$ is restricted. Unsurprisingly, we still observe that the model's estimate of the expert's correctness is uncalibrated, exhibiting overconfidence. The ECE is 7.58%. Subfigure 3.2c plots the distribution of *pointwise* risks: $1 - p_{\mathrm{m}}(\mathbf{x})$. [7] Due to the probabilities being clamped to one, we see a false mode at zero error. In turn, the system is not transparent about the actual risk that decision-makers would encounter.

7: We leave it as an exercise to verify that the *pointwise* risk estimates of the expert for $0-1$ loss are given as $1 - p_{\mathrm{m}}(\boldsymbol{x})$.

Summary of the Chapter

We formally study the problem of multiclass learning to defer (L2D) classification problem and analyze its Bayes optimal behavior. We also describe the only *consistent* surrogate loss function w.r.t. to the canonical $0-1$ loss function proposed by Mozannar and Sontag (2020). While the loss function is a *consistent* one w.r.t. the $0-1$ loss function, we show that it learns a 'fragile' Bayes rule, where it fails to appropriately estimate the true confidences required for the Bayes optimal classifier and the rejector. The estimates for the expert's correctness confidence are unbounded; thus, the system is not *confidence calibrated*.

One-vs-All Surrogate Loss for Learning to Defer

4

—This is where Professor Abernathe invites you to set down the story of your own adventure.
—I think we're on it now, said Billy. — Amor Towles, The Lincoln Highway

In the last chapter, we show the 'fragility' of the softmax surrogate loss for multiclass learning to defer (L2D), where it theoretically learns the Bayes optimal rejector and classifier but does not necessarily estimate the class probabilities and expert's correctness probability to attain the Bayes rule. We demonstrate that the problematic quantity is the expert's correctness probability, which in theory, is unbounded. In this chapter, we propose an alternative surrogate loss function for L2D that is also a *consistent* surrogate loss function w.r.t to the canonical $0-1$ loss for L2D. Our loss function is based on reductions using the error correcting output codes that resemble the popular One-vs-All parameterization (a.k.a. One-vs-Rest) in machine learning. Thus, we call our loss function the One-vs-All (OvA) surrogate loss for L2D. We will continue using the notation from previous chapters.

4.1 Reductions using Error Correcting Output Codes

We also consider L2D as a general classification problem and aim to find a function $h : \mathcal{X} \to \mathcal{Y}^{\perp}$, mapping input $\boldsymbol{x} \in \mathcal{X}$ to either one of the output labels or to the deferral decision. We note that such a function h trivially models both the rejector and the classifier for an L2D problem, for example, the rejector $r_c : \boldsymbol{x} \mapsto \mathbb{I}[h(\boldsymbol{x}) \in \{\perp\}]$, and the classifier $h_c : \boldsymbol{x} \mapsto h(\boldsymbol{x})$ when $r_c(\boldsymbol{x}) = 0$. To this end, we aim to find h, or (h_c, r_c) that minimizes $\mathcal{R}^{L_{0-1}}_{\mathcal{D},\mathrm{m}}[r_c, h_c]$, and ideally attains $\mathcal{R}^{L_{0-1},*}_{\mathcal{D},\mathrm{m}}$. It's now an obvious fact that computationally minimizing $\mathcal{R}^{L_{0-1}}_{\mathcal{D},\mathrm{m}}[r_c, h_c]$ is an infeasible problem. So, we employ a surrogate function instead.

To derive our surrogate loss function, we follow the general framework of reductions using error-correcting output codes (ECOC) as described in Section 2.4 in Chapter 2 (Preliminaries). To this end, we need to define our coding matrix $\mathbf{M} \in \{-1, 0, 1\}^{\mathrm{K} \times \hat{\mathrm{K}}}$. We have $\hat{\mathrm{K}} = \mathrm{K} + 1$, and we define $\mathbf{M} \in \{-1, 1\}^{\mathrm{K} \times (\mathrm{K}+1)}$. The construction of $\mathbf{M}$ is as follows: The $\mathrm{K} \times \mathrm{K}$ sub-matrix $\mathbf{M}_{1:\mathrm{K},1:\mathrm{K}}$ has $+1$ along its diagonal and -1 on the off-diagonal. The entries in the $\mathrm{K}+1$-th column are given by the function $m_{y,\mathrm{K}+1} = \nu_y(\mathrm{m}) := (-1 + 2\mathbb{I}[y = \mathrm{m}])$. Thus, our main insight is to construct a set of ECOC matrices.

Having defined the coding matrix $\mathbf{M}$, we can follow the general strategy to decompose this classification problem into $\mathrm{K}+1$ binary classification sub-problems. More specifically, for a surrogate prediction space $\mathcal{V} \subseteq \mathbb{R}$, we learn K surrogate prediction functions $f_y : \mathcal{X} \to \mathcal{V}$, $y \in \mathcal{Y}$, and similarly $f_{\perp} : \mathcal{X} \to \mathcal{V}$, each of which is a binary classifier. Thus, for each $\boldsymbol{x} \in \mathcal{X}$, we get the prediction $f(\boldsymbol{x}) = [f_1(\boldsymbol{x}), \ldots, f_{\mathrm{K}}(\boldsymbol{x}), f_{\perp}(\boldsymbol{x})] \in \mathcal{V}^{\mathrm{K}+1}$. This means that, eventually, we are learning a prediction function $f : \mathcal{X} \to \mathcal{V}^{\mathrm{K}+1}$.

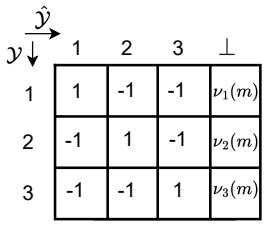


Figure 4.1: *An example of the ECOC matrix construction.* We have $\mathcal{Y} = [3]$, $\hat{\mathcal{Y}} = \mathcal{Y}^{\perp} = [3] \cup \{\perp\}$, and $\nu_y(m) = (-1 + 2\mathbb{I}[y = m])$

.

Now, if we use $\phi : \{-1, 1\} \times \mathscr{V} \to \mathbb{R}_+$ as the binary loss function to learn these K + 1 binary classifiers, we can use $\mathbf{M}$ to write the closed-form expression of the final multiclass surrogate loss $\psi_{\text{OvA}} : \mathscr{Y} \times \mathscr{V}^{\text{K}+1} \times \mathscr{M} \to \mathbb{R}_+$ this whole process eventually minimizes as follows:

$$\begin{aligned}&\psi_{\text{OvA}}(f_1, \ldots, f_K, f_\perp; y, m) = \\ &\quad \phi[f_y(\boldsymbol{x})] + \sum_{y' \in \mathscr{Y}, y' \neq y} \phi[-f_{y'}(\boldsymbol{x})] \; + \\ &\quad \phi[-f_\perp(\boldsymbol{x})] + \mathbb{I}[m = y]\,(\phi[f_\perp(\boldsymbol{x})] - \phi[-f_\perp(\boldsymbol{x})])\end{aligned} \tag{4.1}$$

We derive the above closed-form expression in Appendix A.2.2.

4.2 One-vs-All Surrogate Loss for L2D

Our proposed One-vs-All surrogate loss for L2D is, thus, ψ_{OvA}. The decoding function g for a prediction function returned by an L2D classification algorithm working with ϕ_{OvA} is again the arg max function, i.e. $g \circ f : \boldsymbol{x} \mapsto \arg\max_{i \in \mathscr{Y}^\perp} f_i(\boldsymbol{x})$. This gives us the rejector $r(\boldsymbol{x}) = \mathbb{I}[g \circ f(\boldsymbol{x}) = \text{K} + 1]$, and the classifier $h(\boldsymbol{x}) = g \circ f(\boldsymbol{x})$ when $r(\boldsymbol{x}) = 0$. Our formulation is, thus, the OvA analog of Mozannar and Sontag's (2020) softmax-based loss. The f-functions are entirely the same; the difference is in how they are combined. The classifier and rejector are computed exactly the same as in the softmax case. We can also introduce a re-weighting parameter that is analogous to α in Mozannar and Sontag's (2020) loss by re-weighting the first two terms in Equation 4.1 when the expert is correct.

We next prove that ψ_{OvA} is a *consistent* surrogate loss w.r.t. L_{0-1} for the L2D classification problem. As stated above, ψ_{OvA} can be seen as a one-vs-all analog of ϕ_{SM}. However, the argument for *consistency* of ψ_{OvA} is non-trivial. We cannot construct our consistency proof in the same direct manner as Mozannar and Sontag (2020). When we differentiate with respect to a particular $f(\boldsymbol{x})$, the other f's drop from the OvA loss (but not from the softmax loss). Thus, to analyse the *consistency* of ψ_{OvA} w.r.t. L_{0-1}, we first study the *calibration* of the surrogate loss ψ_{OvA} w.r.t. the target los L_{0-1}. We can show that ψ_{OvA} is *calibrated* w.r.t. L_{0-1} as Theorem 4.2.1 establishes.

Theorem 4.2.1 *For a strictly proper binary composite loss ϕ with a well-defined continuous inverse link function γ^{-1}, ψ_{OvA} (Equation 4.1) is a calibrated surrogate for the canonical $0-1$ learning to defer loss (Equation 3.3)).*

This is our main result, and we prove it in Appendix A.1.4. We show that the *pointwise* minimizer f^* of the *inner* ψ_{OvA}-*risk* $\mathscr{C}^{\psi_{\text{OvA}}}_{\mathscr{D},\text{m}}[f]$ together with the decoding function g as defined above, agree with the Bayes optimal classifier and rejector pair. Having established the *calibration* of ψ_{OvA} w.r.t L_{0-1}, the next corollary gives us the result.

Corollary 4.2.2 *Assume that $f \in \mathscr{F}$, where $\mathscr{F}$ is the hypothesis class of all*

measurable functions. Minimizability (Definition 2.1.3) is then satisfied for ψ_{OvA}*, and it follows that* ψ_{OvA} *is a consistent surrogate for the* $0-1$ *learning to defer loss (Equation 3.3).*

Thus, ψ_{OvA} is also a *consistent* loss function for L2D. This means that the minimizer of the proposed loss function ψ_{OvA} over all measurable functions agrees with the Bayes optimal classifier and rejector (Definition 3.1.1). Our result holds for all strictly binary proper composite loss functions ϕ (see Table Section 2.3 on page 9 for a few examples).

We now describe how to get the confidence estimates for $\mathbb{P}(\mathrm{y} = \mathrm{m}|\mathbf{x} = \boldsymbol{x})$ and $\mathbb{P}(\mathrm{y} = y|\mathbf{x} = \boldsymbol{x})$ for each $y \in \mathcal{Y}$ for the surrogate prediction function f^* returned by an L2D classification algorithm working with ψ_{OvA}. However, this is trivial now due to the existence of well-defined inverse *link* function γ^{-1} for the proper composite loss function ϕ. We simply have:

$$\begin{aligned} \gamma_y\left(f^*(\boldsymbol{x})\right) &= \gamma^{-1}\left(f^*(\boldsymbol{x})\right) \\ \gamma_{\mathrm{m}}\left(f^*(\boldsymbol{x})\right) &= \gamma^{-1}\left(f^*(\boldsymbol{x})\right). \end{aligned} \tag{4.2}$$

Here, we define,

$$\gamma_y\left(f^*(\boldsymbol{x})\right) = \gamma^{-1} : \left(f_1^*(\boldsymbol{x}), \ldots, f_{\mathrm{K}}^*(\boldsymbol{x}), f_\perp^*(\boldsymbol{x})\right) \mapsto \gamma^{-1}\left(f_y^*(\boldsymbol{x})\right), y \in \mathcal{Y}^\perp.$$

One thing we can immediately conclude is that both γ_y and γ_{m} have range in $[0, 1]$, i.e. they both assign *valid* probabilities, unlike in the case of ϕ_{SM} where γ_{m} ranges in $[0, \infty)$. One can say that with the application and the properties of γ^{-1}, we are now directly estimating the necessary confidences required for the Bayes optimality of the rejector and classifier for L2D (Definition 3.1.1).

Summary of the Chapter

We derive an alternative loss function for an L2D classification problem based on the reductions using error-correcting output codes. We prove that our loss function is a *consistent* surrogate w.r.t the canonical $0-1$ loss function for L2D. We further describe how to estimate the confidence required for the Bayes optimal behavior of an L2D system.

5 Experiments

—I am pretty sure that we are on our adventure, Emmett. — Amor Towles, The Lincoln Highway

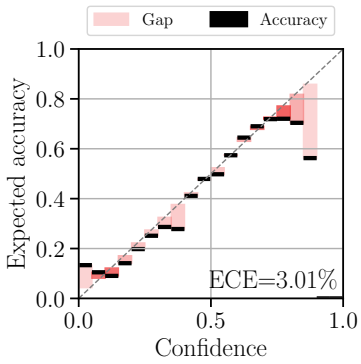
In this chapter, we empirically study the properties of the proposed OvA surrogate loss ψ_{OvA} for L2D and compare them with ϕ_{SM} along with the common baselines for L2D in the literature. We perform two types of experiments. In the first, we verify the *confidence calibration* property of the proposed OvA surrogate loss. We also demonstrate the significance of better confidence calibrated models and the efficacy of ψ_{OvA} compared with ϕ_{SM} in a safety-critical application scenario. In the second type of experiment, we assess the overall system's performance in terms of classification accuracy on tasks ranging from hate speech detection to galaxy classification to skin lesion diagnosis. We compare our OvA-based method to the softmax surrogate method as well as other state-of-the-art methods for L2D in the literature. Although ψ_{OvA} is amenable to any strictly proper binary composite loss function, we use the logistic loss for all our experiments. We report results averaged over re-runs with six different random seeds.

5.1 Assessing Confidence Calibration

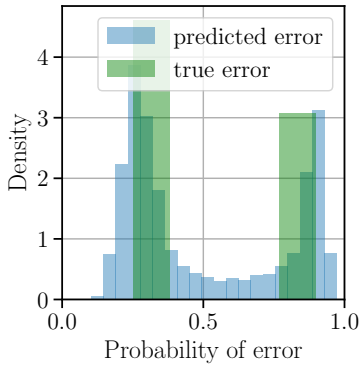
In this section, we assess the confidence calibration property of the proposed OvA surrogate loss. We are interested in estimating the quality of the class probability $p_y = \mathbb{P}(\mathrm{y} = y|\mathbf{x} = \boldsymbol{x})$ and the expert's correctness probability $p_{\mathrm{m}} = \mathbb{P}(\mathrm{m} = \mathrm{y}|\mathbf{x} = \boldsymbol{x})$. We refer the reader to Chapter 3 (Learning to Defer) and Chapter 4 (One-vs-All Surrogate Loss for Learning to Defer) for how to estimate these probabilities for both ϕ_{SM} and ψ_{OvA}. Since p_m can be a degenerate quantity for ϕ_{SM}, we focus special attention to the calibration property of p_m, and we call it confidence calibration w.r.t. expert correctness. We perform experiments on `CIFAR-10` (Krizhevsky, 2009) and `HAM10000` (Tschandl et al., 2018). We now discuss our experimental settings and results in the following sections.

5.1.1 Comparison to the Softmax Loss on `CIFAR-10`

Data, Model, and Training We use the standard train-test splits of `CIFAR-10` (Krizhevsky, 2009). We further partition the training split by 90% − 10% to form training and validation sets, respectively. We simulate the expert demonstrations from the training labels, as described in detail below. We use the same neural network and training settings for both the OvA and softmax methods. Following Mozannar and Sontag (2020), we use a wide residual networks (Zagoruyko and Komodakis, 2016) to parameterize the $f(\boldsymbol{x})$ functions. We train a 28-layer network using stochastic gradient descent (SGD) with momentum and a cosine annealing schedule for the learning rate. We employ early stopping, and terminating training if the validation loss does not improve for 20 epochs. Additional experimental details can be found in Appendix A.3.



(a) Reliability Diagram and ECE



(b) Empirical Dist. of $1 - p_{\mathrm{m}}^{\mathrm{OvA}}(\boldsymbol{x})$

Figure 5.1: *Calibration and Accuracy of OvA Parameterization on CIFAR-10.* Subfigure (a) reports a reliability diagram and the expected calibration error (ECE) for $p_{\mathrm{m}}^{\mathrm{OvA}}(\boldsymbol{x})$ (Eq. 3.7). A darker bin shade means more samples in the bin. Subfigure (b) shows the distribution of risk estimates.

Table 5.1: Subtable (a) reports *ECE* (%) w.r.t expert correctness. We compare confidence-calibration across the three parameterizations considered: OvA, softmax, and proxy ($p_{\perp}$) for the estimator of p_{m}. Subtable (b) reports *ECE* (%). We compare calibration across the two parameterizations: OvA and softmax for the estimators of $\mathbb{P}(\mathrm{y} = y|\mathbf{x} = \boldsymbol{x})$.

Setting	**OvA**	Softmax	Proxy
Both Random	0.53	0.97	**0.04**
Random Expert	**0.68**	3.72	2.83
Random Data	**2.05**	2.07	39.06
Both Useful	**1.68**	3.32	37.15

(a) ECE (%) w.r.t Expert Correctness on `CIFAR-10`

Setting	**OvA**	Softmax
Both Random	0.51	**0.34**
Random Expert	**6.47**	7.22
Random Data	**1.94**	2.36
Both Useful	**6.92**	7.92

(b) ECE (%) w.r.t Classifier Correctness on `CIFAR-10`

,

OvA Method's Calibration We now test our OvA method's calibration in the same experimental setting used to test the softmax method in Section 3.3.1 Chapter 3 (Learning to Defer). To reiterate, the expert has a 75% chance of being correct in the first five classes and a random chance in the last five. Figure 5.2a reports a reliability diagram and the ECE. Compared to the softmax results in Figure 3.2b, our OvA loss produces a model that has an over fifty percent reduction in ECE: 7.58% for softmax, 3.01% for OvA. Figure 5.2b reports the empirical distribution of error estimates: $1 - p_{\mathrm{m}}^{\mathrm{OvA}}(\boldsymbol{x})$. Unlike the corresponding softmax results in Figure 3.2c, the OvA method produces sharper modes nearer to the true error values. Moreover, OvA does not have a false mode at zero.

Comparing Calibration Across Estimators We next test OvA's calibration against not only the softmax but also the proxy function $p_{\perp}$ defined in Section 3.3 Chapter 3 (Learning to Defer). It is possible that the deferral function $p_{\perp}(\mathbf{x})$ is a useful estimator of $\mathbb{P}(\mathrm{m} = \mathrm{y}|\boldsymbol{x})$, despite that theory suggests otherwise. Here the range is no longer a problem because $p_{\perp}(\mathbf{x}) \in [0, 1]$. We consider two types of experts: a useful one and a random one. The useful one is an *oracle* (i.e. always correct) for the first seven classes and predicts randomly for the last three classes. The random expert predicts uniformly over all classes. Moreover, we consider when the data is useful, i.e. the original CIFAR-10 training split, and when it is random, i.e. training labels are uniformly random.

ECE results for the OvA, softmax, and proxy (p_{m}) methods are reported in

Table 5.1a. OvA has the best ECE in all but one case—the one in which both expert and data are random. Yet the $p_\perp$ proxy is clearly not a viable estimator since it has an egregious ECE of 37.15% when both data and experts are useful. Furthermore, its ECE is an even worse 39.06% when the expert is useful and the data is random. In general, the softmax's true estimator p_{m} is competent but still consistently worse than the OvA estimator. We compare the ECE values for the classifier for OvA and softmax in Table 5.1b.

System Accuracy and Coverage Next, we compare the OvA system's accuracy to the softmax's. The expert in this case has a 70% chance of being correct if the image belongs to the classes $[1, k]$ and a random chance if it belongs to classes $[k, 10]$. We then vary k from $k = 2$ to $k = 8$. The left plot in Figure 5.3 shows accuracy vs k. Our OvA model (blue) has a modest but consistent advantage over the softmax model (red).

The right plot in Figure 5.3 reports the accuracy vs coverage, where coverage is the proportion of samples that the system has *not* deferred. *Classifier accuracy* is the accuracy on the non-deferred samples. An L2D system ideally should have high coverage and high accuracy. Again, the results show the OvA method's (blue) advantage at most coverage levels. Note the OvA's significant superiority at low coverage ($0.2 - 0.3$). Here the rejector must carefully choose which instances to pass to the classifier. We believe that OvA's success is due to OvA's superior calibration in estimating when to defer (as the Bayes rule suggests).

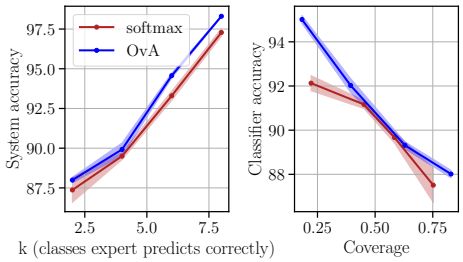


Figure 5.3: *Accuracy and Coverage plot for* `CIFAR-10`. We report the accuracy as a function of an expert with increasing expertise (left) and of varying coverage (right). Coverage means the %ge of samples where $r(\boldsymbol{x}) = 0$.

Effect of Calibration on System's Accuracy Finally, we verify confidence calibration's role in the overall system's accuracy. For the trained one-vs-all model from Figure 5.3, we apply a post-processing calibration technique called *temperature scaling* (Guo et al., 2017a) to further calibrate the rejector. In Figure 5.4, we see that this additional calibration step marginally improves the system's accuracy. This result shows that calibration does positively correlate with accuracy. This result should be unsurprising given that the correct estimation of confidence is sufficient to achieve the Bayes rule for L2D.

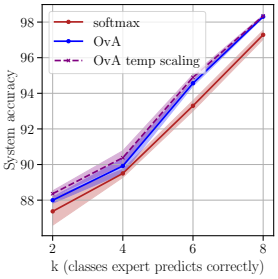
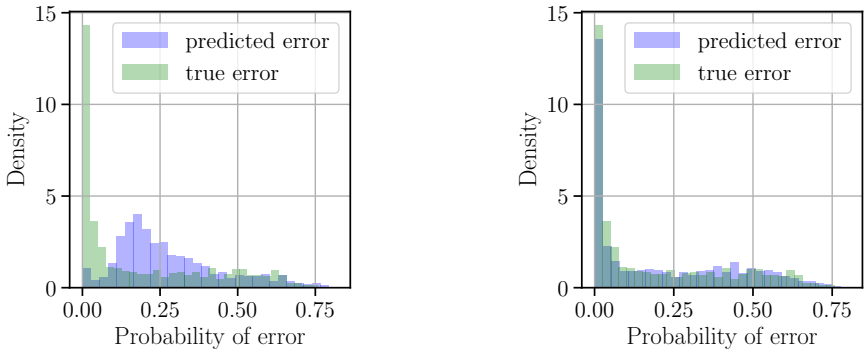


Figure 5.4: *Effect of post-processing calibration for One-Vs-All rejector.* We can see that post-processing calibration of $p_{\mathrm{m}}(\boldsymbol{x})$ further improves the system accuracy. This shows the effect of calibration on the overall system's accuracy for L2D.

5.1.2 Risk Assessment on `HAM10000`

Data, Model, and Expert We again study risk assessment but this time for a high-stakes medical task. `HAM10000` (Tschandl et al., 2018) is a data set of 10,015 dermatoscopic images containing seven categories of human skin lesions. We partition the data into 60% training, 20% validation, and 20% test splits. Each image includes metadata such as age, gender, and diagnosis type of the lesion. For our simulated expert model, we train an 8-layer MLPMixer (Tolstikhin et al., 2021). To simulate the expert having extra information, we input the image metadata into the final feedforward layer. This model has a classification accuracy of 74% (see Table 5.2 for complete classwise performance). For the classifier, we fine-tune a 34-layer residual network (ResNet34) (He et al., 2016), following Tschandl et al. (2020). We use data augmentations such as random cropping, reflection, and horizontal flipping.



(a) Risk for Model Trained with Softmax Surrogate **(b)** Risk for Model Trained with OvA Surrogate (Ours)

Figure 5.5: *Pointwise risk for Softmax vs OvA models on HAM10000.* Subfigure (a) reports the distribution of pointwise risks for the softmax method: $1 - p_{\mathrm{m}}(\boldsymbol{x})$. Subfigure (b) reports the distribution of pointwise risks for the OvA method: $1 - p_{\mathrm{m}}^{\mathrm{OvA}}(\boldsymbol{x})$. We observe markedly more overlap for the latter. The Wasserstein distance between the empirical and true error distributions is 8.02 ± 1.37 for OvA and 26.72 ± 1.77 for softmax.

Table 5.2: *Classwise performance of the simulated MLPMixer expert on* `HAM10000`. We can see that the trained model has non-uniform performance across different classes. The resulting model is still a valid simulation of a real-world expert who might be an expert for some classes (class `nv` for example).

	Classes							
metric	`bkl`	`df`	`mel`	`nv`	`vasc`	`akiec`	`bcc`	weighted avg
precision	0.52	0.33	0.51	0.82	0.27	0.44	0.47	0.71
recall	0.37	0.06	0.21	0.95	0.48	0.39	0.45	0.74
f1-score	0.43	0.10	0.30	0.88	0.34	0.41	0.46	0.71

Results Figure 5.5 visualizes the expert's predicted error and the expert's true error on the `HAM10000` test set. Subfigure (a) shows results for the softmax method and (b) for our OvA method. We restrict $p_{\mathrm{m}}(\boldsymbol{x}) \in (0, 1]$ for the softmax surrogate. The gap between the predicted and true error is substantially reduced for OvA. We confirm this quantitatively by computing the Wasserstein distance between the true and predicted error. The distance is 8.02 ± 1.37 for OvA and 26.72 ± 1.77 for softmax. OvA provides clearly superior estimates of the expert's error. This suggests that the OvA method is more faithful and transparent about the risk associated with asking an expert to make the prediction. This is a desirable quality in high-stake applications like healthcare.

5.2 Assessing Overall System's Accuracy

Data We examine the OvA method's classification error on three real-world tasks: `HAM10000` (Tschandl et al., 2018) for diagnosing skin lesions, `Galaxy-Zoo` (Bamford et al., 2009) for scientific discovery, and `HateSpeech` (Davidson et al., 2017) for detecting offensive language. Following Okati et al. (2021), we use a random sample of 10,000 images for `Galaxy-Zoo`. We use 60% train, 20% validation, and 20% test splits for `HAM10000` and `HateSpeech`.

Baselines We compare the OvA and softmax-based surrogates to three baselines. The first is *differentiable triage* (Okati et al., 2021), a policy-learning method. The other two baselines are confidence-based methods that do not enjoy theoretical guarantees. The two are the *score* baseline (Raghu et al., 2019) and the *confidence* baseline (Bansal et al., 2021a). We note that the differentiable-triage algorithm (Okati et al., 2021) considers the triage level (or *budget*) in the training of the algorithm. Since the expert might be difficult or expensive to access, *budget* is the upper limit on the proportion of samples that can be deferred to the expert. None of the other baselines have this aspect. Thus, to fairly compare all the other methods with the differentiable triage algorithm, we use the same methodology employed by Okati et al. (2021) in their paper (We refer the reader to Appendix C of their paper for more details). For each of the method, we also provide the details below: [1]

1: Here, $p_k(\boldsymbol{x}) = \mathbb{P}(\mathrm{y} = k|\boldsymbol{x}), k \in \mathcal{Y}$.

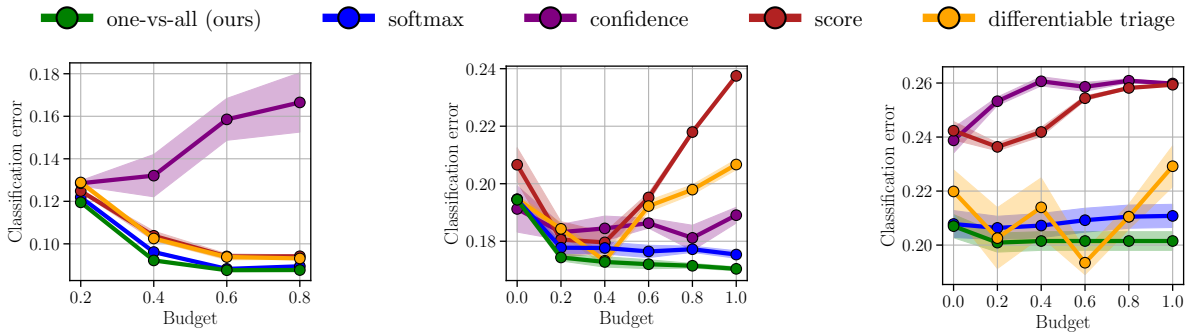
1. **Softmax Surrogate** (Mozannar and Sontag, 2020) : for a *budget* b and the samples size N, it sorts the samples in increasing order of $\max_{k\in[\mathrm{K}]} p_k(\boldsymbol{x}) - p_\perp(\boldsymbol{x})$, and then defers the min $(\lfloor b|\mathrm{N}|\rfloor, n_c)$ where n_c is the number of samples for which $p_\perp(\boldsymbol{x}) \geq \max_{k\in[\mathrm{K}]} p_k(\boldsymbol{x})$.

2. **OvA Surrogate**: we use the same procedure as the softmax method.

3. **Score Baseline** (Raghu et al., 2019): this method first trains a classifier model, and uses the classifier's predictive uncertainty to defer to the expert. Note that this classifier is trained in a regular way, i.e. it doesn't employ any additional procedure for deferral. During test time, it first sorts the dataset of size N in the increasing order of $\max_{k\in[\mathrm{K}]} p_k(\boldsymbol{x})$, and defers to the expert first $\lfloor b|\mathrm{N}|\rfloor$ for the *budget* b. The performance of this method depends on the reliability of the uncertainty estimates the classifier provides. We, therefore, use a post-processing calibration technique called temperature scaling (Guo et al., 2017a) to calibrate the classifier using the validation dataset split.

4. **Confidence Baseline** (Bansal et al., 2021a): this method first estimates $\mathbb{P}(\mathrm{y} = \mathrm{m})$, the probability of the expert being correct. However, this estimate is independent of the input sample $\boldsymbol{x}$, i.e. $\mathbb{P}(\mathrm{y} = |\mathbf{x} = \mathrm{x}) = \mathbb{P}(\mathrm{y} = \mathrm{m})$. Having obtained this estimate, it trains the system sequentially where at each iteration, it uses only min $(\lfloor b\mathrm{N}\rfloor, n_c)$ samples with the lowest value of $\mathbb{P}(\mathrm{y} = \mathrm{m}) - \max_{k\in[\mathrm{K}]} p_k(\boldsymbol{x})$ in the corresponding mini-batch for training. Here, n_c is the number of samples where $\mathbb{P}(\mathrm{y} = \mathrm{m}) > \max_{k\in[\mathrm{K}]} p_k(\boldsymbol{x})$. During test time for the *budget* b, it first sorts the dataset of size N in the increasing order of $\max_{k\in[\mathrm{K}]} p_k(\boldsymbol{x})$, and defers the first min $(\lfloor b|\mathrm{N}|\rfloor, n_c)$ samples to the expert, where n_c denotes the same quantity as before except this time for the test set samples.

5. **Differentiable Triage** (Okati et al., 2021): this is a sequential learning algorithm that first estimates the predictive model for a given *budget* b, and then having learned the model, it approximates the optimal triage policy for the learned model and b. The optimal triage policy is to compare the model's prediction loss and the expert's prediction loss and defer to the expert if the latter is smaller than the former. Therefore, the training algorithm assumes access to the expert's predictive loss as opposed to just the expert's predictions for the surrogate loss methods. Following the original authors,



(a) HateSpeech **(b)** Galaxy-Zoo **(c)** HAM10000

Figure 5.6: *Accuracy on HateSpeech, Galaxy-Zoo, and HAM10000.* The subfigures report the classification error of the OvA method, softmax method, and baselines for three data sets. OvA (green) is competitive in all cases and is superior for `HateSpeech` and `Galaxy-Zoo`.

we use the negative log-likelihood (NLL) loss as the expert's loss. At test time, it uses the learned approximation of the optimal triage policy to defer to the expert.

Models and Experts For these experiments, we closely follow the setup of Okati et al. (2021). Our base model is a 50-layer residual network (ResNet50) for `Galaxy-Zoo`. For `HateSpeech`, we first embed the tweet's text into a 100-dimensional feature vector using *fasttext* (Joulin et al., 2016). Our base model for `HateSpeech` is the text classification convolutional neural network (CNN) developed by Kim (2014). For the surrogate loss methods, we sample the expert demonstrations from the expert model's predictive distribution. For training the surrogate models, we early stop if the validation loss does not improve for 20 epochs. We train the models using Adam (Kingma and Ba, 2015), a cosine-annealed learning rate, and a warm-up period of 5 epochs. For other baselines, we use the same experimental setup as Okati et al. (2021).

Results Figure 5.6 reports the classification accuracy for each data set and each baseline as a function of the *budget*. The OvA surrogate is competitive among all baselines for the range of *budgets* considered. This shows that the OvA does not sacrifice accuracy for improved confidence calibration. Rather, our model enjoys the benefits of both predictive performance and better uncertainty quantification. OvA's performance is also quite stable across random seeds.

5.3 Ambiguity rejection and the one-vs-all surrogate loss

As we saw in the proof of Theorem 4.2.1 in Chapter 4 (One-vs-All Surrogate Loss for Learning to Defer), the One-vs-All surrogate loss can also consider the ambiguity rejection rule (where the system is ambiguous about the actual label for the input). In this section, we consider whether ambiguity rejection happens in practice or not. Note that ambiguity

rejection in OvA surrogate loss arises from estimation errors, as ambiguity rejection means having two output prediction labels with $\eta_y(\boldsymbol{x}) > 1/2$. This contradicts the rules of probabilities and must not happen theoretically. Nevertheless, we study this as a measure of the fitting property of the OvA surrogate loss. We hypothesize that if ambiguity rejection is prominent in an L2D system returned by a classification algorithm minimizing the OvA surrogate loss, it would mean the system demonstrates poor theoretical properties and may not be consistent in practice. To study this, we train an L2D system on the DirtyMNIST dataset (Mukhoti et al., 2021). DirtyMNIST is a concatenation of regular MNIST images and Ambiguous MNIST images, where the latter are synthetically generated images with multiple *valid* output labels. We simulate an expert with perfect expertise in the image samples with unique output labels and predict a random label for the images with multiple labels. We find that only $\approx 1\%$ images were assigned the ambiguity rejection rule at test time. We further test this L2D system on NotMNIST* images to see the effect of out-of-distribution images. We again observe that the system assigned only 0.4% of the samples to ambiguity criteria. These empirical results suggest that the trained L2D system abides by the laws of probabilities and makes significantly fewer approximation errors.

Summary of the Chapter

In this chapter, we empirically verify that our one-vs-all surrogate loss for learning to defer provides well-calibrated confidence estimates compared with the softmax surrogate method. Furthermore, experiments on three real-world tasks (hate speech detection, galaxy classification, and skin lesion detection) show that our method's accuracy is comparable (and often better) than the softmax surrogate method and other baselines for learning to defer.

* http://yaroslavvb.blogspot.com/2011/09/notmnist-dataset.html

Calibration of Learning to Defer to Multiple Experts

6

Let the road rise up to meet you, say the Irish, and that's what was happening to the intrepid travelers on the Lincoln Highway. It was rising up to meet each and every one of them, whether they were headed east, headed west, or going around in circles. — Amor Towles, The Lincoln Highway

So far in this thesis, we have studied learning to defer systems with one expert. Yet in many critical applications, it is common to involve multiple experts. For example, in healthcare, serious illnesses require the patient to be treated by multiple specialists. In this chapter, we extend the framework of single expert learning to defer framework to allow for multiple experts. We will see that it's easy to extend single expert learning to defer to allow for multiple experts, given the theoretical foundations for learning to defer framework in Chapter 3 (Learning to Defer).

Note on Contributions: This chapter is based on the paper: *On the Calibration of Learning to Defer Multiple Experts*, co-authored with Daniel Barrejón and Eric Nalisnick, and presented at the ICML 2022 Workshop on Human-Machine Collaboration and Teaming held in Baltimore, USA.

6.1 Learning to Defer to Multiple Experts

Same as before, we have $\mathcal{X}$ as the input space, $\mathcal{Y} = [K]$ as the output label space, and distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$. $\mathbf{x}$ and y denote the random variables over $\mathcal{X}$, with $\boldsymbol{x}$ and y as their realizations respectively. Let there be J experts, and denote each expert's prediction space as $\mathcal{M}_j$, and we continue with the assumption that $\mathcal{M}_j = \mathcal{Y}$. We again have m_j as the random variable associated with the jth expert, m_j as its realization. We have $m_j \sim \mathbb{P}_j\left(\mathrm{m}_j|\mathbf{x}, \mathrm{y}\right)$. We assume access to the finite sample $S = \{\boldsymbol{x}_i, y_i, m_{i,1}, \ldots, m_{i,J}\}_{i=1}^{\mathrm{N}}$. Again, the goal is to learn two functions: the classifier and the rejector. In L2D with one expert, the rejector makes a binary decision—to defer or not—but in multi-expert L2D, the rejector also must choose to which expert to assign the instance. Thus, while the classifier takes the same form as the single expert L2D, the rejector is now $r : \mathcal{X} \to \{0, 1, \ldots, J\}$. When $r(\mathbf{x}) = 0$, the classifier makes the decision in a typical way. When $r(\mathbf{x}) = j$, the classifier abstains and defers the decision to the jth expert. Our focus is again on the $0-1$ loss for L2D. Now based on Definition 3.1.1 for the Bayes optimal rejector and classifier for single expert L2D, it's easy to argue for the Bayes optimal rejector and classifier for the multi-experts L2D.

According to Definition 3.1.1, we should defer to the expert only if the probability of the expert making the correct prediction is greater than that of the classifier. This argument can be extended to a multi-expert setting that, among all the J experts, the system should compare the confidence of each of the experts and the classifier, and defer to jth expert if the probability of $j$the expert making the correct prediction

is greater than that of the classifier. Thus, we get the following Bayes optimal classifier $h^*(\boldsymbol{x})$ and rejector $r^*(\boldsymbol{x})$:

$$h^*(\boldsymbol{x}) = \arg\max_{y \in \mathcal{Y}} \mathbb{P}(\mathrm{y} = y | \mathbf{x} = \boldsymbol{x}),$$
$$r^*(\boldsymbol{x}) = \begin{cases} 0 \text{ if } \mathbb{P}(\mathrm{y} = h^*(\boldsymbol{x}) | \mathbf{x} = \boldsymbol{x}) > \mathbb{P}(\mathrm{m}_{j'} = \mathrm{y} | \mathbf{x} = \boldsymbol{x}) \ \ \forall j' \\ \arg\max_{j \in [1,J]} \mathbb{P}(\mathrm{m}_j = \mathrm{y} | \mathbf{x} = \boldsymbol{x}, \mathrm{y} = y) \quad \text{otherwise,} \end{cases} \tag{6.1}$$

where $\mathbb{P}(\mathrm{y}|\mathbf{x} = \boldsymbol{x})$ is the probability of the label under the data generating process, and $\mathbb{P}(\mathrm{m}_j = \mathrm{y}|\mathbf{x} = \boldsymbol{x}, \mathrm{y} = y)$ is the probability that the jth expert is correct.

We next see how to extend the softmax surrogate loss and the OvA surrogate loss for this multi-experts L2D setting. For both the cases, we have the prediction space $\mathcal{Y}^\perp := \mathcal{Y} \cup \{\perp_1, \ldots, \perp_J\}$ where $\perp_j$ denotes the decision to defer to the jth expert. Similar as before, we use the surrogate prediction space $\mathcal{V} = \mathbb{R}^{\mathrm{K+J}}$, and learn a surrogate prediction function $f : \mathcal{X} \to \mathbb{R}^{\mathrm{K+J}}$, which can be considered *pointwise* to give K + J f functions: $f_1, \ldots, f_K, f_{\perp,1}, \ldots f_{\perp,J}$.

Softmax Surrogate Loss The softmax surrogate loss function extended to the multi-expert setting is then given as:

$$\phi_{\mathrm{SM}}\left(f_1, \ldots, f_K, f_{\perp,1}, \ldots, f_{\perp,J}; y, m_1, \ldots, m_J\right) =$$
$$-\log\left(\frac{\exp\{f_y(\boldsymbol{x})\}}{\sum_{y' \in \mathcal{Y}^\perp} \exp\{f_{y'}(\boldsymbol{x})\}}\right)$$
$$-\sum_{j=1}^{J} \mathbb{I}[m_j = y] \ \log\left(\frac{\exp\{f_{\perp,j}(\boldsymbol{x})\}}{\sum_{y' \in \mathcal{Y}^\perp} \exp\{f_{y'}(\boldsymbol{x})\}}\right).$$

As for the decoding function g, we use the same arg max function. The classifier is obtained by taking the maximum over $k \in [1, K]$: $\hat{y} = h(\boldsymbol{x}) = \arg\max_{k \in [1,K]} f_k(\boldsymbol{x})$. The rejection function is similarly formulated as

$$r(\boldsymbol{x}) = \begin{cases} 0 \ \text{ if } \ f_{h(\boldsymbol{x})} > f_{\perp,j'} \ \ \forall j' \in [1, J] \\ \arg\max_{j \in [1,J]} f_{\perp,j}(\boldsymbol{x}) \quad \text{otherwise.} \end{cases}$$

One-vs-All Surrogate Loss OvA surrogate loss too can be straightforwardly extended to the multi-expert setting:

$$\psi_{\mathrm{OvA}}(f_1, \ldots, f_K, f_{\perp,1}, \ldots, f_{\perp,J}; y, m_1, \ldots, m_J) =$$
$$\phi[f_y(\boldsymbol{x})] + \sum_{y' \in \mathcal{Y}, y' \neq y} \phi[-f_{y'}(\boldsymbol{x})] \ + \sum_{j=1}^{J} \phi[-f_{\perp,j}(\boldsymbol{x})]$$
$$+ \sum_{j=1}^{J} \mathbb{I}[m_j = y] \left(\phi[f_{\perp,j}(\boldsymbol{x})] - \phi[-f_{\perp,j}(\boldsymbol{x})]\right)$$

where $\phi : \{-1, 1\} \times \mathbb{R} \mapsto \mathbb{R}_+$ is a binary surrogate loss. We define the classifier and rejector exactly as in the softmax case.

It's now a straightforward exercise to see that both the softmax surrogate and OvA surrogate are *consistent* surrogates for learning to defer w.r.t the canonical $0-1$ loss function.

6.2 Confidence calibration of Expert Confidence

For both types of L2D parameterizations, we are again interested in studying the *confidence calibration* of the system (Dawid, 1982). Specifically, we are interested in the model's ability to estimate $\mathbb{P}(\mathrm{m}_j = \mathrm{y}|\boldsymbol{x})$. This quantity is crucial not only for the system's ability to correctly defer but is also useful for interpretability and safety—to quantify what the model thinks the human knows. We saw in earlier chapters that the softmax surrogate loss can result in poor estimators of this quantity in practice, despite having valid Bayes optimal solutions. In this chapter, we examine each parameterization's behavior in the multi-expert formulation. For the prediction function f^* returned by the L2D classification algorithm working with each of these surrogates, we next describe how to get the estimator of $\mathbb{P}\left(\mathrm{m}_j = \mathrm{y}|\mathbf{x} = \boldsymbol{x}\right)$ for each of the surrogates.

6.2.1 Softmax Parameterization

We define $p_{\perp,j}\left(f^*\left(\boldsymbol{x}\right)\right)$ as

$$p_{\perp,j}\left(f^*\left(\boldsymbol{x}\right)\right) = \frac{\exp\{f^*_{\perp,j}\left(\boldsymbol{x}\right)\}}{\sum_{y' \in \mathcal{Y}^\perp} \exp\{f^*_{y'}\left(\boldsymbol{x}\right)\}}.$$

Then, the suitable inverse *link* function γ_{m_j} function $\gamma_j : \mathbb{R}^{\mathrm{K+J}} \to [0,1]$ to estimate $\mathbb{P}\left(\mathrm{m}_j = \mathrm{y}|\mathbf{x} = \boldsymbol{x}\right)$ is given as

$$\gamma_{\mathrm{m}_j}\left(f^*\left(\boldsymbol{x}\right)\right) = \frac{p_{\perp,j}\left(f^*\left(\boldsymbol{x}\right)\right)}{1 - \sum_{j'=1}^{J} p_{\perp,j'}\left(f^*((\boldsymbol{x})\right)}. \tag{6.2}$$

We derive this in Appendix A.2.3. One can see that due to the denominator involving the quantity $p_{\perp,j}\left(f^*\left(\boldsymbol{x}\right)\right)$ for all experts, there is dependence across the estimators. Same as γ_{m} for the single expert setting, we can clearly see that γ_{m_j} is a 'fragile' estimator, as it is not guaranteed to range in the required $[0,1]$ range. For instance, for $p_{\perp,j}\left(f^*\left(\boldsymbol{x}\right)\right) > 0$, as $\sum_{j'=1}^{J} p_{\perp,j'}\left(f^*\left(\boldsymbol{x}\right)\right)$ approaches one, the estimate of $\mathbb{P}\left(\mathrm{m}_j = \mathrm{y}|\mathbf{x} = \boldsymbol{x}\right)$ will go to infinity.

6.2.2 One-vs-All Parameterization

For the OvA formulation, the inverse *link* function γ^{-1} associated with the strictly proper composite loss ϕ directly gives us the estimate for $\mathbb{P}\left(\mathrm{m}_j = \mathrm{y}|\mathbf{x} = \boldsymbol{x}\right)$, i.e.

$$\gamma_{\mathrm{m}_j}\left(f^*\left(\boldsymbol{x}\right)\right) = \gamma^{-1}\left(f^*\left(\boldsymbol{x}\right)\right).$$

Here, we define,

$$\gamma_{\mathrm{m}_j}\left(f^*\left(\boldsymbol{x}\right)\right) = \gamma^{-1} : \left(f^*_1\left(\boldsymbol{x}\right), \ldots, f^*_{\perp,J}\left(\boldsymbol{x}\right)\right) \mapsto \gamma^{-1}\left(f^*_{\perp,j}\left(\boldsymbol{x}\right)\right).$$

6.3 Experiments

We perform experiments on synthetic data and on the CIFAR-10 (Krizhevsky, 2009) dataset. We use ECE as the measure of confidence calibration. Since the softmax parameterization can result is probability estimates greater than one, we cap confidences at 1.0 to calculate ECE in all experiments. First, we study the effect of gradually increasing the number of experts on the overall calibration of the system. Second, we examine how different experts' behavior affects other expert estimates. Our results suggest that systems trained with the softmax surrogate exhibit degradation in calibration as the number of experts increases. Furthermore, other experts in the committee significantly affect the calibration of other experts.

6.3.1 Datasets and Models

Mixture of Gaussians For the synthetic dataset, we generate a mixture of Gaussians (`MoG`) with 4 clusters. The data is plotted in Figure 6.1. It shows the severe overlap between cluster 2 and cluster 3, and thus these clusters represent where the expert's advice might be required. The other two clusters have a small overlap and can be discriminated by a simple classifier. For the classifier, we use a small feedforward neural network with four layers. We train it using stochastic gradient descent (SGD) with early stopping (look-ahead of 20 epochs).

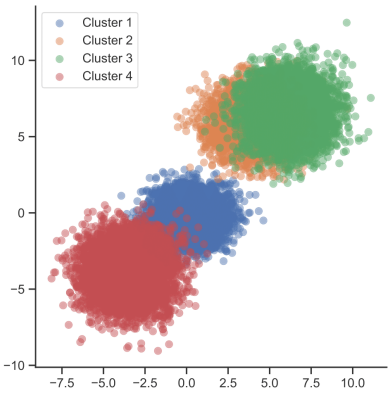


Figure 6.1: *Generated Mixture of Gaussian (MoG) dataset.* The dataset represents the varying level of complexity for the simple classifier with cluster 2 and cluster 3 demonstrating severe overlap conducive to querying the expert for correct prediction. The other two clusters are easy to be learned by the classifier.

`CIFAR-10` For the experiments using CIFAR-10, we use the canonical train-test split (Krizhevsky, 2009). We partition the training split 90% − 10% to form training and validation sets, respectively. We use a wide residual network (Zagoruyko and Komodakis, 2016) to parameterize the $f(\boldsymbol{x})$ functions. We train a 28-layer network using SGD with momentum and a cosine annealing schedule for the learning rate. We again employ early stopping with 20 look-ahead epochs.

We first examine calibration under an increasing number of experts—from 1 to 8. For the `MoG` dataset, the experts are oracles if an instance belongs to either cluster #3 or #4 and predict randomly over all classes otherwise. For the `CIFAR-10` dataset, the experts are an oracle for the first 5 classes and predict randomly over all 10 classes otherwise.

The results are reported in Figure 6.2 (a, b, d, e). Firstly, examine subfigures (b) and (e), which report the system accuracy to ensure both parameterizations are well-performing. We see that the OvA parameterization is slight to moderately superior in all cases. Moving on to the calibration results, the average ECE (across the J deferral functions) is reported in subfigures (a) and (d). We see that the OvA parameterization (orange) is roughly stable w.r.t. expert size, but the softmax's (blue) average ECE tends to increase. This behavior is expected for the softmax according to Equation 6.2. With the addition of more experts, the denominator becomes smaller, leading to overconfident (and degenerate) estimates for $\mathbb{P}(\mathrm{m}_j = \mathrm{y}|\boldsymbol{x})$. However, we do see some cancellation effects with the addition of the second expert in Figure 6.2 (d). This can be

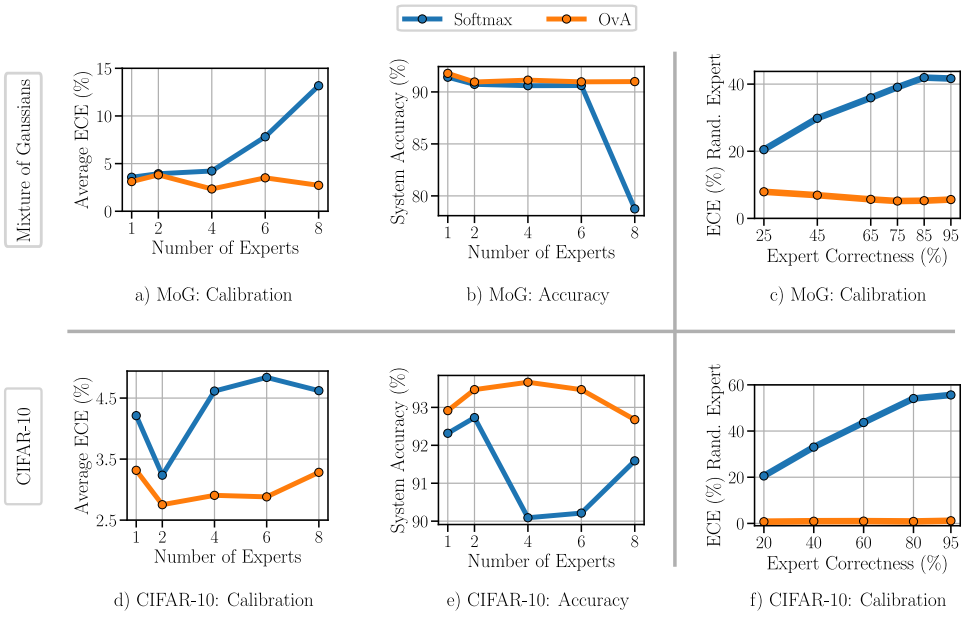


Figure 6.2: *Calibration and System Accuracy on Simulated Data and* `CIFAR-10`. The first column reports ECE under an increasing number of experts, the second column the system accuracy, and the third column the ECE to show the dependence across experts. The top row shows results for the mixture of Gaussians simulation, and the bottom row shows results for `CIFAR-10`.

explained by the fact that adding more experts constrains the confidence allocation to multiple experts (due to the tied nature of the softmax parameterization). But the effect dissipates for 4 or more experts, with the average ECEs continuing to increase.

6.3.2 Expert Dependence

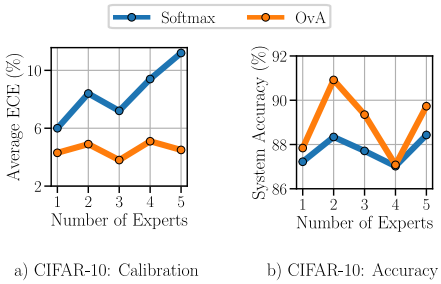
We further aim to assess confidence calibration when there is a gap in expert quality. We simulate four experts with one always being random and the other three have an increased probability of correctness (20% - 95%). For the `MoG` dataset, three experts will increase their probability of being correct on two of the clusters and predict randomly for the other two. For `CIFAR-10`, three experts increase their probability of being correct in the first five classes and predict randomly for the other ten classes. We hypothesize that for the softmax, the calibration of the random expert will increase when the probability of correctness for the other three experts increases due to the tied parameterization. We conjecture that no such dependence will be present in the OvA results.

The results are reported in the third column of Figure 6.2. We see that the ECE for the random expert dramatically increases for both datasets for the softmax parameterization (blue), reaching values above 40%. Yet for OvA (orange), the ECE is nearly flat for both datasets due to its explicit independence across experts. This supports our hypothesis from above that the softmax parameterization will skew per-expert estimation due to its dependencies across experts.

6.3.3 Specialized Experts

For our final experiment, we examine calibration when the experts have non-overlapping expertise.

For `CIFAR-10`, each expert is simulated to be an oracle in two of the ten classes. Figure 6.3 reports the average ECE across experts (a) and the system accuracy (b) as the number of specialized experts increases. For system accuracy, both methods are competitive, with OvA (orange) having a slight edge. For ECE, OvA is again clearly superior by being stable across the number of experts. Thus we see that despite the experts having independent expertise, the softmax parameterization still accumulates calibration errors.



a) CIFAR-10: Calibration b) CIFAR-10: Accuracy

Figure 6.3: *Calibration and System Accuracy on `CIFAR-10` for expert dependence.* Subfigure (a) reports the ECE when an increasing number of specialized experts are incorporated. Subfigure (b) reports the system accuracy under the same conditions.

Summary of the Chapter

We extend the single expert learning to defer framework in the case of multiple experts. We show that the softmax surrogate and the OvA surrogate loss functions can be easily extended to give consistent surrogates w.r.t. $0-1$ loss function for multi-expert learning to defer. We then examine the confidence calibration w.r.t the expert correctness for these methods. Our experiments on simulated and real data suggest that the softmax parameterization of multi-expert L2D exhibits calibration error, especially for an increasing number of experts. The OvA parameterization, on the other hand, is much more stable for multiple experts. We further find that the softmax method causes mis-calibration to propagate between the estimates of expert correctness while the latter's parameterization does not.

Related Work 7

—Welcome, weary travelers, he called across the fire. From where do you hail? — Amor Towles, The Lincoln Highway

This thesis studies learning to defer (L2D) and its confidence calibration properties both in single expert and multi-expert settings. Learning to defer is a promising learning paradigm to foster trust and reliability in artificial intelligence (AI) systems, especially in critical application scenarios. Moreover, it also enables *division of labor* and human-AI complementarity. In this chapter, we survey works in the same vein starting with rejection learning, general human-AI collaboration, complementarity, calibration of confidence estimates, and its role in the trustworthiness and reliability of an AI system.

7.1 Learning to Reject and Learning to Defer

Learning to defer framework can be seen as a generalized version of the learning to reject (a.k.a. rejection learning) problem in machine learning. Rejection learning is a framework in which a classifier can abstain from making a prediction and incurs a fixed cost c, instead of a sample dependent loss ℓ_{exp} for abstaining and querying the expert in L2D. The problem dates back to (at least) Chow (1957)'s work on studying the optimal rejection rule for a fixed c, famously referred to as the Chow rule,

$$h^*(\boldsymbol{x}) = \begin{cases} \text{reject} \;\; \text{if} \;\; \max_{y \in \mathcal{Y}} \mathbb{P}(\mathrm{y} = y | \mathbf{x} = \boldsymbol{x}) \leq 1 - \mathrm{c} \\ \arg\max_{y \in \mathcal{Y}} \mathbb{P}(\mathrm{y} = y | \mathbf{x} = \boldsymbol{x}) \quad \text{otherwise} \,. \end{cases}$$

Chow (1970) further analyzed the trade-off between this rejection rate c and the overall accuracy. Due to its potential in addressing misclassification costs in consequential decision-making applications (where the cost of a classifier making a wrong decision is very high), these initial works have simulated a range of follow-up approaches. Obviously, now, these works' major focus is to design *consistent* algorithms for rejection learning. These works can be broadly categorized into two types: *confidence-based* (Bartlett and Wegkamp, 2008; Yuan and Wegkamp, 2010; Jiang et al., 2018; Grandvalet et al., 2009; Ramaswamy et al., 2018; Ni et al., 2019) and the *classifier-rejector* (Cortes et al., 2016a; Cortes et al., 2016b) approaches. Given the estimates of $\mathbb{P}(\mathrm{y} = y | \mathbf{x} = \boldsymbol{x})$, we trivially have the optimal rejection learning classifier. So the confidence-based methods aim to design *consistent* algorithms by trying to estimate the class probabilities. Building upon the theoretical background we provide in Chapter 2 (Preliminaries), Yuan and Wegkamp (2010) showed that standard binary classification algorithms based on strictly proper composite losses like logistic loss, exponential loss, and least squares loss give the consistent algorithms for rejection learning. However, it's not directly trivial to design such algorithms for general multiclass classification problems. Ramaswamy et al. (2018) extend these results for a general classification case and

provide consistent algorithms for $c \in [0, 1/2]$. In practice, confidence methods consider the prediction uncertainty (Hendrycks and Gimpel, 2017; Gal and Ghahramani, 2016; Lakshminarayanan et al., 2017) and abstain from making a prediction if the classifier's confidence is less than some threshold. Thus, these methods have a simple rejector — a fixed threshold. Classifier-rejector methods, on the other hand, jointly learn two functions, a classifier, and a rejector, usually from two different hypothesis sets. Cortes et al. (2016a) and Cortes et al. (2016b) argue that classifier-rejector methods generalize confidence-based methods and result in more powerful algorithms in practice when working with limited hypothesis class $\mathscr{H}$. First and foremost, they circumvent directly estimating class probabilities which are difficult to computationally estimate (Guo et al., 2017b). The classifier-rejector approach has been well-studied for binary classification and resulted in theoretical guarantees (Cortes et al., 2016a; Cortes et al., 2016b). Ni et al. (2019) was the first to seriously study the multi-class formulation and found that the existing theory was hard to extend to this more general case. Recently, Charoenphakdee et al. (2021) proposed a surrogate loss for rejection learning for general classification, taking inspiration from cost-sensitive learning.

=-1For safety-critical applications, rejection learning is a promising paradigm. However, its learning procedure completely ignores the downstream experts who will eventually make decisions for the rejected samples. For instance, the downstream decision-maker could be just as inaccurate as the classifier. To this end, Madras et al. (2018) introduced an adaptive rejection learning framework termed *learning to defer*. L2D aims to directly model the interaction between the (usually human) decision-makers and the autonomous system. More formally, they build upon the work of Cortes et al. (2016a) to introduce a rejector function $r(\boldsymbol{x})$ to model whether to defer to the expert ($r(\boldsymbol{x}) = 1$) or not ($r(\boldsymbol{x}) = 0$). This results in a mixture-of-experts type loss (as we saw in Equation 3.1 in Chapter 3 (Learning to Defer)). Raghu et al. (2019) approaches the same problem by confidence-based methods by learning a classifier and comparing the expert's certainty and the classifier's certainty, deferring if the latter is lower. Wilder et al. (2020) use the same mixture of experts framework as Madras et al. (2018) and apply the same confidence-based deferral policy as Raghu et al. (2019).

Mozannar and Sontag (2020) study the L2D multi-class classification problem in generality, finding the algorithms proposed by Madras et al. (2018) as *inconsistent*. They also study the limitation of confidence-based approaches (Raghu et al., 2019). Moreover, they propose the first consistent loss for multiclass L2D, asserting the importance of consistent algorithms for learning to defer. Okati et al. (2021) studied the problem of L2D in more general settings than classification and derived results on the optimal deferral policy. Their result is the same as the result we state in Corollary 3.1.2, except their results are derived from the constrained optimization point of view while we look at the problem from the lens of decision theory. They also propose a gradient-based algorithm that learns an approximation of this optimal deferral policy for any predictive model. Keswani et al. (2021) consider learning to defer setting where a system could defer to multiple experts and propose a surrogate loss for this. However, their surrogate loss function is not consistent. Liu et al. (2022) build upon the work of Mozannar and Sontag (2020) to consider uncertainty in

classifier's predictions to decide the decision of deferral. They propose a two-stage algorithm, where first, an ensemble of classifiers is trained, and in the second stage, an L2D system takes as input the predictions from the classifiers as well as their uncertainties to decide whether to let the classifier make the prediction or defer to the expert. In this way, their work bridges the confidence-based and classifier-rejector approaches to learning to defer. Parbhoo et al. (2021) also consider uncertainty in predictions to decide *when to defer* in sequential decision-making applications.

7.2 Human-AI Complementarity and Learning to Defer

In a first of its kind of work in human-AI complementarity, Raghu et al. (2019) consider the problem of algorithmic triage and argue that machine learning systems can achieve significantly higher performance by designing them for both — prediction and triage. They show that such a system can achieve significantly higher performance than just the classifier or the (human) decision-maker making the prediction, even when each one is substantially better on average. Considering the promise of algorithmic triage to achieve better systems and enable human-AI complementarity, several works have explored machine learning systems with decision makers in mind (De et al., 2021; De et al., 2020; Bansal et al., 2021a; Kerrigan et al., 2021; Pandya et al., 2019; Donahue et al., 2022; Vodrahalli et al., 2021). Bansal et al. (2021b) defines complementarity as a scenario whenever the combined human-AI system has a strictly lower loss (or higher performance) than either of those alone. In this direction, contemporary works have used complementarity and collaboration as explicit objectives of the system. For example, Bansal et al. (2021a) optimizes the expected utility of the machine and the expert working together as a team and presents a confidence-based learning algorithm for this. Most recently, Donahue et al. (2022) formally studied the theoretical conditions when human-AI systems can achieve complementarity. They also give impossibility results when a human-AI system can never achieve complementarity. Their theoretical result shows that complementarity is easier to achieve when human and algorithm errors are highly variable across samples. A similar result was also proved in an algorithmic triage setup in Okati et al. (2021). These results explain how a framework like learning to defer naturally enables complementarity and *division of labor*. By knowing what the human knows, the model is free to adapt itself to complement the humans. The model can concentrate on providing the predictions for samples with a lower error rate than the humans, thus enabling complementarity. While these works define complementarity in terms of average error of the whole system, Charusaie et al. (2022) recently defined complementarity in terms of learning predictors that improve on human weaknesses explicitly and design deferral algorithms to this end.

7.3 Confidence calibration and Learning to Defer

Learning to defer is a framework suitable for critical applications, like the medical domain, autonomous driving, etc. Critical applications require not just point predictions but also accurate quantification of their predictive uncertainty. To enable successful human-AI collaboration, factors like transparency, trust, and fairness are crucial as well (Cramer et al., 2008; Kizilcec, 2016; Madras et al., 2018). While well-calibrated uncertainty estimates have a long history to reason about the reliability of a machine learning system (Gal et al., 2016; Bröcker and Smith, 2007; Zadrozny and Elkan, 2002; Zhang et al., 2020), Bhatt et al. (2021), most recently, argue the importance of uncertainty for fairness, transparency, decision-making, and trust in automated AI systems. Most contemporary works in learning to defer have focused on the overall performance of a system. In this thesis, we are the first to investigate the quality of uncertainty estimates provided by the learning to defer system. Our investigation is crucial from two points: 1. Having well-calibrated confidence estimates fosters trust, transparency, and reliability in the learning to defer system. 2. Getting well-calibrated confidence estimates is vital for well-behaved learning to defer system as the Bayes optimal rule for learning to defer relies on these estimates. For our investigation, we build upon the notion of confidence calibration (Guo et al., 2017a) — a well-studied measure of the quality of confidence estimates in machine learning literature. We are unaware of other works investigating the quality of the confidence estimates in learning to defer.

Summary of the Chapter

In this chapter, we discuss related works in learning to defer in the context of human-AI complementarity, rejection learning, and confidence calibration.

8 Discussion

Many years before, Abacus had come to the conclusion that the greatest of heroic stories have the shape of a diamond on its side. Beginning at a fine point, the life of the hero expands outward through youth as he begins to establish his strengths and fallibilities, his friendships and enmities. Proceeding into the world, he pursues exploits in grand company, accumulating honors and accolades. But at some untold moment, the two rays that define the outer limits of this widening world of hale companions and worthy adventures simultaneously turn a corner and begin to converge. The terrain our hero travels, the cast of characters he meets, the sense of purpose that has long propelled him forward all begin to narrow—to narrow toward that fixed and inexorable point that defines his fate.

— Amor Towles, The Lincoln Highway

In consequential human-AI applications, it is vital that the system be reliable and trusted by the human. A well-calibrated system—a good forecaster—can help engender this trust. The focus of this thesis was to study the confidence calibration properties of the learning to defer systems. We studied the surrogate loss methods that result in *consistent* algorithms for learning to defer. We find that the previously (and the only) consistent surrogate loss method results in *degenerate* confidence estimates for the expert correctness. We remedy this problem by proposing an alternative surrogate loss for learning to defer that is also consistent. We also extend the surrogate loss methods to provide consistent algorithms for learning to defer in the case of multiple experts. We conclude this thesis by discussing crucial points related to the contributions of this thesis and highlighting possible future directions moving forward.

8.1 Discussion

Applicability of Post-hoc confidence calibration techniques Mis-calibration (in particular, overconfidence) is common in modern-day machine learning systems (Guo et al., 2017a; Ovadia et al., 2019). There are a range of post-hoc techniques designed to fix mis-calibration in classifiers, e.g. *temperature scaling* (Guo et al., 2017a), *Dirichlet calibration* (Kull et al., 2019), *top-label calibration* (Gupta and Ramdas, 2022). These techniques employ a calibration map (Vaicenavicius et al., 2019): a simple transformation that is applied to the confidence estimates to re-calibrate them. Such a map is fitted on a held-out validation set using some goodness-of-fit measure, e.g. log-likelihood. Our experiments suggest that both the softmax surrogate and one-vs-all surrogate method have comparable ECEs (one-vs-all is slightly better; refer Table 5.1b). One can also apply these post-hoc calibration techniques to further confidence calibrate them for both methods. However, due to the range problem in expert's correctness confidence estimates provided by the softmax surrogate method and interdependence of its $\boldsymbol{f}$ functions, we do not know of a general procedure for defining and fitting a calibration map for the L2D setting. One can fit such a calibration map to the one-vs-all method though (and we did that in Section 5.1.1 in Chapter 5 (Experiments)).

Distribution calibration and the one-vs-all surrogate This thesis proposes a one-vs-all surrogate loss function for learning to defer. We show that the learning to defer algorithm minimizing the proposed loss function results in well-calibrated confidence estimates. Yet, one major downside of the proposed one-vs-all formulation is that we can no longer compute normalized probabilities for all classes. Rather, we only estimate the probability of each output label independently. Hence, we can evaluate the confidence calibration of the OvA classifier but not its distribution calibration. Distribution calibration aims to calibrate the whole prediction distribution — all the samples that have the predictive distribution $p \in \Delta^{K-1}$ (Δ^{K-1} is a K-dimensional probability simplex), the true distribution over output labels should be p. However, distribution calibration is nearly impossible to achieve in practice (Zhao et al., 2021a). Thus, we argue that the OvA formulation is a worthwhile trade-off for having an appropriate estimator for $\mathbb{P}(\mathrm{m} = \mathrm{y}|\boldsymbol{x})$ and to achieve confidence calibration.

Theory vs Practice w.r.t Consistency Consistency of a learning algorithm means that the solution returned by the algorithm agrees with the Bayes optimal solution. In this thesis, we study the consistent surrogate loss for learning to defer and propose another surrogate loss that is also provably consistent. Both surrogate losses have theoretical guarantees that their minimizer over all measurable functions agrees with the Bayes solution. However, in practice, we work with a fixed hypothesis class. So, there is a gap in theory and practice.

Nevertheless, we empirically verify the property of the proposed one-vs-all surrogate loss to see that it results in fewer estimation errors and abiding by the laws of probability (refer to Section in Chapter 5 (Experiments)). However, it's an empirical observation. In the future, it would be interesting to establish theoretical results to compare the consistency property of both the surrogates in a limited hypothesis size and limited sample size setting. Unfortunately, we didn't find time while writing this thesis to work on this.

Confidence calibration in practice The major focus of this thesis is to study the calibration of the confidence estimates provided by the consistent surrogate methods for learning to defer. One quantity of particular interest in this thesis is the probability of an expert's correctness. It is crucial for two reasons: 1) The Bayes optimal decision depends on it, and 2) it fosters faithfulness and transparency in safety-critical applications. By correctly estimating the expert's confidence in making the right decision, one can enable trust and reliability in the system. This thesis and the prior works make a simplifying assumption that the expert's predictions are coming from $m \sim \mathbb{P}(\mathrm{m}|\mathbf{x}, \mathrm{y})$ (where our usual notation holds). Yet in practical situations, expert(s) might have access to extra information $z \sim \mathrm{z}$ that is difficult to computationally model or acquire, and expert predictions are obtained as $m \sim \mathbb{P}(\mathrm{m}|\mathbf{x}, \mathrm{y}, \mathrm{z})$. One can argue that this would make estimating an expert's true uncertainties difficult. Thus, it is an infeasible problem where we want to estimate the true (or calibrated) uncertainties for Bayes optimality and transparency. But, one can't computationally estimate them due to this extra information assumption. Fortunately, estimating true uncertainties in learning to defer is not necessary to achieve consistent algorithms. Thus, in the future, it would

be interesting to define other notions of calibration of the confidence estimates that do not aim to estimate the true uncertainties but still enable faithfulness, transparency, and trust in general decision-making. One such notion is that of *decision calibration* proposed by Zhao et al. (2021b).

Learning to defer with multiple experts In this thesis, we, for the first time, provide consistent algorithms for learning to defer in the case of multiple experts by straightforwardly extending the consistent surrogate methods for learning to defer to a single expert. The resulting algorithms can choose from a set of multiple experts whom to defer. However, in many practical scenarios, it is common for many experts *together* to decide the outcome. Our framework is restricted because it does not allow synthesizing decisions from multiple experts and just defers to one expert who has more chance of providing the correct predictions. Many practical applications benefit from asking multiple experts, mainly due to the different expertise of each expert. In the future, it would be helpful to provide consistent algorithms that allow synthesizing predictions from multiple experts and establish synergies among experts who might be operating under different assumptions and have access to different information or expertise. We are not aware of any works in this direction.

A Appendix

A.1 Proofs

A.1.1 Proof of Lemma 2.2.1

Proof Credits: Worked out together with Tigernach Feehilly (Utrecht University) and Simone Astarita (University of Amsterdam) in the Mastermath Machine Learning Theory class.

Given the function $h^*_{\mathcal{D}}$ as defined in the statement of the lemma, and any other function h, we aim to show that $\mathcal{R}^{\ell_{0-1}}_{\mathcal{D}}[h^*_{\mathcal{D}}] \leq \mathcal{R}^{\ell_{0-1}}_{\mathcal{D}}[h]$. We have

$$\mathcal{R}^{\ell_{0-1}}_{\mathcal{D}}[h] = \mathbb{E}_{(\boldsymbol{x},y)\sim(\mathbf{x},\mathrm{y})}\left[\mathbb{I}\left[y \neq h\left(\boldsymbol{x}\right)\right]\right] = \mathbb{E}_{\boldsymbol{x}\sim\mathbf{x}}\left[\mathbb{E}_{y\sim\mathrm{y}|\mathbf{x}=\boldsymbol{x}}\left[\mathbb{I}\left[y \neq h(x)\right]\right]\right] = \mathbb{E}_{\boldsymbol{x}\sim\mathbf{x}}\left[\mathbb{P}\left(y \neq h\left(\boldsymbol{x}\right)|\mathbf{x}=\boldsymbol{x}\right)\right]. \quad \text{(A.1)}$$

We now compare $\mathbb{P}\left(y \neq h\left(\boldsymbol{x}\right)|\mathbf{x}=\boldsymbol{x}\right)$ and $\mathbb{P}\left(y \neq h^*_{\mathcal{D}}\left(\boldsymbol{x}\right)|\mathbf{x}=\boldsymbol{x}\right)$. Denoting $\eta\left(\boldsymbol{x}\right) = \mathbb{P}\left(\mathrm{y}=1|\mathbf{x}=\boldsymbol{x}\right)$, we have

$$\begin{aligned}\mathbb{P}\left(y \neq h^*_{\mathcal{D}}\left(\boldsymbol{x}\right)|\mathbf{x}=\boldsymbol{x}\right) &= \mathbb{I}[\eta(\boldsymbol{x}) \geq 1/2]\cdot\mathbb{P}(\mathrm{y}=-1|\mathbf{x}=\boldsymbol{x}) + \mathbb{I}[\eta(\boldsymbol{x}) < 1/2]\cdot\mathbb{P}(\mathrm{y}=1|\mathbf{x}=\boldsymbol{x})\\ &= \mathbb{I}[\eta(\boldsymbol{x}) \geq 1/2]\cdot(1-\eta\left(\boldsymbol{x}\right)) + \mathbb{I}[\eta(\boldsymbol{x}) < 1/2]\cdot\eta\left(\boldsymbol{x}\right)\\ &= \min\{\eta\left(\boldsymbol{x}\right), 1-\eta\left(\boldsymbol{x}\right)\}.\end{aligned}$$

And

$$\begin{aligned}\mathbb{P}\left(y \neq h\left(\boldsymbol{x}\right)|\mathbf{x}=\boldsymbol{x}\right) &= \mathbb{P}(h(\boldsymbol{x})=1|\mathbf{x}=\boldsymbol{x})\cdot\mathbb{P}(\mathrm{y}=-1|\mathbf{x}=\boldsymbol{x}) + \mathbb{P}(h(\boldsymbol{x})=-1|\mathbf{x}=\boldsymbol{x})\cdot\mathbb{P}(\mathrm{y}=1|\mathbf{x}=\boldsymbol{x})\\ &= \mathbb{P}(h(\boldsymbol{x})=1|\mathbf{x}=\boldsymbol{x})\cdot(1-\eta(\boldsymbol{x})) + \mathbb{P}(h(\boldsymbol{x})=-1|\mathbf{x}=\boldsymbol{x})\cdot\eta(\boldsymbol{x})\\ &\geq \mathbb{P}\left(h\left(\boldsymbol{x}\right)=1|\mathbf{x}=\boldsymbol{x}\right)\cdot\min\{\eta\left(\boldsymbol{x}\right), 1-\eta\left(\boldsymbol{x}\right)\} + \mathbb{P}\left(h\left(\boldsymbol{x}\right)=-1|\mathbf{x}=\boldsymbol{x}\right)\cdot\min\{\eta\left(\boldsymbol{x}\right), 1-\eta\left(\boldsymbol{x}\right)\}\\ &= \min\{\eta\left(\boldsymbol{x}\right), 1-\eta\left(\boldsymbol{x}\right)\}.\end{aligned}$$

Thus, we have $\mathbb{P}\left(y \neq h\left(\boldsymbol{x}\right)|\mathbf{x}=\boldsymbol{x}\right) \geq \mathbb{P}\left(y \neq h^*_{\mathcal{D}}\left(\boldsymbol{x}\right)|\mathbf{x}=\boldsymbol{x}\right) \ \forall \boldsymbol{x} \in \mathbf{x}$. Combined with Equation A.1 and considering that expectation is a linear operator, we now conclude that

$$\mathcal{R}^{\ell_{0-1}}_{\mathcal{D}}[h^*_{\mathcal{D}}] \leq \mathcal{R}^{\ell_{0-1}}_{\mathcal{D}}[h].$$

□

A.1.2 Proof of Theorem 3.1.1

The loss $L(y, \hat{y}, \hat{r}, m)$ in Equation 3.1 is

$$L(y, \hat{y}, \hat{r}, m) = \begin{cases} \ell_{\text{exp}}(y, m), & \text{if } \hat{r} = 1 \\ \ell_{\text{clf}}(y, \hat{y}), & \text{if } \hat{r} = 0. \end{cases}$$

The goal in the general classification problem is to learn a function $h : \mathcal{X} \to \mathcal{Y}^{\perp}$, where $\hat{y} = h(\boldsymbol{x})$. In what follows, we consider h as first modeling a probabilistic decision rule, i.e. $h(\boldsymbol{x}) = g(\boldsymbol{\delta}(\hat{y}|\boldsymbol{x}))$ where g is some simple mapping rule from the probability simplex $\boldsymbol{\delta}(\hat{y}|\boldsymbol{x}) \in \Delta^{\text{K}}$ to $\hat{\mathcal{Y}}$. Here, $\boldsymbol{\delta}(\hat{y}|\boldsymbol{x}) = [\delta(\hat{y} = 1|\boldsymbol{x}), \delta(\hat{y} = 2|\boldsymbol{x}), \ldots, \delta(\hat{y} = \text{K}|\boldsymbol{x}), \delta(\hat{y} = \text{K}+1|\boldsymbol{x})]$, where $\delta(\hat{y} = i|\boldsymbol{x})$ is the probability of making the prediction $\hat{y} = i$ for $\boldsymbol{x} \sim \mathbf{x}$. Note that we are using K + 1 and $\perp$ interchangeably for the deferral decision. Thus, our goal is now to learn the optimal $\boldsymbol{\delta}(\hat{y}|\boldsymbol{x})$. To assess the quality of $\boldsymbol{\delta}(\hat{y}|\boldsymbol{x})$, we use the standard expected-risk (*L-risk*), as written below:

$$\mathcal{R}^{L}_{\mathcal{D}}[\boldsymbol{\delta}(\hat{y}|\boldsymbol{x})] = \sum_{i=1}^{\text{K}} \sum_{j=1}^{\text{K}+1} \int_{x} \delta(\hat{y} = j|\boldsymbol{x}) \, \mathbb{P}(\text{y} = i) \, \ell(\text{y} = i, \hat{y} = j) \, \mathbb{P}(\boldsymbol{x}|\text{y} = i) \, d\boldsymbol{x},$$

where $\ell : (y, \hat{y}) \mapsto \mathbb{R}_{+}$ is a general loss function, i runs over the output label space, j runs over the output prediction label space. We can further expand it based on the definition of the loss function L above as follows:

$$\mathcal{R}^{L}_{\mathcal{D}}[\boldsymbol{\delta}(\hat{y}|\boldsymbol{x})] = \sum_{i=1}^{\text{K}} \int_{x} \left(\sum_{j=1}^{\text{K}} \delta\left(\hat{y}_j|\boldsymbol{x}\right) \ell_{\text{clf}}(i, j) + \sum_{m=1}^{\text{K}} \delta(\hat{y}_{\text{K}+1}|\boldsymbol{x}) \, \ell_{\text{exp}}(i, m) \, \mathbb{P}(m|\boldsymbol{x}, \text{y} = i) \right) \mathbb{P}(\text{y} = i) \, \mathbb{P}(\boldsymbol{x}|\text{y} = i) \, d\boldsymbol{x},$$

where we have used shorthand $\delta\left(\hat{y}_j|\boldsymbol{x}\right)$ to denote $\delta(\hat{y} = j|\boldsymbol{x})$, and $\mathbb{P}(m|\boldsymbol{x}, \text{y} = i) = \mathbb{P}(\text{m} = m|\boldsymbol{x}, \text{y} = i)$. Next, we define

$$w_{i,j} = \ell_{\text{clf}}(i, j) \quad \text{and}$$
$$w_{i,\perp} = \sum_{m=1}^{\text{K}} \ell_{\text{exp}}(i, m) \, \mathbb{P}(m|\boldsymbol{x}, \text{y} = i).$$

Here, $w_{i,j}$ denotes the cost associated with making a decision $\hat{y} = j$ when the actual output label is $\text{y} = i$. Similarly, $w_{i,\perp}$ denotes the cost associated with deferring to the expert when the actual output label is $\text{y} = i$. Plugging in these quantities, We rewrite $\mathcal{R}^{L}_{\mathcal{D}}[\boldsymbol{\delta}(\hat{y}|\boldsymbol{x})]$ below:

$$\mathcal{R}^{L}_{\mathcal{D}}[\boldsymbol{\delta}(\hat{y}|\boldsymbol{x})] = \sum_{i=1}^{\text{K}} \int_{x} \left(\sum_{j=1}^{\text{K}} \delta\left(\hat{y}_j|\boldsymbol{x}\right) w_{i,j} + \delta(\hat{y}_{\text{K}+1}|\boldsymbol{x}) \, w_{i,\perp} \right) \mathbb{P}(\text{y} = i) \, \mathbb{P}(\boldsymbol{x}|\text{y} = i) \, d\boldsymbol{x},$$

Having written the expected risk in the above form, we can use the machinery from Chow (1957), and decompose $\mathcal{R}^{L}_{\mathcal{D}}[\boldsymbol{\delta}(\hat{y}|\boldsymbol{x})]$ as follows:

$$\mathcal{R}^{L}_{\mathcal{D}} = \mathcal{R}^{L,\perp}_{\mathcal{D}} + \mathcal{R}^{L,\boldsymbol{\delta}}_{\mathcal{D}}$$

where

$$\mathcal{R}_{\mathcal{D}}^{L,\perp} = \sum_{i=1}^{\mathrm{K}} \mathbb{P}\left(\mathrm{y} = i\right) \cdot w_{i,\perp}$$
$$\mathcal{R}_{\mathcal{D}}^{L,\boldsymbol{\delta}} = \int_{\boldsymbol{x}} \sum_{j=1}^{\mathrm{K}+1} \delta\left(\hat{y}_j|\boldsymbol{x}\right) \cdot Z_j\left(\boldsymbol{x}\right) d\boldsymbol{x}, \quad \text{and}$$
$$Z_j\left(\boldsymbol{x}\right) = \sum_{i=1}^{\mathrm{K}} \left(w_{i,j} - w_{i,\perp}\right) \mathbb{P}\left(\boldsymbol{x}\right) \mathbb{P}\left(\mathrm{y} = i|\boldsymbol{x}\right), j \in \{1, 2, \ldots, \mathrm{K}, \perp\}.$$

To simplify the notation we have dropped the dependence on $\boldsymbol{\delta}$ in the above quantities. We can observe that we have no control over $\mathcal{R}_{\mathcal{D}}^{L,\perp}$. However, we can control $\mathcal{R}_{\mathcal{D}}^{L,\boldsymbol{\delta}}$ by controlling the decision rule $\boldsymbol{\delta}$. We have $Z_\perp(\boldsymbol{x}) = 0$, and also it holds that

$$\mathcal{R}_{\mathcal{D}}^{L,\boldsymbol{\delta}} \geq \int_{\boldsymbol{x}} \min_j \left[Z_j\left(\boldsymbol{x}\right)\right] d\boldsymbol{x},$$

where the equality holds iff $\delta\left(\hat{y}_k|\boldsymbol{x}\right) = 1.0$ for $k = \arg\min_j Z_j\left(\boldsymbol{x}\right)$. Thus, the optimal rule is to deterministically (i.e. with confidence 1.0) choose $k \in \{1, 2, \ldots, \mathrm{K}, \perp\}$ with the minimum $Z_j\left(\boldsymbol{x}\right)$.

This means that choosing j for which the $Z_j(x)$ is the smallest minimizes the expected risk. Given that $Z_\perp = 0$, this means that the classifier predicts when the minimum $Z_j\left(\boldsymbol{x}\right)$ is negative. Thus, deferral happens when $Z_j(x)$ is positive for all j, i.e., we can define the the optimal classifier $h^*(x)$ and the rejector $r^*(x)$ as below:

$$h^*(\boldsymbol{x}) = \underset{j \in \{1,\ldots,\mathrm{K}\}}{\arg\min}\, Z_j(\boldsymbol{x})$$
$$r^*(\boldsymbol{x}) = \mathbb{I}\left[Z_j(\boldsymbol{x}) \geq 0; \forall j \in \{1, \ldots, \mathrm{K}\}\right].$$

□

A.1.3 Proof of Corollary 3.1.2

Theorem 3.1.1 says that the optimal rejector can be characterized as

$$r^*(\boldsymbol{x}) = \mathbb{I}\left[Z_j(\boldsymbol{x}) \geq 0; \forall j \in \{1, \ldots, K\}\right].$$

By the definition of $Z_j\left(\boldsymbol{x}\right)$, the optimal rejection rule (or rejector) can be written as:

$$r^*(\boldsymbol{x}) = \begin{cases} 1 \text{ if } \sum_{i=1}^{\mathrm{K}} w_{i,j} \mathbb{P}\left(\mathrm{y} = i|\boldsymbol{x}\right) \geq \sum_{i=1}^{\mathrm{K}} w_{i,\perp} \mathbb{P}\left(\mathrm{y} = i|\boldsymbol{x}\right) \ \forall j \in \{1, \ldots, K\} \\ 0 \ \text{ otherwise.} \end{cases}$$

Or equivalently,

$$r^*\left(\boldsymbol{x}\right) = \begin{cases} 1 \text{ if } \mathbb{E}_{\mathrm{y}|\mathbf{x}}\left[\ell_{\mathrm{clf}}\left(y, j\right)\right] \geq \mathbb{E}_{\mathrm{y}|\mathbf{x}}\left[\mathbb{E}_{\mathrm{m}|\mathbf{x},\mathrm{y}}\left[\ell_{\mathrm{exp}}\left(m, j\right)\right]\right] \ \forall j \in \{1, \ldots, K\} \\ 0 \text{ otherwise.} \end{cases}$$

□

A.1.4 Proof of Theorem 4.2.1

For K + 1 surrogate prediction function $f_1(\mathbf{x}), \ldots, f_K(\mathbf{x}), f_\perp(\mathbf{x})$, and the binary classification surrogate $\phi : \{-1, 1\} \times \mathbb{R} \to \mathbb{R}_+$, the proposed one-vs-all (OvA) surrogate has the following pointwise form:

$$\begin{aligned}&\psi_{\text{OvA}}(f_1, \ldots, f_K, g_\perp; y, m) = \\ &\quad \phi[f_y(\boldsymbol{x})] \; + \; \phi[-f_\perp(\boldsymbol{x})] \; + \sum_{y' \in \mathcal{Y}, y' \neq y} \phi[-f_{y'}(\boldsymbol{x})] \; + \mathbb{I}[m = y]\,(\phi[f_\perp(\boldsymbol{x})] - \phi[-f_\perp(\boldsymbol{x})])\,.\end{aligned}$$

We consider the pointwise *inner* ψ_{OvA}*-risk* for some $\mathbf{x} = \boldsymbol{x}$ written as follows:

$$\mathbb{E}_{\mathrm{y}|\mathbf{x}=\boldsymbol{x}} \mathbb{E}_{\mathrm{m}|\mathbf{x}=\boldsymbol{x},\mathrm{y}=y} \psi_{\text{OvA}}(f_1, \ldots, f_K, f_\perp; y, m). \tag{A.2}$$

We simplify the *inner* ψ_{OvA}*-risk* by expanding both the expectations below:

$$\begin{aligned}&\mathbb{E}_{\mathrm{y}|\mathbf{x}=\boldsymbol{x}} \mathbb{E}_{\mathrm{m}|\mathbf{x}=\boldsymbol{x},\mathrm{y}=y} \psi_{\text{OvA}}(f_1, \ldots, f_K, f_\perp; y, m) = \\ &\quad \mathbb{E}_{\mathrm{y}|\mathbf{x}=\boldsymbol{x}} \Bigg[\phi(f_y(\boldsymbol{x})) + \phi(-f_\perp(\boldsymbol{x})) + \sum_{y' \in \mathcal{Y}, y' \neq y} \phi(-f_{y'}(\boldsymbol{x})) \\ &\qquad + \sum_{m \in \mathcal{Y}} \mathbb{P}(\mathrm{m} = m | \mathbf{x} = \boldsymbol{x}, \mathrm{y} = y) \mathbb{I}[m = y] [\phi(f_\perp(\boldsymbol{x})) - \phi(-f_\perp(\boldsymbol{x}))] \Bigg].\end{aligned}$$

Expanding the outer expectation, and $\eta_y(\boldsymbol{x}) = \mathbb{P}(\mathrm{y} = y | \mathbf{x} = \boldsymbol{x})$, we have:

$$\begin{aligned}&\mathbb{E}_{\mathrm{y}|\mathbf{x}=\boldsymbol{x}} \mathbb{E}_{\mathrm{m}|\mathbf{x}=\boldsymbol{x},\mathrm{y}=y} \psi_{\text{OvA}}(f_1, \ldots, f_K, f_\perp; \boldsymbol{x}, y, m) = \\ &\quad \sum_{y \in \mathcal{Y}} \eta_y(\boldsymbol{x}) \left[\phi\left(f_y(\boldsymbol{x})\right) + \sum_{y' \in \mathcal{Y}, y' \neq y} \phi\left(-f_{y'}(\boldsymbol{x})\right) \right] + \phi\left(-f_\perp(\boldsymbol{x})\right) \\ &\quad + \sum_{y \in \mathcal{Y}} \eta_y(\boldsymbol{x}) \sum_{m \in \mathcal{Y}} \mathbb{P}(\mathrm{m} = m | \mathbf{x} = \boldsymbol{x}, \mathrm{y} = y) \mathbb{I}[m = y] [\phi(f_\perp(\boldsymbol{x})) - \phi(-f_\perp(\boldsymbol{x}))] \\ &\quad = \sum_{y \in \mathcal{Y}} \eta_y(\boldsymbol{x}) \left[\phi\left(f_y(\boldsymbol{x})\right) + \sum_{y' \in \mathcal{Y}, y' \neq y} \phi\left(-f_{y'}(\boldsymbol{x})\right) \right] + \phi\left(-f_\perp(\boldsymbol{x})\right) \\ &\quad + \sum_{y \in \mathcal{Y}} \eta_y(\boldsymbol{x}) \sum_{m \in \mathcal{Y}} \mathbb{P}(\mathrm{m} = y | \mathbf{x} = \boldsymbol{x}, \mathrm{y} = y) [\phi(f_\perp(\boldsymbol{x})) - \phi(-f_\perp(\boldsymbol{x}))] \\ &\quad = \sum_{y \in \mathcal{Y}} \eta_y(\boldsymbol{x}) \left[\phi\left(f_y(\boldsymbol{x})\right) + \sum_{y' \in \mathcal{Y}, y' \neq y} \phi\left(-f_{y'}(\boldsymbol{x})\right) \right] + \phi\left(-f_\perp(\boldsymbol{x})\right) \\ &\quad + \underbrace{\sum_{y \in \mathcal{Y}} \eta_y(\boldsymbol{x}) \sum_{m \in \mathcal{Y}} \mathbb{P}(\mathrm{m} = y | \mathbf{x} = \boldsymbol{x}, \mathrm{y} = y)}_{\mathbb{P}(\mathrm{y} = \mathrm{m} | \mathbf{x} = \boldsymbol{x})} [\phi(f_\perp(\boldsymbol{x})) - \phi(-f_\perp(\boldsymbol{x}))] \\ &\quad = \sum_{y \in \mathcal{Y}} \eta_y(\boldsymbol{x}) \left[\phi\left(f_y(\boldsymbol{x})\right) + \sum_{y' \in \mathcal{Y}, y' \neq y} \phi\left(-f_{y'}(\boldsymbol{x})\right) \right] + \phi\left(-f_\perp(\boldsymbol{x})\right) \\ &\quad + \mathbb{P}(\mathrm{y} = \mathrm{m} | \mathbf{x} = \boldsymbol{x}) [\phi(f_\perp(\boldsymbol{x})) - \phi(-f_\perp(\boldsymbol{x}))] \\ &\quad = \sum_{y \in \mathcal{Y}} \eta_y(\boldsymbol{x}) \left[\phi\left(f_y(\boldsymbol{x})\right) + \sum_{y' \in \mathcal{Y}, y' \neq y} \phi\left(-f_{y'}(\boldsymbol{x})\right) \right] + \mathbb{P}(\mathrm{y} = \mathrm{m} | \mathbf{x} = \boldsymbol{x}) \phi(f_\perp(\boldsymbol{x})) \\ &\quad + (1 - \mathbb{P}(\mathrm{y} = \mathrm{m} | \mathbf{x} = \boldsymbol{x})) \phi(-f_\perp(\boldsymbol{x}))\,.\end{aligned}$$

Using the usual notation $p_{\mathrm{m}}(\boldsymbol{x}) = p(\mathrm{y} = \mathrm{m}|\mathbf{x} = \boldsymbol{x})$, we can further rewrite the above equation in the following form,

$$\mathbb{E}_{\mathrm{y}|\mathbf{x}=\boldsymbol{x}}\mathbb{E}_{\mathrm{m}|\mathbf{x}=\boldsymbol{x},\mathrm{y}=y}\psi_{\mathrm{OvA}}(f_1,\ldots,f_{\mathrm{K}},f_\perp;y,m) = \sum_{y\in\mathcal{Y}}\left[\eta_y(\boldsymbol{x})\,\phi\left(f_y(\boldsymbol{x})\right) + \left(1-\eta_y(\boldsymbol{x})\right)\phi\left(-f_y(\boldsymbol{x})\right)\right] + p_{\mathrm{m}}(\boldsymbol{x})\,\phi\left(f_\perp(\boldsymbol{x})\right) + (1-p_{\mathrm{m}})\,\phi\left(-f_\perp(\boldsymbol{x})\right). \tag{A.3}$$

The above expression says that we have K + 1 binary classification problems where the *inner* ϕ-*risk* for the i^{th} binary classification problem is given as $\eta_y(\boldsymbol{x})\,\phi\left(f_y(\boldsymbol{x})\right) + \left(1-\eta_y(\boldsymbol{x})\right)\phi\left(-f_y(\boldsymbol{x})\right)$ when $i \in [K]$ and $p_{\mathrm{m}}(\boldsymbol{x})\,\phi\left(f_\perp(\boldsymbol{x})\right) + (1-p_{\mathrm{m}}(\boldsymbol{x}))\,\phi\left(-f_\perp(x)\right)$ when $i \in \{\mathrm{K}+1\}$. This means that the pointwise minimizer of the inner ψ_{OvA} *-risk* can be analyzed in terms of the pointwise minimizer of the *inner* ϕ-*risk* for each of the K + 1 binary classification problems we have. Denote the minimizer of pointwise *inner* ψ_{OvA}-*risk* as $\boldsymbol{f}^*$, then the above decomposition means f_i^* corresponds to the minimizer of the *inner* ϕ-*risk* for the ith binary classification problem.

We know that the Bayes solution for the binary classification problem is $\operatorname{sign}\left(\eta(\boldsymbol{x}) - \frac{1}{2}\right)$ where $\eta(\boldsymbol{x})$ denotes $p(\mathrm{y} = 1|\mathbf{x} = \boldsymbol{x})$. Now when the binary surrogate loss ϕ is a strictly proper composite loss for binary classification, by the property of strictly proper composite losses, we have $\operatorname{sign}(f_y^*(\boldsymbol{x}))$ would agree with the Bayes solution of the Binary classification (refer Equation 2.9 Chapter 2 (Preliminaries)), i.e. $f_y^*(\boldsymbol{x}) > 0$ if $\eta_y(\boldsymbol{x}) > \frac{1}{2}$. And similarly $f_\perp^*(\boldsymbol{x}) > 0$ if $p_{\mathrm{m}}(\boldsymbol{x}) > \frac{1}{2}$. Furthermore, we have the existence of a continuous and increasing inverse link function γ^{-1} for the binary surrogate ϕ with the property that $\gamma^{-1}\left(f_y^*(\boldsymbol{x})\right)$ would converge to $\eta_y(\boldsymbol{x})$. Similarly, $\gamma^{-1}\left(f_\perp^*(\boldsymbol{x})\right)$ would converge to $p_{\mathrm{m}}(\boldsymbol{x})$ (refer Equation 2.10 Chapter 2 (Preliminaries)).

Using the above information, we can establish the Bayes optimal decision for this minimizer $\boldsymbol{f}^*$ using following cases.

Case 1: If we have $f_y^*(\boldsymbol{x}) > 0$ and $f_\perp^*(\boldsymbol{x}) > 0$ for some $y \in \mathcal{Y}$. Note that we cannot have $y \neq y'$ both belonging to [K] such that $f_y^*(\boldsymbol{x}) > 0$ and $f_{y'}^*(\boldsymbol{x}) > 0$. Because this would imply $\eta_y(\boldsymbol{x}) > \frac{1}{2}$ and $\eta_{y'}(\boldsymbol{x}) > \frac{1}{2}$ which contradicts the rules of probabilities. Thus, theoretically, only one such $y \in \mathcal{Y}$ is possible such that $f_y^*(\boldsymbol{x}) > 0$. And if we take the prediction for our L2D problem as $\arg\max_{k\in[\mathrm{K}+1]} f_k^*(\boldsymbol{x})$, our prediction would correspond to the Bayes optimal decision, i.e. if

$$\begin{aligned} & f_y^*(\boldsymbol{x}) < f_\perp^*(\boldsymbol{x}) \quad \forall y \in \mathcal{Y} \\ \implies & \gamma^{-1}\left(f_y^*(\boldsymbol{x})\right) < \gamma^{-1}\left(f_\perp^*(\boldsymbol{x})\right) \quad \forall y \in \mathcal{Y} \\ \implies & \eta_y(\boldsymbol{x}) < p_{\mathrm{m}}(\boldsymbol{x}) \quad \forall y \in \mathcal{Y}. \end{aligned}$$

Thus, such if $f_\perp^*(\boldsymbol{x}) > f_y^*(\boldsymbol{x})$ such that $f_\perp^*(\boldsymbol{x}) > 0$, $f_y^*(\boldsymbol{x}) > 0$, then the prediction following the decision rule $\arg\max_{k\in[\mathrm{K}+1]} g_k^*(\boldsymbol{x})$ would correspond with the Bayes optimal rule

$$r(\boldsymbol{x}) = \mathbb{I}\left[\max_{y\in\mathcal{Y}} \eta_y(\boldsymbol{x}) < p_{\mathrm{m}}(\boldsymbol{x})\right].$$

Case 2: In this case, if $\nexists y \in \mathcal{Y}$ s.t. $f_y^*(\boldsymbol{x}) > 0$, but $f_\perp^*(\boldsymbol{x}) > 0$, then the same argument as above implies the decision with the Bayes optimal rule.

Case 3: if $\exists y \in \mathcal{Y}$ s.t. $f_y^*(\boldsymbol{x}) > 0$, but $f_\perp^*(\boldsymbol{x}) < 0$, then the same argument as above implies the decision with the Bayes optimal rule. In this case, we will have $r(\boldsymbol{x}) = 0$, and the classifier's prediction would correspond with the regular Bayes optimal classifier, i.e. $\arg\max_{y\in\mathcal{Y}} \eta_y(\boldsymbol{x})$.

Case 4: In this case, if $\nexists y \in \mathcal{Y}$ s.t. $f_y^*(\boldsymbol{x}) > 0$, and also $f_\perp^*(\boldsymbol{x}) < 0$. This situation invokes the common 'none of the above' classification rule for One-vs-All classifiers. However, γ^{-1} would still measure $\eta_y(\boldsymbol{x})$ and $p_{\mathrm{m}}(\boldsymbol{x})$.

Thus, the cases above imply that the minimizer of the pointwise *inner* ψ*-risk* gives the Bayes optimal classifier and rejection prediction for $\mathbf{x} = \boldsymbol{x}$. Thus, the surrogate loss ϕ is calibrated for $0-1$ L2D loss function.

A.2 Derivations

A.2.1 Derivation of inverse *link* functions for $\phi_{\mathbf{SM}}$

From the proof of Theorem 1 of Mozannar and Sontag (2020), we have for the Bayes Optimal $f^*(\boldsymbol{x})$:

$$\frac{\mathbb{P}(\mathrm{y} = y|\boldsymbol{x})}{1 + \mathbb{P}(\mathrm{m} = \mathrm{y}|\boldsymbol{x})} = \frac{\exp f_y^*(\boldsymbol{x})}{\sum_{y' \in \mathcal{Y}^\perp} \exp f_{y'}^*(\boldsymbol{x})}$$
$$\frac{\mathbb{P}(\mathrm{m} = \mathrm{y}|\boldsymbol{x})}{1 + \mathbb{P}(\mathrm{m} = \mathrm{y}|\boldsymbol{x})} = \frac{\exp f_\perp^*(\boldsymbol{x})}{\sum_{y' \in \mathcal{Y}^\perp} \exp f_{y'}^*(\boldsymbol{x})}.$$

where we denote $\mathbb{P}(\mathrm{y} = y|\mathbf{x} = \boldsymbol{x}) = \gamma_y(f^*(\boldsymbol{x}))$, and $\mathbb{P}(\mathrm{m} = \mathrm{y}|\mathbf{x} = \boldsymbol{x}) = \gamma_{\mathrm{m}}(f^*(\boldsymbol{x}))$. From the second expression above, we easily have

$$\gamma_{\mathrm{m}}(f^*(\boldsymbol{x})) = \mathbb{P}(\mathrm{m} = \mathrm{y}|\mathbf{x} = \boldsymbol{x}) = \frac{\exp\{f_\perp^*(\boldsymbol{x})\}}{\sum_{y' \in \mathcal{Y}} \exp\{f_{y'}^*(\boldsymbol{x})\}}.$$

Plugging in the expression of $\mathbb{P}(\mathrm{m} = \mathrm{y}|\mathbf{x} = \boldsymbol{x})$ in the first expression, we have

$$\gamma_y(f^*(\boldsymbol{x})) = \mathbb{P}(\mathrm{y} = y|\mathbf{x} = \boldsymbol{x}) = \frac{\exp\{f_y^*(\boldsymbol{x})\}}{\sum_{y' \in \mathcal{Y}} \exp\{f_{y'}^*(\boldsymbol{x})\}}.$$

A.2.2 Closed-form expression for $\psi_{\mathbf{OvA}}$

We derive the closed-form expression for surrogate loss ψ_{OvA} using the procedure described in Section 2.4 in Chapter 2 (Preliminaries) for the code matrix $\mathbf{M}$ defined in Section 4.1. For the surrogate prediction space $\mathbb{R}$, and $f_y : \mathcal{X} \to \mathbb{R}, y \in \mathcal{Y}$ and $f_\perp : \mathcal{X} \to \mathbb{R}$ and $\boldsymbol{f}(\mathbf{x}) = [f_1(\mathbf{x}), \ldots, f_\perp(\mathbf{x})]$, we can use $\mathbf{M}$ to derive the closed form expression for the surrogate loss $\psi_{\mathrm{OvA}} : \mathcal{Y} \times \mathbb{R}^{\hat{\mathrm{K}}+1} \times \mathcal{M} \to \mathbb{R}_+$ as follows:

1. **Case 1:** $\psi_{\mathrm{OvA}}(\boldsymbol{f}; y, m)$ for y such that $\mathbb{I}[y \neq \mathrm{m}] = 1$
   In this case, we can follow the definition of $\mathbf{M}$ to gather that $m_{yj} = 1$ only if $j = y$. Thus, we get

$$\psi_{\mathrm{OvA}}(\boldsymbol{f}; y, m) = \phi\left[f_y(\boldsymbol{x})\right] + \sum_{\substack{y' \in \mathcal{Y} \cup \{\perp\} \\ y' \neq y}} \phi\left[-f_{y'}(\boldsymbol{x})\right]$$

2. **Case 2:** $\psi_{\mathrm{OvA}}(\boldsymbol{g}; y, m)$ for y such that $\mathbb{I}[y = m] = 1$
   In this case, we have $m_{yy} = 1$ as well as $m_{y\perp} = 1$ where $\perp$ denotes the index $(\mathrm{K}+1)$. Thus,

$$\psi_{\mathrm{OvA}}(\boldsymbol{f}; y, m) = \phi\left[f_y(\boldsymbol{x})\right] + \phi\left[f_\perp(\boldsymbol{x})\right] + \sum_{y' \in \mathcal{Y}, y' \neq y} \phi\left[-f_{y'}(\boldsymbol{x})\right]$$

.

Finally, we can combine both the cases to get

$$\psi_{\text{OvA}}(\boldsymbol{f}; y, m) = \phi\left[f_y(\boldsymbol{x})\right] + \phi\left[-f_\perp(\boldsymbol{x})\right] + \sum_{y' \in \mathcal{Y}, y' \neq y} \phi\left[-f_{y'}(\boldsymbol{x})\right] + \mathbb{I}[m = y]\left(\phi\left[f_\perp(\boldsymbol{x})\right] - \phi\left[-f_\perp(\boldsymbol{x})\right]\right).$$

where $\phi : \{-1, 1\} \times \mathbb{R} \to \mathbb{R}_+$ is a binary classification surrogate loss, and $\phi\left[f_y(\boldsymbol{x})\right] = \phi(1, f_y(\boldsymbol{x}))$. Similarly, $\phi\left[-f_y(\boldsymbol{x})\right] = \phi(-1, f_y(\boldsymbol{x}))$.

A.2.3 Derivation of inverse *link* functions for ϕ_{SM} in multi-experts setting

We follow the proof of Theorem 2 of Mozannar and Sontag (2020) to write the risk, denoted as $L\left(f_1, \ldots, f_{\perp,J}; \boldsymbol{x}, y, m_1, \ldots, m_J\right)$, for multi-expert learning to defer as follows:

$$L\left(f_1, \ldots, f_{\perp,J}; \boldsymbol{x}, y, m_1, \ldots, m_J\right) = \\ -\sum_{y \in \mathcal{Y}} \eta_y(\boldsymbol{x}) \log\left(\frac{\exp\{f_y(\boldsymbol{x})\}}{\sum_{y' \in \mathcal{Y}^\perp} \exp\{f_{y'}(\boldsymbol{x})\}}\right) - \sum_{j=1}^{J} \mathbb{P}(\mathrm{m}_j = \mathrm{y}|\boldsymbol{x}) \log\left(\frac{\exp\{f_{\perp,j}(\boldsymbol{x})\}}{\sum_{y' \in \mathcal{Y}^\perp} \exp\{f_{y'}(\boldsymbol{x})\}}\right).$$

We take the partial derivatives with respect to each f function and set them to 0. Placing in the optimal classifier h^*, and taking derivative with respect to f_j and setting it to 0, we get the desired relationship for the optimal f_j^*:

$$\frac{\mathbb{P}(\mathrm{m}_j = \mathrm{y}|\boldsymbol{x})}{1 + \sum_{j'=1}^{J} \mathbb{P}(\mathrm{m}_{j'} = \mathrm{y}|\boldsymbol{x})} = \frac{\exp\{f_{\perp,j}^*(\boldsymbol{x})\}}{\sum_{y' \in \mathcal{Y}^\perp} \exp\{f_{y'}^*(\boldsymbol{x})\}}.$$

Denote the RHS of the above equation as $p_{\perp,j}(f^*(\boldsymbol{x}))$. Since we have J equations, one for each expert, we can uniquely solve for $\mathbb{P}(\mathrm{m}_j = \mathrm{y}|\boldsymbol{x})$ as:

$$\mathbb{P}(\mathrm{m}_j = \mathrm{y}|\boldsymbol{x}) = \frac{p_{\perp,j}(f^*(\boldsymbol{x}))}{1 - \sum_{j'=1}^{J} p_{\perp,j'}(f^*(\boldsymbol{x}))}.$$

A.3 Additional Experimental Details

Below we provide more details on our experimental set-up.

`CIFAR-10` For the experiments on `CIFAR-10`, we use 28-layer Wide Residual Networks Zagoruyko and Komodakis, 2016 without using any data augmentation techniques following Mozannar and Sontag (2020). We use SGD with a momentum of 0.9, weight decay $5e-4$, and initial learning rate of 0.1. We further use cosine annealing learning rate schedule. We monitor validation loss, and employ early stopping to terminate the training if the loss doesn't improve for 20 epochs. The datasets are standardized to have 0 mean and unit variance. We train the models with a batch size of 1024. These experimental settings apply to both the Softmax Surrogate and the One-vs-All surrogate loss.

`HAM10000` To simulate the expert, we train an 8-layer MLPMixer model Tolstikhin et al., 2021. We make use of the publicly available code * for MLPMixer model. We resize the `HAM10000` images to 224×224 for our experiments. The 8-layer model has patch size of 16, expansion factor 2, and the dimensionality of the features to be 128. We train this model with Adam optimization algorithm with a learning rate of 0.001, weight decay of $5e-4$. We further use cosine annealing learning rate schedule with a warm-up period of

* https://github.com/jaketae/mlp-mixer/

5 epochs. The model is trained with a batch size of 1024, again with early stopping with a patience of 20 epochs. Since our goal was to simulate the real-world expert, we did not do extensive hyperparameter search for the expert model. For our main model on `HAM10000`, we finetune ResNet34 model. The training settings are same for the surrogate loss methods for `CIFAR-10` experiments.

For our other baselines, we use the code made available by the respective authors.

References

Mozannar, Hussein and David A. Sontag (2020). 'Consistent Estimators for Learning to Defer to an Expert'. In: *International Conference on Machine Learning* (cited on pages 3, 2, 12, 14, 15, 17, 18, 20, 22, 26, 36, 47, 48).

Zoabi, Yazeed, Shira Deri-Rozov, and Noam Shomron (2021). 'Machine learning-based prediction of COVID-19 diagnosis based on symptoms'. In: *npj Digital Medicine* (cited on page 1).

Kadampur, Mohammad Ali and Sulaiman Al Riyaee (2020). 'Skin cancer detection: Applying a deep learning based model driven architecture in the cloud for classifying dermal cell images'. In: *Informatics in Medicine Unlocked* (cited on page 1).

Zhong, Haoxi, Zhipeng Guo, Cunchao Tu, Chaojun Xiao, Zhiyuan Liu, and Maosong Sun (2018). 'Legal judgment prediction via topological learning'. In: *Conference on Empirical Methods in Natural Language Processing* (cited on page 1).

Chalkidis, Ilias, Ion Androutsopoulos, and Nikolaos Aletras (2019). 'Neural Legal Judgment Prediction in English'. In: *Meeting of the Association for Computational Linguistics* (cited on page 1).

Grigorescu, Sorin, Bogdan Trasnea, Tiberiu Cocias, and Gigel Macesanu (2020). 'A survey of deep learning techniques for autonomous driving'. In: *Journal of Field Robotics* (cited on page 1).

Hendrycks, Dan and Thomas G. Dietterich (2019). 'Benchmarking Neural Network Robustness to Common Corruptions and Perturbations'. In: *International Conference on Learning Representations* (cited on page 1).

Nguyen, Anh, Jason Yosinski, and Jeff Clune (2015). 'Deep neural networks are easily fooled: High confidence predictions for unrecognizable images'. In: *Conference on Computer Vision and Pattern Recognition* (cited on page 1).

Chow, C. K. (1957). 'An optimum character recognition system using decision functions'. In: *IRE Transactions on Electronic Computers* (cited on pages 1, 12, 35, 43).

Madras, David, Toniann Pitassi, and Richard Zemel (2018). 'Predict Responsibly: Improving Fairness and Accuracy by Learning to Defer'. In: *Advances in Neural Information Processing Systems* (cited on pages 1, 2, 36, 38).

Tschandl, Philipp et al. (2020). 'Human–computer collaboration for skin cancer recognition'. In: *Nature Medicine*, pp. 1–6 (cited on pages 1, 24).

Ramaswamy, H. G., Ambuj Tewari, and Shivani Agarwal (2018). 'Consistent algorithms for multiclass classification with an abstain option'. In: *Electronic Journal of Statistics* (cited on pages 2, 35).

Okati, Nastaran, Abir De, and Manuel Gomez-Rodriguez (2021). 'Differentiable Learning Under Triage'. In: *Advances in Neural Information Processing Systems* (cited on pages 2, 25–27, 36, 37).

Steinwart, Ingo (2007). 'How to Compare Different Loss Functions and Their Risks'. In: *Constructive Approximation* (cited on page 6).

Bartlett, Peter, Michael Jordan, and Jon McAuliffe (2006). 'Convexity, classification, and risk bounds'. In: *Journal of the American Statistical Association* (cited on page 7).

Reid, Mark D. and Robert C. Williamson (2009). 'Surrogate Regret Bounds for Proper Losses'. In: *International Conference on Machine Learning* (cited on page 9).

Reid, Mark D. and Robert C. Williamson (2010). 'Composite Binary Losses'. In: *Journal of Machine Learning Research* (cited on page 9).

Buja, Andreas, Werner Stuetzle, and Yi Shen (2005). 'Loss Functions for Binary Class Probability Estimation and Classification: Structure and Applications'. In: *Technical Report* (cited on page 9).

Dietterich, Thomas G. and Ghulum Bakiri (1995). 'Solving Multiclass Learning Problems via Error-Correcting Output Codes'. In: *Journal of Artificial Intelligence Research* (cited on page 10).

Langford, John, Tti-Chicago, Jl@hunch Net, and Alina Beygelzimer (2005). 'Sensitive error correcting output codes.' In: *Conference on Learning Theory* (cited on page 10).

Allwein, Erin L., Robert E. Schapire, and Yoram Singer (2001). 'Reducing Multiclass to Binary: A Unifying Approach for Margin Classifiers'. In: *Journal of Machine Learning Research* (cited on page 10).

Ramaswamy, Harish G., Balaji Srinivasan Babu, Shivani Agarwal, and Robert C. Williamson (2014). 'On the Consistency of Output Code Based Learning Algorithms for Multiclass Learning Problems'. In: *Conference on Learning Theory* (cited on pages 10, 11).

Cortes, Corinna, Giulia DeSalvo, and Mehryar Mohri (2016a). 'Learning with rejection'. In: *International Conference on Algorithmic Learning Theory* (cited on pages 12, 35, 36).
Cortes, Corinna, Giulia DeSalvo, and Mehryar Mohri (2016b). 'Boosting with Abstention'. In: *Advances in Neural Information Processing Systems* (cited on pages 12, 35, 36).
Ling, Charles X. and Victor S. Sheng (2010). 'Cost-Sensitive Learning'. In: *Encyclopedia of Machine Learning* (cited on page 15).
Vaicenavicius, Juozas, David Widmann, Carl Andersson, Fredrik Lindsten, Jacob Roll, and Thomas Schön (2019). 'Evaluating model calibration in classification'. In: *Conference on Artificial Intelligence and Statistics* (cited on pages 16, 17, 39).
Dawid, A Philip (1982). 'The well-calibrated Bayesian'. In: *Journal of the American Statistical Association* (cited on pages 17, 31).
Krizhevsky, Alex (2009). 'Learning Multiple Layers of Features from Tiny Images'. In: *Technical Report* (cited on pages 17, 22, 32).
Tschandl, Philipp, Cliff Rosendahl, and Harald Kittler (2018). 'The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions'. In: *Scientific data* (cited on pages 22, 24, 25).
Zagoruyko, Sergey and Nikos Komodakis (2016). 'Wide residual networks'. In: *British Machine Vision Conference* (cited on pages 22, 32, 48).
Guo, Chuan, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger (2017a). 'On Calibration of Modern Neural Networks'. In: *International Conference on Machine Learning* (cited on pages 24, 26, 38, 39).
Tolstikhin, Ilya O. et al. (2021). 'MLP-Mixer: An all-MLP architecture for vision'. In: *Advances in Neural Information Processing Systems* (cited on pages 24, 48).
He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2016). 'Deep residual learning for image recognition'. In: *Conference on Computer Vision and Pattern Recognition* (cited on page 24).
Bamford, Steven P. et al. (2009). 'Galaxy Zoo: the dependence of morphology and colour on environment*'. In: *Monthly Notices of the Royal Astronomical Society* (cited on page 25).
Davidson, Thomas, Dana Warmsley, Michael W. Macy, and Ingmar Weber (2017). 'Automated Hate Speech Detection and the Problem of Offensive Language'. In: *International AAAI Conference on Web and Social Media* (cited on page 25).
Raghu, Maithra, Katy Blumer, Greg Corrado, Jon Kleinberg, Ziad Obermeyer, and Sendhil Mullainathan (2019). 'The algorithmic automation problem: Prediction, triage, and human effort'. In: *arXiv preprint arXiv:1903.12220* (cited on pages 26, 36, 37).
Bansal, Gagan, Besmira Nushi, Ece Kamar, Eric Horvitz, and Daniel S. Weld (2021a). 'Is the Most Accurate AI the Best Teammate? Optimizing AI for Teamwork'. In: *AAAI Conference on Artificial Intelligence* (cited on pages 26, 37).
Joulin, Armand, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hervé Jégou, and Tomás Mikolov (2016). 'FastText.zip: Compressing text classification models'. In: *arXiv preprint arXiv:1612.03651* (cited on page 27).
Kim, Yoon (2014). 'Convolutional Neural Networks for Sentence Classification'. In: *Conference on Empirical Methods in Natural Language Processing* (cited on page 27).
Kingma, Diederik P. and Jimmy Ba (2015). 'Adam: A Method for Stochastic Optimization'. In: *International Conference on Learning Representations* (cited on page 27).
Mukhoti, Jishnu, Andreas Kirsch, Joost van Amersfoort, Philip HS Torr, and Yarin Gal (2021). 'Deterministic Neural Networks with Appropriate Inductive Biases Capture Epistemic and Aleatoric Uncertainty'. In: *arXiv preprint arXiv:2102.11582* (cited on page 28).
Chow, C. (1970). 'On optimum recognition error and reject tradeoff'. In: *IEEE Transactions on Information Theory* (cited on page 35).
Bartlett, Peter L. and Marten H. Wegkamp (2008). 'Classification with a Reject Option Using a Hinge Loss'. In: *Journal of Machine Learning Research* (cited on page 35).
Yuan, Ming and Marten Wegkamp (2010). 'Classification Methods with Reject Option Based on Convex Risk Minimization'. In: *Journal of Machine Learning Research* (cited on page 35).
Jiang, Heinrich, Been Kim, Melody Y. Guan, and Maya Gupta (2018). 'To Trust or Not to Trust a Classifier'. In: *Advances in Neural Information Processing Systems* (cited on page 35).

Grandvalet, Yves, Alain Rakotomamonjy, Joseph Keshet, and Stéphane Canu (2009). 'Support Vector Machines with a Reject Option'. In: *Advances in Neural Information Processing Systems* (cited on page 35).
Ni, Chenri, Nontawat Charoenphakdee, Junya Honda, and Masashi Sugiyama (2019). 'On the Calibration of Multiclass Classification with Rejection'. In: *Advances in Neural Information Processing Systems* (cited on pages 35, 36).
Hendrycks, Dan and Kevin Gimpel (2017). 'A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks'. In: *International Conference on Learning Representations* (cited on page 36).
Gal, Yarin and Zoubin Ghahramani (2016). 'Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning'. In: *International Conference on Machine Learning* (cited on page 36).
Lakshminarayanan, Balaji, Alexander Pritzel, and Charles Blundell (2017). 'Simple and Scalable Predictive Uncertainty Estimation Using Deep Ensembles'. In: *Advances in Neural Information Processing Systems* (cited on page 36).
Guo, Chuan, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger (2017b). 'On Calibration of Modern Neural Networks'. In: *International Conference on Machine Learning* (cited on page 36).
Charoenphakdee, Nontawat, Zhenghang Cui, Yivan Zhang, and Masashi Sugiyama (2021). 'Classification with Rejection Based on Cost-sensitive Classification'. In: *International Conference on Machine Learning* (cited on page 36).
Wilder, Bryan, Eric Horvitz, and Ece Kamar (2020). 'Learning to Complement Humans'. In: *International Joint Conference on Artificial Intelligence* (cited on page 36).
Keswani, Vijay, Matthew Lease, and Krishnaram Kenthapadi (2021). 'Towards unbiased and accurate deferral to multiple experts'. In: *AAAI/ACM Conference on AI, Ethics, and Society* (cited on page 36).
Liu, Jessie, Blanca Gallego, and Sebastiano Barbieri (2022). 'Incorporating uncertainty in learning to defer algorithms for safe computer-aided diagnosis'. In: *Scientific Reports* (cited on page 36).
Pre-emptive Learning to Defer for Sequential Medical Decision-Making Under Uncertainty (2021) (cited on page 37).
De, Abir, Nastaran Okati, Ali Zarezade, and Manuel Gomez-Rodriguez (2021). 'Classification Under Human Assistance'. In: *AAAI Conference on Artificial Intelligence* (cited on page 37).
De, Abir, Paramita Koley, Niloy Ganguly, and Manuel Gomez-Rodriguez (2020). 'Regression Under Human Assistance'. In: *AAAI Conference on Artificial Intelligence* (cited on page 37).
Kerrigan, Gavin, Padhraic Smyth, and Mark Steyvers (2021). 'Combining Human Predictions with Model Probabilities via Confusion Matrices and Calibration'. In: *Advances in Neural Information Processing Systems* (cited on page 37).
Pandya, Ravi, Sandy H. Huang, Dylan Hadfield-Menell, and Anca D. Dragan (2019). 'Human-AI Learning Performance in Multi-Armed Bandits'. In: *AAAI/ACM Conference on AI, Ethics, and Society* (cited on page 37).
Donahue, Kate, Alexandra Chouldechova, and Krishnaram Kenthapadi (2022). 'Human-Algorithm Collaboration: Achieving Complementarity and Avoiding Unfairness'. In: *ACM Conference on Fairness, Accountability, and Transparency* (cited on page 37).
Vodrahalli, Kailas, Tobias Gerstenberg, and James Zou (2021). 'Do Humans Trust Advice More if it Comes from AI? An Analysis of Human-AI Interactions'. In: *arXiv preprint arXiv:2107.07015* (cited on page 37).
Bansal, Gagan, Tongshuang Sherry Wu, Joyce Zhou, Raymond Fok, Besmira Nushi, Ece Kamar, Marco Tulio Ribeiro, and Daniel S. Weld (2021b). 'Does the Whole Exceed its Parts? The Effect of AI Explanations on Complementary Team Performance'. In: *CHI Conference on Human Factors in Computing Systems* (cited on page 37).
Charusaie, Mohammad-Amin, Hussein Mozannar, David Sontag, and Samira Samadi (2022). 'Sample Efficient Learning of Predictors that Complement Humans'. In: *International Conference on Machine Learning* (cited on page 37).
Cramer, Henriette, Vanessa Evers, Satyan Ramlal, Maarten van Someren, Lloyd Rutledge, Natalia Stash, Lora Aroyo, and Bob J. Wielinga (2008). 'The effects of transparency on trust in and acceptance of a content-based art recommender'. In: *User Modeling and User-Adapted Interaction* (cited on page 38).
Kizilcec, René F. (2016). 'How Much Information? Effects of Transparency on Trust in an Algorithmic Interface'. In: *CHI Conference on Human Factors in Computing Systems* (cited on page 38).
Gal, Yarin et al. (2016). 'Uncertainty in deep learning'. In: (cited on page 38).
Bröcker, Jochen and Leonard A. Smith (2007). 'Increasing the Reliability of Reliability Diagrams'. In: *Weather and Forecasting* (cited on page 38).

Zadrozny, Bianca and Charles Elkan (2002). 'Transforming Classifier Scores into Accurate Multiclass Probability Estimates'. In: *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (cited on page 38).

Zhang, Yunfeng, Q. Vera Liao, and Rachel K. E. Bellamy (2020). 'Effect of Confidence and Explanation on Accuracy and Trust Calibration in AI-Assisted Decision Making'. In: *Conference on Fairness, Accountability, and Transparency* (cited on page 38).

Bhatt, Umang et al. (2021). 'Uncertainty as a Form of Transparency: Measuring, Communicating, and Using Uncertainty'. In: *AAAI/ACM Conference on AI, Ethics, and Society* (cited on page 38).

Ovadia, Yaniv, Emily Fertig, Jie Ren, Zachary Nado, D. Sculley, Sebastian Nowozin, Joshua V. Dillon, Balaji Lakshminarayanan, and Jasper Snoek (2019). 'Can You Trust Your Model's Uncertainty? Evaluating Predictive Uncertainty under Dataset Shift'. In: *Advances in Neural Information Processing Systems* (cited on page 39).

Kull, Meelis, Miquel Perello-Nieto, Markus Kängsepp, Telmo Silva Filho, Hao Song, and Peter Flach (2019). 'Beyond temperature scaling: Obtaining well-calibrated multiclass probabilities with Dirichlet calibration'. In: *Advances in Neural Information Processing Systems* (cited on page 39).

Gupta, Chirag and Aaditya Ramdas (2022). 'Top-label calibration and multiclass-to-binary reductions'. In: *International Conference on Learning Representations* (cited on page 39).

Zhao, Shengjia, Michael Kim, Roshni Sahoo, Tengyu Ma, and Stefano Ermon (2021a). 'Calibrating Predictions to Decisions: A Novel Approach to Multi-Class Calibration'. In: *Advances in Neural Information Processing Systems* (cited on page 40).

Zhao, Shengjia, Michael P. Kim, Roshni Sahoo, Tengyu Ma, and Stefano Ermon (2021b). 'Calibrating Predictions to Decisions: A Novel Approach to Multi-Class Calibration'. In: *Advances in Neural Information Processing Systems* (cited on page 41).