



Uncertainty in predictions of deep learning models for fine-grained classification

Titouan Lorieul

► To cite this version:

Titouan Lorieul. Uncertainty in predictions of deep learning models for fine-grained classification. Machine Learning [cs.LG]. Université de Montpellier (UM), FRA., 2020. English. NNT: . tel-03040683v1

HAL Id: tel-03040683

<https://theses.hal.science/tel-03040683v1>

Submitted on 4 Dec 2020 (v1), last revised 1 Oct 2021 (v2)

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

THÈSE POUR OBTENIR LE GRADE DE DOCTEUR DE L'UNIVERSITÉ DE MONTPELLIER

En Informatique

École doctorale : Information, Structures, Systèmes

Unité de recherche LIRMM

Incertitude des prédictions dans les modèles d'apprentissage profonds appliqués à la classification fine

Présentée par Titouan LORIEUL

Le 02/12/2020

Sous la direction de Alexis JOLY

Devant le jury composé de

Patrick GALLINARI, Professeur, Sorbonne Université

Willem WAEGEMAN, Associate Professor, Ghent University

Joseph SALMON, Professeur, Université de Montpellier

Jana WALDCHEN, Group Leader, Max Planck Institute for Biogeochemistry

Dennis SHASHA, Professeur, New York University

Alexis JOLY, Directeur de recherche, Inria

Rapporteur

Rapporteur

Examineur

Examineur

Examineur

Directeur de thèse



UNIVERSITÉ
DE MONTPELLIER

Abstract

Deep neural networks have shown dramatic improvements in a lot of supervised classification tasks. Such models are usually trained with the objective to ultimately minimize the top-1 error rate. Although this approach is very powerful, it averages out the uncertainty of individual samples and does not capture if on a given data point this prediction is reliable or not and why.

In real-world scenarios, it can actually be impossible – even for an oracle – to determine the exact label of a given data item because it does not, by itself, contain sufficient evidence to decide between several similar classes. Unlike multi-task classification where each data sample is associated with several labels, here, each item corresponds to exactly one label but this latter is uncertain. For instance, an image of a plant leaf might not be enough to distinguish between several possible species sharing the same leaf morphology. In fine-grained classification problems, most data samples intrinsically contain a certain amount of such label ambiguity even if they are associated with a single hard label. Furthermore, the model itself introduces additional uncertainty in the prediction because it is learned using a finite training dataset. This uncertainty is expected to be progressively reduced by increasing the training set size contrary to the intrinsic ambiguity of the data items which is theoretically irreducible.

The goal of this PhD is to study these two types of uncertainties in a decision-theoretic framework. To do so, we propose to move away from the classic top-1 prediction error rate which solely requires to estimate the most probable class. Instead, we pick decision frameworks that force the model to learn more structure about the existing uncertainty. In particular, we focus on two frameworks: (i) adding the opportunity for the classifier to refuse to answer, usually referred to as *classification with reject option*, and (ii) allowing the classifier to output a set of possible labels rather than a single one, which is known as *set-valued classification*.

We first study how uncertainty information can be exploited to tackle classification with reject option. In this framework, the predictor is a pair containing a classifier and a rejector. By fixing the classifier and focusing on the rejector, we can study how uncertainty information about the classifier can be leveraged to hopefully build a better rejection criterion. Unfortunately, we empirically show that it is difficult to separate both forms of uncertainty and recombine them properly. Based on this observation, we then focus on the first form of uncertainty, task ambiguity, and study natural frameworks to handle it: set-valued classification. There are several ways to predict sets. The most naive approach is to predict the K most probable classes. However, this assumes that all the samples have the same level of ambiguity which is known to be wrong in most cases. Instead, we propose to use average- K : the predictor can output sets of different sizes but on average their size must be equal to K . We then generalize to other adaptive set-valued classification approaches and propose a framework unifying most of them. In particular, we show several ways to construct such classifiers depending on the constraints on the error rate and on the set size and study their relative advantages and weaknesses.

Résumé en français

Les réseaux neuronaux profonds ont permis des améliorations spectaculaires dans de nombreuses tâches de classification supervisées. Ces modèles sont généralement entraînés avec pour objectif final de minimiser le taux d'erreur en top 1. Bien que cette approche soit très puissante, elle moyenne l'incertitude des échantillons individuels et ne permet pas de savoir si, sur un point de données donné, cette prévision est fiable ou non et pourquoi.

Dans des cas réels, il peut être impossible (même pour un oracle) de déterminer l'étiquette exacte d'un échantillon donné car il ne contient pas, en soi, de preuves suffisantes pour trancher entre plusieurs classes similaires. Contrairement à la classification multitâche où chaque échantillon de données est associé à plusieurs étiquettes, ici, chaque donnée correspond exactement à une classe, mais cette dernière est incertaine. Par exemple, une image d'une feuille de plante peut ne pas suffire à distinguer plusieurs espèces possibles partageant la même morphologie de feuille. Dans les problèmes de classification à grain fin, la plupart des échantillons de données contiennent intrinsèquement un certain niveau de cette ambiguïté sur l'étiquette, même s'ils sont associés à une seule vraie étiquette. En outre, le modèle lui-même introduit une incertitude supplémentaire dans ses prédictions car il est entraîné à l'aide d'un jeu de données d'apprentissage fini. Cette incertitude devrait être progressivement réduite en augmentant la taille de cette ensemble d'apprentissage, contrairement à l'ambiguïté intrinsèque des données qui est théoriquement irréductible.

L'objectif de ce doctorat est d'étudier ces deux types d'incertitudes dans le cadre de la théorie de la décision. Pour ce faire, nous proposons de mettre de côté le taux d'erreur de prédiction en top 1 classique qui ne nécessite que l'estimation de la classe la plus probable. Nous proposons plutôt de nous intéresser à des cadres décisionnels qui forcent le modèle à mieux apprendre la structure de l'incertitude existante. En particulier, nous nous concentrons sur deux cadres : (i) ajouter la possibilité pour le classifieur de refuser de répondre, généralement appelé *classification avec option de rejet*, et (ii) en autorisant au classifieur de produire un ensemble d'étiquettes possibles plutôt qu'une seule, ce qui est connu sous le nom de *prédiction d'ensembles*.

Nous étudions d'abord comment l'information d'incertitude peut être exploitée pour traiter la classification avec option de rejet. Dans cette configuration, le prédicteur est une paire comprenant un classifieur et un rejeteur. En fixant le classifieur et en étudiant le rejeteur, nous pouvons étudier comment l'information d'incertitude concernant le classifieur peut être exploitée pour éventuellement construire un meilleur critère de rejet. Malheureusement, nous montrons empiriquement qu'il est difficile de séparer les deux formes d'incertitude et de les recombinaison correctement. Sur la base de cette observation, nous nous concentrons ensuite sur la première forme d'incertitude, l'ambiguïté de la tâche, et étudions un cadre naturel pour la gérer : la prédiction d'ensemble. Il existe plusieurs façons de prédire des ensembles. L'approche la plus naïve consiste à prédire les K classes les plus probables. Toutefois, cela suppose que tous les échantillons présentent le même niveau d'ambiguïté, ce qui est connu pour être faux dans

la plupart des cas. Nous proposons plutôt d'utiliser une approche moyenne- K : le prédicteur peut produire des ensembles de taille différente, mais en moyenne leur taille doit être égale à K . Nous généralisons ensuite à d'autres approches adaptatives de prédiction d'ensembles et proposons un cadre unifiant la plupart d'entre elles. En particulier, nous montrons plusieurs façons de construire de tels classifieurs en fonction des contraintes sur le taux d'erreur et sur la taille de l'ensemble et étudions leurs avantages et faiblesses relatifs.

Contents

Notation	1
1 Introduction	3
1.1 Illustrative use case: Pl@ntNet project, citizen science for biodiversity monitoring	4
1.2 Predictive uncertainty in classification	7
1.3 Summary of contributions	12
1.4 Publications	14
2 Positioning and state-of-the-art	15
2.1 Supervised image classification	15
2.2 Predictive uncertainty in classification	19
2.3 Making decisions in presence of uncertainty	24
3 How to know when not to trust a classifier?	27
3.1 From usual classification to classification with reject option	28
3.2 Learning to reject when the classifier is fixed: theoretical study	29
3.3 What uncertainty is needed for rejection?	35
3.4 Confidence-based criteria	37
3.5 Leveraging disagreement information	40
3.6 Experiments	42
3.7 Conclusion	44
4 Average-K versus top-K classification	45
4.1 From top- K to average- K classification	47

4.2	Introductory examples	52
4.3	When is top- K classification optimal?	54
4.4	Quantifying the usefulness of average- K	60
4.5	Consistent estimation procedures	64
4.6	Experiments on real datasets	70
4.7	Conclusion	79
5	A unified framework for set-valued classification	83
5.1	A general framework for set-valued classifiers	84
5.2	Main set-valued classification frameworks	87
5.3	Empirical comparison of the frameworks	94
5.4	Summary of the set-valued classification formulations	101
5.5	Conclusion	104
6	Conclusion and perspectives	107
6.1	Synthesis of results and general perspectives	107
6.2	Applicative perspectives	108
	Bibliography	115

Notation

$:=$	Equal by definition
$\mathbb{1}_P$	Indicator function, equals to 1 if proposition P is true and 0 otherwise
\mathbb{N}	Set of natural numbers
\mathbb{R}	Set of real numbers
$(x)^+$	$= \max(x, 0)$, the positive part of scalar x
$\ \mathbf{x}\ _1$	$= \sum_i x_i $, the ℓ_1 -norm of vector \mathbf{x}
$\ \mathbf{x}\ _2$	$= \sqrt{\sum_i x_i^2}$, the ℓ_2 -norm of vector \mathbf{x}
$ A $	Cardinality of set A
$\mathcal{P}(A)$	Power set (set of all subsets) of A
$A \times B$	Cartesian product of sets A and B
$A \Delta B$	$= (A \setminus B) \cup (B \setminus A)$, symmetric difference of sets A and B
\mathcal{X}	Input/instance set
\mathcal{Y}	Output/label set
$\mathcal{X}^{\mathcal{Y}}$	Set of functions mapping \mathcal{X} to \mathcal{Y}
$f[A]$	Image of set A under function f
$f^{-1}[A]$	Preimage (or inverse image) of set A under function f
f^{-1}	(Generalized) inverse of function f
Δ_C	Probability simplex of dimension C
\mathbb{P}	Probability of a random variable
\mathbb{E}	Expectation of a random variable

Scalars are denoted with lowercase letters (e.g. x) while vectors are written in bold case (e.g. \mathbf{x}). The i th element of a vector \mathbf{x} is denoted x_i . Uppercase is used to denote random variable, e.g. X , while bold uppercase is used for random vectors, e.g. \mathbf{X} . The probability of a random variable X is denoted \mathbb{P}_X , the subscript can be dropped if there is no ambiguity on which random variable is concerned and the previous probability is then denoted \mathbb{P} . The expected value of a function f of a random variable X is denoted $\mathbb{E}_X[f(X)]$. Here again, the subscript might be dropped if the context is explicit. Finally, throughout this manuscript, we will use the following generalized inverse of a right-continuous non-increasing function f denoted f^{-1} defined as

$$f^{-1}(t) := \inf \{x \in \mathbb{R} : f(x) \leq t\}.$$

This notation must not be confused with the preimage of a set A under f denoted $f^{-1}[A]$.

I

Introduction

Contents

1.1	Illustrative use case: Pl@ntNet project, citizen science for biodiversity monitoring	4
1.1.1	General presentation	5
1.1.2	Forms of uncertainty in the different components	6
1.2	Predictive uncertainty in classification	7
1.2.1	Multi-class classification framework	7
1.2.2	Different types of uncertainties in classification	8
1.2.3	What to do with this information?	9
1.3	Summary of contributions	12
1.4	Publications	14

Artificial neural networks, predictive models inspired by biological neural systems, have been studied since the first half of the 20th century. One of the highlights of this period was the discovery of the perceptron algorithm by [Rosenblatt \(1957\)](#). Despite these first successes, they faced criticism due to major theoretical limitations of the proposed methods which were thought by many at the time to be insuperable. Among those critiques, one of the most prominent and influential was the one by [Minsky and Papert \(1969\)](#). Eventually, these methods were superseded by symbolic approaches like expert systems during the second half of the 20th century ([Cardon et al., 2018](#), see [Figure 1.1](#)). Although progress was made at the end of the century, it is only in 2012 with AlexNet winning ImageNet visual classification challenge ([Krizhevsky et al., 2012](#)) that neural networks came back to the forefront. Nowadays usually referred to under the name of *deep learning* ([LeCun et al., 2015](#)), these approaches predominate the wider field of *machine learning* ([Jordan and Mitchell, 2015](#)) to which they belong. They have even taken their revenge on symbolic methods by “stealing” their terminology: the use of the term *artificial intelligence* in a broad audience these days usually refers to the successes of deep learning methods.

Their success is largely due to the approach taken by machine learning, i.e. learning to perform a task by generalizing from a set of examples. Learning is formalized in a very pragmatic way: the machine does not need to understand what is relevant for this task but is required to perform correct predictions as often as possible. Some simple models in machine learning are interpretable and allow to explain how they made their predictions. However, deep

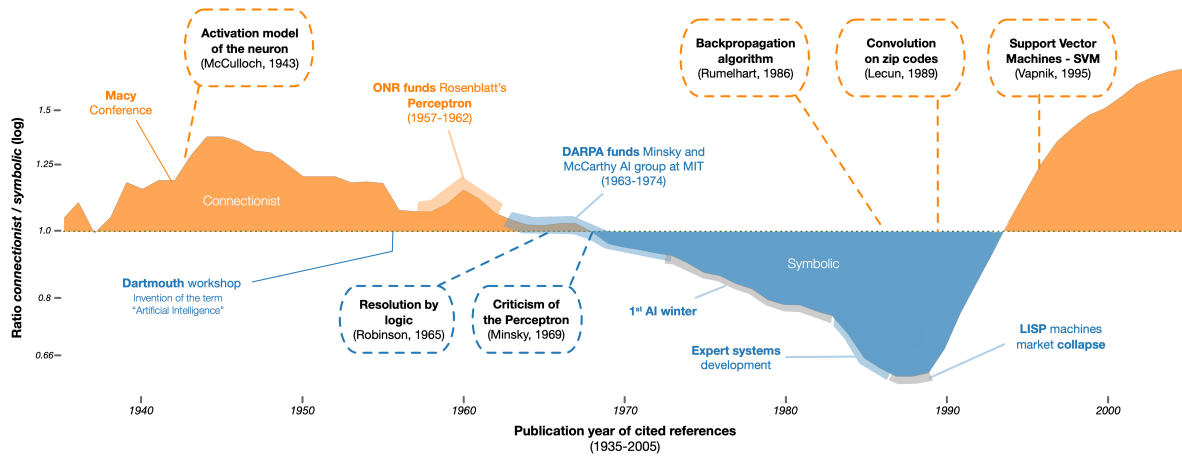


Figure 1.1: Relative number of publications related to connectionist approaches (machine learning, neural networks, etc.) versus symbolic ones (expert systems, etc.) through time. Taken from [Cardon et al. \(2018\)](#) (under license CC BY 4.0).

learning models, by combining numerous layers of neurons more or less connected with each other, are much harder to interpret. By trading explicability for predictive power, these models have been shown to be very effective for tackling tasks which were out of reach before. For these reasons, their usage spreads to an ever-increasing range of fields.

Nowadays, deep learning models are used in medicine where they are helpful for medical diagnosis ([Kononenko, 2001](#)) and medical image analysis ([Greenspan et al., 2016](#); [Shen et al., 2017](#)). They are also employed in chemistry for drug design, material property prediction, protein structure prediction, and numerous other use cases ([Goh et al., 2017](#); [Mater and Coote, 2019](#)). In physics, among others, they enable the discovery of new particles in high-energy particle colliders ([Baldi et al., 2014](#)). In geophysics, they permit the exploitation of increasingly large datasets and they accelerate simulations useful to study Earth geoscience ([Bergen et al., 2019](#)). In Earth system science, they are being integrated and coupled with physical modeling to gain further understanding of these complex systems ([Reichstein et al., 2019](#); [de Bezenac et al., 2019](#)). In computational biology, they allow to analyze vast volumes of cell images and to extract knowledge from complex data in regulatory genomics ([Angermueller et al., 2016](#)). In ecology, they are useful to estimate biodiversity from a wide array of sources which is interesting for scientific purposes but also helps for decision-making in managing and conserving natural resources ([Christin et al., 2019](#)).

This list seems dizzyingly endless... It highlights the wide variety of usages of machine learning ranging from data collection and annotation to scientific knowledge extraction by way of modeling complex systems and computer-assisted decision-making. Several of these usages can be found in the project within which this PhD took place, the Pl@ntNet project, which we present next.

1.1 Illustrative use case: Pl@ntNet project, citizen science for biodiversity monitoring

This PhD has been carried out within the Pl@ntNet project which we describe here. The contributions of this thesis are not specific to this project and have broader applications. This project serves as a motivating use case and illustrates which were the main questions related to

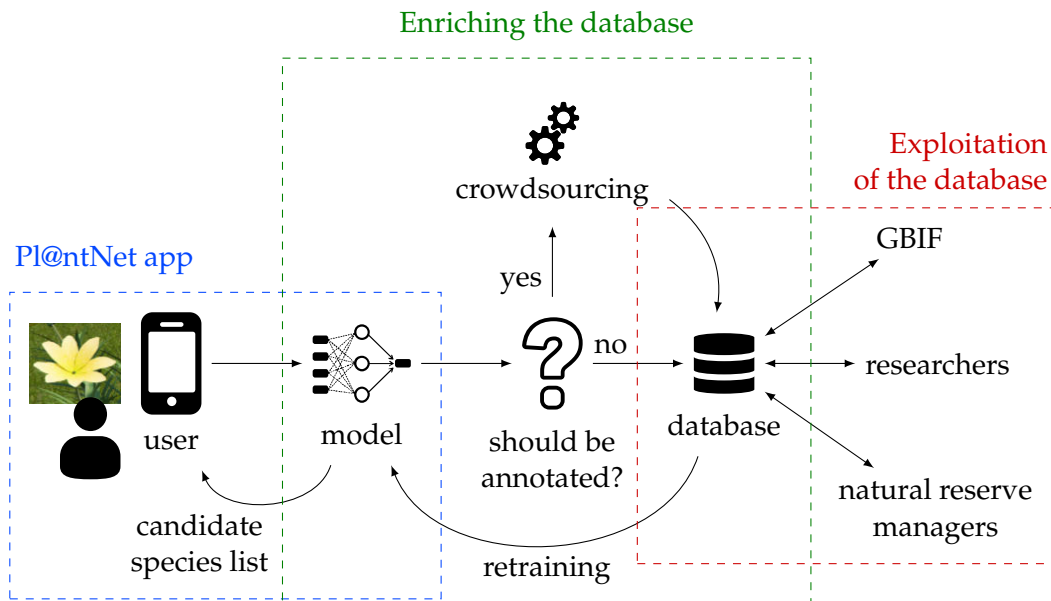


Figure 1.2: General pipeline of the Pl@ntNet project and its three main components.

uncertainty addressed in this PhD.

1.1.1 General presentation

The goal of the Pl@ntNet project¹ is to monitor the biodiversity (Joly et al., 2016). Its general pipeline is shown in Figure 1.2. We briefly describe its three main components.

Pl@ntNet app The main component is a plant recognition application for smartphones which is freely available for download. It allows anyone to take a picture of a plant and to send it to a server for recognition. This recognition is performed by a neural network (Affouard et al., 2017). The user then gets in return a list of candidate species each of them associated with a confidence score and an adapted illustrative picture to help the user make his decision. The user can then validate a species and share his observation if he desires to. Users of the app have a wide variety of profiles. It is used as an educational tool but it also finds a practical utility for professionals as it can help decision-making. For instance, some farmers use it to identify a plant in their field unknown to them. They can then take a more informed decision on how to deal with it – e.g. leave it, remove it, apply a treatment, etc.

Crowdsourcing platform The second component is a human-in-the-loop module. It aims at improving the collected annotations gathered from the queries sent by the users. This dataset is (i) used to enhance the model by re-training it on a bigger and more complete dataset, and, (ii) it is made available for the needs of stakeholders such as researchers and natural reserve managers. Because of the huge volume of data (over two million observations were shared in 2019²), this annotation can not be done solely by expert botanists. At first, the data is filtered and only images for which the model is uncertain goes through the human annotation process, the rest is automatically annotated by the model. Even after this step, the volume of data selected for manual annotation remains too large. To overcome this issue, a crowdsourcing platform

¹<https://plantnet.org/en/>

²<https://identify.plantnet.org/stats>

called *ThePlantGame*³ allows users with all levels of expertise to participate in the annotation effort through a serious game (Servajean et al., 2016). Once sufficiently many votes are gathered for an observation and a consensus emerges on the species name, the annotation is considered to be trust-worthy and enriches the global database.

Data sharing Finally, the last component is the sharing of the collected database. Because of its volume and its richness, this data is valuable for a lot of different stakeholders and partners. For instance, researchers in botany and ecology are interested in modeling and monitoring the species distribution through time (Botella et al., 2018). To do so, they typically want to query the dataset in order, for instance, to retrieve all the observations of species from a given list. Other stakeholders, such as natural reserve managers use this data as an aid for decision-making in conservation policy via species inventory (Bonnet et al., 2020). Finally, part of this data is shared via the Global Biodiversity Information Facility (GBIF)⁴. This international network aims at providing open access to time and localization data of observed species from all forms of life all around the world. This data is available for researchers and policymakers. Pushing data to GBIF requires to meet their standards, in particular, only the most confident annotations must be sent.

1.1.2 Forms of uncertainty in the different components

We now dive into the different components. We highlight the natural questions related to uncertainty that emerges in those blocks and list some related approaches which allow to handle them.

Plant recognition model

Q1.1: How to predict the list of candidate species for a given query which would help the user to make his final decision?

- ➔ *Set-valued classification* provides principled ways to predict sets instead of a single species depending on the wished properties of the produced sets.

Q1.2: Could we provide additional feedback to the user to help him reduce the list?

- ➔ *Knowledge extraction* could be used to learn the visual confusions between the proposed species from which it would be possible to determine what is the most discriminating organ – for the given list of species – that a user should send a photo of.

Q1.3: Should an answer always be returned?

- ➔ *Classification with reject option* could be used to refuse to answer if the model is likely to commit an error;
- ➔ *Out-of-distribution detection* could permit to reject irrelevant queries (e.g. do not contain plants, too noisy, etc.) and those different from the training data which would corrupt the quality of the database;
- ➔ *Open set classification* could allow treating separately queries containing plants representing species not (yet) recognized by the model, these observations could be useful to increase the list of species covered by the model.

Human re-annotation

³<http://theplantgame.com>

⁴<https://www.gbif.org/what-is-gbif>

Q2.1: How to select the observations which require human annotation?

- ➔ *Classification with reject option* allows to filter observations for which we do not trust the prediction proposed by the model;
- ➔ *Active learning*, on the other hand, aims at selecting the samples which will be the most useful to re-train the model with.

Q2.2: How to select which users should vote on which observation depending on their expertise? How many votes are required before accepting the collective annotation?

- ➔ *Crowdsourcing* allows to automatically learn the individual expertise of the users and combine them to efficiently obtain a high confidence annotation.

Data sharing

Q2.1: How to select the observations that can be safely shared?

- ➔ *Classification with reject option* can select only the most confident observations.

Q2.2: Which are the most trust-worthy observations for knowledge extraction? For decision-making? Is it the same selection process for both?

- ➔ *Classification with reject option*, if a stakeholder wants very confident annotations, can select only those;
- ➔ *Set-valued classification*, if a stakeholder wants to retrieve all the observations of a given species with high recall (e.g. to detect the presence of invasive which might require investigation), provides a way to do so.

Q2.3: What is the impact of this selection process on future studies?

- ➔ *Sample bias correction* takes into account the additional bias that this selection might introduce – such as filtering out less represented species;
- ➔ *Sensitivity analysis* studies how the conclusions of a study are sensitive to changes in its inputs;
- ➔ *Uncertainty quantification* quantifies how the uncertainty in the inputs propagates to the conclusions.

As can be seen, uncertainty appears in a variety of fashions through the different components. Even when the decisions to be taken seem to be similar – such as rejecting or selecting observations at different stages of the pipeline – the context is different with distinct constraints and the resulting decisions may thus differ.

1.2 Predictive uncertainty in classification

We have illustrated in the previous section the importance of properly handling uncertainty in practical scenarios. We now more concretely and formally present what are the different forms of uncertainty occurring in classification and how to deal with it.

1.2.1 Multi-class classification framework

In *multi-class classification* setting (Shalev-Shwartz and Ben-David, 2014), we are given data samples $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$ called the *training data* where each pair (x_i, y_i) is composed of an input x and a label y . Typically, in our case, x is a picture of a plant while y is the species

name. The pairs (x_i, y_i) are supposed to be independently sampled from a probability measure $\mathbb{P}_{\mathbf{X}, Y}$.

The aim of classification is to learn a classifier – a function taking new inputs x and outputting labels y – which generalizes well on unseen data. This is formalized by measuring the expected error rate:

$$R(h) := \mathbb{E}_{\mathbf{X}, Y} [\mathbb{1}_{Y \neq h(\mathbf{X})}] = \mathbb{P}_{\mathbf{X}, Y} [Y \neq h(\mathbf{X})].$$

This quantity is referred to as a *risk* and captures how often we are likely to make a wrong prediction in the future. It can be minimized for every input x , in which case the resulting optimal classifier predicts the most probable class for every x :

$$h^*(x) := \arg \max_k \mathbb{P}_{\mathbf{X}, Y} [Y = k \mid \mathbf{X} = x].$$

This probability is the conditional distribution of labels Y given an input x denoted $\eta(x)$,

$$\eta_k(x) := \mathbb{P}_{\mathbf{X}, Y} [Y = k \mid \mathbf{X} = x].$$

In practice this conditional probability $\eta(x)$ is unknown. Instead, we have to use the training data to build an estimator \hat{h} of the optimal predictor h^* .

In its formulation, the classification problem does not explicitly refer to uncertainty but rather formalizes the idea of learning a function which makes good predictions on average. It thus only refers to a *risk* which is an expected error rate on new unseen samples: the uncertainty is only seen as an averaged quantity. On the contrary, in our case, we are interested in *predictive uncertainty* (Hüllermeier and Waegeman, 2019): we want to estimate the uncertainty of our prediction $\hat{h}(x)$ for every given input x and not solely on average. This requires redefining the task we want to solve.

1.2.2 Different types of uncertainties in classification

In general, uncertainty is categorized into two forms (Der Kiureghian and Ditlevsen, 2009):

- *aleatoric uncertainty*: uncertainty arising from the intrinsic randomness of the underlying process, it is considered to be *irreducible*;
- *epistemic uncertainty*: uncertainty caused by a lack of knowledge, it is considered to be *reducible* given additional information.

As noted by Der Kiureghian and Ditlevsen (2009), this distinction without context is quite arbitrary. It is the context which determines which part of the uncertainty can be reduced and which one can not. In the supervised classification setting, we can reformulate these as respectively:

- *task ambiguity*: uncertainty intrinsic to the task due to the ambiguity between classes – it is independent of the model's choice and estimation, in particular, it can not be reduced by collecting more data;
- *model uncertainty*: uncertainty arising from the model's choice and estimation – due to the finite size of the available training data, the model class has to be constrained and it can only be approximately fitted – collecting more data can reduce this uncertainty.

In the case of Pl@ntNet, looking at the way the data is collected helps us understand how task ambiguity emerges. This latent data-acquisition process is shown in Figure 1.3. Typically, a

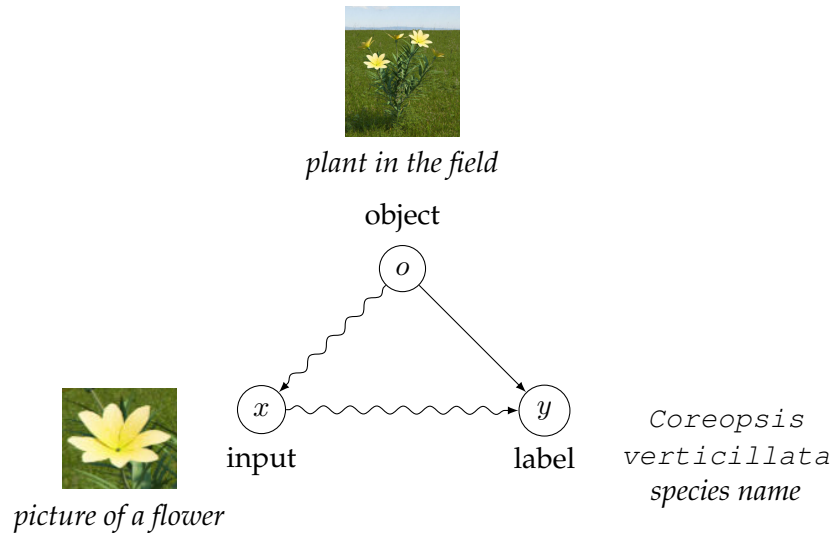


Figure 1.3: Latent data-acquisition process in Pl@ntNet application: a plant is observed in the field but only a picture of it is sent to the classification model. This introduces some uncertainty which propagates to the determination of the species name only from the picture.

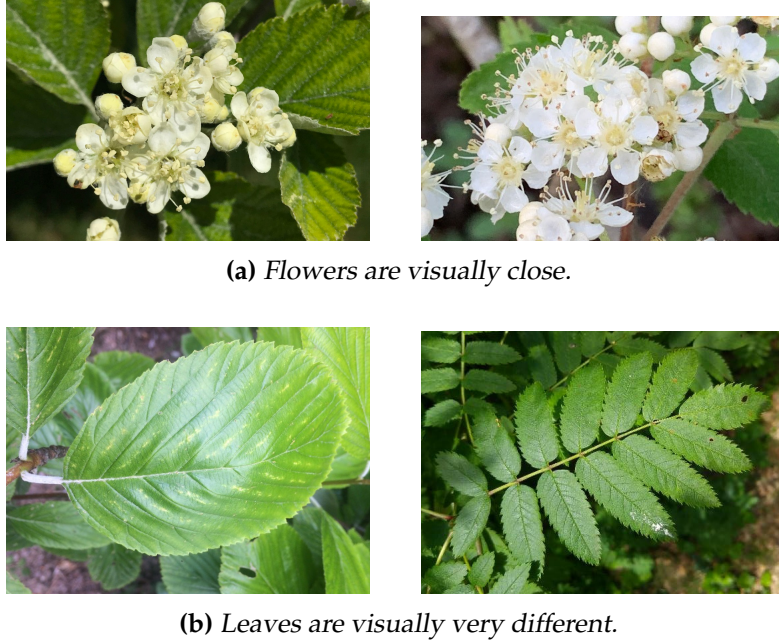
plant specimen is observed in the field. This plant has a single correct label – its species name – associated to it. If an expert could observe this specimen, he could give the exact species name with very high confidence. However, by taking a photo of an organ of the plant, some discriminative information is destroyed and it might be hard to recover the exact species name, even for an expert. To summarize, an object o – in our case the plant on the field – is observed, however, the input data collected x actually comes from measurements of this object – a picture of this plant – which introduces uncertainty on the true label y – the species name.

We illustrate this with a simple example in the case of plant species recognition shown in Figure 1.4. Here, we compare two species: *Sorbus aria* and *Sorbus aucuparia*. If we look solely at the flowers, the two species seem very close visually. However, the leaves have very different structures: the first one has simple leaves whereas the other has compound leaves. This difference makes it very easy to discriminate between these two species by looking at the leaves. The two concepts – *Sorbus aria* and *Sorbus aucuparia* – are thus very well defined, there is no ambiguity between them, and they are easily distinguishable if we could observe the specimen on the field. However, given solely an image of an organ, we might not always be able to determine the exact species. The image acquisition procedure has injected some ambiguity in the problem which can not be removed without additional information.

On the other hand, model uncertainty mainly comes from the fact that the model is learned using a training dataset. This data is limited in size and, for some species, there is not enough images for the model to correctly and reliably recognize them. While this uncertainty is reduced by enriching the dataset with new annotated images, in practice, it is important to detect its presence to identify when the output of the model should not be trusted.

1.2.3 What to do with this information?

If we assume we have a perfect knowledge of the problem and of the existing uncertainties, what should we do with this information? What type of decisions should we take depending on the uncertainty? Actually, the first question hidden in the previous one is: how do we represent these different types of uncertainties? This is a broad question, we will not cover it here as a lot



(a) Flowers are visually close.

(b) Leaves are visually very different.

Figure 1.4: In the case of plants, different organs might have different levels of ambiguity. Here, we compare flowers and leaves from *Sorbus aria* (left column) and *Sorbus aucuparia* (right column). If we could observe these specimens on the field, there would not be any uncertainty. However, the image acquisition procedure can introduce some ambiguity.

of alternatives exists (Hüllermeier and Waegeman, 2019). But as we are interested in the final decision, we will focus on the uncertainty in the predictions.

An omniscient knowledge of the problem means that one knows exactly the data-generating process modeled as $\mathbb{P}_{\mathbf{X},Y}$ which can be decomposed as the marginal probability of the input space, $\mathbb{P}_{\mathbf{X}}$, and the conditional probability of the class given the input

$$\eta_k(\mathbf{x}) := \mathbb{P}_{\mathbf{X},Y} [Y = k \mid \mathbf{X} = \mathbf{x}].$$

This last probability is actually tied to the irreducible error, indeed the minimum error rate of a classifier – which is achieved by the classifier always predicting the most probable class – is equal to

$$\inf_h R(h) = 1 - \mathbb{E}_{\mathbf{X}} \left[\max_k \eta_k(\mathbf{X}) \right].$$

This quantity, $\eta(\mathbf{x})$, thus captures the task ambiguity. When we try to estimate it using an estimator $\hat{\eta}(\mathbf{x})$, we are uncertain of the exact value of $\eta(\mathbf{x})$. This epistemic uncertainty can be modeled as a probability distribution over the possible values taken by this quantity. In the classification setting, $\eta(\mathbf{x})$ is a categorical distribution, i.e. $\forall k, \eta_k(\mathbf{x}) \in [0, 1]$ and $\sum_k \eta_k(\mathbf{x}) = 1$. Therefore, it is a point on the probability simplex and epistemic uncertainty is represented as a probability distribution over this space. The resulting representation is illustrated in Figure 1.5.

Figure 1.6 shows different examples illustrating the main forms of uncertainty which can occur in classification. In some cases, we can be very confident that there is no ambiguity about the class (Figure 1.6a): we can thus trustfully predict this class. In other cases, we are very confident that there is an ambiguity (Figure 1.6b): we are sure that predicting a single class – even if it is the most probable one – will induce a high error rate. Finally, we can also be unsure about the value of the ambiguity (Figure 1.6c). In which case, it is hard to tell which is the correct class to predict and how much error this prediction will induce.

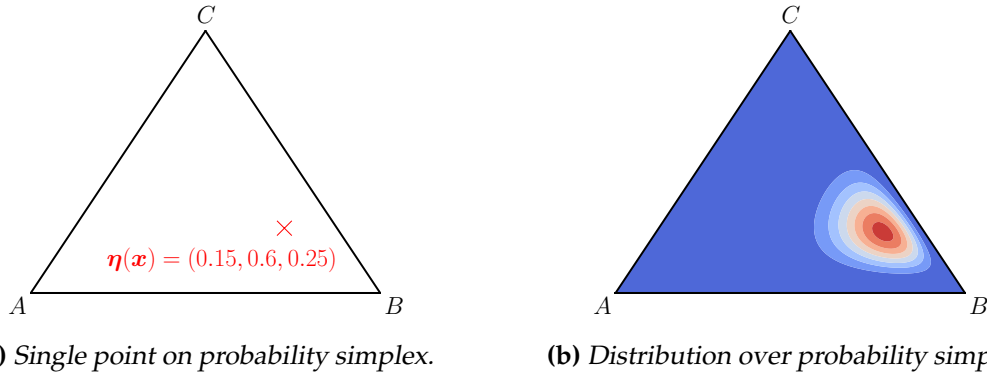


Figure 1.5: Uncertainty representation we use here. In this case, we have 3 classes: A , B and C . A point on the probability simplex, shown as a triangle, represents a value of the aleatoric uncertainty $\eta(x)$. It's exact value is not known in practice. The uncertainty over its value is represented as a distribution over the simplex.

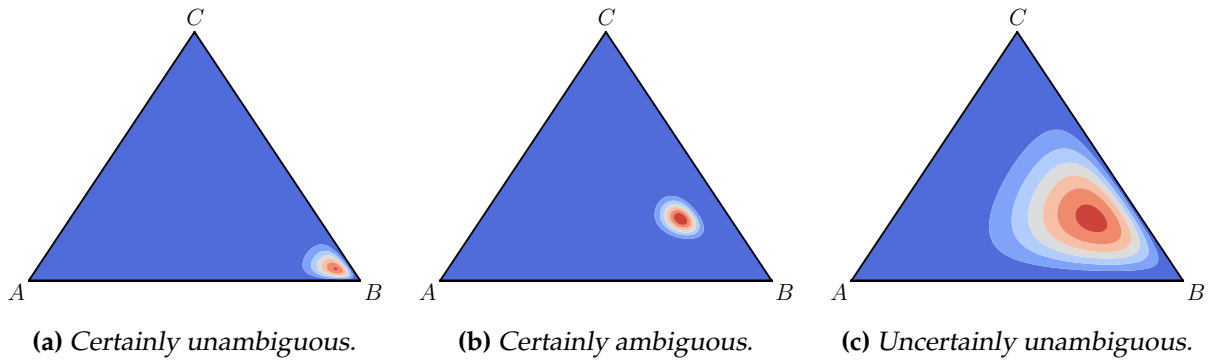



Figure 1.6: Examples of different possible uncertainty over the simplex.


Being able to quantify and estimate the uncertainty as in the examples above is interesting, but, in practice, this raises a question: how can we know that we are correctly estimating this uncertainty information? It is hard to tell in practice if one does a good job at estimating and separating the different forms of uncertainties without a good metric. Unfortunately, as we will see, finding such a measure of performance assessing the quality of our estimators is hard to find in practice.

This brings us to another question: even if we could accurately capture and measure the uncertainty, what should we do with this information? As illustrated with the Pl@ntNet use case, plenty of different decisions can be taken with this additional information. In this PhD, we will focus on two main categories of such decisions which encapsulate a lot of them. The first one consists of *refusing to answer* in presence of uncertainty, while the second one predicts *sets of candidate labels* instead of taking a bet and predicting a single class. They are illustrated in Figure 1.7. As we will see, there exist several possible formalizations for each of these problems depending on the scenario considered and its constraints. Interestingly, some of these formulations are associated with a measure of performance which can be used as an indirect way to assess the quality of uncertainty estimators. They can thus be exploited to tackle the issue of measuring the performance of estimators of uncertainty.


$$h : \mathcal{X} \rightarrow \mathcal{Y}$$


$$\mapsto \text{Sorbus aria}$$

(a) Standard classification.

$$h : \mathcal{X} \rightarrow \mathcal{Y} \cup \{\text{"I don't know"}\}$$


$$\mapsto \text{"I don't know"}$$

$$h : \mathcal{X} \rightarrow \mathcal{P}(\mathcal{Y})$$


$$\mapsto \{\text{Sorbus aucuparia}, \text{Sorbus aria}\}$$

(b) Classification with reject option.

(c) Set-valued classification.

Figure 1.7: Standard classification setting can be modified in order to make the predictions of the classifier closer to that of a human when in presence of task ambiguity.

1.3 Summary of contributions

This manuscript studies the two previously stated problems and their relationship with uncertainty information. We will keep in mind that the methods we are interested in should be compatible with the constraints of the Pl@ntNet project. These constraints are not specific to this project and the methods proposed can be applied to a much broader scope of use cases.

In particular, we focus on image classification which requires the usage of deep learning models to achieve state-of-the-art performance. As we will see, these methods are not well understood theoretically. We will thus use them mainly as black boxes which we will not try to open and investigate. On the contrary, we will rather use them in their standard settings without trying to play with their architectures, the learning procedure, etc.

Another constraint comes from the fact that we need to deal with these different uncertainty problems in a single pipeline. We can not afford to have specialized models trained entirely for each of them, not solely for computational concerns, but rather because the requirements can change through time and the exact specifications are not known beforehand. We thus need a flexible system that allows handling all of them at the same time. This is in accordance with the previous constraint as we will focus on methods that are built around those models once the training was performed.

The manuscript is organized as follows:

- In [Chapter 2](#), we complete this introduction by providing a general positioning and state-of-the-art related to uncertainty estimation which completes this introduction. We briefly recall the statistical learning framework and present how neural networks have become the standard method for image classification. We then detail what are the current limits of our theoretical understanding of these models and why we will essentially consider them as black boxes in this thesis. Next, we develop the previous discussion of [Section 1.2.2](#) on the different sources of task ambiguity to stress out that this uncertainty occurs more often than expected thus highlighting the need to explicitly handle it in practice. Finally, we present in more detail the different scenarios of [Section 1.1.2](#) and elaborate on how they differ from the settings we focus on in this manuscript.
- In [Chapter 3](#), we study how uncertainty information can be exploited to tackle classification

with reject option. In this framework, the predictor is a pair containing a classifier and a rejector. We focus on a specific instance of this framework in which the classifier is already trained and we solely need to learn the rejector to filter its decisions. We first study this setting and show that it is connected to binary cost-sensitive classification and bipartite ranking tasks in which the label corresponds to whether the classifier has made an error for a given input or not. This essentially means that an efficient rejector needs to learn the ordering of the point-wise error rate of the classifier. We then analyze what uncertainty information is useful for rejection and show that, actually, both task ambiguity and model uncertainty are important. We show why some (but not all) confidence-based criterion – exploiting the scoring function used by the classifier to make its decision – are consistent for rejection. In particular, this is the case of scoring functions learned using negative log-likelihood, highlighting the usefulness of probability calibration approaches based on this loss. Furthermore, our analysis shows that dispersion-based criteria, although commonly used in practice, are not consistent for rejection when in presence of task ambiguity. If confidence-based and dispersion-based criteria essentially capture, respectively, task ambiguity and model uncertainty, it would seem reasonable to combine them to build a better rejector. Unfortunately, we empirically show that it is difficult to separate both forms of uncertainty and recombine them properly.

- In [Chapter 4](#), we focus on the first form of uncertainty, task ambiguity, and study natural frameworks to handle it: set-valued classification. There are several ways to predict sets, the most naive approach being to predict the K most probable classes. This method, known as top- K classification, however, assumes that all the samples have the same level of ambiguity which is known to be wrong in most cases. Instead, we propose to use average- K : the predictor can output sets of different sizes but on average their size must be equal to K . We study when average- K is most beneficial compared to top- K . In particular, we introduce and illustrate a notion of heterogeneity of ambiguity, different from the variance, which quantifies this gain. We then propose natural estimation procedures based on strongly proper losses for top- K and average- K and show that they are in fact consistent. These results imply that such set-valued classifiers can be directly built from the scores predicted by neural networks without modifying and adapting their training process which we verify empirically. Finally, we carry out experiments on a variety of real-world datasets that show that average- K classification always gives better results than top- K in practice.
- In [Chapter 5](#), we study other adaptive set-valued classification formulations proposed in the literature. Because there exists a wide variety of such formulations, we introduce a general framework unifying most of them and allowing their joint analysis. These approaches are characterized by a trade-off between two quantities: error rate and set size, which can be measured either point-wisely or on average. We propose to set this trade-off by formulating set-valued classification as a constrained optimization problem enforcing constraints on one of this quantity and minimizing the other. There is not a good or a bad way to set this trade-off, we show that it actually depends on the scenario considered. For each of these formulations, we give their optimal set-valued predictor which can be expressed either as a thresholding strategy on the conditional probability or as a (potentially) adaptive top- K . We show how to estimate them and analyze empirically how their constraints are satisfied in practice. We then study their relative advantages and weaknesses on real-world data. Finally, we introduce hybrid approaches mixing several of the previous constraints in order to mitigate the weaknesses of the different formulations.

1.4 Publications

The work presented in this manuscript resulted in the following papers which were submitted to journals:

- Lorieul, T. and Joly, A. (2020). Leveraging uncertainty information to reject predictions made by a previously trained classifier
- Lorieul, T., Joly, A., and Shasha, D. (2020). Average-K classification: when and how to predict adaptive confidence sets rather than top-K
- Chzhen, E., Denis, C., Hebiri, M., and Lorieul, T. (2020). Set-valued classification—overview via a unified framework

Other works published during this PhD which are not included in this manuscript are listed here:

- Lorieul, T., Pearson, K. D., Ellwood, E. R., Goëau, H., Molino, J.-F., Sweeney, P. W., Yost, J. M., Sachs, J., Mata-Montero, E., Nelson, G., Soltis, P. S., Bonnet, P., and Joly, A. (2019). Toward a large-scale and deep phenological stage annotation of herbarium specimens: Case studies from temperate, tropical, and equatorial floras. *Applications in Plant Sciences*, 7(3):e01233
- Pearson, K. D., Nelson, G., Aronson, M. F. J., Bonnet, P., Brenskelle, L., Davis, C. C., Denny, E. G., Ellwood, E. R., Goëau, H., Heberling, J. M., Joly, A., Lorieul, T., Mazer, S. J., Meineke, E. K., Stucky, B. J., Sweeney, P., White, A. E., and Soltis, P. S. (2020). Machine learning using digitized herbarium specimens to advance phenological research. *BioScience*, 70(6):610–620
- Cole, E., Deneu, B., Lorieul, T., Servajean, M., Botella, C., Morris, D., Jojic, N., Bonnet, P., and Joly, A. (2020). The GeoLifeCLEF 2020 dataset. *arXiv preprint arXiv:2004.04192*
- Deneu, B., Lorieul, T., Cole, E., Servajean, M., Botella, C., Morris, D., Jojic, N., Bonnet, P., and Joly, A. (2020). Overview of LifeCLEF location-based species prediction task 2020 (GeoLifeCLEF). *CLEF task overview*
- Joly, A., Goëau, H., Kahl, S., Botella, C., De Castaneda, R. R., Glotin, H., Cole, E., Champ, J., Deneu, B., Servajean, M., Lorieul, T., Vellinga, W.-P., Stöter, F.-R., Durso, A., Bonnet, P., and Müller, H. (2020a). LifeCLEF 2020 teaser: Biodiversity identification and prediction challenges. In *European Conference on Information Retrieval*, pages 542–549. Springer
- Joly, A., Goëau, H., Kahl, S., Deneu, B., Servajean, M., Cole, E., Picek, L., Ruiz de Castañeda, R., Bolon, I., Durso, A., Lorieul, T., Botella, C., Glotin, H., Champ, J., Eggel, I., Vellinga, W.-P., Bonnet, P., and Müller, H. (2020b). Overview of LifeCLEF 2020: A system-oriented evaluation of automated species identification and species distribution prediction. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction*, pages 342–363. Springer International Publishing
- Lorieul, T. and Joly, A. (2019). Vers un désenchevêtrement de l’ambiguïté de la tâche et de l’incertitude du modèle pour la classification avec option de rejet à l’aide de réseaux neuronaux. In *Conférence sur l’Apprentissage automatique (CAp)*

II

Positioning and state-of-the-art

Contents

2.1	Supervised image classification	15
2.1.1	Statistical learning theory: supervised classification setting	15
2.1.2	Image classification: from hand-crafted features to neural networks	17
2.1.3	Limits of our theoretical understanding of neural networks	17
2.2	Predictive uncertainty in classification	19
2.2.1	Different types of uncertainties	19
2.2.2	Sources of task ambiguity	20
2.2.3	Sources of model uncertainty	22
2.3	Making decisions in presence of uncertainty	24
2.3.1	Prediction rejection and sample selection frameworks	25
2.3.2	Set prediction frameworks	26

2.1 Supervised image classification

The aim of this section is to provide a brief overview of supervised image classification and to present the limitations of the current state-of-the-art methods, i.e. of neural networks. It is not meant to be exhaustive but rather to set the global framework in which we place ourselves in the thesis.

2.1.1 Statistical learning theory: supervised classification setting

In this manuscript, we use the theoretical framework provided by the statistical learning theory (Shalev-Shwartz and Ben-David, 2014) and we focus on the *multi-class classification* setting. In this framework, we are given data samples $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$. Each pair (x_i, y_i) is composed of an *input* or *instance* x_i and a *label* y_i . Each input x_i comes from an input set \mathcal{X} and each label y_i from an output set \mathcal{Y} , i.e. $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$. Typically, in our case, $\mathcal{X} \subset \mathbb{R}^d$ is the set of images while $\mathcal{Y} = \{1, \dots, C\}$, where C the number of classes, is the set of species we want to

classify. The joint space $\mathcal{X} \times \mathcal{Y}$ is supposed to be a probabilistic space with probability measure $\mathbb{P}_{\mathbf{X},Y}$ and the data is sampled from it, i.e. $(\mathbf{x}_i, y_i) \sim \mathbb{P}_{\mathbf{X},Y}$. This joint measure can be decomposed into the marginal distribution measure over \mathcal{X} , $\mathbb{P}_{\mathbf{X}}$, and the conditional distribution of Y given an input \mathbf{x} denoted $\boldsymbol{\eta}(\mathbf{x}) = (\eta_1(\mathbf{x}), \dots, \eta_C(\mathbf{x}))$ and equal to

$$\forall k \in \mathcal{Y}, \quad \eta_k(\mathbf{x}) := \mathbb{P}_{\mathbf{X},Y} [Y = k \mid \mathbf{X} = \mathbf{x}].$$

The set $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)$ is called the *training data*. The aim is to learn a classifier – a function f of the form $f : \mathcal{X} \rightarrow \mathcal{Y}$ – which generalizes well on unseen data. This is formalized by minimizing the 0/1-risk defined as follows,

$$R(f) := \mathbb{E}_{\mathbf{X},Y} [\mathbb{1}_{Y \neq f(\mathbf{X})}] = \mathbb{P}_{\mathbf{X},Y} [Y \neq f(\mathbf{X})].$$

This risk can be minimized for every input $\mathbf{x} \in \mathcal{X}$, in which case the resulting minimizer is called the *Bayes classifier* and is equal to

$$f^*(\mathbf{x}) \in \arg \max_k \eta_k(\mathbf{x}). \quad (2.1)$$

Thus, minimizing the risk implies predicting the most probable class for every instance \mathbf{x} . The minimal risk is equal to

$$\inf_{f \in \mathcal{F}} R(f) = R(f^*) = 1 - \mathbb{E}_{\mathbf{X}} \left[\max_k \eta_k(\mathbf{X}) \right].$$

In practice the conditional probability $\boldsymbol{\eta}(\mathbf{x})$ is unknown and, in general, minimizing the previous risk over all the functions $f \in \mathcal{F} = \mathcal{X}^{\mathcal{Y}}$ is hard without any prior knowledge. This is known as the *no-free-lunch theorem* (Wolpert, 1996; Shalev-Shwartz and Ben-David, 2014). To overcome this difficulty, it is necessary to introduce some *bias* (Mitchell, 1980), this bias is often referred to as *inductive bias* or *learning bias*. One way to do so is to restrict the set of functions to a *hypothesis class* $\mathcal{H} \subset \mathcal{F}$ and the aim is to find the classifier $h \in \mathcal{H}$ which minimizes the risk. As we do not have complete knowledge of the data-generating process $\mathbb{P}_{\mathbf{X},Y}$, we can build an estimator \hat{h} of the minimizer using the training data. The risk of this estimator \hat{h} can then be decomposed as:

$$R(\hat{h}) = \underbrace{\inf_{f \in \mathcal{F}} R(f)}_{\text{irreducible error}} + \underbrace{\left(\inf_{h \in \mathcal{H}} R(h) - \inf_{f \in \mathcal{F}} R(f) \right)}_{\text{approximation error}} + \underbrace{\left(R(\hat{h}) - \inf_{h \in \mathcal{H}} R(h) \right)}_{\text{estimation error}}. \quad (2.2)$$

- The *irreducible error* is the minimum risk achievable by any classifier, it is due to the intrinsic noise of the problem;
- The *approximation error* quantifies if the choice of the hypothesis \mathcal{H} allows to correctly approximate the true risk minimizer f^* ;
- The *estimation error* measures how well the optimal hypothesis from \mathcal{H} is estimated from the learning algorithm using only the training data.

The control of the two last terms has been extensively studied in the literature. In particular, there exists a trade-off between those two which is often referred to as the *complexity trade-off*. Intuitively, the more complex a hypothesis class, the wider variety of the functions it can approximate well, and the lower the approximate error. On the other hand, a complex hypothesis class requires more training samples for a learning algorithm to find its risk minimizer, thus

increasing the estimation error. A lot of different mathematical complexity measures tied to this notion of learnability have been proposed in the literature. For instance, one of the earliest measures was *VC-dimension* which is purely combinatorial and independent of the data (Vapnik and Chervonenkis, 1971). Another important and finer measure is *Rademacher complexity* which, on the contrary, is data-dependent (Bartlett and Mendelson, 2002; Koltchinskii and Panchenko, 2002). We will not describe in detail these notions here but they are important tools to study the generability of learning algorithms.

2.1.2 Image classification: from hand-crafted features to neural networks

In image classification, the structure of the data is particular. Indeed, images typically have a high dimensionality: for instance, 256×256 color images have a dimension of $3 \times 256 \times 256 \approx 2.10^5$. However, this high dimensionality hides the fact that neighbor pixels are highly correlated. Images thus typically require to use an adapted representation space.

Historically, hand-crafted features such as *histograms of oriented gradients* (HOG) (Dalal and Triggs, 2005), *scale-invariant feature transform* (SIFT) (Lowe, 1999), *local binary patterns* (LBP) (Ojala et al., 1994), and others were first proposed. These features attempted to use prior knowledge to reduce the dimensionality of the images while keeping as much discriminative information as possible. Supervised classifiers such as *support vector machines* (SVM) (Cristianini et al., 2000) were then learned from this representation space.

The second wave of image classification algorithms used mid-level representation approaches. They consisted of the dense extraction of the previous hand-crafted features on local patches of the images. A global representation was then learned from these features using unsupervised learning techniques such as *K-means clustering* (Lloyd, 1982) or *Gaussian mixture models* (GMM). These led to representations such as *bag of visual words* (BoVW) (Csurka et al., 2004) and *Fisher vectors* (FV) (Sánchez et al., 2013) among others. A classifier is then applied on top of this learned representation.

From this point, research has gone towards increasing the supervision of the representation learning and jointly learning the whole classification pipeline. Eventually, this led to using deep learning models (LeCun et al., 2015). The first success of neural networks in image classification occurred in 2012 when Krizhevsky et al. (2012) won the ImageNet visual classification challenge (Russakovsky et al., 2015). Since then, deeper and deeper models were developed lowering the state-of-the-art error rate on this challenge with models such as VGG (Simonyan and Zisserman, 2015), Inception (Szegedy et al., 2015), ResNet (He et al., 2016), and more.

We will not detail here how such models work as it is in constant evolution and a lot of resources providing introductions to such methods already exist such as LeCun et al. (2015); Goodfellow et al. (2016). Instead, in the next section, we present the main limitations of our understanding of those models from a theoretical point-of-view.

2.1.3 Limits of our theoretical understanding of neural networks

From a theoretical point-of-view, little is known about why neural networks work so well in practice. This comes from the fact that the generic theoretical tools developed to analyze classical machine learning algorithms are not directly applicable to neural networks. In this section, we describe the limits of our understanding of this type of model which we split into three main

categories: optimization, approximation, and generability.

Optimization Before learnability considerations, the first difficulty raised by neural networks is the optimization procedure used. The optimization algorithm used, stochastic gradient descent (SGD), is known to find the optimal solution for convex optimization problems (Robbins and Monro, 1951; Bottou et al., 2018). However, in the case of neural nets, the optimization problem is highly non-convex with a lot of suboptimal local minima (Li et al., 2018). It is thus surprising at first that SGD manages to minimize the loss even on the training set – before talking about generalizing well on unseen data.

In fact, by default, SGD performs poorly (Bengio et al., 2007). Nevertheless, by carefully setting the random initialization (Glorot and Bengio, 2010), choosing an adapted activation function such as ReLU (Glorot et al., 2011) and/or using momentum (Sutskever et al., 2013), suddenly, it becomes very efficient. The design choices of the architecture of the network has also a great impact on optimization. Indeed, the depth of the network and the presence of skip connections can have a major influence on the trainability of those models (Li et al., 2018). To accelerate training, some adaptive gradient descent approaches have been proposed. These methods try to adaptively modify the learning rate according to the history of steps such as RMSProp (Tieleman and Hinton, 2012), AdaGrad (Duchi et al., 2011), Adam (Kingma and Welling, 2014), etc. However, it has been shown that they could fail or be significantly worse than SGD even on real-world datasets (Wilson et al., 2017).

From a theoretical point-of-view, it has been shown that it is possible to make assumptions under which there are no poor local minima, i.e. every minimum is actually a global minimum (Soudry and Carmon, 2016; Kawaguchi, 2016). However, these assumptions are not matched in practice and thus there still exists a gap between theory and practice.

These different points highlight the difficulty of the analysis of the training procedure using classical optimization tools.

Approximation Neural networks with a single hidden layer are known to be *universal approximators*: they can approximate any (Borel measurable) function with arbitrary precision if given sufficiently many parameters (Hornik et al., 1989). Similar works characterizing the approximation properties of deep neural networks were also published (Montufar et al., 2014). These results, however, hide the fact that the number of parameters required to approximate some functions depends exponentially on the expected precision.

To fill this limitation, a line of works has focused on understanding what class of functions can be *efficiently* – i.e. with relatively few parameters – approximated by deep networks but not by shallow ones. It has for instance been shown by Poggio et al. (2017) that deep networks approximate well a type of composition of functions: hierarchically local compositional functions. Those works usually start by taking a class of functions of interest before checking if neural networks approximate this class efficiently. On the contrary, another line of work uses tools from *approximation theory* to capture what is the *exact* class of functions efficiently approximated by neural networks and to study its properties (Gribonval et al., 2019).

For a more complete review of the literature of approximation properties of neural networks, we refer to Gühring et al. (2020).

Generability As said previously, in statistical learning theory, a key concept related to generability is the complexity of the function classes. These tools were applied to neural networks and their complexity has been analyzed using different measures. In particular, it is known

that the VC-dimension of those neural networks is high and increases both with the number of weights and the number of layers (Bartlett et al., 2019). Moreover, it has empirically been shown that neural networks can fit data samples which are randomly labeled (Zhang et al., 2017). This tends to show that the Rademacher complexity of these models is also large. However, these complexity measures, by considering at the same weight all the functions in the hypothesis class \mathcal{H} , allow only to carry out a worst-case analysis. Typically, a learning algorithm can have a better performance by selecting functions in a favorable subset of \mathcal{H} . The generability of these algorithms is better analyzed using local complexity measures such as the *local Rademacher complexity* (Bartlett et al., 2005; Koltchinskii, 2006).

In the case of deep learning, the hope is that the training procedure, and in particular the stochasticity of SGD, introduces an implicit bias towards a better generalizing subset of \mathcal{H} . In fact, it has been shown empirically that, without this stochasticity, the model tends to converge to sharp minima (Keskar et al., 2017; Hoffer et al., 2017). Although converging to such minima does not necessarily imply that the model will generalize poorly (Dinh et al., 2017), explicitly biasing the training procedure towards large valleys and flat minima helps to generalize well (Hochreiter and Schmidhuber, 1997; Chaudhari et al., 2019).

Other works search for sufficient conditions for good generalization with neural networks and try to empirically check if SGD enforces such conditions. This line of work follows the seminal paper of Bartlett (1997) which has shown that controlling the ℓ_1 -norm of the flatten vector of all weights of a neural network allows to generalize well (Bartlett, 1997). Unfortunately, in practice, this norm, for a neural network trained using the standard procedure, can be large. Besides, it can be shown that we can reparameterize the model to increase arbitrarily the ℓ_1 -norm while preserving the function it represents. Some works have thus proposed and analyzed different norms which, if controlled, can also be shown to generalize well (Neyshabur et al., 2015a,b; Bartlett et al., 2017; Golowich et al., 2019; Liang et al., 2019). Although the learning procedure was not found to implicitly control such norms, adding an explicit regularization based on these norms was shown to be helpful.

To summarize, studying neural networks requires theoreticians to develop and use more complex tools than for other machine learning models. This lack of clear understanding results in absence of theoretical guarantees on the properties of the learned model. In particular, in our case, it is difficult to make reasonable assumptions on the representation learned and on the scores predicted by such models for individual samples which could have been exploited to derive uncertainty information. For these reasons, we will essentially consider those models as black boxes.

2.2 Predictive uncertainty in classification

In the previous section, we have presented the supervised classification framework. We now present the different forms of uncertainty arising in such problems and describe their sources. This will allow us to better decide what we should do depending on the uncertainty present in a given task.

2.2.1 Different types of uncertainties

As stated in the introduction, in general, uncertainty is categorized into two forms (Der Kiureghian and Ditlevsen, 2009):

- *aleatoric uncertainty*: uncertainty raising from the intrinsic randomness of the underlying process, it is considered to be *irreducible*;
- *epistemic uncertainty*: uncertainty caused by a lack of knowledge, it is considered to be *reducible* given additional information.

This dichotomy depends heavily on the context and what we are allowed to do. In the supervised classification setting, we can analyze the existing uncertainties by looking at the decomposition of the error rate of a predictor \hat{h} of Equation (2.2) which we recall here:

$$R(\hat{h}) = \underbrace{\inf_{f \in \mathcal{F}} R(f)}_{\text{irreducible error}} + \underbrace{\left(\inf_{h \in \mathcal{H}} R(h) - \inf_{f \in \mathcal{F}} R(f) \right)}_{\text{approximation error}} + \underbrace{\left(R(\hat{h}) - \inf_{h \in \mathcal{H}} R(h) \right)}_{\text{estimation error}}.$$

Each of these error rate terms can be associated with a different form of uncertainty which we will detail in the rest of this section:

1. *intrinsic uncertainty*: each task has an intrinsic ambiguity which generates an irreducible classification error;
2. *approximation uncertainty*: there is an uncertainty in the choice of the hypothesis class \mathcal{H} , this choice is based on prior knowledge which might be incomplete and result in an approximate error;
3. *estimation uncertainty*: the limited amount of data available to estimate the best model in the hypothesis class \mathcal{H} generates an uncertainty in the choice of the best hypothesis resulting in an estimation error.

In most classification scenarios, the first uncertainty is irreducible and thus is often referred to as *aleatoric* uncertainty while the two others are reducible and thus together they form the *epistemic* uncertainty (Kendall and Gal, 2017; Hüllermeier and Waegeman, 2019). These uncertainties are found under different names in the literature. For instance, Hüllermeier and Waegeman (2019) uses “model uncertainty” and “approximation uncertainty” for, respectively, the second and third types of uncertainty. On the other hand, Gal (2016) uses respectively “structure uncertainty” and “uncertainty in model parameters” for them and “model uncertainty” to refer to them when they are grouped together.

In most of the cases, the first term is merely considered as an irreducible *noise* term (Kendall and Gal, 2017). A classical assumption to obtain fast convergence rates in statistical learning theory actually consists in assuming that the ambiguity is small (Tsybakov et al., 2004; Boucheron et al., 2005). In this thesis, we will rather consider scenarios in which this ambiguity is non-negligible and thus must be correctly handled. For this reason, we will call this uncertainty *task ambiguity* while referring to the other two as *model uncertainty*. In the next section, we describe rather common scenarios where task ambiguity appears.

2.2.2 Sources of task ambiguity

In this section, we present different sources of task ambiguity arising when collecting data for multi-class classification tasks. We introduce a general latent data-acquisition process model which is illustrated in Figure 2.1. This model assumes that an object of interest o is observed, e.g. a plant in the field, and that it is associated with a single label y , e.g. the plant species. A measurement x is made from this object, e.g. a picture of the plant. Ideally, all the information is preserved in the data-acquisition procedure and the correct label y can be exactly extracted

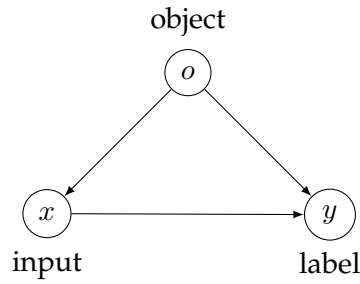


Figure 2.1: Latent data-acquisition process model we use to analyze the different sources of task ambiguity in classification.

from the input x . However, in practice, all the links of Figure 2.1 are susceptible to introduce uncertainty in this process. We analyze four main sources here, they are shown in Figure 2.2 and illustrated in Figure 2.3.

Label noise (Figure 2.2a) When collecting the label, some noise might be introduced. This occurs for instance when the annotator is not an expert which can happen if there are too much data and not enough experts available to annotate it. We refer the reader to Frénay and Verleysen (2013) for more details on this source of task ambiguity. We will leave this source apart for the rest of the manuscript and rather focus on the other sources.

Measurement noise (Figure 2.2b) A single unambiguous object is measured with its correct label collected but the measurement introduces some noise. It corresponds to the example developed in the introduction (Figure 1.4). Here, we show another example from ImageNet in Figure 2.3a: a single animal is present in the picture, however, because its head is cropped, it is not possible to recover its exact name. In this case, the image acquisition procedure has injected some ambiguity in the problem.

Multiple objects uncertainty (Figure 2.2c) When collecting the data, several (unambiguous) objects are present but a single one is annotated. The measurement captures information from all the objects but the label contains only partial information. This illustrated in Figure 2.3b: several fruits are present in the photo, choosing one will be imprecise. In image classification, this is typically the case in single-positive multi-label datasets: it is intrinsically a multi-label problem but only a single positive label is collected rendering this a multi-class classification task.

Non-exclusive concepts (Figure 2.2d) The classes we want to detect are not always non-exclusive concepts. In such a case, several labels actually correspond to a given object. An example is shown in Figure 2.3c: the picture contains a single car which is both a sports car and a convertible car. This typically corresponds to cases where we want to detect attributes when several can match the object. Like the previous case, it is also intrinsically a multi-label problem but where we collect only one positive label.

In our plant recognition scenario, all of the above sources are present. Indeed, part of the data is annotated via a crowdsourcing system that can inject label noise. The pictures taken by users typically destroy part of the discriminative information needed to recover the exact species name. The photos can also contain several different plants, sometimes due to user negligence, but also because some plants can be hard to isolate on the field. Finally, species taxonomy is in constant evolution: separated species at some point in time can be considered as synonyms later on or a single species can be split into several ones. Thus, in some sense, the species are

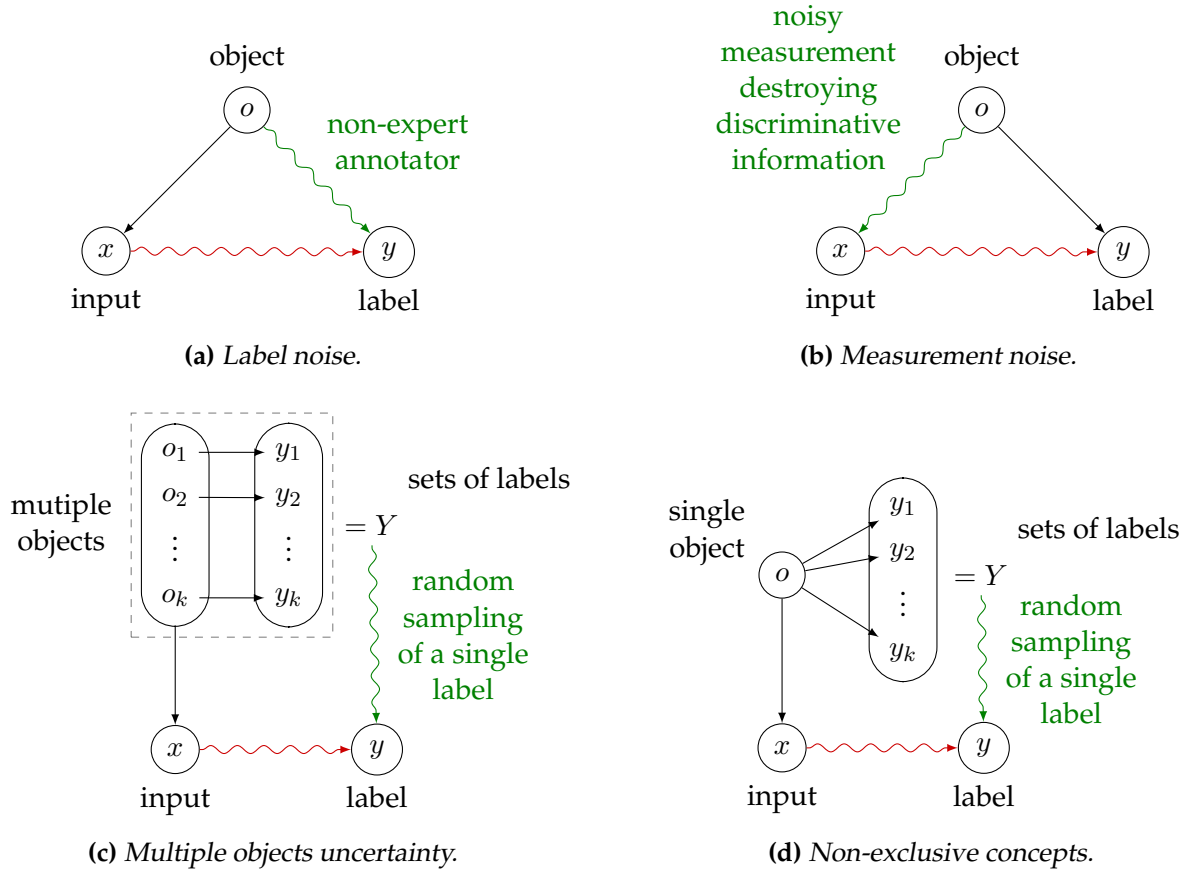


Figure 2.2: Four possible sources of ambiguity in the data-acquisition process. Black straight arrows indicate the absence of uncertainty while wavy arrows indicate the presence of uncertainty. Green arrows show the source of the uncertainty while red arrows highlight the resulting uncertainty.

not always exclusive concepts. However, of all these sources of task ambiguity, some are more prevalent than others. In our case, and in fine-grained visual classification problems in general, we argue that measurement noise is the most present and thus it will be the one we focus on.

In any case, the task ambiguity is directly linked to the data-generating process $\mathbb{P}_{\mathbf{X},Y}$. It is easy to formalize it as it is entirely captured by the conditional probability

$$\eta_k(\mathbf{x}) = \mathbb{P}[Y = k \mid \mathbf{X} = \mathbf{x}].$$

It is thus clear that this uncertainty can be represented using a categorical distribution over the classes for each instance \mathbf{x} .

2.2.3 Sources of model uncertainty

While task ambiguity is easy to define and is captured entirely by the conditional probability, model uncertainty is more complex to formalize. For this reason, we propose to define model uncertainty as all the uncertainty which is not part of the task ambiguity. This definition hides various sources of model uncertainty which fit in two main categories: (i) the uncertainty arising from the choice of the learning bias, and (ii) the one resulting from the choice of the model within this bias. This distinction actually goes beyond the decomposition into approximation and estimation uncertainty previously presented.

(a) **impala**, gazelle, hartebeest(b) **banana**, orange, Granny Smith(c) **sports car**, convertible

Figure 2.3: Examples from ImageNet (Russakovsky et al., 2015) showing various sources of task ambiguity. These examples contain either an intrinsically ambiguous image containing a single object associated with a single class (a), multiple objects (b), or a single object with multiple matching attributes (c). The official annotation associated is shown in bold.

Learning bias uncertainty As stated previously, every learning algorithm has to make assumptions referred to as bias (Mitchell, 1980; Wolpert, 1996). In our setting, there are two main types of assumptions: (i) the restriction of classification functions considered by choosing a hypothesis class \mathcal{H} , and (ii) the enforcement of an ordering of the hypotheses in \mathcal{H} by an (implicit or explicit) regularization.

These assumptions influence both the approximation capacity and generability of the resulting model. Indeed, if the chosen \mathcal{H} is too restrictive, it will not allow fitting models close to the real optimal classifier. On the other hand, choosing a hypothesis class \mathcal{H} too wide and complex will hinder the generability of the learned model to new data. Beyond the choice of the hypothesis class, the addition of an implicit or explicit regularization also influences the generability of the resulting model. Indeed, assume that we choose linear models as hypothesis class and that the problem is indeed linearly separable. In most cases, many linear models will fit the training data, however, some will generalize better than others. The learning algorithms can inject biases towards those by adding regularization schemes. For instance, some algorithms maximize the margin such as support vector machines (Cristianini et al., 2000), some enforce sparsity such as ℓ_1 -regularization which is useful when some features are irrelevant for the task (Ng, 2004), others control the size of the parameters such as ℓ_2 -norm regularization (Tikhonov, 1943), etc. The choice of the learning bias is subject to uncertainty: we can not be sure beforehand if we make the correct assumptions.

Typically, a learning algorithm possesses hyperparameters such as the strength of the regularization, the complexity of the hypothesis class, and others, which must be set but are hard to fix manually. Moreover, several competing families of learning algorithms might be suited for a task. Thus, in practice, the inductive bias is not entirely fixed beforehand, it is rather chosen by picking a learning algorithm among a fixed set of candidates. To compare these candidates, *model selection* methods have been derived (Hastie et al., 2009). These techniques – such as bootstrap, cross-validation, leave-one-out – analyze competing models by fitting them on a subset of the training data and measuring their generalization data on the hold-out subset. These methods allow to compare a fixed set of candidate learning algorithms, however, there exist approaches for automated bias selection (Gordon and Desjardins, 1995). This dynamic choice of the learning bias is also one of the goals of *meta-learning* (Vilalta and Drissi, 2002). The existence of these methods stresses the intrinsically uncertain nature of picking the correct bias: because, in general, we can not be sure beforehand of the good learning bias, some principled data-driven methods have been derived.

We will not detail here methods to represent this uncertainty. We refer to Hüllermeier and

Waegeman (2019) which presents some of them such as *credal inference* and *hierarchical Bayes*.

Model choice uncertainty The second form of model uncertainty results from the choice of the parameters of the model within this learning bias. Firstly, during training, we are given a specific sampling of the training data on which we apply our learning algorithm. Another sampling of the training data could have made us choose different parameters for this same model. There is thus an uncertainty related to the fact that we have only access to a finite sampling to use as training data. Furthermore, some training algorithms are intrinsically stochastic. For instance, because of the non-convexity of the optimization problem, training the same neural network using different random initializations and different orderings of the training samples results in different parameters. These models are complementary as they all fit the training data and have a similar average generalization performance.

There are several ways to represent and estimate this second type of model uncertainty. We will only briefly present the main approach applied to neural networks: Bayesian methods (Gelman et al., 2013). These approaches consist in setting an a priori distribution on the parameters modeling our a priori knowledge of the problem. This knowledge is then updated when observing new data in a principled way using Bayes' rule. This method allows modeling the uncertainty in the parameters as a complete posterior distribution over the parameter space. From this distribution, we can make a prediction for a given sample by marginalizing out the uncertainty, an operation known as *inference*. Such Bayesian methods have been applied to neural networks for a long time with some early works such as the works by MacKay (1992) and Neal (1996). Quickly after neural networks came back to forefront, these approaches – denoted *Bayesian neural networks* (BNN) – reappeared with works from Kingma and Welling (2014); Rezende et al. (2014); Blundell et al. (2015); Gal and Ghahramani (2016). The main difficulty with BNNs is that the posterior distribution of the parameters can not be exactly computed. Instead, it is required to use specific approximate Bayesian inference methods. Furthermore, the prior chosen, a standard Gaussian distribution, is likely to be inappropriate and requires to use a trick breaking the Bayesian theory – scaling the prediction using a low temperature – in order to improve the predictive performance of those models (Wenzel et al., 2020). This is thus still an active research area.

To summarize, model uncertainty is harder to define than task ambiguity, moreover, there is also no unique way to represent it. We will consider that, in our setting, the learning bias uncertainty is smaller than the model choice uncertainty. When referring to model uncertainty, we will thus focus mainly on model choice uncertainty.

2.3 Making decisions in presence of uncertainty

The final objective of estimating the uncertainty present in a task is to adapt one's decisions to it. In this thesis, we focus on two different types of decisions: refusing to answer by acknowledging the lack of knowledge ("I do not know") or providing several answers when unsure rather than being forced to take a single shot which will likely be wrong. These different decisions amount to adapting the predictors from usual classifiers of the form,

$$h : \mathcal{X} \rightarrow \mathcal{Y}$$

to predictors of the form,

$$h : \mathcal{X} \rightarrow \mathcal{Y} \cup \{\text{"I don't know"}\}, \quad \text{and,} \quad h : \mathcal{X} \rightarrow \mathcal{P}(\mathcal{Y}).$$

where $\mathcal{P}(\mathcal{Y})$ represents the power set of \mathcal{Y} , i.e. the set of all sets of labels.

There are several distinct scenarios in which these types of predictors are useful. In this manuscript, we consider the previous decisions in the same setting as multi-class classification, i.e. the data is issued by the same data-generating process and is sampled from the joint distribution $\mathbb{P}_{\mathbf{X}, \mathbf{Y}}$. In this case, these two decision frameworks are respectively called *classification with reject option* and *set-valued classification* which we will present in more length later in the manuscript. In this section, we detail the other settings proposed in the literature and highlight how they differ from the ones we focus on in the thesis.

2.3.1 Prediction rejection and sample selection frameworks

Several frameworks consist in rejecting or selecting samples. They differ from one another in the reasons or the context in which they reject. We list and briefly describe them here.

Classification with reject option Classification with reject option allows a predictor to refuse to answer to a query in order to reduce its error rate. In particular, the rejection is performed at test time and test data is assumed to be i.i.d. with the training data, i.e. $(\mathbf{X}, \mathbf{Y}) \sim \mathbb{P}_{\mathbf{X}, \mathbf{Y}}$. It is thus an *in-distribution rejection* framework. In this case, as we will see later in this thesis, the error rate is due to suboptimal classification and to the intrinsic task ambiguity. This setting has been studied for a long time, one of the earliest works being the one of [Chow \(1957\)](#), and since then a growing literature has been developed with works ranging from theoretical formulation, statistical analysis to practical algorithm, see among others [Chow \(1970\)](#); [Herbei and Wegkamp \(2006\)](#); [Bartlett and Wegkamp \(2008\)](#); [Cortes et al. \(2016\)](#); [Ramaswamy et al. \(2018\)](#).

Out-of-distribution detection Contrary to classification with reject option, in out-of-distribution detection, the aim is to reject samples that come from a different distribution than the training data ([Hendrycks and Gimpel, 2017](#); [Liang et al., 2018](#); [DeVries and Taylor, 2018](#)). In this setting, the data-generating process is unchanged for training samples, however, at test time, there are two sources of data. Some samples are i.i.d. with the training data, i.e. $\mathbf{X} \sim \mathbb{P}_{\mathbf{X}}$, while others come from a different distribution $\mathbb{Q}_{\mathbf{X}}$, i.e. $\mathbf{X} \sim \mathbb{Q}_{\mathbf{X}}$. The goal is then to detect the test samples generated by the out-distribution $\mathbb{Q}_{\mathbf{X}}$. For an overview of methods targeted to neural networks, see [Bulusu et al. \(2020\)](#).

Novelty / anomaly / outlier detection These frameworks are related to out-of-distribution detection as they attempt to detect out-of-distribution or abnormal samples ([Chandola et al., 2009](#); [Pimentel et al., 2014](#)). The main difference is that some abnormal samples are available during training and each training sample has an additional “normal” / “abnormal” label. This problem can thus be seen as a binary classification problem with an additional constraint: abnormal samples are very rare, typically only a handful is available. It is thus an extremely imbalanced classification problem rendering usual classification algorithms ineffective. This particular classification setting is usually referred to as *one-class classification*. For a survey of methods targeted to neural networks, see [Chalapathy and Chawla \(2019\)](#).

Open set classification Another related problem is open set classification ([Scheirer et al., 2012](#); [Bendale and Boulton, 2016](#); [Geng et al., 2020](#)). Contrary to usual *closed set* classification where all the classes are known at training time, in this framework, new classes unknown in the training data can appear at test time. The goal is then to detect these unknown classes when they show up and deal with them.

Active learning A last related approach is active learning (Cohn et al., 1994; Settles, 2012). Supervised classification requires to have access to a large amount of annotated data. This annotation can be costly and time-consuming. Moreover, it can be ineffective: annotating samples for which the model is already confident about is a waste of time. Based on this observation, active learning aims to select the samples which annotation is the most useful for the model. Although it is not exactly the same objective, under some assumptions, there are some links with classification with reject option as shown by El-Yaniv and Wiener (2012); Gelbhart and El-Yaniv (2019).

2.3.2 Set prediction frameworks

Similarly, there exist different classification frameworks involving predicting sets of candidate labels rather than single labels. We briefly expose them here and highlight their differences.

Set-valued classification As explained previously, in set-valued classification, the predictor is not a single-label classifier of the form $h : \mathcal{X} \rightarrow \mathcal{Y}$ but rather a mapping from the input space to the set of all subsets of labels, i.e. $h : \mathcal{X} \rightarrow \mathcal{P}(\mathcal{Y})$. The rest of the classification setting is unchanged. In particular, the data-generating process is the same and samples are generated using $(\mathbf{X}, Y) \sim \mathbb{P}_{\mathbf{X}, Y}$. The labels we are given during training and testing are not sets but single classes from \mathcal{Y} . Given an instance $\mathbf{x} \in \mathcal{X}$, the label is thus sampled using the conditional probability, i.e. $y \sim \boldsymbol{\eta}(\mathbf{x})$ where $\forall k \in \mathcal{Y}, \eta_k(\mathbf{x}) = \mathbb{P}[Y = k \mid \mathbf{X} = \mathbf{x}]$. We will give a more detailed overview of these approaches in Chapter 5.

Multi-label classification and extensions At the first glance set-valued classification seems to be directly connected to multi-label classification (Dembczyński et al., 2012; Zhang and Zhou, 2013). This is a recurrent source of confusion due to the fact that in both settings a set of label candidates is predicted by the classifier. However, the crucial difference with set-valued classification lies in the data-generating process. Indeed, in multi-label classification, each input $\mathbf{x} \in \mathcal{X}$ is associated with a set of labels $\mathbf{y} \in \mathcal{P}(\mathcal{Y})$. Thus the labels are no longer sampled from the conditional probability over the label space \mathcal{Y} but are rather sampled from the conditional probability over the powerset of \mathcal{Y} . Some extensions of multi-label classification were proposed. In particular, *multi-label learning with missing labels* consider the case where not all the labels are known and some – positive and negative – labels are missing. A special case is *presence-only data* (Hastie et al., 2009; Mac Aodha et al., 2019) in which a single positive label is collected. In this case, the data-generating process is exactly the same as multi-class classification but we want to recover the latent set of labels. Unlike more general formulations of multi-label classification, this last framework is thus tightly related to set-valued classification.

Conformal prediction Seminal ideas of set-valued classification appear in the works by Vovk et al. (2005) on conformal prediction theory. This theory provides strong guarantees on the error rate: the error rate of the conformal predictor for a new data point given a set of training samples can be controlled solely with an exchangeability assumption. This framework however works in a different setting than set-valued classification. Indeed, it is targeted to the *online transductive* setting: the predictions are made sequentially using the aggregation of all previous data which arrives as a stream. Once a prediction is made on a new instance \mathbf{x} , we are given its true label which we can use for the next prediction. This is opposed to the *offline inductive* setting considered by set-valued classification where a prediction rule is learned on training data before being applied to test data.

III

How to know when not to trust a classifier?

Contents

3.1	From usual classification to classification with reject option	28
3.1.1	Supervised classification formulation	28
3.1.2	Adding a reject option	28
3.2	Learning to reject when the classifier is fixed: theoretical study	29
3.2.1	Fixed cost rejection as a binary cost-sensitive classification problem	29
3.2.2	Linking rejection cost with rejection rate	31
3.2.3	Global rejection as a bipartite ranking task	32
3.3	What uncertainty is needed for rejection?	35
3.3.1	Point-wise error rate decomposition	36
3.3.2	Special case: absence of task ambiguity	36
3.4	Confidence-based criteria	37
3.4.1	A priori, confidence-based criteria provide no guarantees for rejection	37
3.4.2	Negative log-likelihood minimization is consistent for rejection	38
3.5	Leveraging disagreement information	40
3.5.1	Disagreement-based criteria	40
3.5.2	In general, disagreement-based criteria are inconsistent	41
3.5.3	Integrating disagreement to confidence-based criteria	41
3.6	Experiments	42
3.6.1	Confidence-based criteria	42
3.6.2	Disagreement-based and fusion criteria	43
3.7	Conclusion	44

The standard classification formulation aims to minimize the error rate of a classifier of the form $h : \mathcal{X} \rightarrow \mathcal{Y}$, i.e., for each given input $x \in \mathcal{X}$, it predicts a single class. In practice, it is useful to associate a measure of uncertainty to this decision to enable a practitioner to choose whether or not he should trust it. Classification with reject option is a formalization of this decision problem where the decision is now not only to predict a class but also to decide whether this

prediction should be kept or not. In this chapter, we show how this framework can be used to analyze how uncertainty information can be exploited to reject the predictions made by a classifier.

3.1 From usual classification to classification with reject option

The task of refuting the answer provided by a classifier can be nicely formalized in the framework of classification with reject option. In this section, we briefly recall the formulation of classic classification and show how it is simply extended to provide a reject option elucidating when we should not trust the prediction of the classifier.

3.1.1 Supervised classification formulation

Recall that, in the classification scenario, a task is defined given an input space \mathcal{X} and a discrete output space $\mathcal{Y} = \{1, \dots, C\}$ where C is the number of classes. The goal is, given an input $\mathbf{x} \in \mathcal{X}$, to predict a label $y \in \mathcal{Y}$ which amounts in learning a function $h : \mathcal{X} \rightarrow \mathcal{Y}$, the classifier function. The spaces \mathcal{X} and \mathcal{Y} are supposed to be probability spaces with joint probability measure $\mathbb{P}_{\mathbf{X}, Y}$ which can be decomposed into the marginal probability measure over \mathcal{X} , denoted $\mathbb{P}_{\mathbf{X}}$, and the conditional probability of y given \mathbf{x} . This former probability is also called the regression function and is denoted $\eta(\mathbf{x})$ such that

$$\forall k \in \mathcal{Y}, \quad \eta_k(\mathbf{x}) := \mathbb{P}[Y = k \mid \mathbf{X} = \mathbf{x}].$$

The goal is to learn a classifier h which minimizes the risk function defined as

$$R(h) := \mathbb{E}_{\mathbf{X}, Y} [\mathbb{1}_{Y \neq h(\mathbf{X})}] = \mathbb{P}_{\mathbf{X}, Y} [Y \neq h(\mathbf{X})], \quad (3.1)$$

where $\mathbb{1}_P$ is equal to 1 if the logical proposition P is true and to 0 otherwise. This risk is minimized by the Bayes optimal classifier h^* which is equal to

$$h^*(\mathbf{x}) := \arg \max_k \eta_k(\mathbf{x}). \quad (3.2)$$

3.1.2 Adding a reject option

Classification with reject option ([Herbei and Wegkamp, 2006](#); [Cortes et al., 2016](#)) consists in providing a pair of functions (h, r) : a classifier $h : \mathcal{X} \rightarrow \mathcal{Y}$, as in the classical classification setting, associated with a rejector $r : \mathcal{X} \rightarrow \{0, 1\}$. The global predictor is then

$$(h, r)(\mathbf{x}) := \begin{cases} h(\mathbf{x}) & \text{if } r(\mathbf{x}) = 0, \\ \mathbb{R} & \text{otherwise,} \end{cases}$$

where \mathbb{R} denotes a refusal to answer.

The risk of [Equation \(3.1\)](#) needs to be adapted. This is done by fixing a rejection cost $\lambda \in [0, 1]$: the cost of rejecting is equal to λ whereas the cost of accepting to answer but making an error is left to 1. This results in the following risk:

$$R_\lambda(h, r) := \mathbb{E}_{\mathbf{X}, Y} [\mathbb{1}_{Y \neq h(\mathbf{X})} \mathbb{1}_{r(\mathbf{X})=0} + \lambda \mathbb{1}_{r(\mathbf{X})=1}]. \quad (3.3)$$

If we define the *error rate* $\mathcal{E}(h, r)$ and the *rejection rate* $\mathcal{R}(h, r)$ as

$$\mathcal{E}(h, r) := \mathbb{P}_{\mathbf{X}, Y} [Y \neq h(\mathbf{X}), r(\mathbf{X}) = 0], \text{ and,} \quad (3.4)$$

$$\mathcal{R}(h, r) := \mathbb{P}_{\mathbf{X}} [r(\mathbf{X}) = 1], \quad (3.5)$$

this risk can be written as

$$R_\lambda(h, r) = \mathcal{E}(h, r) + \lambda \mathcal{R}(h, r).$$

It is thus a trade-off between these two quantities defined using a linear combination with parameter λ .

This risk is minimized for the Bayes optimal pair (h^*, r^*) ,

$$h^*(\mathbf{x}) := \arg \max_k \eta_k(\mathbf{x}), \quad r^*(\mathbf{x}) := \begin{cases} 1 & \text{if } \max_k \eta_k(\mathbf{x}) < 1 - \lambda, \\ 0 & \text{otherwise.} \end{cases} \quad (3.6)$$

It can be noted that the optimal classifier h^* is the same as for the usual classification task given in Equation (3.2). A sample \mathbf{x} is rejected if the likelihood of its most probable class, $\max_k \eta_k(\mathbf{x})$, is too low, or said differently, if the error rate of the optimal prediction, $1 - \max_k \eta_k(\mathbf{x})$, is too high. Thus, samples that are too ambiguous – in the sense that $\max_k \eta_k(\mathbf{x})$ is low – are those that are rejected.

In the next section, Section 3.2, we study how the classification with reject option framework can be used to analyze the two-step procedure in which case the classifier is trained and we only have to learn the rejector. This will enable us to then study, in Section 3.3, what type of uncertainty is necessary for rejection. Which will, in turn, be helpful to examine two types of hand-crafted criteria: confidence-based rejectors, in Section 3.4, and disagreement-based criteria, in Section 3.5.

3.2 Learning to reject when the classifier is fixed: theoretical study

In this section, we focus on a specific instance of classification with reject option in which the classifier is already trained and we want to learn to reject its decisions when appropriate. This scenario is appealing for studying rejection criteria built on uncertainty information as we will see in the next sections. More broadly speaking, this setting is of interest on its own. Indeed, it is common practice, in deep learning, for instance, to use off-the-shelf pre-trained models which are shared via model zoos. Learning an additional rejector on top of this classifier can be interesting to filter out its most untrustworthy predictions.

3.2.1 Fixed cost rejection as a binary cost-sensitive classification problem

We show in the next theorem that fixing the classifier h and only optimize the risk of Equation (3.3) with respect to the rejector r is exactly equivalent to a binary cost-sensitive classification task (Elkan, 2001). Indeed, it states that, formalized this way, rejecting amounts to learn to predict the random variable (r.v.) $E = \mathbb{1}_{h(\mathbf{X}) \neq Y}$, i.e. the r.v. corresponding to a prediction error, with a penalty of λ in case of false rejection and of $1 - \lambda$ in case of false acceptance.

Theorem 3.2.1 (Cost-sensitive classification form). *Let $R_\lambda(r; h)$ be the following risk*

$$R_\lambda(r; h) := \mathbb{E}_{\mathbf{X}, E} \left[\lambda \mathbb{1}_{E=0, r(\mathbf{X})=1} + (1 - \lambda) \mathbb{1}_{E=1, r(\mathbf{X})=0} \right]. \quad (3.7)$$

When there is no ambiguity over h , we will simply write $R_\lambda(r) := R_\lambda(r; h)$.

This risk is related to the original rejection risk $R_\lambda(h, r)$ of Equation (3.3) by

$$R_\lambda(h, r) = R_\lambda(r; h) + \lambda \mathbb{P}_{\mathbf{X}, Y} [Y \neq h(\mathbf{X})].$$

Thus, fixing the classifier h and minimizing $R_\lambda(h, r)$ of Equation (3.3) only with respect of the rejector r is exactly equal to minimizing $R_\lambda(r; h)$.

Proof. The risk of Equation (3.3) can be written as

$$\begin{aligned} R_\lambda(h, r) &= \mathbb{E}_{\mathbf{X}, Y} [\mathbb{1}_{h(\mathbf{X}) \neq Y} \mathbb{1}_{r(\mathbf{X})=0} + \lambda \mathbb{1}_{r(\mathbf{X})=1}] \\ &= \mathbb{E}_{\mathbf{X}, Y} [(\lambda - \mathbb{1}_{h(\mathbf{X}) \neq Y}) \mathbb{1}_{r(\mathbf{X})=1}] + \mathbb{E}_{\mathbf{X}, Y} [\mathbb{1}_{h(\mathbf{X}) \neq Y}] \\ &= \mathbb{E}_{\mathbf{X}, E} [(\lambda - \mathbb{1}_{E=1}) \mathbb{1}_{r(\mathbf{X})=1}] + \mathbb{P}_{\mathbf{X}, Y} [Y \neq h(\mathbf{X})] \\ &= \mathbb{E}_{\mathbf{X}, E} [\lambda \mathbb{1}_{E=0, r(\mathbf{X})=1} + (\lambda - 1) \mathbb{1}_{E=1} (1 - \mathbb{1}_{r(\mathbf{X})=0})] + \mathbb{P}_{\mathbf{X}, Y} [Y \neq h(\mathbf{X})] \\ &= \mathbb{E}_{\mathbf{X}, E} [\lambda \mathbb{1}_{E=0, r(\mathbf{X})=1} + (1 - \lambda) \mathbb{1}_{E=1, r(\mathbf{X})=0}] \\ &\quad + (\lambda - 1) \mathbb{P}_{\mathbf{X}, Y} [Y \neq h(\mathbf{X})] + \mathbb{P}_{\mathbf{X}, Y} [Y \neq h(\mathbf{X})] \\ &= R_\lambda(r; h) + \lambda \mathbb{P}_{\mathbf{X}, Y} [Y \neq h(\mathbf{X})]. \end{aligned}$$

The last term being independent of r , if we assume that h is fixed then only the first term has to be minimized. This thus concludes the proof. \square

The conditional probability of this task, denoted $\eta_E(\mathbf{x})$, is equal to

$$\eta_E(\mathbf{x}) := \mathbb{P}_{\mathbf{X}, E} [E = 1 \mid \mathbf{X} = \mathbf{x}] = \mathbb{P}_{\mathbf{X}, Y} [Y \neq h(\mathbf{X}) \mid \mathbf{X} = \mathbf{x}]. \quad (3.8)$$

It is thus the point-wise error rate of the classifier h . Using this quantity, the risk $R_\lambda(r; h, \mathbf{x})$ for a given input \mathbf{x} can be rewritten as

$$\begin{aligned} R_\lambda(r; h, \mathbf{x}) &= \lambda(1 - \eta_E(\mathbf{x})) \mathbb{1}_{r(\mathbf{x})=1} + (1 - \lambda) \eta_E(\mathbf{x}) \mathbb{1}_{r(\mathbf{x})=0} \\ &= [\lambda(1 - \eta_E(\mathbf{x})) - (1 - \lambda) \eta_E(\mathbf{x})] \mathbb{1}_{r(\mathbf{x})=1} + (1 - \lambda) \eta_E(\mathbf{x}) \\ &= (\lambda - \eta_E(\mathbf{x})) \mathbb{1}_{r(\mathbf{x})=1} + (1 - \lambda) \eta_E(\mathbf{x}). \end{aligned}$$

This risk is thus minimized point-wisely by following Bayes optimal rejector,

$$r^*(\mathbf{x}) := \begin{cases} 1 & \text{if } \eta_E(\mathbf{x}) > \lambda, \\ 0 & \text{otherwise.} \end{cases} \quad (3.9)$$

To summarize, when the classifier is fixed, rejecting amounts to filtering out samples for which the classification error rate is above the rejection cost λ . This formulation is thus very appealing as this rejection cost directly sets an upper bound constraint on the tolerated point-wise classification error.

In practice, we are interested in quantifying how much larger is the risk of a given rejector \hat{r} compared to the one of the optimal r^* . This is captured by the notion of *rejection regret* denoted $\text{Reg}_{\text{rej}, \lambda}(\hat{r})$ and defined as

$$\text{Reg}_{\text{rej}, \lambda}(\hat{r}) := R_\lambda(\hat{r}) - R_\lambda(r^*).$$

The following proposition gives its expression and shows that the rejection regret is higher when misrejecting a sample for which the error rate is far away from the constraint λ .

Proposition 3.2.2. *The regret $\text{Reg}_{\text{rej},\lambda}$ can be expressed as*

$$\text{Reg}_{\text{rej},\lambda}(\hat{r}) = \mathbb{E}_{\mathbf{X}} \left[|\eta_E(\mathbf{X}) - \lambda| \mathbb{1}_{\hat{r}(\mathbf{X}) \neq r^*(\mathbf{X})} \right].$$

Proof. Using the definition of the cost-sensitive risk of Equation (3.7), we have

$$\begin{aligned} \text{Reg}_{\text{rej},\lambda}(\hat{r}) &= \lambda \mathbb{E}_{\mathbf{X},E} \left[\mathbb{1}_{E=0,\hat{r}(\mathbf{X})=1} - \mathbb{1}_{E=0,r^*(\mathbf{X})=1} \right] \\ &\quad + (1-\lambda) \mathbb{E}_{\mathbf{X},E} \left[\mathbb{1}_{E=1,\hat{r}(\mathbf{X})=0} - \mathbb{1}_{E=1,r^*(\mathbf{X})=0} \right] \\ &= \lambda \mathbb{E}_{\mathbf{X}} \left[(1 - \eta_E(\mathbf{X})) (\mathbb{1}_{\hat{r}(\mathbf{X})=1} - \mathbb{1}_{r^*(\mathbf{X})=1}) \right] \\ &\quad + (1-\lambda) \mathbb{E}_{\mathbf{X}} \left[\eta_E(\mathbf{X}) (\mathbb{1}_{\hat{r}(\mathbf{X})=0} - \mathbb{1}_{r^*(\mathbf{X})=0}) \right] \\ &= \mathbb{E}_{\mathbf{X}} \left[\left\{ \lambda(1 - \eta_E(\mathbf{X})) - (1-\lambda)\eta_E(\mathbf{X}) \right\} (\mathbb{1}_{\hat{r}(\mathbf{X})=1} - \mathbb{1}_{r^*(\mathbf{X})=1}) \right] \\ &= \mathbb{E}_{\mathbf{X}} \left[(\lambda - \eta_E(\mathbf{X})) (\mathbb{1}_{\hat{r}(\mathbf{X})=1} - \mathbb{1}_{r^*(\mathbf{X})=1}) \right] \\ &= \mathbb{E}_{\mathbf{X}} \left[(\lambda - \eta_E(\mathbf{X})) (\mathbb{1}_{\hat{r}(\mathbf{X})=1,r^*(\mathbf{X})=0} - \mathbb{1}_{\hat{r}(\mathbf{X})=0,r^*(\mathbf{X})=1}) \right]. \end{aligned}$$

Using the fact that $r^*(\mathbf{X}) = 1 \Leftrightarrow \eta_E(\mathbf{X}) \geq \lambda$, we can conclude as

$$\begin{aligned} \text{Reg}_{\text{rej},\lambda}(\hat{r}) &= \mathbb{E}_{\mathbf{X}} \left[|\lambda - \eta_E(\mathbf{X})| \mathbb{1}_{\hat{r}(\mathbf{X})=1,r^*(\mathbf{X})=0} + |\lambda - \eta_E(\mathbf{X})| \mathbb{1}_{\hat{r}(\mathbf{X})=0,r^*(\mathbf{X})=1} \right] \\ &= \mathbb{E}_{\mathbf{X}} \left[|\lambda - \eta_E(\mathbf{X})| \mathbb{1}_{\hat{r}(\mathbf{X}) \neq r^*(\mathbf{X})} \right]. \end{aligned}$$

□

3.2.2 Linking rejection cost with rejection rate

It is not always easy to determine what value for the rejection cost λ one should pick. On the other hand, it can be more intuitive in practice to force the rejection rate $\mathcal{R}(h, r)$ to be lower than a chosen value $R \in [0, 1]$. With that in mind, the rejection problem can be reformulated under the following constrained optimization problem (where the classifier h is fixed):

$$\begin{aligned} \min_r \quad & \mathcal{E}(h, r) \\ \text{s.t.} \quad & \mathcal{R}(h, r) \leq R \end{aligned} \tag{3.10}$$

This formulation has been studied in the binary classification case in Denis and Hebiri (2020). In this section, we generalize this analysis to the multiclass case.

We show here that there is actually a direct relationship between the cost-based formulation (3.3) and the constrained formulation (3.10). The rejection rate of the optimal rejector r^* of Equation (3.9) is given by the complementary cumulative distribution of error rates $\eta_E(\mathbf{X})$ (Equation (3.8)) denoted $\bar{F}(\lambda)$:

$$\mathcal{R}(h, r^*) = \bar{F}(\lambda) := \mathbb{P}_{\mathbf{X}} [\eta_E(\mathbf{X}) > \lambda].$$

To compute what threshold λ to apply in order to reach a specific value of reject rate R , we need to inverse this function. Because \bar{F} is not invertible in general, we need to consider the following generalized inverse function (Reiss, 1989),

$$\bar{F}^{-1}(R) := \inf \{ \lambda \in [0, 1] \mid \bar{F}(\lambda) \leq R \}.$$

This generalized inverse satisfies

$$\bar{F}(\bar{F}^{-1}(R)) \leq R.$$

with equality if \bar{F} is continuous at $\bar{F}^{-1}(R)$. It thus guaranteed to satisfy the constraint of (3.10). The next proposition proves that this quantity is indeed the threshold minimizing the error rate and thus proves the equivalence between the two formulations.

Assumption A (Continuity assumption). *We assume that the distribution of $\eta_E(\mathbf{X})$ is continuous.*

This assumption thus implies that

$$\bar{F}(\bar{F}^{-1}(R)) = R. \quad (3.11)$$

Proposition 3.2.3. *For a fixed classifier h , under [Assumption A](#), an optimal solution of the constraint problem (3.10) is achieved for*

$$r^*(\mathbf{x}) := \mathbb{1}_{\eta_E(\mathbf{x}) > \bar{F}^{-1}(R)}.$$

The constrained problem (3.10) has thus the same minimizer as the cost-sensitive formulation (3.7) with $\lambda = \bar{F}^{-1}(R)$.

Proof. Let r be any rejector with rejection rate below R , i.e. $\mathcal{R}(h, r) \leq R$. To simplify the notations, we denote λ_R the threshold $\lambda_R = \bar{F}^{-1}(R)$. We have

$$\begin{aligned} \mathcal{E}(h, r) - \mathcal{E}(h, r^*) &= \mathbb{P}_{\mathbf{X}, Y} [Y \neq h(\mathbf{X}), r(\mathbf{X}) = 0] - \mathbb{P}_{\mathbf{X}, Y} [Y \neq h(\mathbf{X}), r^*(\mathbf{X}) = 0] \\ &= \mathbb{E}_{\mathbf{X}} [\eta_E(\mathbf{X})(\mathbb{1}_{r(\mathbf{X})=0} - \mathbb{1}_{r^*(\mathbf{X})=0})] \\ &= \mathbb{E}_{\mathbf{X}} [(\eta_E(\mathbf{X}) - \lambda_R)(\mathbb{1}_{r(\mathbf{X})=0} - \mathbb{1}_{r^*(\mathbf{X})=0}) + \lambda_R(\mathbb{1}_{r(\mathbf{X})=0} - \mathbb{1}_{r^*(\mathbf{X})=0})] \\ &= \mathbb{E}_{\mathbf{X}} [(\eta_E(\mathbf{X}) - \lambda_R)(\mathbb{1}_{r(\mathbf{X})=0, r^*(\mathbf{X})=1} - \mathbb{1}_{r(\mathbf{X})=1, r^*(\mathbf{X})=0})] \\ &\quad + \lambda_R \mathbb{E}_{\mathbf{X}} [\mathbb{1}_{r^*(\mathbf{X})=1} - \mathbb{1}_{r(\mathbf{X})=1}] \\ &= \mathbb{E}_{\mathbf{X}} [|\eta_E(\mathbf{X}) - \lambda_R| \mathbb{1}_{r(\mathbf{X}) \neq r^*(\mathbf{X})}] + \lambda_R (\mathcal{R}(h, r^*) - \mathcal{R}(h, r)). \end{aligned}$$

Because of [Assumption A](#) and its consequence ([Equation \(3.11\)](#)), we have $\mathcal{R}(h, r^*) = R$ and thus the second term is positive. We can thus conclude that the above quantity is positive and that

$$\mathcal{E}(h, r) \geq \mathcal{E}(h, r^*),$$

which concludes the proof. □

3.2.3 Global rejection as a bipartite ranking task

In general, we might not know what cost λ to use or what rejection rate we are aiming for beforehand. In this case, we can aim at learning a global rejector that will work for each of these values.

To do so, we consider rejectors based on a scoring function $s : \mathcal{X} \rightarrow \mathbb{R}$ and a threshold $t \in \mathbb{R}$ and defined as

$$r_t(\mathbf{x}) = \mathbb{1}_{s(\mathbf{x}) > t}.$$

This scoring function can be seen as an *uncertainty* scoring function as the higher its value the more likely we are to reject it. As in the previous section, the rejection rate of this type of rejector is given by the complementary cumulative distribution of scores,

$$\mathcal{R}(h, r_t) = \bar{F}_s(t) := \mathbb{P}_{\mathbf{X}} [s(\mathbf{X}) > t],$$

and the threshold t needed to attain a given rejection rate of R is given by the following generalized inverse,

$$\bar{F}_s^{-1}(R) := \inf\{t \in \mathbb{R} \mid \bar{F}_s(t) \leq R\}.$$

To simplify the rest of this section, we will make the following simplifying assumption.

Assumption B (Absence of ties). *We assume that, almost surely, there is no tie, i.e.*

$$\mathbb{P}_{\mathbf{X}, \mathbf{X}'} [s(\mathbf{X}) = s(\mathbf{X}')] = 0.$$

This assumption is introduced merely for technical reasons as all the results of this section still hold in presence of ties. This would however require to use a stochastic rejector which would break the ties uniformly at random and to add a term in the quantities introduced hereafter in a similar fashion as [Menon and Williamson \(2016\)](#). For the sake of clarity, we assume the absence of such ties.

Optimizing the trade-off between error and rejection rate amounts to optimize the functional

$$t \mapsto (\mathcal{R}(h, r_t), \mathcal{E}(h, r_t)).$$

In general, this is a difficult problem. An easier approach is to minimize the area-under-the-curve of this functional denoted $\text{AUC}(h, s)$:

$$\text{AUC}(h, s) = \int_{R=0}^1 \mathcal{E}(h, r_{t_R}) \, dR$$

where $t_R = \bar{F}_s^{-1}(R)$ is the threshold on s achieving a rejection rate of R . The following proposition shows that minimizing AUC can be expressed as a ranking problem on the scoring function in which we want erroneous samples to have a higher uncertainty score value s than all the other samples.

Proposition 3.2.4. *The AUC of the error-rate/rejection-rate curve can be expressed as*

$$\text{AUC}(h, s) = \mathbb{P}_{(\mathbf{X}, Y), \mathbf{X}'} [s(\mathbf{X}') \geq s(\mathbf{X}), Y \neq h(\mathbf{X})].$$

Proof. Let U be a uniform variable on $[0, 1]$, then $\bar{F}_s^{-1}(U)$ is distributed as $s(\mathbf{X})$ according to the quantile transformation ([Reiss, 1989](#), Lemma 1.2.4). Using this, we have

$$\begin{aligned} \text{AUC}(h, s) &= \int_{R=0}^1 \mathcal{E}(h, r_{t_R}) \, dR \\ &= \mathbb{E}_U \left[\mathbb{P}_{\mathbf{X}, Y} [Y \neq h(\mathbf{X}), s(\mathbf{X}) \leq \bar{F}_s^{-1}(U)] \right] \\ &= \mathbb{P}_{(\mathbf{X}, Y), \mathbf{X}'} [Y \neq h(\mathbf{X}), s(\mathbf{X}) \leq s(\mathbf{X}')]. \end{aligned}$$

□

Note that using this AUC was proposed by [Nadeem et al. \(2009\)](#) to compare rejection methods in the context of healthcare. However, the formalization presented here was not provided and the metric was only described to carry out experiments. In particular, this expression of the AUC as well as the next results are new.

The next results show that minimizing AUC when the classifier is fixed is in fact equivalent to solving a bipartite ranking problem. Bipartite ranking ([Cl  men  on et al., 2008](#); [Agarwal, 2014](#))

consists in learning a score function from instances with binary labels. The goal is to give higher scores to positive instances – in our case, erroneous samples – than to negative ones – correct samples. More formally, the aim is to minimize the following risk called *ranking risk* which, with our notations and in absence of ties, is defined as¹:

$$R_{\text{rank}}(s) = \mathbb{P}_{(\mathbf{X}, E), (\mathbf{X}', E')} [(E - E')(s(\mathbf{X}) - s(\mathbf{X}')) < 0].$$

This risk counts an ordering error if \mathbf{X} is an erroneous sample while \mathbf{X}' is not, but, the uncertainty score of \mathbf{X}' is higher than that of \mathbf{X} (and vice-versa). The following theorem shows the exact connection between AUC minimization and bipartite ranking.

Theorem 3.2.5 (Bipartite ranking form). *We have*

$$\text{AUC}(h, s) = \frac{1}{2} R_{\text{rank}}(s) + \frac{1}{2} \mathbb{P}_{\mathbf{X}, Y} [Y \neq h(\mathbf{X})]^2.$$

Optimizing the AUC of the error-rate/rejection-rate curve with a fixed classifier is thus strictly equivalent to a bipartite ranking problem where the label of interest is the error r.v. E .

In order to prove this theorem, we need the following lemma.

Lemma 3.2.6. *For two i.i.d. random variables Z and Z' , we have*

$$\mathbb{P}_{Z, Z'} [Z' \geq Z] = \frac{1}{2} (1 + \mathbb{P}_{Z, Z'} [Z = Z']).$$

Proof.

$$\begin{aligned} \mathbb{P}_{Z, Z'} [Z' \geq Z] &= 1 - \mathbb{P}_{Z, Z'} [Z' < Z] \\ &= 1 + \mathbb{P}_{Z, Z'} [Z = Z'] - \mathbb{P}_{Z, Z'} [Z' \leq Z] \\ &= 1 + \mathbb{P}_{Z, Z'} [Z = Z'] - \mathbb{P}_{Z, Z'} [Z \leq Z'], \end{aligned}$$

because Z and Z' are i.i.d., which concludes the proof. \square

The proof of the theorem can now be derived.

Proof of Theorem 3.2.5. Denoting $p_E := \mathbb{P}_E [E = 1] = \mathbb{P}_{\mathbf{X}, Y} [Y \neq h(\mathbf{X})]$, from Proposition 3.2.4, we have

$$\begin{aligned} \text{AUC}(h, s) &= \mathbb{P}_{(\mathbf{X}, E), \mathbf{X}'} [s(\mathbf{X}') \geq s(\mathbf{X}), E = 1] \\ &= p_E \mathbb{P}_{(\mathbf{X}, E), \mathbf{X}'} [s(\mathbf{X}') \geq s(\mathbf{X}) \mid E = 1] \\ &= p_E \left\{ (1 - p_E) \mathbb{P}_{(\mathbf{X}, E), (\mathbf{X}', E')} [s(\mathbf{X}') \geq s(\mathbf{X}) \mid E = 1, E' = 0] \right. \\ &\quad \left. + p_E \mathbb{P}_{(\mathbf{X}, E), (\mathbf{X}', E')} [s(\mathbf{X}') \geq s(\mathbf{X}) \mid E = E' = 1] \right\} \end{aligned}$$

¹An alternative definition using a conditional probability can also be found in the literature, in which case the risk is defined as $R'_{\text{rank}}(s) = \mathbb{P}_{(\mathbf{X}, E), (\mathbf{X}', E')} [(E - E')(s(\mathbf{X}) - s(\mathbf{X}')) < 0 \mid E \neq E']$. The link between the two risks is then given by $R_{\text{rank}}(s) = 2p_E(1 - p_E)R'_{\text{rank}}(s)$ where $p_E = \mathbb{P}_E [E = 1]$.

Using [Lemma 3.2.6](#) and [Assumption B](#), we obtain

$$\begin{aligned}
 \text{AUC}(h, s) &= p_E(1 - p_E) \mathbb{P}_{(\mathbf{X}, E), (\mathbf{X}', E')} [s(\mathbf{X}') \geq s(\mathbf{X}) \mid E = 1, E' = 0] + \frac{p_E^2}{2} \\
 &= \mathbb{P}_{(\mathbf{X}, E), (\mathbf{X}', E')} [s(\mathbf{X}') \geq s(\mathbf{X}), E = 1, E' = 0] + \frac{p_E^2}{2} \\
 &= \mathbb{P}_{(\mathbf{X}, E), (\mathbf{X}', E')} [s(\mathbf{X}') > s(\mathbf{X}), E = 1, E' = 0] + \frac{p_E^2}{2} \\
 &= \frac{1}{2} \mathbb{P}_{(\mathbf{X}, E), (\mathbf{X}', E')} [(s(\mathbf{X}) - s(\mathbf{X}'))(E - E') < 0] + \frac{p_E^2}{2}.
 \end{aligned}$$

□

The link with ROC AUC and bipartite ranking is a widely known result, see for instance [Cl  men  on et al. \(2008\)](#) for proof. Here, the AUC computed is not the ROC AUC but the area under another curve, it is thus a new result.

Now that the link between rejection and bipartite ranking is established, we can exploit all that is known for that task and apply it to rejection. To have an overview of some results, we refer the reader to [Agarwal \(2014\)](#); [Werner \(2019\)](#). We will here briefly give some useful results in our context. In particular, it is widely known that the Bayes optimal scoring function is given by

$$s^*(\mathbf{x}) := \eta_E(\mathbf{x}). \quad (3.12)$$

More interestingly, the following result gives the expression of global rejection regret denoted $\text{Reg}_{\text{rank}}(s)$ and defined as

$$\text{Reg}_{\text{rank}}(s) := R_{\text{rank}}(s) - R_{\text{rank}}(s^*).$$

Proposition 3.2.7 ([Agarwal \(2014\)](#), Theorem 11). *In absence of ties, the regret of the rank loss Reg_{rank} is given by*

$$\text{Reg}_{\text{rank}}(s) = \mathbb{E}_{\mathbf{X}, \mathbf{X}'} \left[|\eta_E(\mathbf{X}) - \eta_E(\mathbf{X}')| \mathbb{1}_{(s(\mathbf{X}) - s(\mathbf{X}'))(\eta_E(\mathbf{X}) - \eta_E(\mathbf{X}')) < 0} \right].$$

This result implies that any scoring function s preserving the order of η_E will be optimal. Indeed, the previous regret is null if and only if

$$\begin{aligned}
 (s(\mathbf{X}) - s(\mathbf{X}'))(\eta_E(\mathbf{X}) - \eta_E(\mathbf{X}')) &\geq 0 \\
 \Updownarrow \\
 s(\mathbf{X}) \geq s(\mathbf{X}') \text{ if and only if } \eta_E(\mathbf{X}) &\geq \eta_E(\mathbf{X}').
 \end{aligned}$$

To summarize these results in our context, when rejecting the predictions of a fixed classifier, we need to be able to find an uncertainty measure that is monotonically correlated to the point-wise error rate $\eta_E(\mathbf{x})$. In the next section, we analyze this error rate in order to understand it better and to see what type of measure we should look for.

3.3 What uncertainty is needed for rejection?

As shown in the previous section, in [Equation \(3.9\)](#) and [Equation \(3.12\)](#), the optimal strategy depends on the point-wise error rate $\eta_E(\mathbf{x})$. In this section, we first analyze more finely this quantity to understand its relationship with the different forms of uncertainty. This will be useful to analyze rejection criteria later on.

3.3.1 Point-wise error rate decomposition

The point-wise error rate $\eta_E(\mathbf{x})$ of a given classifier h is given by

$$\begin{aligned}\eta_E(\mathbf{x}) &= 1 - \eta_{h(\mathbf{x})}(\mathbf{x}) \\ &= \sum_k \mathbb{1}_{h(\mathbf{x})=k} (1 - \eta_k(\mathbf{x})).\end{aligned}$$

It can also be expressed differently by introducing the following quantities:

1. the error rate of the optimal classifier, denoted $\eta_E^*(\mathbf{x})$ and equal to

$$\eta_E^*(\mathbf{x}) := 1 - \max_k \eta_k(\mathbf{x});$$

2. the relative additional error rate induced by predicting class k instead of the most probable class, denoted $\Delta_k(\mathbf{x})$ and defined as

$$\begin{aligned}\Delta_k(\mathbf{x}) &:= \eta_E(\mathbf{x}) - \eta_E^*(\mathbf{x}) \\ &= \max_{k'} \eta_{k'}(\mathbf{x}) - \eta_k(\mathbf{x}).\end{aligned}$$

Using these quantities, the classification error rate can be rewritten as

$$\eta_E(\mathbf{x}) = \underbrace{\eta_E^*(\mathbf{x})}_{\text{irreducible classification error}} + \underbrace{\sum_{k \neq h^*(\mathbf{x})} \mathbb{1}_{h(\mathbf{x})=k} \Delta_k(\mathbf{x})}_{\text{reducible classification error}}. \quad (3.13)$$

The first term is the error rate of the optimal Bayes classifier h^* . It is irreducible in the sense that no classifier can have an error rate lower than this quantity. It is thus directly related to the intrinsic ambiguity of the task and to *aleatoric* uncertainty. The second term, which is also positive as by definition $\Delta_k \geq 0$, depends on a potential misclassification of h , it is thus reducible by finding a better classifier. It depends on two types of uncertainties. First, it is influenced by the uncertainty in the choice of h : given more data, the number of suboptimal decisions should decrease and this second term should tend to zero. This is related to model uncertainty and *epistemic* uncertainty. However, this uncertainty is weighted by the difference of error rates $\Delta_k(\mathbf{x})$: even if we make a suboptimal prediction, in presence of intrinsic task ambiguity, the resulting error rate is not necessarily high because the prediction can actually be possible. It is thus also related to *aleatoric* uncertainty.

Note that, if $h = h^*$, we do recover the same rejector as in Equation (3.6). It is interesting to add that, in general, making an suboptimal classification decision for a sample \mathbf{x}_1 (i.e. $\max_k \eta_k(\mathbf{x}_1) > \eta_{h(\mathbf{x}_1)}(\mathbf{x}_1)$) can actually lead to a lower error rate than predicting the optimal class for a very ambiguous sample \mathbf{x}_2 (i.e. $\max_k \eta_k(\mathbf{x}_2) \approx \max_{k \neq k^*} \eta_k(\mathbf{x}_2)$) if $\eta_{h(\mathbf{x}_1)}(\mathbf{x}_1) > \max_k \eta_k(\mathbf{x}_2)$. This can happen for instance if for \mathbf{x}_1 there is an ambiguity between two classes with nearly the same probability, whereas for \mathbf{x}_2 this ambiguity is among more classes, say three for instance.

3.3.2 Special case: absence of task ambiguity

In this case, also known as the *noise-free* setting, the conditional probability $\eta(\mathbf{x})$ is of the form:

$$\exists k \mid \eta_k(\mathbf{x}) = 1 \quad \text{and} \quad \forall k' \neq k, \eta_{k'}(\mathbf{x}) = 0.$$

This implies that the optimal classifier has a null error rate, i.e. $R(h^*) = 0$. In this setting, the expression of point-wise error rate of classifier h is simpler as it is equal to

$$\eta_E(\mathbf{x}) = \mathbb{1}_{h(\mathbf{x}) \neq h^*(\mathbf{x})}.$$

All the error is now reducible: making a suboptimal prediction will always generate an error of 1 while taking the optimal decision will not incur any error.

Building an optimal rejector thus amounts in being able to detect when we make a prediction error. Note that it is not because we are able to detect such errors that we will know how to correct them: except in the binary case, detecting an error will not allow us to directly conclude what is the correct class to predict.

3.4 Confidence-based criteria

Now that we have analyzed the rejection problem, we study concrete criteria to tackle it. In this section, we focus on a form of such criteria, *confidence-based criteria* (Bartlett and Wegkamp, 2008; Ramaswamy et al., 2018; Ni et al., 2019). Assume the classifier h is based on a scoring function $f : \mathcal{X} \rightarrow \mathbb{R}^C$ by $h(\mathbf{x}) = \arg \max_k f_k(\mathbf{x})$. Can we build a rejection criterion based on this scoring function? The natural approach is to build a rejection criterion based on the uncertainty score $s(\mathbf{x}) = -\max_k f_k(\mathbf{x})$: the higher the confidence predicted for a sample, the less likely we will reject it. In this section, we study what it takes for a scoring function to provide a good confidence-based criterion.

3.4.1 A priori, confidence-based criteria provide no guarantees for rejection

Independently of its performance for classification, the scoring function f on its own does not guarantee to give a score useful for rejection. Indeed, it can be arbitrarily good or bad: a given classifier h can be associated with a scoring function f which can be chosen to attain the bounds on the AUC.

For instance, the lower bound is attained by picking

$$f_k(\mathbf{x}) = \begin{cases} \eta_k(\mathbf{x}) & \text{if } k = h(\mathbf{x}), \\ -1 & \text{otherwise.} \end{cases}$$

as in this case $\arg \max_k f_k(\mathbf{x}) = h(\mathbf{x})$ and $s(\mathbf{x}) = -\eta_k(\mathbf{x})$ which preserves the ranking of $\eta_E(\mathbf{x})$. Thus, by choosing confidence-based criteria, we do not limit ourselves and we can theoretically build an optimal rejector.

On the other hand, the upper bound is attained by picking the following scoring function:

$$f_k(\mathbf{x}) = \begin{cases} -\eta_k(\mathbf{x}) & \text{if } k = h(\mathbf{x}), \\ -1 & \text{otherwise,} \end{cases}$$

as in this case $\arg \max_k f_k(\mathbf{x}) = h(\mathbf{x})$ and $s(\mathbf{x}) = \eta_k(\mathbf{x})$ which is totally the inverse ranking of $\eta_E(\mathbf{x})$. This means that using confidence-based criteria does not guarantee to do better than the worst possible rejector.

Thus, a priori, using confidence-based criteria neither limits nor advantages us. However, if some structure in the confidence score f is added when the classifier is learned, then this

might be leveraged by the rejection criterion. In the rest of this section, we show that models optimizing negative log-likelihood provides this additional structure and are consistent for rejection.

3.4.2 Negative log-likelihood minimization is consistent for rejection

Assume that we estimate the conditional probability η using an estimator $\hat{\eta}$. From this estimator, we can estimate the error rate η_E using

$$\hat{\eta}_E(\mathbf{x}) = 1 - \hat{\eta}_{h(\mathbf{x})}(\mathbf{x}).$$

From this, we can use the *plug-in estimator* by using this estimator as an uncertainty score for rejection. This raises a question: how good is this rejector based on the estimation error of $\hat{\eta}_E$? To answer this question, we give an upper bound Reg_{rank} in terms of the estimation error of $\hat{\eta}_E$ measured as $\mathbb{E}_{\mathbf{X}} [|\hat{\eta}_E(\mathbf{X}) - \eta_E(\mathbf{X})|]$. This bound is a classic result from bipartite ranking which proof can be found in [Agarwal \(2014\)](#).

Proposition 3.4.1 ([Agarwal \(2014\)](#), Corollary 12). *The regret Reg_{rank} can be upper bounded by*

$$\text{Reg}_{\text{rank}}(\hat{\eta}_E) \leq 2\mathbb{E}_{\mathbf{X}} [|\hat{\eta}_E(\mathbf{X}) - \eta_E(\mathbf{X})|].$$

To use this plug-in estimator, we need to estimate η . One approach to do so is to fit the model using the *negative log-likelihood* which is known to be a strictly proper loss ([Gneiting and Raftery, 2007](#)), i.e. it is minimized only when the model predicts η . The negative log-likelihood loss is defined as

$$l_{\log}(k, \hat{\eta}) = -\log \hat{\eta}_k.$$

Its regret is defined as

$$\text{Reg}_{\log}(\hat{\eta}) = \mathbb{E}_{\mathbf{X}, Y} [l_{\log}(Y, \hat{\eta})] - \mathbb{E}_{\mathbf{X}, Y} [l_{\log}(Y, \eta)].$$

The following result states that, given a classifier based on a confidence score, if this score is calibrated using negative log-likelihood then it is useful and consistent for rejection.

Proposition 3.4.2. *The following bound holds:*

$$\text{Reg}_{\text{rank}}(\hat{\eta}_E) \leq \sqrt{2\text{Reg}_{\log}(\hat{\eta})}.$$

where $\hat{\eta}_E(\mathbf{x}) = 1 - \hat{\eta}_{h(\mathbf{x})}(\mathbf{x})$.

Negative log-likelihood calibration is thus consistent for rejection.

To prove this proposition, we need the following lemma.

Lemma 3.4.3. *Define the regret of the negative log-likelihood of the rejection task as*

$$\text{Reg}_{\log}^E(\hat{\eta}_E) = \mathbb{E}_{\mathbf{X}, E} [l_{\log}(E, \hat{\eta}_E)] - \mathbb{E}_{\mathbf{X}, E} [l_{\log}(E, \eta_E)].$$

This regret can be upper bounded using

$$\text{Reg}_{\log}^E(\hat{\eta}_E) \leq \text{Reg}_{\log}(\hat{\eta}),$$

where $\hat{\eta}_E(\mathbf{x}) = 1 - \hat{\eta}_{h(\mathbf{x})}(\mathbf{x})$.

Proof. The regret of the negative log-likelihood can be written as

$$\begin{aligned}\text{Reg}_{\log}(\hat{\eta}) &= \mathbb{E}_{\mathbf{X}} \left[-\sum_k \eta_k \log \hat{\eta}_k + \sum_k \eta_k \log \eta_k \right] \\ &= \mathbb{E}_{\mathbf{X}} [\text{D}_{\text{KL}}(\eta(\mathbf{X}), \hat{\eta}(\mathbf{X}))]\end{aligned}$$

where D_{KL} is the Kullback-Leibler divergence (KL-divergence), $\text{D}_{\text{KL}}(p, q) = \sum_k p_k \log \frac{p_k}{q_k}$. Similarly, for the rejection log loss,

$$\text{Reg}_{\log}^E(\hat{\eta}_E) = \mathbb{E}_{\mathbf{X}} [\text{D}_{\text{KL}}(\eta_E(\mathbf{X}), \hat{\eta}_E(\mathbf{X}))],$$

where the KL-divergence is now binary.

We now use the following intermediate result:

$$\text{D}_{\text{KL}}(p, q) \geq \text{D}_{\text{KL}}(p_k, q_k).$$

where $\text{D}_{\text{KL}}(p_k, q_k) = p_k \log \frac{p_k}{q_k} + (1 - p_k) \log \frac{1 - p_k}{1 - q_k}$. The proof is based on the convexity of the function $x \mapsto x \log x$. By setting $\lambda_{k'} = \frac{q_{k'}}{1 - q_k}$ which satisfies $\lambda_{k'} \geq 0$ and $\sum_{k' \neq k} \lambda_{k'} = 1$, we have:

$$\begin{aligned}\text{D}_{\text{KL}}(p, q) &= p_k \log \frac{p_k}{q_k} + \sum_{k' \neq k} p_{k'} \log \frac{p_{k'}}{q_{k'}} \\ &= p_k \log \frac{p_k}{q_k} + (1 - q_k) \sum_{k' \neq k} \lambda_{k'} \left(\frac{p_{k'}}{q_{k'}} \log \frac{p_{k'}}{q_{k'}} \right) \\ &\geq p_k \log \frac{p_k}{q_k} + (1 - q_k) \left(\sum_{k' \neq k} \lambda_{k'} \frac{p_{k'}}{q_{k'}} \right) \log \left(\sum_{k' \neq k} \lambda_{k'} \frac{p_{k'}}{q_{k'}} \right) \\ &= p_k \log \frac{p_k}{q_k} + \left(\sum_{k' \neq k} p_{k'} \right) \log \left(\frac{\sum_{k' \neq k} p_{k'}}{1 - q_k} \right) \\ &= \text{D}_{\text{KL}}(p_k, q_k).\end{aligned}$$

To conclude the proof, we apply this result with $p = \eta(\mathbf{X})$, $q = \hat{\eta}(\mathbf{X})$ and $k = h(\mathbf{X})$, use the fact that $\text{D}_{\text{KL}}(p_k, q_k) = \text{D}_{\text{KL}}(1 - p_k, 1 - q_k)$, and then take the expectation. \square

Proof of Proposition 3.4.2. From Proposition 3.4.1 and using the convexity of $x \mapsto x^2$ and the previous lemma, we have

$$\begin{aligned}\text{Reg}_{\text{rank}}(\hat{\eta}_E) &\leq 2\mathbb{E}_{\mathbf{X}} [|\hat{\eta}_E(\mathbf{X}) - \eta_E(\mathbf{X})|] \\ &= 2\sqrt{\mathbb{E}_{\mathbf{X}} [|\hat{\eta}_E(\mathbf{X}) - \eta_E(\mathbf{X})|^2]} \\ &\leq 2\sqrt{\mathbb{E}_{\mathbf{X}} [|\hat{\eta}_E(\mathbf{X}) - \eta_E(\mathbf{X})|^2]}\end{aligned}$$

From Pinsker's inequality (Boucheron et al., 2013, Theorem 4.19) or using the fact the binary negative log-likelihood is 4-strongly proper (Agarwal, 2014), we have

$$\text{D}_{\text{KL}}(\eta_E(\mathbf{X}), \hat{\eta}_E(\mathbf{X})) \geq 2|\hat{\eta}_E(\mathbf{X}) - \eta_E(\mathbf{X})|^2.$$

Using this result and the previous lemma, we obtain

$$\begin{aligned}\text{Reg}_{\text{rank}}(\hat{\eta}_E) &\leq 2\sqrt{\mathbb{E}_{\mathbf{X}} \left[\frac{1}{2} \text{D}_{\text{KL}}(\eta_E(\mathbf{X}), \hat{\eta}_E(\mathbf{X})) \right]} \\ &= \sqrt{2\text{Reg}_{\log}^E(\hat{\eta}_E)} \\ &\leq \sqrt{2\text{Reg}_{\log}(\hat{\eta})},\end{aligned}$$

which concludes the proof. \square

In this section, we have focused on scores calibrated using negative log-likelihood because they appear naturally in a lot of commonly used models such as in neural networks that interest us here. However, this does not mean that no other loss would work. We will see in [Section 3.6](#) how scoring functions outputted by neural networks behave in practice.

3.5 Leveraging disagreement information

In the previous section, we described *confidence-based* criteria. These criteria exploit the scores used by the classifier to make its decision. Another important family of methods is *disagreement-based* criteria which we describe now.

3.5.1 Disagreement-based criteria

Disagreement-based criteria can not be formalized with as much precision as confidence-based criteria. Instead, their common characteristic is that they exploit the disagreement among a committee of models.

From a theoretical perspective, several learning algorithms using this idea were proposed. One of the earliest approaches was proposed by [Freund et al. \(2004\)](#) which consists in measuring the disagreement among the predictions of the classifiers in the hypothesis class \mathcal{H} weighted by their training error rate. Another line of work is the one by [El-Yaniv and Wiener \(2010\)](#); [Wiener and El-Yaniv \(2011, 2015\)](#) which proposes to only select the hypotheses of low training error rate and to detect when they disagree. However, these different algorithms can not be used in practice as they require to scan all the classifiers in \mathcal{H} which is not feasible from a computational perspective. In active learning, similar methods were derived ([Cohn et al., 1994](#); [Settles, 2012](#)) and more practical methods known as *query-by-committee* approaches ([Seung et al., 1992](#); [Freund et al., 1997](#); [Abe, 1998](#)) proposed to exploit the stochasticity of some learning algorithms to sample hypothesis of low training error. The disagreement is then computed from this ensemble.

We propose to try to briefly formalize those criteria by assuming that there exists a probability distribution over the hypothesis space \mathcal{H} which allows to sample hypotheses of low training error. In our case, as in the previous section, we assume that classification is based on a scoring function. We restrict ourselves to the case where this scoring function is an estimator $\hat{\eta}$ of the conditional probability η . We thus assume that we have probability distribution $\mathbb{P}_{\hat{\eta}}$ which allows to sample estimators $\hat{\eta}$ resulting in classifiers $h(\mathbf{x}) = \arg \max_k \hat{\eta}_k(\mathbf{x})$ of low training error rate.

Denoting $\bar{\eta}$ the mean estimated conditional probability, i.e.

$$\bar{\eta}(\mathbf{x}) := \mathbb{E}_{\hat{\eta}} [\hat{\eta}(\mathbf{x})],$$

the global classifier is defined as

$$\bar{h}(\mathbf{x}) := \arg \max_k \bar{\eta}_k(\mathbf{x}).$$

The main disagreement measures found in the literature can then be defined as follows:

1. the probability of disagreement with the prediction (related to the approach proposed by [El-Yaniv and Wiener \(2010\)](#); [Wiener and El-Yaniv \(2011, 2015\)](#)):

$$s_{\text{prob}}^{\text{dis}}(\mathbf{x}) := \mathbb{P}_{\hat{\eta}} \left[\arg \max_k \hat{\eta}_k(\mathbf{x}) \neq \bar{h}(\mathbf{x}) \right],$$

2. the variance of the estimated probability of the predicted class (Gal and Ghahramani, 2016):

$$s_{\text{var}}^{\text{dis}}(\mathbf{x}) := \text{Var}(\hat{\eta}_{\bar{h}(\mathbf{x})}(\mathbf{x})) = \mathbb{E}_{\hat{\eta}} \left[(\hat{\eta}_{\bar{h}(\mathbf{x})}(\mathbf{x}) - \bar{\eta}_{\bar{h}(\mathbf{x})}(\mathbf{x}))^2 \right],$$

3. the expected KL-divergence between individual estimators and mean estimator (Settles, 2012):

$$s_{\text{KL}}^{\text{dis}}(\mathbf{x}) := \mathbb{E}_{\hat{\eta}} [\text{D}_{\text{KL}}(\hat{\eta}(\mathbf{x}), \bar{\eta}(\mathbf{x}))] = \mathbb{E}_{\hat{\eta}} \left[\sum_k \hat{\eta}_k(\mathbf{x}) \log \frac{\hat{\eta}_k(\mathbf{x})}{\bar{\eta}_k(\mathbf{x})} \right].$$

In practice, we have only access to a sampling $\hat{\eta}^{(i)}$ of m estimators from $\mathbb{P}_{\hat{\eta}}$ and we can only compute the empirical version of the previous disagreement measures. There are several ways to build this ensemble, for instance, using boosting or bagging (Abe, 1998). With neural networks, a common approach is to train the same model several times using a different random initialization and a different sampling sequence of the training set. Other approaches were proposed but this method was shown to be very effective (Ashukha et al., 2020), we will thus stick to it.

3.5.2 In general, disagreement-based criteria are inconsistent

Disagreement-based criteria are asymptotically inconsistent. As the training set size increases, the distribution $\mathbb{P}_{\hat{\eta}}$ will concentrate and eventually converge to the real conditional probability, i.e. we have

$$\mathbb{P}_{\hat{\eta}} \longrightarrow \delta_{\eta},$$

where δ_{η} is the Dirac measure concentrating all the distribution mass exactly at η . In this case, all the disagreement-based criteria presented above satisfy

$$s_{\text{prob}}^{\text{dis}}(\mathbf{x}) \rightarrow 0, \quad s_{\text{var}}^{\text{dis}}(\mathbf{x}) \rightarrow 0, \quad s_{\text{KL}}^{\text{dis}}(\mathbf{x}) \rightarrow 0,$$

i.e. they all tend to zero, their minimal value.

Thus, asymptotically, they all tend to a null value as the uncertainty in the choice of the models in \mathcal{H} decreases. These criteria mainly capture model uncertainty. However, in general, there is some task ambiguity and such criteria are unable to capture it even when it becomes predominant compared to model uncertainty.

Note that, in absence of task ambiguity, such criteria can be useful. Recall that, in this case, we only need to estimate

$$\eta_E(\mathbf{x}) = \mathbb{1}_{\bar{h}(\mathbf{x}) \neq h^*(\mathbf{x})}.$$

In fact, the theoretical works using them often focus on this setting (El-Yaniv and Wiener, 2010) or assume that this ambiguity is low (Freund et al., 2004; Wiener and El-Yaniv, 2011, 2015).

3.5.3 Integrating disagreement to confidence-based criteria

Based on the previous observations, if dispersion-based criteria capture model uncertainty, at first, confidence-based criteria can be supposed to mainly focus on task ambiguity. Therefore, it can be hoped that disagreement-based criteria could be useful even in the presence of ambiguity when used in conjunction with confidence-based criteria. These two types of criteria thus seem complementary and a natural question is: can we create a fusion criterion mixing both approaches?

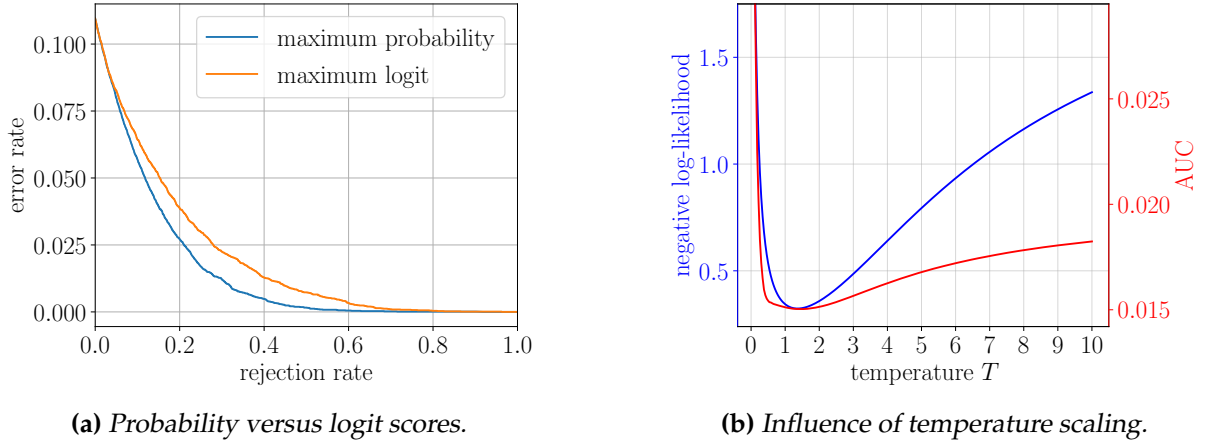


Figure 3.1: Influence of calibration on the performance of confidence-based criteria for rejection measured on CIFAR-10. Figure (a) compares scores in probability and logit spaces by plotting the error rate / rejection rate curve for the associated confidence-based criteria, the lower the better. Figure (b) analyzes the influence of the calibration of the probabilities by plotting the area under the error rate / rejection rate curve when applying temperature scaling with a varying parameter, the lower the better.

We propose to use the expected estimated error rate of individual classifiers defined as

$$s^{\text{fus}}(\mathbf{x}) := 1 - \mathbb{E}_h [\bar{\eta}_{h(\mathbf{x})}(\mathbf{x})] = 1 - \sum_k \mathbb{P}_h [h(\mathbf{x}) = k] \bar{\eta}_k(\mathbf{x}).$$

This criterion assumes that $\bar{\eta}(\mathbf{x})$ is an unbiased estimator of $\eta(\mathbf{x})$ and thus allows to estimate the aleatoric uncertainty. The model uncertainty is then taken into account by marginalizing over the predictions made by the models in the ensemble. Note that, this time, when $\mathbb{P}_{\hat{\eta}} \rightarrow \delta_{\eta}$, we have $s^{\text{fus}}(\mathbf{x}) \rightarrow 1 - \max_k \eta_k(\mathbf{x}) = \eta_E(\mathbf{x})$. It thus corrects the inconsistency of the disagreement-based criteria.

We have tried other fusion criteria but as they showed similar results, we decided to present only this one for clarity.

3.6 Experiments

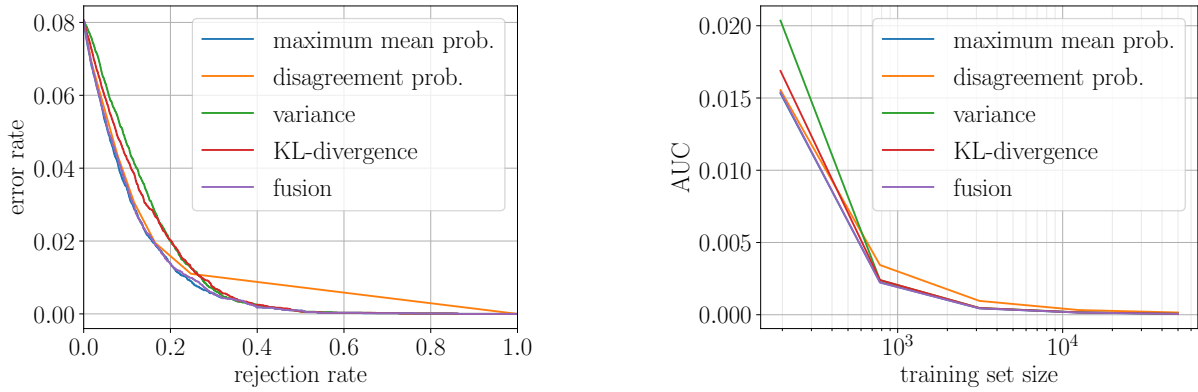
We now perform experiments on real-world datasets to compare these criteria in a practical scenario.

3.6.1 Confidence-based criteria

We start by analyzing confidence-based criteria. We carry out experiments on CIFAR-10 (Krizhevsky and Hinton, 2009) using a simple convolutional neural network. It is trained to minimize the negative log-likelihood on the training set and achieves an error rate of 11% on the test set.

Due to the usage of negative log-likelihood, the probabilities predicted by the neural network are thus natural candidates for confidence-based rejectors. In practice, a neural network outputs scores $\mathbf{z}(\mathbf{x})$ in the logit space ($\mathbf{z}(\mathbf{x}) \in \mathbb{R}^C$) which are transformed into probabilities using

$$\hat{\eta}_k(\mathbf{x}) = \text{softmax}(\mathbf{z}(\mathbf{x}))_k,$$



(a) CIFAR-10: ambiguous case.

(b) MNIST: non-ambiguous case.

Figure 3.2: Comparison of disagreement-based and fusion criteria on two datasets: a dataset with task ambiguity, CIFAR-10, and a non-ambiguous one, MNIST. Figure (a) shows the error rate / rejection rate curve on CIFAR-10, the lower the better. Figure (b) shows the area under this curve on MNIST for a variable training set size, the lower the better.

where $\text{softmax}(\mathbf{z})_k = \frac{e^{z_k}}{\sum_{k'} e^{z_{k'}}$. Note that the softmax transform does not change the ordering of the logits, however, as shown in Figure 3.1a, the scores in the logit space are not well calibrated for rejection compared to those in the probability space. For confidence-based criteria, it is thus important to use probabilities.

The second plot studies the influence of a posteriori probability calibration on the performance of the rejector. According to Section 3.4, calibrated probabilities are guaranteed to allow to build optimal rejector. We test this observation by using a posteriori probability calibration known as temperature scaling (Guo et al., 2017) which allows to preserve the ordering of the probabilities and thus fits into the setting of Section 3.4. It consists into learning a temperature parameter T and updating the estimated probabilities using

$$\hat{\eta}'_k(\mathbf{x}) = \text{softmax}(\mathbf{z}(\mathbf{x})/T)_k.$$

The temperature is set by minimizing the negative log-likelihood on a calibration set. In Figure 3.1b, we show how this parameter influences the quality of the rejection criterion. It shows that, in fact, the area under the error rate / rejection rate curve and the negative log-likelihood are highly correlated, as suggested in Section 3.4. However, the AUC is less sensitive to higher values of temperature than the negative log-likelihood. This means that, if the model outputs (moderately) overconfident estimates of the conditional probability, it will not impact heavily the quality of the rejection criterion. Finally, the minimum of the AUC is attained for a temperature close to the default temperature of one, i.e. when additional calibration is not performed. When using the optimal temperature, the curve of the rejector based on the calibrated probabilities is indistinguishable from the one of the original probabilities in Figure 3.1a. This suggests that the probability scores outputted by the neural network, although not a perfectly probability calibrated score, is well calibrated for rejection purposes.

3.6.2 Disagreement-based and fusion criteria

In order to study the performance of disagreement-based criteria, we train an ensemble of 10 neural networks in a similar fashion than for confidence-based criteria. This ensemble is built using different initializations and different sampling sequences of the training set.

In Figure 3.2, the different disagreement-based and fusion criteria presented in Section 3.5 are compared to a confidence-based criterion computed on the mean probability of the ensemble. It is known that CIFAR-10 is intrinsically ambiguous: on some samples, even human annotators disagree on the exact class (Peterson et al., 2019). In Figure 3.2a, we can see that disagreement-based criteria perform badly compared to the baseline set by the confidence-based criterion. Moreover, the best performing disagreement-based criterion, the disagreement probability, suffers from estimation issues and requires a lot of models in the ensemble to work properly. Unfortunately, the fusion criterion, although close, does not manage to beat the baseline. When in presence of ambiguity, the confidence-based criterion thus sets a strong baseline.

To assess what happens when task ambiguity is absent, we perform an experiment on MNIST (LeCun et al., 1998). On this dataset, state-of-the-art models achieve a very low error rate (around 0.5% (Ciregan et al., 2012)), this shows that the ambiguity is virtually absent from this task. As shown in Figure 3.2b, in this case, disagreement-based criteria perform well. In fact, by varying the training set size, we can see that, on this dataset, all criteria converge to zero: they are thus consistent for rejection. Surprisingly, the confidence-based baseline remains the best criterion. This simple rejection criterion sets a strong baseline independently of the type and amount of uncertainty present. This means that it captures both the task ambiguity and the model uncertainty at the same time. It is thus harder to disentangle these quantities than expected.

3.7 Conclusion

In this chapter, we studied how uncertainty information can be exploited to reject some of the predictions made by a classifier which incur a high error rate. To this end, we proposed to study a particular instance of classification with reject option framework: learning a rejector when the classifier is already trained and considered fixed. We showed that this amounts to learning an uncertainty score that preserves the ordering of the point-wise error rate of the classifier. We then focused on the case where the rejector is a hand-crafted criterion because it provides a way to analyze how we could estimate the point-wise error rate directly and in particular how to exploit uncertainty information for rejection purposes. By analyzing the point-wise error rate, we showed that both task ambiguity and model uncertainty are important. We empirically showed that rejecting based on the highest probability predicted by a neural network performs consistently well, independently of the amount of aleatoric and epistemic uncertainty. This suggests that it captures both uncertainties at the same time. We attempted to build criteria that separated both uncertainties and recombined them properly however experiments showed that this is more difficult than expected.

In this chapter, we have focused on a particular rejection framework which implies refusing to answer when the error rate is high. This requires to build rejectors which capture both types of uncertainties. Finding frameworks which would necessitate to estimate only a single uncertainty could be interesting to compare to. For example, an alternative rejection framework could be to detect when the classifier makes a suboptimal prediction inducing an increased error rate with respect to the optimal classifier. This formulation might rely less strongly on the task ambiguity.

In the next chapter, we focus on task ambiguity and try to provide principled approaches to handle it. In particular, we study ways to predict a set of candidate labels instead of rejecting an ambiguous sample.

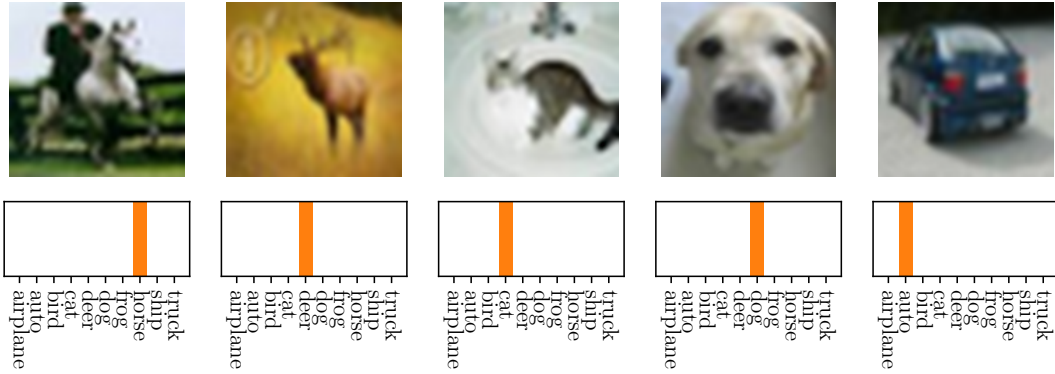
IV

Average- K versus top- K classification

Contents

4.1	From top-K to average-K classification	47
4.1.1	Multi-class classification	48
4.1.2	Top- K classification	48
4.1.3	Average- K classification	49
4.1.4	Contributions	51
4.2	Introductory examples	52
4.3	When is top-K classification optimal?	54
4.3.1	Characterization of the optimality of top- K classification	54
4.3.2	Discussion of Proposition 4.3.1: the notion of overlap of ambiguity	57
4.3.3	Proof of Proposition 4.3.1	58
4.4	Quantifying the usefulness of average-K	60
4.4.1	A lower bound on the adaptive gain	60
4.4.2	Discussion of the notion of dispersion strength	61
4.5	Consistent estimation procedures	64
4.5.1	Plug-in estimators and their regret bounds	65
4.5.2	Proper losses and strongly proper losses	67
4.6	Experiments on real datasets	70
4.6.1	Practical estimation of top- K and average- K classifiers	70
4.6.2	Experiments on altered real data: potential of average- K classification	74
4.6.3	Experiments on unaltered real-world datasets	77
4.7	Conclusion	79

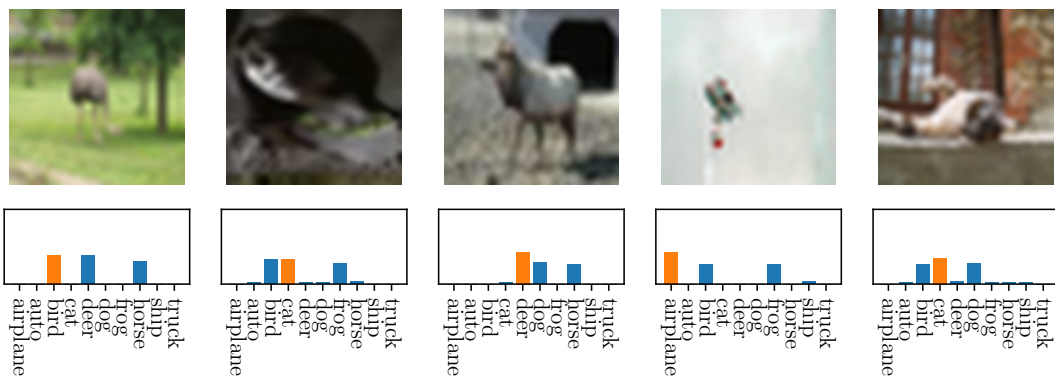
Consider the problem of assigning a label to an object. In applications where the label noise is low, predicting a single label works well. However, in a wide range of applications, at least some data items may be ambiguous due to noise or occlusion and even expert human annotators may disagree on what the true label should be. In such cases, human experts may prefer either to refuse to answer or to give a list of possible answers, filtering out the classes that certainly are wrong.



(a) Non-ambiguous examples.



(b) Ambiguity between two classes.



(c) Ambiguity between three classes.

Figure 4.1: Various degrees of ambiguity according to human annotators on CIFAR-10 collected in [Peterson et al. \(2019\)](#). The orange class corresponds to the most probable class according to these annotators.

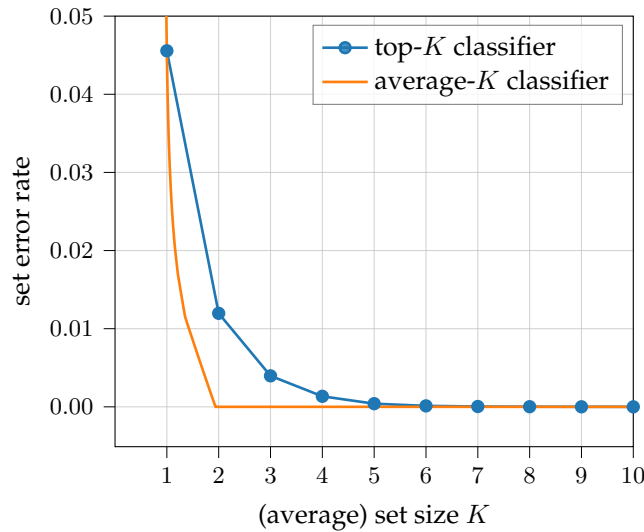


Figure 4.2: Comparison of error rate of top- K and of average- K strategy studied in this chapter computed on the human uncertainty annotations of CIFAR-10H (Peterson et al., 2019). Lower is better.

Researchers have found that even on simple tasks, such as CIFAR-10, humans sometimes disagree (Peterson et al., 2019). This is illustrated in Figure 4.1 which shows images with different levels of disagreement according to annotators. The first row displays non-ambiguous samples, whereas the second and third rows contain images with, respectively, two and three possible classes. In all cases, a single object is present in the image, however, for some samples, the low image resolution does not allow to determine precisely what this object is¹. However, in most cases, the observer can still easily rule out most classes, so filtering is still possible even when there is ambiguity.

In this chapter, we study an adaptive classification approach where predictors are allowed to act like human experts by responding with a list of candidate labels. The size of the list may vary depending on the sample being classified but is constrained to be equal to a certain size K on average. We denote this *average- K classification* as a generalization of *top- K classification*. In our problem setting, as in Figure 4.1, there is a single correct label, but the input provided may be ambiguous. Our problem is thus a different task from multi-label classification (Zhang and Zhou, 2013) where there are multiple true answers, all of them correct, which are given during training. Figure 4.2 shows how, on CIFAR-10H, an adaptive average- K strategy has the potential to lower the error rate compared to always predicting the same number of classes for each sample.

The chapter is organized as follows. After formalizing the average- K classification task, we determine the contexts in which the adaptive strategy is most beneficial compared to top- K classification. Moreover, we give estimation procedures to estimate those strategies and show that they are consistent. Finally, we perform experiments on real-world image datasets to show how average- K classification can be helpful in practice.

4.1 From top- K to average- K classification

In this section, we detail the formalization and notations that will be used in the rest of the chapter. We briefly recall the multi-class classification and top- K classification settings, extend

¹CIFAR-10 images have a size of 28x28.

them to the average- K classification task and synthesize the main contributions of the chapter.

4.1.1 Multi-class classification

In the multi-class classification setting (Tewari and Bartlett, 2007), we are given a training dataset $(\mathbf{x}_i, y_i)_{i \in \{1, \dots, n\}}$ where, for all i , the input \mathbf{x}_i belongs to an input space $\mathcal{X} \subset \mathbb{R}^d$, e.g. the space of images, and the output y_i to an output space $\mathcal{Y} = \{1, 2, \dots, C\}$, the set of class labels, where C is the number of classes. The joint space $\mathcal{X} \times \mathcal{Y}$ is a probabilistic space and the data points are sampled from the joint probability measure $\mathbb{P}_{\mathbf{X}, Y}$. This joint probability can, in turn, be decomposed into the marginal probability measure on \mathcal{X} , denoted $\mathbb{P}_{\mathbf{X}}$, and the conditional probability of Y given \mathbf{X} , denoted $\eta(\mathbf{x})$:

$$\eta_k(\mathbf{x}) := \mathbb{P}[Y = k \mid \mathbf{X} = \mathbf{x}].$$

In the classical setting, the aim is to find a classifier $h : \mathcal{X} \rightarrow \mathcal{Y}$ which produces a single prediction for each input and generalizes well on unseen data. This is formalized by trying to minimize the risk defined by the 0-1 error rate,

$$R(h) := \mathbb{E}_{\mathbf{X}, Y} [\mathbb{1}_{Y \neq h(\mathbf{X})}] = \mathbb{P}_{\mathbf{X}, Y} [Y \neq h(\mathbf{X})] \quad (4.1)$$

where $\mathbb{1}_P$ is equal to 1 if the logical proposition P is true and to 0 otherwise.

Any predictor satisfying, for all $\mathbf{x} \in \mathcal{X}$,

$$h^*(\mathbf{x}) \in \arg \max_{k \in \mathcal{Y}} \eta_k(\mathbf{x})$$

minimizes the previous risk, i.e. $R(h^*) = \inf_h R(h)$. They are called Bayes predictors. For simplicity, we usually assume that there is a single minimizer and write $h^*(\mathbf{x}) = \arg \max_k \eta_k(\mathbf{x})$. In this case, we have

$$R(h^*) = 1 - \mathbb{E}_{\mathbf{X}} \left[\max_k \eta_k(\mathbf{X}) \right].$$

In general, the minimizer does not achieve a risk of zero. This occurs only when $\max_k \eta_k(\mathbf{x}) = 1$ almost everywhere, i.e. when task ambiguity is absent. However, as previously shown, in a lot of practical datasets, such as CIFAR-10, ImageNet, and others, there is some intrinsic ambiguity in the task and we have $R(h^*) > 0$. From a top-1 classification perspective, this risk is irreducible: it is the lowest achievable risk. It is however possible to reduce it by moving away from top-1 classification. That is by predicting sets.

4.1.2 Top- K classification

The most natural and direct extension to top-1 classification is to take the top- K most probable classes and make this set as the new prediction. In this case, the same number of labels K is predicted for each sample.

The learning problem can be formulated as follows. Denoting $\mathcal{P}(\mathcal{Y})$ the power set of \mathcal{Y} , i.e. the set of all subsets of \mathcal{Y} , the classifier S is now of the form $S : \mathcal{X} \rightarrow \mathcal{P}(\mathcal{Y})$: it takes values from \mathcal{X} and predicts a set of labels from \mathcal{Y} . Moreover, it has the following additional constraint:

$$\forall \mathbf{x} \in \mathcal{X}, |S(\mathbf{x})| = K.$$

The objective, the risk of Equation (4.1), is then modified to minimize the set error rate:

$$\mathcal{E}(S) := \mathbb{E}_{\mathbf{X}, Y} [\mathbb{1}_{Y \notin S(\mathbf{X})}] = \mathbb{P}_{\mathbf{X}, Y} [Y \notin S(\mathbf{X})].$$

In this case, this error rate is the top- K error rate.

As expected, this error rate is minimized by predicting the top- K most probable classes for any input x (Lapin et al., 2016). We denote by $\eta_{\sigma_x(k)}(x)$ the re-ranking of $\eta_k(x)$ in decreasing order of probability, i.e. $\eta_{\sigma_x(1)}(x) \geq \eta_{\sigma_x(2)}(x) \geq \dots \geq \eta_{\sigma_x(C)}(x)$. The minimizer of the top- K error rate can then be written as

$$S_K^*(x) := \{\sigma_x(k) : k \in \{1, \dots, K\}\} = \sigma_x[\{1, \dots, K\}]. \quad (4.2)$$

To simplify these notations, in the rest of the document, we will often use the following notation for the re-ordering of $\eta_k(x)$:

$$\tilde{\eta}_k(x) := \eta_{\sigma_x(k)}(x).$$

This top- K classification approach is the most direct generalization of top-1 classification, but there is no reason to believe that, in general, we should predict the same number of classes for all the samples. We thus propose to rather look at adaptive set-valued classifiers which relax this constraint.

4.1.3 Average- K classification

By removing the constraint of always predicting a fixed set size of K , i.e. $|S(x)| = K$, we consider a predictor \mathcal{S} which outputs any subset of \mathcal{Y} . It is of the form

$$\mathcal{S} : \mathcal{X} \rightarrow \mathcal{P}(\mathcal{Y}),$$

where $\mathcal{P}(\mathcal{Y})$ is the power set of \mathcal{Y} . The naive goal is, as previously, to maximize the chance of predicting the good label. (Note that, in general, the predictor \mathcal{S} may output the empty set \emptyset .)

However, only considering the set error rate, i.e.

$$\mathcal{E}(\mathcal{S}) := \mathbb{P}_{\mathbf{X}, Y} [Y \notin \mathcal{S}(\mathbf{X})],$$

is too naive, because the predictor $\mathcal{S}_{\text{all}}(x) = \mathcal{Y}$ – i.e. always predict all classes – is very accurate, but completely uninformative. Thus, this accuracy needs to be balanced with a measure of informativeness $\mathcal{I}(\mathcal{S})$. There are several ways to define these two quantities – accuracy and informativeness – and to measure the trade-off between them. This gives different formulations of the set-valued classification problem which are adapted to different scenarios. We refer the reader to the next chapter, Chapter 5, for more information on the different existing approaches.

Because we are interested in a generalization of top- K , we propose to relax the constraint of a fixed set size by changing it to a constraint on the *average* set size which we take as measure of informativeness:

$$\mathcal{I}(\mathcal{S}) := \mathbb{E}_{\mathbf{X}} [|\mathcal{S}(\mathbf{X})|].$$

This gives the following optimization problem

$$\begin{aligned} \min_{\mathcal{S}} \quad & \mathbb{P}_{\mathbf{X}, Y} [Y \notin \mathcal{S}(\mathbf{X})] \\ \text{s.t.} \quad & \mathbb{E}_{\mathbf{X}} [|\mathcal{S}(\mathbf{X})|] \leq \mathcal{K} \end{aligned} \quad (4.3)$$

where the constraint on the average set size, $\mathcal{K} > 0$, is not limited to integers but can take any real value. This setting corresponds exactly to the setting studied by [Denis and Hebiri \(2017\)](#). In the rest of this section, we recall the main results that we will be needing for our work.

Firstly, this constrained optimization problem has the same solution as the following risk minimization problem which was proposed by [Ha \(1997a\)](#):

$$\begin{aligned} R_\lambda(\mathcal{S}) &:= \mathcal{E}(\mathcal{S}) + \lambda \mathcal{I}(\mathcal{S}) \\ &= \mathbb{E}_{\mathbf{X}, Y} \left[\mathbb{1}_{Y \notin \mathcal{S}(\mathbf{X})} + \lambda |\mathcal{S}(\mathbf{X})| \right] \end{aligned}$$

where $\lambda \in \mathbb{R}^+$ is a parameter related to \mathcal{K} in a way that we will explain later in this subsection. Note that this risk formulation assumes that the cost of predicting additional labels increases proportionally to some cost coefficient $\lambda > 0$.

Given that

$$\begin{aligned} R_\lambda(\mathcal{S}) &= \mathbb{E}_{\mathbf{X}} \left[\left(1 - \sum_{k \in \mathcal{S}(\mathbf{X})} \eta_k(\mathbf{x}) \right) + \sum_{k \in \mathcal{S}(\mathbf{X})} \lambda \right] \\ &= \mathbb{E}_{\mathbf{X}} \left[1 + \sum_{k \in \mathcal{S}(\mathbf{X})} (\lambda - \eta_k(\mathbf{x})) \right], \end{aligned}$$

the optimal Bayes predictor \mathcal{S}_λ^* of this risk can be easily derived and is equal to

$$\mathcal{S}_\lambda^*(\mathbf{x}) := \{k \in \mathcal{Y} : \eta_k(\mathbf{x}) \geq \lambda\}.$$

The link between the cost λ and average set size \mathcal{K} is given by the following function

$$G_\eta(\lambda) := \sum_k \mathbb{P}_{\mathbf{X}} [\eta_k(\mathbf{X}) \geq \lambda]. \quad (4.4)$$

Here, $G_\eta(\lambda)$ is exactly the average set size obtained when using the threshold λ , i.e.

$$G_\eta(\lambda) = \mathbb{E}_{\mathbf{X}} [|\mathcal{S}_\lambda^*(\mathbf{X})|].$$

In order to compute the threshold corresponding to a given average set size, we need to “inverse” the previous function. However, in general, it is not invertible and we need to consider the following generalized inverse function,

$$G_\eta^{-1}(\mathcal{K}) := \inf \{\lambda \in [0, 1] : G_\eta(\lambda) \leq \mathcal{K}\}. \quad (4.5)$$

Thus, in general, the solution of our optimization problem defined in [Equation \(4.3\)](#) can be expressed as

$$\mathcal{S}_\mathcal{K}^*(\mathbf{x}) = \{k \in \mathcal{Y} : \eta_k(\mathbf{x}) \geq G_\eta^{-1}(\mathcal{K})\}. \quad (4.6)$$

The optimal strategy that minimizes the error rate for a given constraint on the average set size consists of a simple thresholding on the values of the conditional probabilities $\mathbb{P}[Y = k | \mathbf{X} = \mathbf{x}]$.

As we will see, it is useful to make the following continuity assumption.

Assumption C (Continuity assumption). *For all $k \in \{1, \dots, C\}$, the cumulative distribution function (CDF) of η_k denoted F_{η_k} , defined as $F_{\eta_k}(t) = \mathbb{P}_{\mathbf{X}} [\eta_k(\mathbf{X}) \leq t]$, is continuous.*

This assumption requires only the probabilities of the classes not to be concentrated at some precise values: they should always be some variations present, although as small as wanted but not null. Note that it does not forbid the presence of jumps in these probabilities: some values of the probabilities may never occur. This assumption is important as it guarantees that the previous optimal predictor of Equation (4.6) satisfies $\mathcal{I}(\mathcal{S}_K^*) = \mathcal{K}$. That is to say, the average size of the sets predicted by the optimal classifier \mathcal{S}_K^* is indeed equal to \mathcal{K} . For this reason, we denote such classifiers *average- \mathcal{K} classifiers*.

Finally, a final important result which we will be using later concerns the regret of such predictors. In particular, it shows that the optimal classifier \mathcal{S}_K^* is indeed the average- \mathcal{K} classifier with the lowest error rate.

Proposition 4.1.1 (Denis and Hebiri (2017), Proposition 4). *Under Assumption C, for any average- \mathcal{K} predictor \mathcal{S}_K , i.e. having exactly an average set size of \mathcal{K} , its regret compared to the optimal predictor \mathcal{S}_K^* is equal to*

$$\mathcal{E}(\mathcal{S}_K) - \mathcal{E}(\mathcal{S}_K^*) = \sum_k \mathbb{E}_X \left[\left| \eta_k(\mathbf{X}) - G_\eta^{-1}(\mathcal{K}) \right| \mathbb{1}_{k \in \mathcal{S}_K(\mathbf{X}) \triangle \mathcal{S}_K^*(\mathbf{X})} \right]$$

where \triangle is the symmetric difference between two sets.

Note that, in the case where $\mathcal{K} = K$, because a top- K classifier is also an average- K classifier, this result also implies that the error rate of the optimal average- K classifier is necessarily lower than that of the optimal top- K classifier.

4.1.4 Contributions

The main goal of this chapter is to study the relative usefulness of average- K compared to top- K classification. Therefore, we will compare the two following optimal predictors:

1. optimal top- K classifier (Section 4.1.2):

$$S_K^*(\mathbf{x}) = \{\sigma_{\mathbf{x}}(k) : k \in \{1, \dots, K\}\}, \quad (4.7)$$

where $\sigma_{\mathbf{x}}$ is a permutation of $\{1, \dots, C\}$ such that $\eta_{\sigma_{\mathbf{x}}(1)}(\mathbf{x}) \geq \eta_{\sigma_{\mathbf{x}}(2)}(\mathbf{x}) \geq \dots \geq \eta_{\sigma_{\mathbf{x}}(C)}(\mathbf{x})$;

2. optimal average- \mathcal{K} classifier (Section 4.1.3):

$$\mathcal{S}_K^*(\mathbf{x}) = \{k \in \mathcal{Y} : \eta_k(\mathbf{x}) \geq \lambda_K\}, \quad (4.8)$$

where $\lambda_K = G_\eta^{-1}(\mathcal{K})$.

In particular, we will compare both classifiers for the same fixed average set size of $\mathcal{K} = K$. This chapter tackles the following questions:

1. For which problems is a top- K strategy $S_K^*(\mathbf{x})$ optimal w.r.t. the adaptive counterpart $\mathcal{S}_K^*(\mathbf{x})$?
2. For which problems does this average- K strategy $\mathcal{S}_K^*(\mathbf{x})$ achieve a lower error rate than the fixed set size one $S_K^*(\mathbf{x})$ and by how much?
3. How can one build consistent estimators $\hat{S}_K(\mathbf{x})$ and $\hat{\mathcal{S}}_K(\mathbf{x})$ for these two strategies?
4. Given a dataset, will an adaptive strategy $\hat{\mathcal{S}}_K(\mathbf{x})$ always outperform the top- K one $\hat{S}_K(\mathbf{x})$ in practice?

We will address these questions in order.

In [Section 4.2](#), we start by giving some introductory toy examples to show cases where top- K and average- K are useful. These examples serve as illustrations and will be referred to in the next sections.

In [Section 4.3](#), we study when is S_K^* is optimal, or stated differently when is \mathcal{S}_K^* useless with respect to S_K^* . In particular, we characterize the problems for which this happens by giving an interpretable necessary and sufficient condition involving \mathbb{P}_X and η_k . We show that it is connected to a notion of heterogeneity of the task ambiguity. Using the introductory examples, we show that this notion of heterogeneity is not trivial: it is not captured by the variance of the ambiguity but rather depends on the overlap of the distributions of the ordered conditional probabilities $\tilde{\eta}_K$ and $\tilde{\eta}_{K+1}$.

In [Section 4.4](#), we theoretically study when \mathcal{S}_K^* provides a gain over $S_K^*(x)$ and quantify by how much. In particular, we will provide a lower bound on this improvement which highlights for which problems the adaptive strategy will be most useful. This lower bound is expressed with a quantification of the heterogeneity of the ambiguity which we call *dispersion strength*. This quantity captures both the *weight* and the *magnitude* of the previously-mentioned overlap of the distributions of the ordered conditional probabilities.

Next, in [Section 4.5](#), we provide estimation procedures and prove their consistency, both for top- K and average- K classification. These procedures are natural and intuitive, they are based on plug-in rules and strongly proper losses minimization. To prove this result, we derive new plug-in regret bounds and generalize strongly proper losses to the multi-class case. In particular, this allows to prove that building top- K and average- K classifiers from the scores of a model learned by minimizing the negative log-likelihood is consistent which is a new result.

Finally, in [Section 4.6](#), using these procedures, we study how the previous theoretical findings apply to real-world image datasets. First, we study the estimation procedures coupled with classic neural network training and show their efficiency under various conditions. Then, experiments are carried out on large-scale datasets to show the usefulness of the adaptive average- K strategy on concrete problems. Surprisingly, average- K *always* outperforms top- K , even when the amount of training data is limited.

4.2 Introductory examples

In this section, we present two toy examples and study how top- K and average- K perform in these cases. These examples are deliberately contrived to train our intuition. We'll use them in the next sections to illustrate both definitions and results. The first example shows a case where average- K does not improve upon top- K whereas the second one shows a case where it does.

For these examples, we consider applications with $C = 6$ classes, where the goal, given an input, is to return a small set containing the proper class. We denote the different classes using letters from A to F .

The first example, Example 1, is illustrated in [Figure 4.3](#). Here, the 6 classes are grouped by pairs: $\{A, B\}$, $\{C, D\}$, and $\{E, F\}$. In each case, two classes have high, roughly equal, probabilities with some variability while all other classes have low and negligible probabilities. This is illustrated in the first column containing some samples with their associated conditional probability $\eta_k(x) = \mathbb{P}[Y = k \mid X = x]$: the first sample x_1 is ambiguous between A and B ,

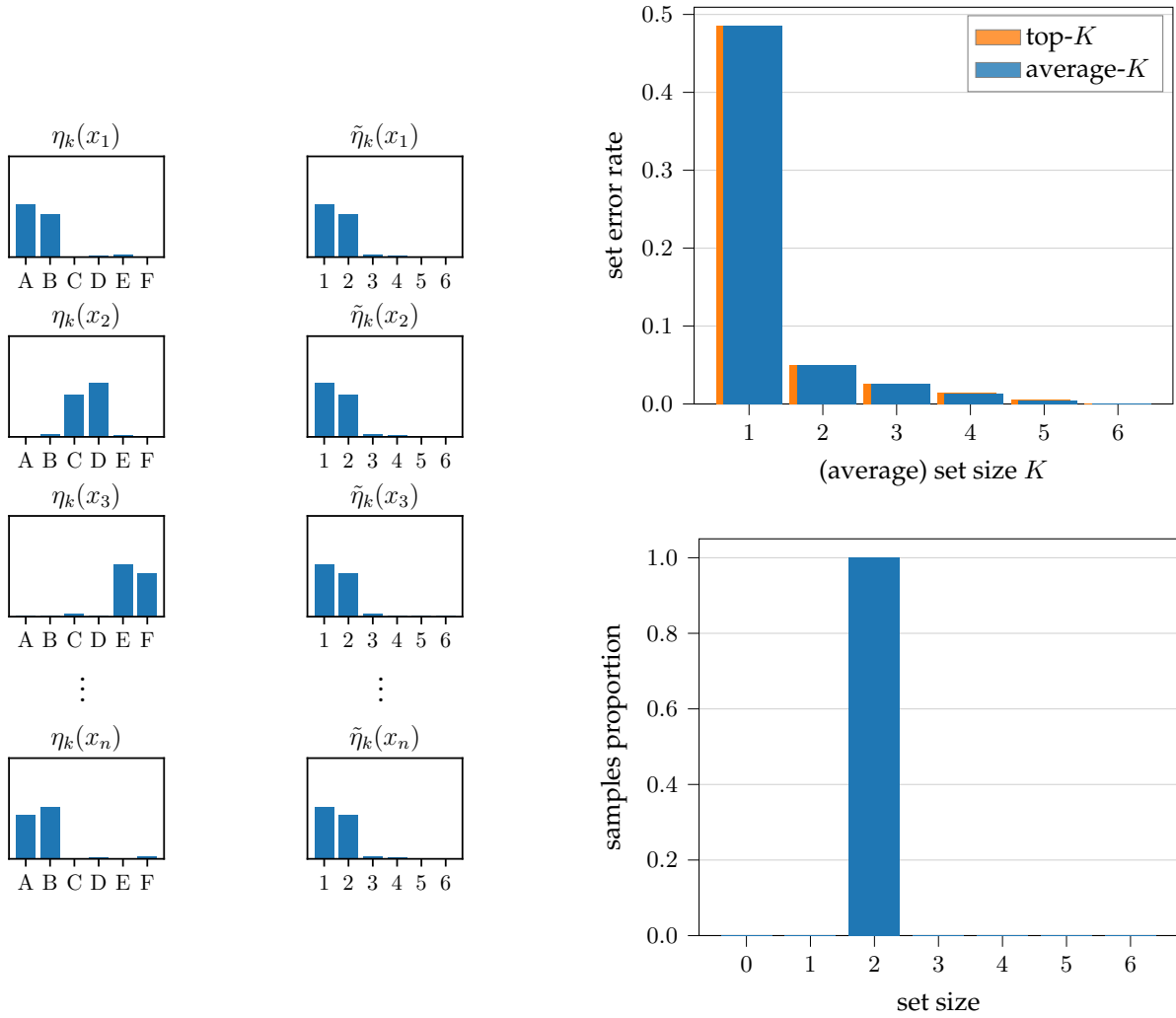


Figure 4.3: Illustration of Example 1. The first column contains samples with their associated conditional probabilities η_k over the classes. The second column orders the classes by the height of the bars in the histogram, $\tilde{\eta}_k$. For example, when classes C and D have nearly all the probability then $\tilde{\eta}_k$ simply reprises the probabilities of those two classes, in descending order of probability. The top-right figure plots the error rate for different K values for top- K and average- K . The bottom-right figure shows the distribution of the predicted set size of average- K strategy for $K = 2$. In this case, the classes are grouped by pairs and there is no gain in using average- K classifiers. A set size of 2 is best in all cases. See the text for more details.

while the second one x_2 between C and D , x_3 between E and F , etc. The second column shows these conditional probabilities reordered in descending order, denoted $\tilde{\eta}_K$. In each case, the two most probable classes are much more likely than the other ones. In this scenario, top-2 and average-2 have the same error rate. Average-2 will always choose two classes. This is illustrated in the top-right and bottom-right figures.

The second example, Example 2, is illustrated in Figure 4.4. Here, for some samples, there is no ambiguity. One class, A , has all the probability. In other cases, there is an ambiguity between two classes, B and C , and in still other cases, there is confusion between three classes, D , E and F . Each of these three cases are equally probable. Within each group of samples, there is a high confusion between the classes but classes outside the group have negligible probability. This is illustrated in the first and second column: the first sample x_1 is un-ambiguous and belongs to class A , while the second one x_2 has the same likelihood for B and C and x_3 the same likelihood for D , E and F , etc. In this scenario, top-2 reduces the error compared to top-1 classification but average-2 reduces the error rate significantly compared to top-2. In fact, as shown in the bottom-right figure, when fixing $K = 2$, the average-2 classifier predicts a balanced number of set of sizes 1, 2 and 3. For each of the groups, it predicts exactly the correct number of classes present. That is why average-2 has a zero error rate.

These two very simple examples raise the question: what makes average- K useful in one case but not in the other one? This is the aim of the next section.

4.3 When is top- K classification optimal?

In this section, we first focus on the optimality of top- K classification with respect to the adaptive strategy. In other words, when is top- K sufficient and when would the adaptive strategy improve on it?

In terms of error rates, as we will see in this section, we have the following relationship between the error rate of classic top-1 classification $\mathcal{E}(S_1^*)$, top- K classification $\mathcal{E}(S_K^*)$ and average- K classification $\mathcal{E}(\mathcal{S}_K^*)$:

$$\mathcal{E}(\mathcal{S}_K^*) \leq \mathcal{E}(S_K^*) \leq \mathcal{E}(S_1^*).$$

A first instructive question is: when is top- K useful compared to top-1 classification? The answer is found by analyzing the error rate of top- K which is given by

$$\mathcal{E}(S_K^*) = 1 - \sum_{k \leq K} \mathbb{E}_X [\tilde{\eta}_k(\mathbf{X})].$$

As we can see, the important quantity here is the average of ambiguities $\mathbb{E}_X [\tilde{\eta}_k(\mathbf{X})]$. Obviously, if there is no ambiguity, top-1 classification is optimal. However, the previous equation is more informative: top- K is solely dependent on *average ambiguities*. Intuitively, the adaptive strategy will thus be relevant when this ambiguity is *heterogeneous*, whereas top- K classification will be optimal when this ambiguity is *homogeneous*. In this section, we start formalizing this notion of heterogeneity by studying when top- K classification is optimal.

4.3.1 Characterization of the optimality of top- K classification

To study the optimality of top- K , we analyze the *adaptive gain* Δ_K defined as

$$\Delta_K := \mathcal{E}(S_K^*) - \mathcal{E}(\mathcal{S}_K^*).$$

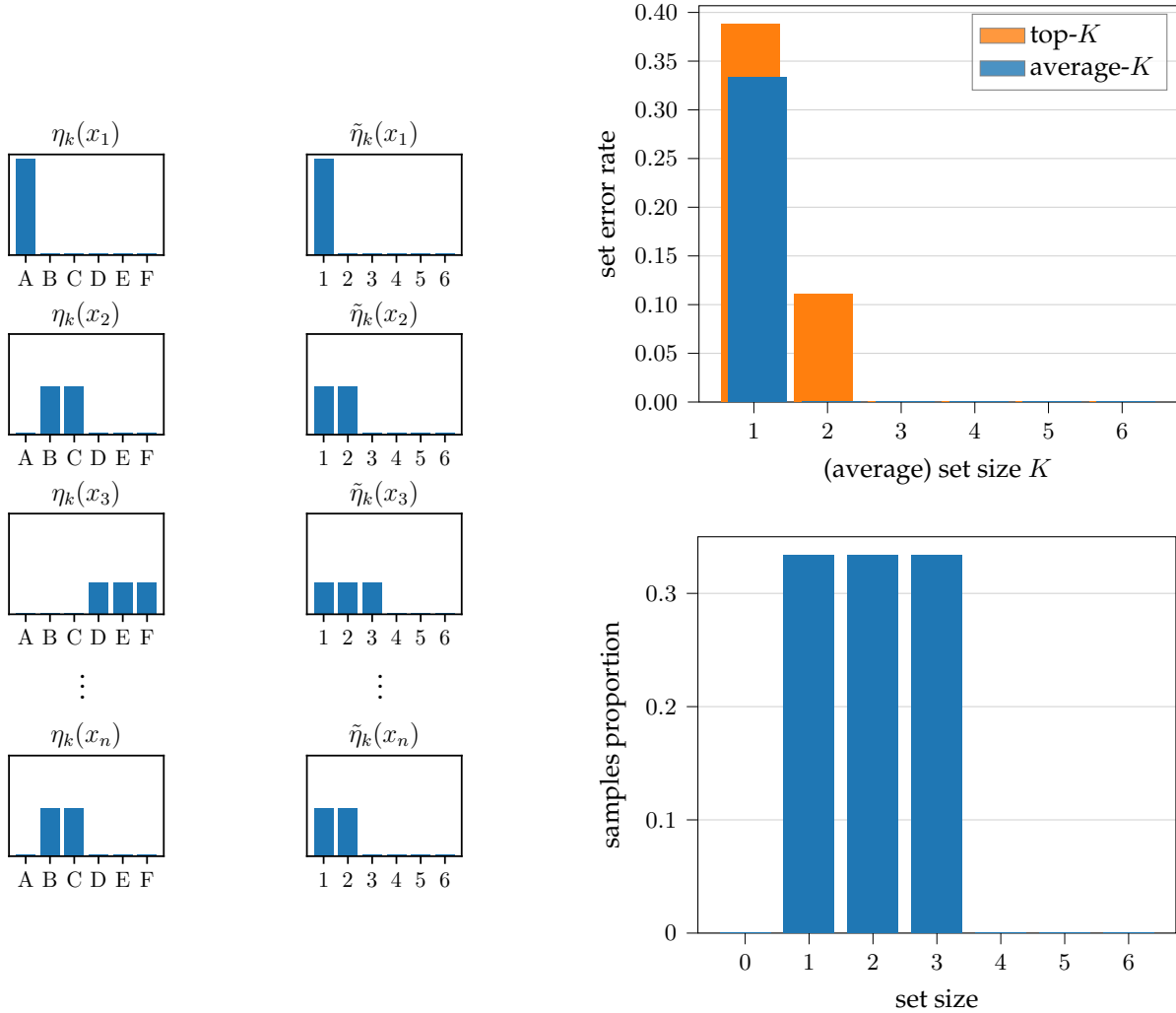


Figure 4.4: Illustration of Example 2. The first column contains four samples with their associated conditional probabilities η_k for each class. The second column reprises the probabilities in descending order of the height of the bars in the corresponding left histogram, $\tilde{\eta}_k$, giving a probability density function of the number of classes found. As it happens, $1/3$ of the time there is no ambiguity, $1/3$ of the time there is ambiguity among two classes, and $1/3$ of the time there is ambiguity among three classes. The top-right figure plots the error rates (vertical axis) for different values of K (horizontal axis) for both the top- K and average- K strategies. The orange shows the additional error when using top- K compared with using average- K . The bottom-right figure shows the distribution of the set sizes of the average- K strategy for $K = 2$, showing that the average- K strategy predicts a set size of 1, 2, or 3, each $1/3$ of the time which accords with the ambiguity distribution. For that reason, average-2 has no errors, whereas top-2 will have an error $1/3$ of the time when there is an ambiguity of 3 ($1/3 * 1/3 = 1/9$). See the text for more details.

To simplify the notation, we will denote the adapted threshold of \mathcal{S}_K^* as λ_K , i.e.

$$\lambda_K = G_\eta^{-1}(K). \quad (4.9)$$

Because top- K classifiers also have an average set size of K , using [Proposition 4.1.1](#), we have

$$\Delta_K = \sum_k \mathbb{E}_{\mathbf{X}} \left[|\eta_k(\mathbf{X}) - \lambda_K| \mathbb{1}_{k \in S_K^*(\mathbf{X}) \Delta \mathcal{S}_K^*(\mathbf{X})} \right] \quad (4.10)$$

which is clearly non-negative. Adaptive gain thus quantifies the benefit of average- K over top- K .

In this section, we focus on characterizing when adaptive average- K prediction falls back to top- K prediction, i.e. when $\Delta_K = 0$. In particular, we want to characterize the problems for which this occurs, i.e. give a result involving properties of $\mathbb{P}_{\mathbf{X}, Y}$ (thus $\mathbb{P}_{\mathbf{X}}$ and η_k).

The following proposition is the main result of this section, it gives a natural necessary and sufficient condition for the usefulness/uselessness of the average- K strategy compared to top- K . In particular, it is expressed in terms of the inverse images of the interval $(0, 1)$ under the conditional density function (CDF) of $\tilde{\eta}_K$ and $\tilde{\eta}_{K+1}$, denoted respectively $F_{\tilde{\eta}_K}^{-1}[(0, 1)]$ and $F_{\tilde{\eta}_{K+1}}^{-1}[(0, 1)]$ and equal to

$$F_{\tilde{\eta}_K}^{-1}[(0, 1)] = \{t \in [0, 1] \mid F_{\tilde{\eta}_K}(t) \in (0, 1)\}$$

and

$$F_{\tilde{\eta}_{K+1}}^{-1}[(0, 1)] = \{t \in [0, 1] \mid F_{\tilde{\eta}_{K+1}}(t) \in (0, 1)\}.$$

Intuitively, these sets tell us where the distributions of $\tilde{\eta}_K$ and $\tilde{\eta}_{K+1}$ are located². In the next subsection, we will show what these quantities mean and what are their values for the introductory examples.

Proposition 4.3.1 (Uselessness characterization). *Under the continuity [assumption C](#), for a fixed $K < C$, the following statements are equivalent:*

1. *The adaptive gain of using average- K is null:*

$$\Delta_K = 0,$$

2. *Top- K and average- K predict the same sets:*

$$S_K^*(\mathbf{X}) = \mathcal{S}_K^*(\mathbf{X}) \text{ almost everywhere,}$$

3. *There is a gap between the support of the distributions of the conditional probabilities $\tilde{\eta}_K$ and $\tilde{\eta}_{K+1}$:*

$$\exists \lambda \in [0, 1] \mid (\tilde{\eta}_{K+1}(\mathbf{X}) < \lambda \leq \tilde{\eta}_K(\mathbf{X}) \text{ almost everywhere}),$$

4. *There is no overlap in the distributions of the conditional probabilities $\tilde{\eta}_K$ and $\tilde{\eta}_{K+1}$:*

$$F_{\tilde{\eta}_K}^{-1}[(0, 1)] \cap F_{\tilde{\eta}_{K+1}}^{-1}[(0, 1)] = \emptyset.$$

Note that, although the statements 1 and 2 seem clearly equivalent at first glance, without the continuity assumption the implication 1 \Rightarrow 2 does not hold. This assumption is thus important in our context. Before providing the proof of [Proposition 4.3.1](#) (in [Section 4.3.3](#)), we discuss in the next subsection its interpretation and consequences.

²These quantities are not equivalent to the support of the distributions but they are rather equal to the interior of the smallest (closed) segment containing the support of the distribution.

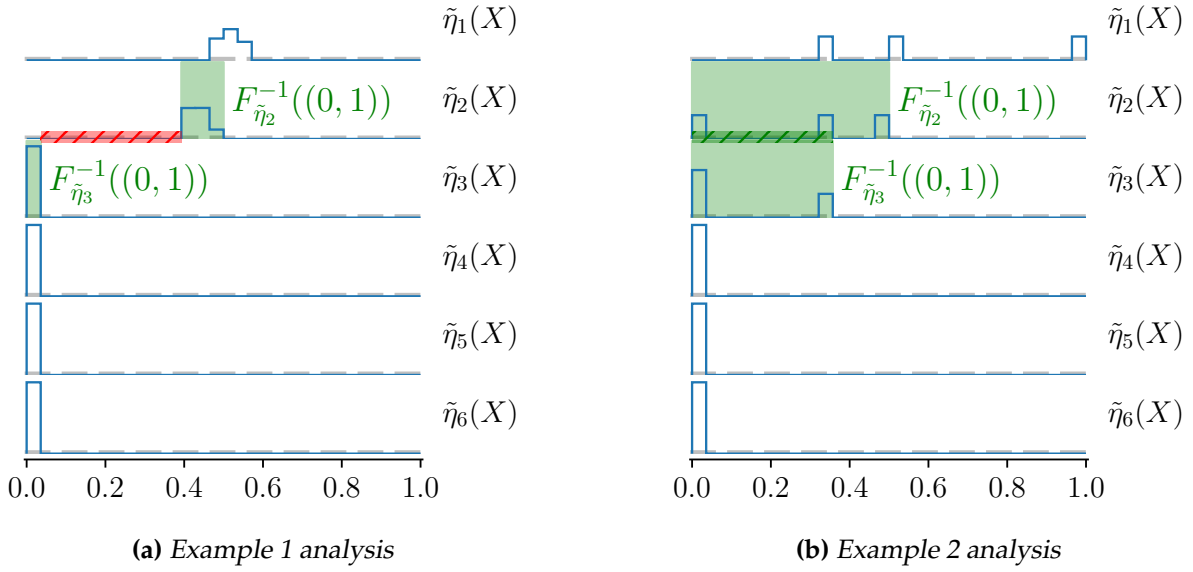


Figure 4.5: Distribution of $\tilde{\eta}_k(X)$ for all k for Example 1 and Example 2. In green, $F_{\tilde{\eta}_K}^{-1}[(0, 1)]$ and $F_{\tilde{\eta}_{K+1}}^{-1}[(0, 1)]$ are shown in both cases. The hatched red area highlights the lack of conditional density overlap in Example 1 whereas the hatched green area shows the overlap in Example 2. In this case, it shows that sometimes the second most likely class is roughly as likely as the third most likely class and sometimes their likelihoods are very different. When their likelihoods are similar, adaptive will return the top three classes. When the two highest probability classes have similar and relatively high probabilities, then adaptive will return only two classes. When the two lower probability classes have very small probabilities, then adaptive will return only one class.

4.3.2 Discussion of Proposition 4.3.1: the notion of overlap of ambiguity

Statements 3 and 4 of Proposition 4.3.1 highlight that the overlap (or its absence) between the conditional density functions (hereafter called "density overlap" or just "overlap") of the different $\tilde{\eta}_k$ is an important characteristic of the problem. In this subsection, we revisit the introductory examples of Section 4.2 and, based on those, show what type of heterogeneity of ambiguity is adaptive gain related to.

In Figure 4.5, the distribution of $\tilde{\eta}_k(X)$ for all k is shown, both for Example 1 and Example 2. Recall that the $\tilde{\eta}_k(X)$ is the probability distribution for the classes in descending order (e.g. $\tilde{\eta}_1(X)$ is the probability distribution for the most likely class). $F_{\tilde{\eta}_K}^{-1}[(0, 1)]$ and $F_{\tilde{\eta}_{K+1}}^{-1}[(0, 1)]$ are shown in green in both cases.

In Example 1, as explained previously, top- K is optimal. However, as shown here, there is a variability of $\tilde{\eta}_K$. Thus, the variance of $\tilde{\eta}_K$ is not a good notion of heterogeneity to capture the usefulness of average- K compared to top- K . On the other hand, the quantities of Proposition 4.3.1 capture this notion well. Indeed, the hatched red area on the figure highlights the lack of conditional density overlap in Example 1: in this case, $F_{\tilde{\eta}_K}^{-1}[(0, 1)] \cap F_{\tilde{\eta}_{K+1}}^{-1}[(0, 1)] = \emptyset$.

In Example 2, by contrast, the hatched green area shows the overlap in Example 2. As noted in the figure caption, sometimes the second most likely class is roughly as likely as the third most likely class and sometimes their likelihoods are very different. When their likelihoods are similar, average- K will return the top three classes. When the two lower probability classes have very small probabilities, then average- K will return only one class. In this case, average- K is more effective than top- K . We will quantify the exact gain in Section 4.4.

These simple examples illustrate that the notion of heterogeneity introduced in [Proposition 4.3.1](#), i.e. the lack of overlap of the distributions of $\tilde{\eta}_K$ and $\tilde{\eta}_{K+1}$ characterizes cases when top- K is optimal. By contrast and somewhat surprisingly, a measure of variability such as the variance, which would have been a natural candidate at first glance, is actually not the appropriate quantity to measure. Even with some variability of the $\tilde{\eta}_k$, top- K can remain optimal. However, as we will see in [Section 4.4](#), a quantification of overlap will characterize the benefit that the average- K strategy confers.

4.3.3 Proof of Proposition 4.3.1

Let us now prove [Proposition 4.3.1](#). To do so, we will prove the different equivalence separately.

Proof of equivalence 1 $\Leftrightarrow 2$. The implication $2 \Rightarrow 1$ is straightforward as

$$\begin{aligned} S_K^*(\mathbf{X}) = \mathcal{S}_K^*(\mathbf{X}) \quad \text{a.s.} &\Rightarrow S_K^*(\mathbf{X}) \triangle \mathcal{S}_K^*(\mathbf{X}) = \emptyset \quad \text{a.s.} \\ &\Rightarrow \Delta_K = 0 \end{aligned}$$

using [Equation \(4.10\)](#).

The converse requires to use the continuity assumption. Indeed, using [Equation \(4.10\)](#), we have

$$\Delta_K = 0 \Rightarrow \forall k, \mathbb{P}_{\mathbf{X}} [\eta_k(\mathbf{X}) = \lambda_K \text{ or } k \notin S_K^*(\mathbf{X}) \triangle \mathcal{S}_K^*(\mathbf{X})] = 1.$$

By continuity assumption, we have $\mathbb{P}_{\mathbf{X}} [\eta_k(\mathbf{X}) = \lambda_K] = 0$, which implies that

$$\forall k, \mathbb{P}_{\mathbf{X}} [k \notin S_K^*(\mathbf{X}) \triangle \mathcal{S}_K^*(\mathbf{X})] = 1.$$

This implies that

$$\begin{aligned} \mathbb{P}_{\mathbf{X}} [S_K^*(\mathbf{X}) \triangle \mathcal{S}_K^*(\mathbf{X}) = \emptyset] &= 1 - \mathbb{P}_{\mathbf{X}} [\exists k \mid k \in S_K^*(\mathbf{X}) \triangle \mathcal{S}_K^*(\mathbf{X})] \\ &\geq 1 - \sum_k \underbrace{\mathbb{P}_{\mathbf{X}} [k \in S_K^*(\mathbf{X}) \triangle \mathcal{S}_K^*(\mathbf{X})]}_{=1 - \mathbb{P}_{\mathbf{X}} [k \notin S_K^*(\mathbf{X}) \triangle \mathcal{S}_K^*(\mathbf{X})] = 0} \end{aligned}$$

thus, we have

$$\mathbb{P}_{\mathbf{X}} [S_K^*(\mathbf{X}) \triangle \mathcal{S}_K^*(\mathbf{X}) = \emptyset] = 1.$$

As, for any given \mathbf{X} , there are only two alternatives, either $S_K^*(\mathbf{X}) \subset \mathcal{S}_K^*(\mathbf{X})$ or $\mathcal{S}_K^*(\mathbf{X}) \subset S_K^*(\mathbf{X})$, we have

$$S_K^*(\mathbf{X}) \triangle \mathcal{S}_K^*(\mathbf{X}) = \emptyset \Leftrightarrow S_K^*(\mathbf{X}) = \mathcal{S}_K^*(\mathbf{X})$$

which allows to conclude the implication

$$\Delta_K = 0 \Rightarrow \mathbb{P}_{\mathbf{X}} [S_K^*(\mathbf{X}) = \mathcal{S}_K^*(\mathbf{X})] = 1.$$

□

Proof of equivalence 2 $\Leftrightarrow 3$. To prove the implication $2 \Rightarrow 3$, we use

$$\begin{aligned} S_K^*(\mathbf{X}) = \mathcal{S}_K^*(\mathbf{X}) \text{ a.s.} &\Rightarrow \mathbb{P}_{\mathbf{X}} [|\mathcal{S}_K^*(\mathbf{X})| = K] = 1 \\ &\Rightarrow \mathbb{P}_{\mathbf{X}} [\tilde{\eta}_{K+1}(\mathbf{X}) < \lambda_K] = \mathbb{P}_{\mathbf{X}} [\tilde{\eta}_K(\mathbf{X}) \geq \lambda_K] = 1 \\ &\Rightarrow \mathbb{P}_{\mathbf{X}} [\tilde{\eta}_{K+1}(\mathbf{X}) < \lambda_K \leq \tilde{\eta}_K(\mathbf{X})] = 1. \end{aligned}$$

as we have

$$\mathbb{P}_{\mathbf{X}} [\tilde{\eta}_{K+1}(\mathbf{X}) < \lambda_K \leq \tilde{\eta}_K(\mathbf{X})] = \mathbb{P}_{\mathbf{X}} [\tilde{\eta}_{K+1}(\mathbf{X}) < \lambda_K] \mathbb{P}_{\mathbf{X}} [\tilde{\eta}_K(\mathbf{X}) \geq \lambda_K].$$

Thus λ_K is a possible value of λ of statement 3.

Conversely, denote Λ the following set:

$$\Lambda = \{\lambda \in [0, 1] \mid \mathbb{P}_{\mathbf{X}} [\tilde{\eta}_{K+1}(\mathbf{X}) < \lambda \leq \tilde{\eta}_K(\mathbf{X})] = 1\}.$$

If $\Lambda \neq \emptyset$, then for all $\lambda \in \Lambda$, we have $G_{\eta}(\lambda) = K$:

$$\begin{aligned} \lambda \in \Lambda &\Rightarrow \begin{cases} \mathbb{P}_{\mathbf{X}} [\tilde{\eta}_K(\mathbf{X}) \geq \lambda] = 1 \\ \mathbb{P}_{\mathbf{X}} [\tilde{\eta}_{K+1}(\mathbf{X}) \geq \lambda] = 0 \end{cases} \\ &\Rightarrow \begin{cases} \forall k \leq K, \mathbb{P}_{\mathbf{X}} [\tilde{\eta}_k(\mathbf{X}) \geq \lambda] = 1 \\ \forall k > K, \mathbb{P}_{\mathbf{X}} [\tilde{\eta}_k(\mathbf{X}) \geq \lambda] = 0 \end{cases} \\ &\Rightarrow G_{\eta}(\lambda) = \sum_k \mathbb{P}_{\mathbf{X}} [\tilde{\eta}_k(\mathbf{X}) \geq \lambda] = K. \end{aligned}$$

As λ_K is defined in Equation (4.9) as the infimum over all λ such that $G_{\eta}(\lambda) \leq K$, we have $\lambda_K \leq \inf \Lambda$. If we assume that $\lambda_K \notin \Lambda$, then necessarily

$$\begin{cases} \mathbb{P}_{\mathbf{X}} [\tilde{\eta}_K(\mathbf{X}) \geq \lambda_K] = 1 \\ \mathbb{P}_{\mathbf{X}} [\tilde{\eta}_{K+1}(\mathbf{X}) \geq \lambda_K] > 0 \end{cases}$$

which implies that $G_{\eta}(\lambda_K) > K$, contradicting the fact that $G_{\eta}(\lambda_K) = K$. Thus, we have $\lambda_K \in \Lambda$ and $\mathbb{P}_{\mathbf{X}} [\tilde{\eta}_K(\mathbf{X}) \geq \lambda_K] = \mathbb{P}_{\mathbf{X}} [\tilde{\eta}_{K+1}(\mathbf{X}) < \lambda_K] = 1$. Finally, this implies that

$$S_K^*(\mathbf{X}) = \mathcal{S}_K^*(\mathbf{X}) \text{ a.s.}$$

which concludes the proof. \square

Proof of equivalence 3 \Leftrightarrow 4. Due to the continuity assumption, we know that the inverse image of open intervals of the CDF functions are also open intervals. We denote them as

$$F_{\tilde{\eta}_K}^{-1}[(0, 1)] = (a, b) \quad \text{and} \quad F_{\tilde{\eta}_{K+1}}^{-1}[(0, 1)] = (c, d)$$

with $a < b$ and $c < d$.

To show the first implication, we start from $F_{\tilde{\eta}_K}^{-1}[(0, 1)] \cap F_{\tilde{\eta}_{K+1}}^{-1}[(0, 1)] = \emptyset$ which implies that $d \leq a$. Setting $\lambda = a$ and using the continuity of $F_{\tilde{\eta}_K}$ and $F_{\tilde{\eta}_{K+1}}$, we have

$$F_{\tilde{\eta}_K}(\lambda) = \mathbb{P}_{\mathbf{X}} [\tilde{\eta}_K(\mathbf{X}) \leq \lambda] = 0 \Rightarrow \mathbb{P}_{\mathbf{X}} [\tilde{\eta}_K(\mathbf{X}) \geq \lambda] = 1$$

and

$$F_{\tilde{\eta}_{K+1}}(\lambda) = \mathbb{P}_{\mathbf{X}} [\tilde{\eta}_{K+1}(\mathbf{X}) \leq \lambda] = 1 \Rightarrow \mathbb{P}_{\mathbf{X}} [\tilde{\eta}_{K+1}(\mathbf{X}) < \lambda] = 1.$$

From that, we can conclude

$$\mathbb{P}_{\mathbf{X}} [\tilde{\eta}_{K+1}(\mathbf{X}) < \lambda \leq \tilde{\eta}_K(\mathbf{X})] = \mathbb{P}_{\mathbf{X}} [\tilde{\eta}_{K+1}(\mathbf{X}) < \lambda] \mathbb{P}_{\mathbf{X}} [\tilde{\eta}_K(\mathbf{X}) \geq \lambda] = 1.$$

Conversely, if such a λ exists, using the continuity assumption, it satisfies

$$\begin{cases} \mathbb{P}_{\mathbf{X}} [\tilde{\eta}_K(\mathbf{X}) \geq \lambda] = 1 \\ \mathbb{P}_{\mathbf{X}} [\tilde{\eta}_{K+1}(\mathbf{X}) \geq \lambda] = 0 \end{cases} \Rightarrow \begin{cases} \lambda \leq a \\ \lambda \geq d \end{cases} \Rightarrow d \leq a$$

which allows to conclude that the intersection is the empty set. \square

4.4 Quantifying the usefulness of average- K

Section 4.3 characterized when top- K is optimal. We showed that a form of heterogeneity is required for the adaptive strategy to improve upon top- K . In this section, we quantify how much we can reduce the error rate using the average- K strategy for some problems compared with top- K . That is the *adaptive gain*. For this, we will quantify the heterogeneity of the ambiguity of the problem using an interpretable quantity, the *dispersion strength*, expressed in terms of $\mathbb{P}_{\mathbf{X},Y}$ (thus $\mathbb{P}_{\mathbf{X}}$ and η).

4.4.1 A lower bound on the adaptive gain

Proposition 4.4.1 gives a lower bound on the adaptive gain Δ_K which partly answers this question. It is based on the *dispersion strength* which is a quantified version of the overlap of distributions of $\tilde{\eta}_k$ studied previously. It measures the heterogeneity of the ambiguity by quantifying how much the level of ambiguity can differ for different inputs.

Definition 4.4.1 (Dispersion strength). *For a fixed K , the dispersion strength of order k denoted $d_{K,k}$ is defined by*

$$d_{K,k} := \mathbb{E}_{\mathbf{X},\mathbf{X}'} \left[(\tilde{\eta}_{K+k}(\mathbf{X}) - \tilde{\eta}_{K+1-k}(\mathbf{X}'))^+ \right],$$

where $a^+ = \max(a, 0)$ is the positive part function.

Note that the difference is computed between conditional probabilities for different X and X' (otherwise, these quantities would always equal to zero). The dispersion strength of order 1, $d_{K,1}$, is equal to

$$d_{K,1} := \mathbb{E}_{\mathbf{X},\mathbf{X}'} \left[(\tilde{\eta}_{K+1}(\mathbf{X}) - \tilde{\eta}_K(\mathbf{X}'))^+ \right].$$

which, using conditional expectations, can be rewritten as

$$\begin{aligned} d_{K,1} &= \mathbb{E}_{\mathbf{X},\mathbf{X}'} [\tilde{\eta}_{K+1}(\mathbf{X}) - \tilde{\eta}_K(\mathbf{X}') \mid \tilde{\eta}_{K+1}(\mathbf{X}) > \tilde{\eta}_K(\mathbf{X}')] \\ &\quad \times \mathbb{P}_{\mathbf{X},\mathbf{X}'} [\tilde{\eta}_{K+1}(\mathbf{X}) > \tilde{\eta}_K(\mathbf{X}')]. \end{aligned}$$

These quantities measure a form of *heterogeneity* of the ambiguity: they quantify how much the level of ambiguity can be different for different inputs. The second term quantifies how often this occurs, on the other hand, the first term quantifies, when it occurs, how strong is this difference. We will discuss this aspect in more length in the next subsection after stating the main result. The dispersion strengths of higher order, $d_{K,k}$, then generalizes this idea and quantify overlaps between further away order conditional probabilities, e.g. for order 2, it compares $\tilde{\eta}_{K+2}$ and $\tilde{\eta}_{K-1}$, etc.

Using this measure of the heterogeneity of ambiguity of the problem, we can show the following proposition giving a lower bound on the adaptive gain Δ_K .

Proposition 4.4.1. *For $K < C$, where C is the number of classes, we have that*

$$\Delta_K \geq d_{K,1} = \mathbb{E}_{\mathbf{X},\mathbf{X}'} \left[(\tilde{\eta}_{K+1}(\mathbf{X}) - \tilde{\eta}_K(\mathbf{X}'))^+ \right],$$

and, more tightly, for $K \leq \frac{C}{2}$,

$$\Delta_K \geq \sum_{k=1}^K d_{K,k} = \sum_{k=1}^K \mathbb{E}_{\mathbf{X},\mathbf{X}'} \left[(\tilde{\eta}_{K+k}(\mathbf{X}) - \tilde{\eta}_{K+1-k}(\mathbf{X}'))^+ \right].$$

Proof. Using Equation (4.10), we can rewrite the adaptive gain Δ_K as

$$\begin{aligned}\Delta_K &= \sum_k \mathbb{E}_{\mathbf{X}} \left[|\eta_k(\mathbf{X}) - \lambda_K| \mathbb{1}_{k \in S_K^*(\mathbf{X}) \Delta \mathcal{S}_K^*(\mathbf{X})} \right] \\ &= \sum_{k \leq K} \mathbb{E}_{\mathbf{X}} \left[(\lambda_K - \tilde{\eta}_k(\mathbf{X})) \mathbb{1}_{\tilde{\eta}_k(\mathbf{X}) < \lambda_K} \right] \\ &\quad + \sum_{k > K} \mathbb{E}_{\mathbf{X}} \left[(\tilde{\eta}_k(\mathbf{X}) - \lambda_K) \mathbb{1}_{\tilde{\eta}_k(\mathbf{X}) \geq \lambda_K} \right] \\ &= \mathbb{E}_{\mathbf{X}} \left[\sum_{k \leq K} (\lambda_K - \tilde{\eta}_k(\mathbf{X}))^+ + \sum_{k > K} (\tilde{\eta}_k(\mathbf{X}) - \lambda_K)^+ \right]\end{aligned}$$

where $(a)^+ = \max(a, 0)$ is the positive part.

From this, we can prove the first lower bound using

$$\begin{aligned}\Delta_K &\geq \mathbb{E}_{\mathbf{X}} \left[(\lambda_K - \tilde{\eta}_K(\mathbf{X}))^+ + (\tilde{\eta}_{K+1}(\mathbf{X}) - \lambda_K)^+ \right] \\ &= \mathbb{E}_{\mathbf{X}, \mathbf{X}'} \left[(\lambda_K - \tilde{\eta}_K(\mathbf{X}'))^+ + (\tilde{\eta}_{K+1}(\mathbf{X}) - \lambda_K)^+ \right] \\ &\geq \mathbb{E}_{\mathbf{X}, \mathbf{X}'} \left[(\tilde{\eta}_{K+1}(\mathbf{X}) - \tilde{\eta}_K(\mathbf{X}'))^+ \right].\end{aligned}$$

The last lower bound comes from the fact that, for any real numbers a , b and c ,

$$(a - b)^+ + (b - c)^+ \geq ((a - b) + (b - c))^+ = (a - c)^+.$$

For the second lower bound of the proposition, we use a tighter lower bound of the sum:

$$\begin{aligned}\Delta_K &= \mathbb{E}_{\mathbf{X}} \left[\sum_{k \leq K} (\lambda_K - \tilde{\eta}_k(\mathbf{X}))^+ + \sum_{k > K} (\tilde{\eta}_k(\mathbf{X}) - \lambda_K)^+ \right] \\ &\geq \mathbb{E}_{\mathbf{X}} \left[\sum_{k=1}^K (\lambda_K - \tilde{\eta}_k(\mathbf{X}))^+ + \sum_{k=K+1}^{2K} (\tilde{\eta}_k(\mathbf{X}) - \lambda_K)^+ \right] \\ &= \sum_{k=1}^K \mathbb{E}_{\mathbf{X}, \mathbf{X}'} \left[(\lambda_K - \tilde{\eta}_{K+1-k}(\mathbf{X}'))^+ + (\tilde{\eta}_{K+k}(\mathbf{X}) - \lambda_K)^+ \right] \\ &\geq \sum_{k=1}^K \mathbb{E}_{\mathbf{X}, \mathbf{X}'} \left[(\tilde{\eta}_{K+k}(\mathbf{X}) - \tilde{\eta}_{K+1-k}(\mathbf{X}'))^+ \right]\end{aligned}$$

using the same lower bound as previously. □

4.4.2 Discussion of the notion of dispersion strength

The aim of this subsection is to give an intuitive explanation of the nature of heterogeneity and to make the dispersion strength concept more understandable. From Section 4.3, we saw that the overlap of distributions of $\tilde{\eta}_K$ and $\tilde{\eta}_{K+1}$ is important to characterize the optimality of top- K . The dispersion strength captures and measures this information.

We start by computing it for Example 2. Recall that, in this example, there are three zones: zone 1 with class A unambiguous, zone 2 with class B and C equally ambiguous, and zone 3 with class D , E , and F equally ambiguous. Each of these zones is equally probable, their

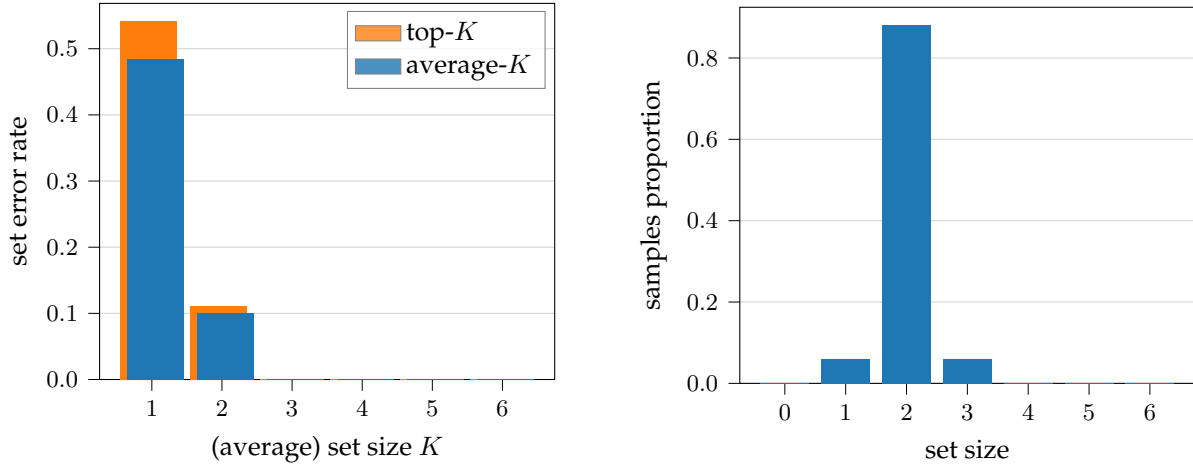


Figure 4.6: Example 3: slight modification of Example 2 where, instead of having three zones of equal weight ($w_1 = w_2 = w_3 = 1/3$), these weights are adapted in order to keep $w_3 = 1/3$ but to unbalance w_1 and w_2 such to have a small $w_1 = 3/100$. The left figure plots the error rate for different K values for top- K and average- K . The right figure shows the distribution of the predicted set size of average- K strategy for $K = 2$. In this case, average-2 predicts the same sets as top-2 for the vast majority of the samples. Average-2 can significantly improve upon top-2 in certain zones, but only occurs for relatively rare cases. See the text for more details.

weight are equal to $w_1 = w_2 = w_3 = 1/3$. Because the three zones are disjoint, the expectation of Definition 4.4.1 can be reduced to the following weighted sum:

$$d_{2,1} = \sum_{i=1}^3 \sum_{j=1}^3 w_i w_j \left(\tilde{\eta}_3^{(i)} - \tilde{\eta}_2^{(j)} \right)^+,$$

where $\tilde{\eta}_3^{(i)}$ (resp. $\tilde{\eta}_2^{(j)}$) is the constant value of $\tilde{\eta}_3(\mathbf{X})$ in zone i (resp. of $\tilde{\eta}_2(\mathbf{X})$ in zone j). Every term of this sum is null except when $i = 3$ and $j = 1$. It is thus equal to

$$d_{2,1} = \underbrace{w_1 w_3}_{\text{weight}} \times \underbrace{(\tilde{\eta}_3^{(3)} - \tilde{\eta}_2^{(1)})}_{\text{magnitude } \delta \geq 0}.$$

This decomposition highlights the general idea of $d_{K,1}$: it captures both how often does $\tilde{\eta}_{K+1}$ have a higher value than $\tilde{\eta}_K$ for different inputs (weight) and, when it occurs, how strong it is (magnitude). We will derive two variations of Example 2 which change one of these quantities at a time to show how it impacts the average-2 error rate. These examples preserve the same top-2 error rate. The computations of their different quantities are given in Table 4.1.

Ex. #	error rates		heterogeneity measure		
	top-2	avg-2	weight $w_1 w_3$	magnitude δ	measure $d_{2,1}$
Ex. 2	11%	0%	1/9	1/3	$1/27 \approx 0.037$
Ex. 3	11%	10%	ϵ	1/3	$\epsilon/3$
Ex. 4	11%	11%	1/9	ϵ	$\epsilon/9$

Table 4.1: Decomposition of the heterogeneity measure for the given examples. We can build examples with ϵ arbitrarily small for Example 3 and Example 4.

First, we derive Example 3 as a simple variant of Example 2 where the weights have been adapted. In particular, we keep $w_3 = 1/3$ to preserve the top-2 error rate. w_1 and w_2 are however unbalanced such to have a small $w_1 = 3/100$. As shown in Table 4.1, this decreases the

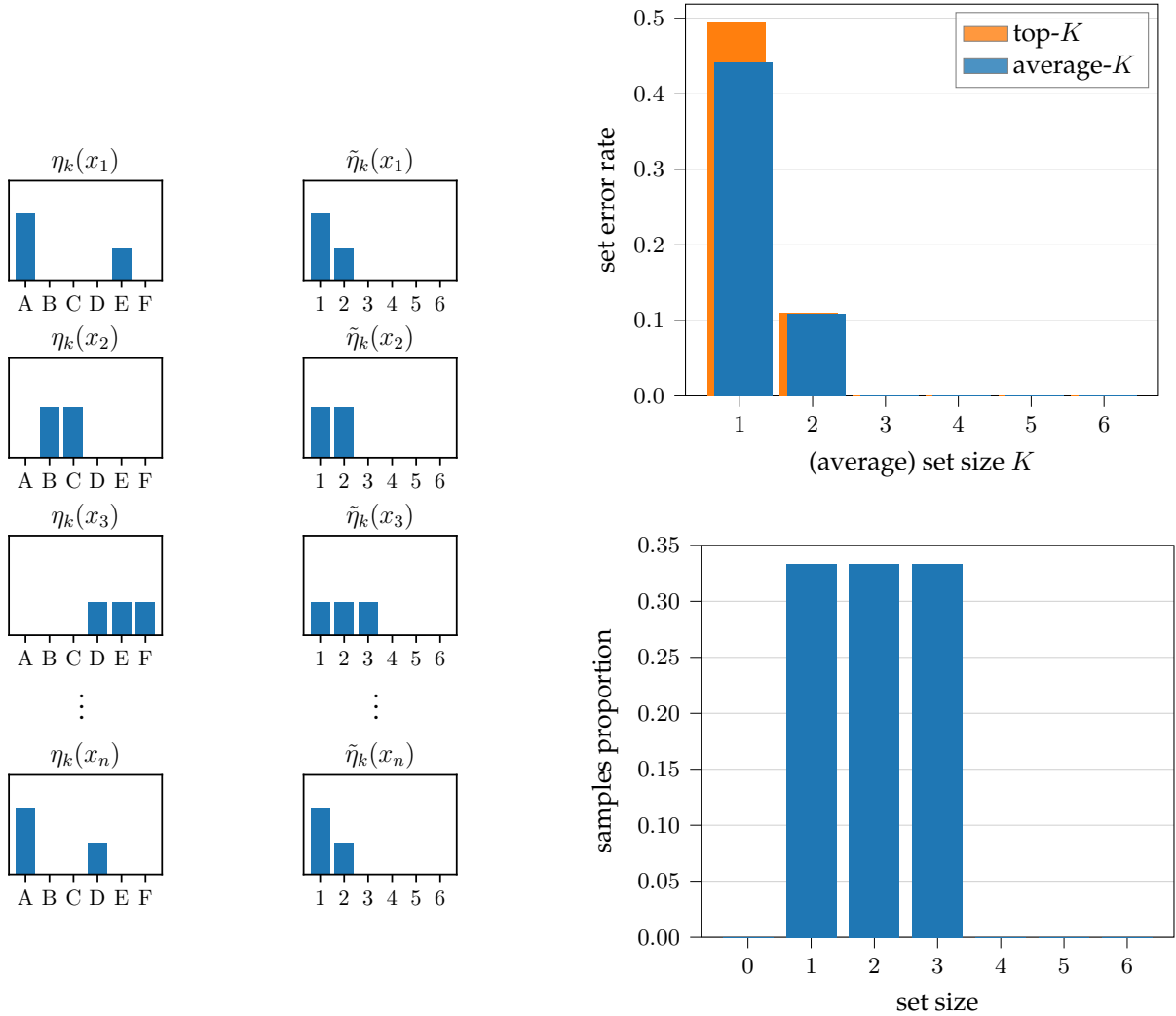


Figure 4.7: Illustration of Example 4 which is a modification of Example 2 where in the case where class A is dominant, instead of being unambiguous, it has a probability close to $2/3$ while one of the other class has a probability around $1/3$ as shown in the first column on the first and last row. The top-right figure plots the error rate for different K values for top- K and average- K . The bottom-right figure shows the distribution of the predicted set size of average- K strategy for $K = 2$. In this case, the sets predicted by average-2 are exactly the same as in Example 2, however, here, they only reduce marginally the error rate. See the text for more details.

heterogeneity of the data. In Figure 4.6, we can see that, in the end, average-2 predicts sets very close to top-2, so average-2 has a small adaptive gain.

The second variant, Example 4, is shown in Figure 4.7. It modifies zone 1 where class A is dominant as follows: class A has a probability close to $2/3$ and one other class has the rest of the probability, i.e. a little less than $1/3$. This does not change the top-2 error rate. The weights w_1, w_2, w_3 are kept equal to $1/3$. As shown in Table 4.1, this results in a much lower magnitude which reduces the heterogeneity of the problem. In this case, as can be seen in Figure 4.7, average-2 often predicts set sizes similar to those of Example 2. In the end, however, the adaptive gain is negligible.

These examples show that both weight and magnitude are important components of the heterogeneity which have a great impact on the gain of average- K over top- K . Proposition 4.4.1 only gives a lower bound, as can be noted in Table 4.1, values can be much smaller than final value of the adaptive gain (see Example 2). However, this result and its analysis provide insights into which problems would benefit most from average- K strategy.

4.5 Consistent estimation procedures

In this section, we will study how to estimate the top- K and average- K predictors developed in Section 4.1. The aim here is not to provide the best possible estimators but rather to give general principled estimation procedures which are consistent and that will allow us to carry out the experiments of Section 4.6. In particular, the estimators we derive here are based on the optimal rules defined in Equations (4.2) and (4.6). To apply these rules, we need an estimator $\hat{\eta}$ of the conditional probability η and directly plug it into the optimal rules. Such estimators are called *plug-in rules*.

In the rest of this section, we will be interested in the *regret* of risks which is a way to quantify how sub-optimal are our estimators. In particular, the top- K regret is defined as

$$\text{Reg}_{\text{top}}(\hat{S}_K) := \mathcal{E}(\hat{S}_K) - \mathcal{E}(S_K^*),$$

where $\hat{S}_K : \mathcal{X} \rightarrow \mathcal{P}(\mathcal{Y})$ is a top- K classifier, i.e. $\forall \mathbf{x}, |\hat{S}_K(\mathbf{x})| = K$. Similarly, the average- K regret is defined as

$$\text{Reg}_{\text{avg}}(\hat{\mathcal{S}}_K) := \mathcal{E}(\hat{\mathcal{S}}_K) - \mathcal{E}(\mathcal{S}_K^*),$$

where $\hat{\mathcal{S}}_K : \mathcal{X} \rightarrow \mathcal{P}(\mathcal{Y})$ is an average- K classifier, i.e. $\mathbb{E}_{\mathbf{X}} [|\hat{\mathcal{S}}_K(\mathbf{X})|] = K$. In our case, \hat{S}_K and $\hat{\mathcal{S}}_K$ are built from $\hat{\eta}$ and their exact definition are given in Section 4.5.1.

In general, it is hard to directly optimize these losses ($\mathcal{E}(\hat{S}_K)$ and $\mathcal{E}(\hat{\mathcal{S}}_K)$). It is thus common to use a *surrogate loss* l which is designed to have nicer optimization properties (convexity for instance). In this chapter, we will focus on *conditional probability estimation* (CPE) losses, that is to say losses of the form $l : \mathcal{Y} \times \Delta_C \rightarrow \mathbb{R}^+$ where Δ_C is the probability simplex of dimension C , i.e. $\Delta_C = \{p \in \mathbb{R}^C \mid \forall k, p_k \geq 0, \sum_k p_k = 1\}$. In this case, its regret is defined as

$$\text{Reg}_l(\hat{\eta}) := \mathbb{E}_{\mathbf{X}, Y} [l(Y, \hat{\eta}(\mathbf{X}))] - \inf_{\hat{\eta}'} \mathbb{E}_{\mathbf{X}, Y} [l(Y, \hat{\eta}'(\mathbf{X}))].$$

One important question when using surrogate losses is to understand how optimizing this loss will eventually minimize the losses that really interest us, i.e. top- K and average- K errors ($\mathcal{E}(\hat{S}_K)$ and $\mathcal{E}(\hat{\mathcal{S}}_K)$). In particular, there are two simple properties that one would want to be

satisfied in general by such losses: *calibration* and *consistency* (Bartlett et al., 2006; Tewari and Bartlett, 2007). A loss is said to be calibrated for a task if the following property is satisfied:

$$\text{Reg}_l(\hat{\eta}) = 0 \quad \Rightarrow \quad \text{Reg}_{\text{task}}(\hat{S}) = 0,$$

i.e. if the minimizers of the surrogate losses also minimize the original loss. It is said to be consistent if, furthermore, given a sequence of estimators $(\hat{\eta}_n)$ from which we build a sequence of set classifiers (\hat{S}_n) , the following property is satisfied:

$$\text{Reg}_l(\hat{\eta}_n) \rightarrow 0 \quad \Rightarrow \quad \text{Reg}_{\text{task}}(\hat{S}_n) \rightarrow 0,$$

i.e. if, given a sequence of estimators converging to the minimizer of the surrogate loss, that sequence of estimators will lead to the construction of optimal set classifiers.

These two properties are, in general, not equivalent. Indeed, although they are known to be equivalent in the binary classification case (Bartlett et al., 2006), it is not the case anymore in the multi-class case (Tewari and Bartlett, 2007). There is thus no reason to believe these properties are equivalent in the problems that interest us here. For top- K classification, Lapin et al. (2016, 2017) proposed some losses which are calibrated for top- K classification but they did not show that they were consistent, Yang and Koyejo (2020) showed in fact that some of their losses were not consistent. For average- K classification, Denis and Hebiri (2017) showed that losses of a certain form are indeed consistent but their result is not applicable to the negative log-likelihood loss, which is by far the most widely used loss in classification with neural networks.

The main aim of this section is to provide consistency results for a special class of losses (strongly proper losses) which contains several widely used losses in practice and in particular the negative log-likelihood. The section is organized as follows.

We first define the plug-in estimators for top- K and average- K estimation and show that they are consistent given consistent estimators of the conditional probability $\eta(x)$. Then, we study *proper losses* and a subclass of such losses called *strongly proper losses* which provides a way to build such estimators of $\eta(x)$, we also show that several widely used losses – such as negative log-likelihood – actually have such a property. In Section 4.6, we will discuss some good practices and show how these results apply in practice when training neural networks.

4.5.1 Plug-in estimators and their regret bounds

The plug-in rule for the top- K classifier is given by

$$\hat{S}_K(x) := \{\hat{\sigma}_x(k) \mid k \in \{1, \dots, K\}\} = \hat{\sigma}_x[\{1, \dots, K\}] \quad (4.11)$$

where $\hat{\sigma}_x$ is a permutation of $\{1, \dots, C\}$ such that $\hat{\eta}_{\hat{\sigma}_x(1)}(x) \geq \hat{\eta}_{\hat{\sigma}_x(2)}(x) \geq \dots \geq \hat{\eta}_{\hat{\sigma}_x(C)}(x)$. For the adaptive strategy, we need to apply a thresholding resulting in the following plug-in estimator:

$$\hat{\mathcal{S}}_K(x) := \{k \in \mathcal{Y} \mid \hat{\eta}_k(x) \geq G_{\hat{\eta}}^{-1}(K)\} \quad (4.12)$$

where $G_{\hat{\eta}}$ and its generalized inverse are defined similarly to G_{η} and G_{η}^{-1} (see Equations (4.4) and (4.5)):

$$G_{\hat{\eta}}(\lambda) := \sum_k \mathbb{P}_{\mathbf{X}} [\hat{\eta}_k(\mathbf{X}) \geq \lambda],$$

$$G_{\hat{\eta}}^{-1}(K) := \inf \{\lambda \in [0, 1] \mid G_{\hat{\eta}}(\lambda) \leq K\}.$$

Note that here the threshold $G_{\hat{\eta}}^{-1}(\mathcal{K})$ is not the same as for the optimal rule, i.e. in general $G_{\hat{\eta}}^{-1}(\mathcal{K}) \neq G_{\eta}^{-1}(\mathcal{K})$. Indeed, as we want to preserve the constraint on the average set size, i.e. that $\mathcal{I}(\hat{\mathcal{S}}_{\mathcal{K}}) = \mathcal{I}(\mathcal{S}_{\mathcal{K}}^*) = \mathcal{K}$, then we need to adapt the threshold on $\hat{\eta}$. For similar reasons than when building $\mathcal{S}_{\mathcal{K}}^*$ for which we made [Assumption C](#), we make the following assumption to guarantee that $\mathcal{I}(\hat{\mathcal{S}}_{\mathcal{K}}) = \mathcal{K}$.

Assumption D (Estimator continuity assumption). *For all $k \in \{1, \dots, C\}$, the cumulative distribution function (CDF) of estimated $\hat{\eta}_k$ denoted $F_{\hat{\eta}_k}$, defined as $F_{\hat{\eta}_k}(t) = \mathbb{P}_{\mathbf{X}} [\hat{\eta}_k(\mathbf{X}) \leq t]$, is continuous.*

Note that this assumption makes the choice of the estimation algorithm important. Indeed, it is satisfied by estimators provided by neural networks but not those provided by random forests for instance.

The following two theorems give upper bounds on the regret of top- K and average- K errors when using the plug-in estimators previously defined. These bounds are expressed in terms of the estimation error of $\hat{\eta}$ as measured by $\mathbb{E}_{\mathbf{X}} [\|\eta(\mathbf{X}) - \hat{\eta}(\mathbf{X})\|_1]$ and thus show that, the better the estimator $\hat{\eta}$ of η , the better the final top- K and average- K classifier.

Theorem 4.5.1. *Let $K \in \{1, \dots, C\}$ be a fixed set size. Given an estimator $\hat{\eta}$ of η , the associated top- K plug-in estimator \hat{S}_K as defined in [Equation \(4.11\)](#) has the following regret bound,*

$$\text{Reg}_{\text{top}}(\hat{S}_K) \leq \mathbb{E}_{\mathbf{X}} [\|\eta(\mathbf{X}) - \hat{\eta}(\mathbf{X})\|_1].$$

Theorem 4.5.2. *Let $\mathcal{K} \in [0, C]$ be a fixed average set size. Given an estimator $\hat{\eta}$ of η , under [Assumption D](#), the associated average- \mathcal{K} plug-in estimator $\hat{\mathcal{S}}_{\mathcal{K}}$ as defined in [Equation \(4.12\)](#) has the following regret bound,*

$$\text{Reg}_{\text{avg}}(\hat{\mathcal{S}}_{\mathcal{K}}) \leq \mathbb{E}_{\mathbf{X}} [\|\eta(\mathbf{X}) - \hat{\eta}(\mathbf{X})\|_1].$$

Proof of Theorem 4.5.1. For readability, as we will do the proof for a fixed \mathbf{x} , we omit \mathbf{x} in the formulas. Using the fact that $\sum_{k \in S_K^*} \hat{\eta}_k - \sum_{k \in \hat{S}_K} \hat{\eta}_k \leq 0$ as a direct consequence of the definition of \hat{S}_K , the point-wise regret can be upper bounded using

$$\begin{aligned} \text{Reg}_{\text{top}}(\hat{S}_K; \mathbf{x}) &= \sum_{k \in S_K^*} \eta_k - \sum_{k \in \hat{S}_K} \eta_k \\ &= \sum_{k \in S_K^*} (\eta_k - \hat{\eta}_k) + \sum_{k \in \hat{S}_K} (\hat{\eta}_k - \eta_k) + \sum_{k \in S_K^*} \hat{\eta}_k - \sum_{k \in \hat{S}_K} \hat{\eta}_k \\ &\leq \sum_{k \in S_K^*} (\eta_k - \hat{\eta}_k) + \sum_{k \in \hat{S}_K} (\hat{\eta}_k - \eta_k) \\ &= \sum_{k \in S_K^* \setminus \hat{S}_K} (\eta_k - \hat{\eta}_k) + \sum_{k \in \hat{S}_K \setminus S_K^*} (\hat{\eta}_k - \eta_k) \\ &\leq \sum_{k \in S_K^* \Delta \hat{S}_K} |\eta_k - \hat{\eta}_k| \\ &\leq \|\eta - \hat{\eta}\|_1. \end{aligned}$$

Taking the expectation over \mathbf{X} concludes the proof. \square

Proof of Theorem 4.5.2. For readability, we omit \mathbf{X} in the formulas. Using [Proposition 4.1.1](#), we have

$$\text{Reg}_{\text{avg}}(\hat{\mathcal{S}}_{\mathcal{K}}) = \mathbb{E}_{\mathbf{X}} \left[\sum_k |\eta_k - \lambda| \mathbb{1}_{k \in \mathcal{S}_{\mathcal{K}}^* \Delta \hat{\mathcal{S}}_{\mathcal{K}}} \right]$$

If $k \in \mathcal{S}_K^* \setminus \hat{\mathcal{S}}_K$, we have $\eta_k \geq \lambda$ and $\hat{\eta}_k < \hat{\lambda}$, thus,

$$|\eta_k - \lambda| = (\eta_k - \hat{\eta}_k) + (\hat{\eta}_k - \hat{\lambda}) + (\hat{\lambda} - \lambda) \leq |\eta_k - \hat{\eta}_k| + (\hat{\lambda} - \lambda).$$

Similarly, if $k \in \hat{\mathcal{S}}_K \setminus \mathcal{S}_K^*$, we have $\eta_k < \lambda$ and $\hat{\eta}_k \geq \hat{\lambda}$, thus,

$$|\eta_k - \lambda| = (\lambda - \hat{\lambda}) + (\hat{\lambda} - \hat{\eta}_k) + (\hat{\eta}_k - \eta_k) \leq |\eta_k - \hat{\eta}_k| + (\lambda - \hat{\lambda}).$$

Finally, this gives,

$$\text{Reg}_{\text{avg}}(\hat{\mathcal{S}}_K) \leq \mathbb{E}_X \left[\|\boldsymbol{\eta} - \hat{\boldsymbol{\eta}}\|_1 + (\hat{\lambda} - \lambda)(|\mathcal{S}_K^* \setminus \hat{\mathcal{S}}_K| - |\hat{\mathcal{S}}_K \setminus \mathcal{S}_K^*|) \right].$$

The second term is in fact equal to $(\hat{\lambda} - \lambda)(|\mathcal{S}_K^*| - |\hat{\mathcal{S}}_K|)$, taking the expectation over X gives $(\hat{\lambda} - \lambda)(\mathcal{I}(\mathcal{S}_K^*) - \mathcal{I}(\hat{\mathcal{S}}_K)) = 0$. Thus, taking the expectation over X concludes the proof. \square

Note that the previous theorems give upper bounds, they thus do not imply that having bad estimators of $\boldsymbol{\eta}$ will necessarily lead to bad estimators \hat{S}_K and $\hat{\mathcal{S}}_K$. However, these bounds are useful to derive the consistency results presented in the next section.

4.5.2 Proper losses and strongly proper losses

Now that we have shown plug-in regret bounds, this section studies losses that allow the construction of estimators of $\boldsymbol{\eta}$, namely proper losses. We will in particular focus on a subset of those losses that satisfy a stronger property: strongly proper losses. The main result of this section, [Theorem 4.5.4](#), shows that such losses are consistent for top- K and average- K classification. We then apply this result to show that negative log-likelihood, Brier score, and others are examples of such losses and are thus also consistent for these tasks.

Throughout this section, we will consider different quantities for a given $\boldsymbol{x} \in \mathcal{X}$, but for simplicity, the dependence on \boldsymbol{x} will be omitted in the equations.

Given a loss $l : \mathcal{Y} \times \Delta_C \rightarrow \mathbb{R}^+$, its conditional risk is defined as

$$L_l(\boldsymbol{\eta}, \hat{\boldsymbol{\eta}}) := \sum_k \eta_k l(k, \hat{\boldsymbol{\eta}}).$$

We denote L_l^* its infimum according to the second variable, i.e.

$$L_l^*(\boldsymbol{\eta}) := \inf_{\boldsymbol{\eta}'} L_l(\boldsymbol{\eta}, \boldsymbol{\eta}').$$

We first start by recalling the definition of (strictly) proper loss ([Gneiting and Raftery, 2007](#)) which were studied and proved to be useful in the context of binary classification ([Reid and Williamson, 2009](#)) and of multi-class classification ([Vernet et al., 2011](#)).

Definition 4.5.1 ((Strictly) proper loss). *A loss $l : \mathcal{Y} \times \Delta_C \rightarrow \mathbb{R}$ is said to be proper if its conditional risk's infimum is attained by $\boldsymbol{\eta}$, i.e.*

$$L_l(\boldsymbol{\eta}, \boldsymbol{\eta}) = L_l^*(\boldsymbol{\eta}).$$

If this infimum is uniquely attained by $\boldsymbol{\eta}$, then the loss is said to be strictly proper.

The following definition is a generalization to the multi-class setting of the strongly proper losses which were introduced by [Agarwal \(2014\)](#) for the binary case in the context of bipartite ranking³.

³To the best of our knowledge, this property of strongly proper losses in the case of multi-class classification was not published before.

Definition 4.5.2 (Strongly proper loss). A loss $l : \mathcal{Y} \times \Delta_C \rightarrow \mathbb{R}$ is said to be μ -strongly proper if it satisfies

$$L_l(\boldsymbol{\eta}, \hat{\boldsymbol{\eta}}) - L_l(\boldsymbol{\eta}, \boldsymbol{\eta}) \geq \frac{\mu}{2} \|\boldsymbol{\eta} - \hat{\boldsymbol{\eta}}\|_1^2,$$

with $\mu > 0$.

It is easy to check that a strongly proper loss is necessarily strictly proper. Indeed, if $\hat{\boldsymbol{\eta}} \neq \boldsymbol{\eta}$, $L_l(\boldsymbol{\eta}, \hat{\boldsymbol{\eta}}) > L_l(\boldsymbol{\eta}, \boldsymbol{\eta})$, and thus $L_l(\boldsymbol{\eta}, \boldsymbol{\eta}) = L_l^*(\boldsymbol{\eta})$. Therefore, we can rewrite the previous definition as a lower bound on the point-wise regret of l :

$$\text{Reg}_l(\hat{\boldsymbol{\eta}}; \mathbf{x}) \geq \frac{\mu}{2} \|\boldsymbol{\eta}(\mathbf{x}) - \hat{\boldsymbol{\eta}}(\mathbf{x})\|_1^2.$$

The converse is not true in general: a strictly proper loss might not be strongly proper.

Note that, when applying this definition in the binary case, $\|\boldsymbol{\eta} - \hat{\boldsymbol{\eta}}\|_1$ becomes $2|\eta - \hat{\eta}|$ resulting in a multiplicative factor 4 for the parameter μ compared to the definition of [Agarwal \(2014\)](#). Note also that, because of norm equivalence in the finite dimension, the actual norm used in the definition is not a restriction: one can easily adapt the norm by changing the parameter μ accordingly. We will illustrate this with examples of such losses later in this section. The choice of the ℓ_1 -norm is justified by the following result.

The following proposition shows how the estimation error of $\hat{\boldsymbol{\eta}}$ is controlled by the regret of a strongly proper loss.

Proposition 4.5.3. Let l be a μ -strongly proper loss. Then, we have

$$\mathbb{E}_{\mathbf{X}} [\|\boldsymbol{\eta}(\mathbf{X}) - \hat{\boldsymbol{\eta}}(\mathbf{X})\|_1] \leq \sqrt{\frac{2}{\mu} \text{Reg}_l(\hat{\boldsymbol{\eta}})}$$

Proof. Using the convexity of the function $x \mapsto x^2$ and the definition of strongly proper losses, we have

$$\begin{aligned} \mathbb{E}_{\mathbf{X}} [\|\boldsymbol{\eta}(\mathbf{X}) - \hat{\boldsymbol{\eta}}(\mathbf{X})\|_1] &\leq \sqrt{\mathbb{E}_{\mathbf{X}} [\|\boldsymbol{\eta}(\mathbf{X}) - \hat{\boldsymbol{\eta}}(\mathbf{X})\|_1^2]} \\ &\leq \sqrt{\frac{2}{\mu} \mathbb{E}_{\mathbf{X}} [L_l(\boldsymbol{\eta}(\mathbf{X}), \hat{\boldsymbol{\eta}}(\mathbf{X})) - L_l^*(\boldsymbol{\eta}(\mathbf{X}))]} \\ &= \sqrt{\frac{2}{\mu} \text{Reg}_l(\hat{\boldsymbol{\eta}})}. \end{aligned}$$

□

This proposition allows us to prove the consistency of strongly proper losses with respect to the set-valued classification problems we consider in this chapter.

Theorem 4.5.4. Any strongly proper loss is consistent for top- K and average- K classification.

Proof. This is a direct consequence of the bounds of [Theorems 4.5.1](#) and [4.5.2](#) and of [Proposition 4.5.3](#). Considering a sequence of estimators $(\hat{\boldsymbol{\eta}}_n)$ such that $\text{Reg}_l(\hat{\boldsymbol{\eta}}_n)$ tends towards zero will force the regret $\text{Reg}_{\text{task}}(\hat{\mathcal{S}}_n)$ to also tend to zero. □

We now give some examples of such losses showing that several well-known losses are actually strongly proper.

Example 4.5.1 (Negative log-likelihood). *The negative log-likelihood loss (NLL) defined as*

$$l_{\log}(k, \hat{\eta}) = -\log \hat{\eta}_k$$

is 1-strongly proper.⁴ It is thus consistent for the set-valued classification problems we consider in this chapter.

This is a direct consequence of Pinsker's inequality (Boucheron et al., 2013, Theorem 4.19) which states that

$$D_{\text{KL}}(\boldsymbol{\eta}, \hat{\boldsymbol{\eta}}) \geq 2\|\boldsymbol{\eta} - \hat{\boldsymbol{\eta}}\|_{TV}^2 = \frac{1}{2}\|\boldsymbol{\eta} - \hat{\boldsymbol{\eta}}\|_1^2$$

where D_{KL} is the Kullback-Leibler divergence and $\|\cdot\|_{TV}$ is the total variation distance. Simply noting that the point-wise regret is equal to

$$\text{Reg}_{\log}(\hat{\boldsymbol{\eta}}; \mathbf{x}) = D_{\text{KL}}(\boldsymbol{\eta}(\mathbf{x}), \hat{\boldsymbol{\eta}}(\mathbf{x})),$$

concludes the proof by Definition 4.5.2.

Example 4.5.2 (Brier score). *Let the square loss, a.k.a. Brier score, defined as*

$$l_{\text{sq}}(k, \hat{\eta}) = \frac{1}{2} \sum_{k'} (\mathbb{1}_{k'=k} - \hat{\eta}_{k'})^2.$$

For this loss, the point-wise regret is equal to

$$\text{Reg}_{\text{sq}}(\hat{\boldsymbol{\eta}}; \mathbf{x}) = \frac{1}{2}\|\boldsymbol{\eta}(\mathbf{x}) - \hat{\boldsymbol{\eta}}(\mathbf{x})\|_2^2 \geq \frac{1}{2C}\|\boldsymbol{\eta}(\mathbf{x}) - \hat{\boldsymbol{\eta}}(\mathbf{x})\|_1^2.$$

It is thus $\frac{1}{C}$ -strongly proper and consistent for the set-valued classification problems we consider in this chapter.

Example 4.5.3 (One-versus-all strongly proper losses). *Let $l_b : \{-1, +1\} \times [0, 1] \rightarrow \mathbb{R}$ be a binary μ -strongly proper loss (as defined by Agarwal (2014)) and define the following one-versus-all loss*

$$l_{\text{ova}}(k, \hat{\eta}) = l_b(+1, \hat{\eta}_k) + \sum_{k' \neq k} l_b(-1, \hat{\eta}_{k'}).$$

We have

$$\begin{aligned} L_{\text{ova}}(\boldsymbol{\eta}, \hat{\boldsymbol{\eta}}) &= \sum_k \eta_k l_b(+1, \hat{\eta}_k) + \sum_{k, k' \neq k} \eta_k l_b(-1, \hat{\eta}_{k'}) \\ &= \sum_k \eta_k l_b(+1, \hat{\eta}_k) + \sum_{k, k' \neq k} \eta_{k'} l_b(-1, \hat{\eta}_k) \\ &= \sum_k \eta_k l_b(+1, \hat{\eta}_k) + \sum_k (1 - \eta_k) l_b(-1, \hat{\eta}_k) \\ &= \sum_k L_b(\eta_k, \hat{\eta}_k). \end{aligned}$$

It is easy to check that

$$\begin{aligned} \text{Reg}_{\text{ova}}(\hat{\boldsymbol{\eta}}; \mathbf{x}) &= \sum_k \text{Reg}_b(\hat{\eta}_k; \mathbf{x}) \\ &\geq \sum_k \frac{\mu}{2} (\eta_k(\mathbf{x}) - \hat{\eta}_k(\mathbf{x}))^2 \\ &= \frac{\mu}{2} \|\boldsymbol{\eta}(\mathbf{x}) - \hat{\boldsymbol{\eta}}(\mathbf{x})\|_2^2 \\ &\geq \frac{\mu}{2C} \|\boldsymbol{\eta}(\mathbf{x}) - \hat{\boldsymbol{\eta}}(\mathbf{x})\|_1^2. \end{aligned}$$

Thus, losses built from univariate μ -strongly proper losses are $\frac{\mu}{C}$ -strongly proper and thus consistent.

⁴See the previous comments for why, in the definition of Agarwal (2014) in the binary case, it is 4-strongly proper.

Now that we have shown the consistency of the proposed estimation procedures, we can experiment them on real-world datasets.

4.6 Experiments on real datasets

In this section, we carry out experiments on image classification datasets to see how well they accord with the previous theoretical results. First, we study the proposed estimation procedures empirically to see how they behave in practice. Then, in order to illustrate the potential of average- K , we carry out controlled experiments showing how different forms of noise impact the relative performance of top- K classification and its adaptive counterpart. We then compare top- K and average- K on ImageNet and on fine-grained classification datasets to highlight how real-world datasets can benefit from average- K classification. We chose to restrict ourselves to image datasets for these experiments because their ambiguity is visually evident. However, top- K and average- K can be applied to any type of data.

4.6.1 Practical estimation of top- K and average- K classifiers

From a theoretical standpoint, when the conditional probabilities of the classes given the input are exactly known, optimal average- K classifiers are always at least as good as optimal top- K ones. In this subsection, we study what happens in practice when we have only estimators of such quantities resulting in classifiers which are thus sub-optimal.

As we are interested in the estimation procedures, independently of a specific value of K , we will regularly measure the *mean top- K error* and the *mean average- K error*, where the mean is taken over all integer values of K :

$$\begin{aligned}\text{mean top-}K \text{ error} &= \frac{1}{C} \sum_{K=1}^C \mathcal{E}(\hat{S}_K), \\ \text{mean average-}K \text{ error} &= \frac{1}{C} \sum_{K=1}^C \mathcal{E}(\hat{\mathcal{J}}_K),\end{aligned}$$

where \hat{S}_K and $\hat{\mathcal{J}}_K$ are the estimators defined in [Equations \(4.11\) and \(4.12\)](#).

Throughout this section, we will perform experiments on the CIFAR-10 dataset ([Krizhevsky and Hinton, 2009](#)), which is a standard image classification dataset containing 60,000 color images of size 32x32 illustrating 10 classes. Although the classes are sufficiently distinct to render the original task quite simple, there is appreciable ambiguity, as shown in [Figure 4.1](#), because the image resolution is low. It is thus a good dataset to perform these experiments.

Estimation using neural networks

Using neural networks has become the standard approach in image classification. However, while efficient in practice, the training of such models is not fully understood from a theoretical point-of-view. In order to see how the proposed estimation procedures for top- K and average- K behave with such models, we perform standard training on CIFAR-10 to see what happens.

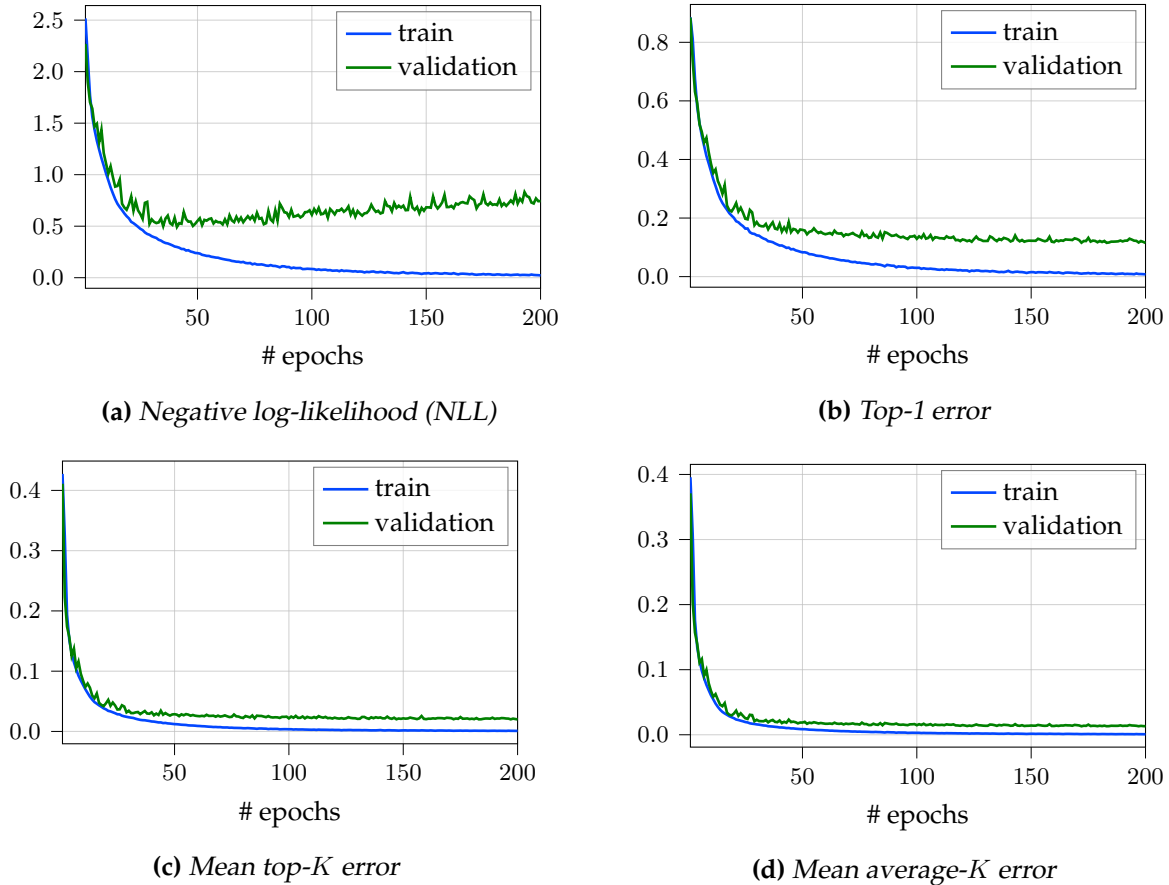


Figure 4.8: Evolution of the different metrics through the training process of a neural network on CIFAR-10. During training, we are optimizing the negative log-likelihood (NLL) on the training set.

First, we train a ResNet-44 (He et al., 2016) using the standard training procedure detailed in the referenced paper. In Figure 4.8, the learning curves for different metrics are shown. It is known that, although the negative log-likelihood (NLL) measured on the validation can increase through training as shown in Figure 4.8a, the top-1 error continues to decrease as shown in Figure 4.8b. Interestingly, and perhaps more surprisingly, the same comment can be made for the estimators of top- K and average- K as can be seen respectively in Figure 4.8c and Figure 4.8d.

This shows that, in spite of the analysis in Section 4.5, carefully monitoring NLL on the validation set is in fact not necessary to obtain good estimators of top- K and average- K classifiers. In the end, using training strategies similar to those used to minimize errors for top-1 classification is sufficient. This means that virtually nothing needs to be changed in the training. Even a pre-trained model can directly be used for top- K and average- K prediction.

Moreover, this empirical finding suggests that the usual training procedures for neural networks, although not necessarily learning a probability-calibrated model, still learn useful scores for average- K . In turn, this suggests that the relative ordering of the conditional probability for each class and each sample is often preserved in the learned scores.

In order to further improve the learned classifiers, we use ensembles of neural networks trained using different initialization and mini-batch sampling. The impact of training with several models in the ensemble is shown in Figure 4.9. Both top- K and average- K benefit significantly from using ensembles. Surprisingly, average- K seems to require fewer models than top- K to stabilize.

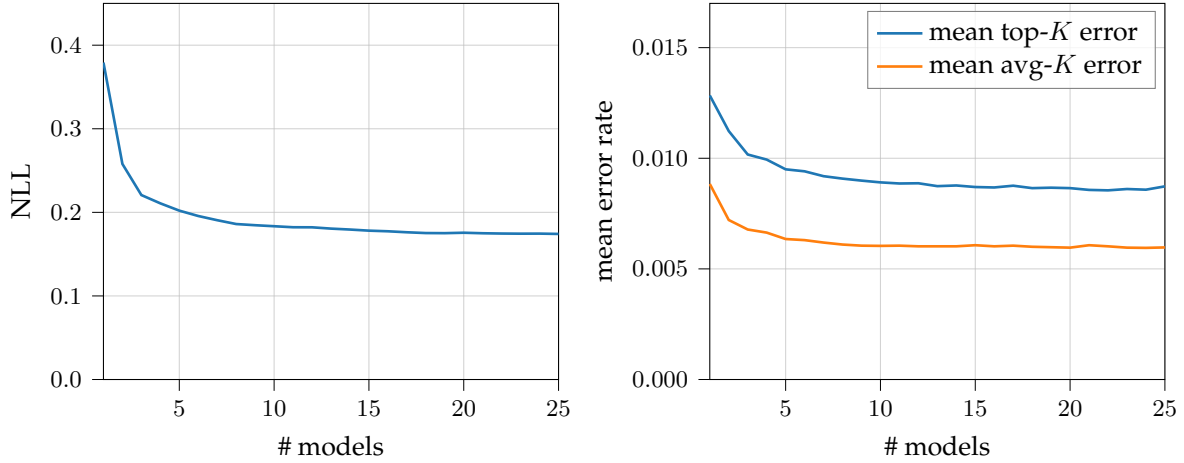


Figure 4.9: Impact of the number of models present in the ensemble (CIFAR-10). More models reduce the error.

Impact of *a posteriori* probability calibration

In this section, we study the impact of probability calibration methods used once the training is completed. These methods aim to improve the estimated probabilities outputted by an existing model (Flach, 2017). They thus seem relevant for top- K and average- K classification.

We restrict our analysis to extensions to the multi-class case of Platt’s scaling (Platt et al., 1999) proposed in Guo et al. (2017) as these methods were shown to outperform other approaches when used with neural networks. We briefly recall them here. They consist in taking a model already trained and learning a logistic regression on the logits. That is to say, if we denote by $\mathbf{z}(\mathbf{x})$ the scores in the logits space predicted by a model for a given \mathbf{x} , the original estimator of the conditional probability $\hat{\eta}_k$ is computed using

$$\hat{\eta}_k(\mathbf{x}) = \text{softmax}(\mathbf{z}(\mathbf{x}))_k,$$

where $\text{softmax}(\mathbf{z})_k = \frac{e^{z_k}}{\sum_{k'} e^{z_{k'}}$. This estimator is replaced by $\hat{\eta}'_k$ computed as

$$\hat{\eta}'_k(\mathbf{x}) = \text{softmax}(W\mathbf{z}(\mathbf{x}) + b)_k,$$

where (W, b) is a new linear model fitted by minimizing the negative log-likelihood (NLL) on a calibration set. We call this calibration *full matrix scaling*. If W is a diagonal matrix, it is called *vector scaling*. Finally, it is called *temperature scaling* if the calibration simply consists in learning a single temperature parameter T and the resulting estimator is computed as

$$\hat{\eta}'_k(\mathbf{x}) = \text{softmax}(\mathbf{z}(\mathbf{x})/T)_k.$$

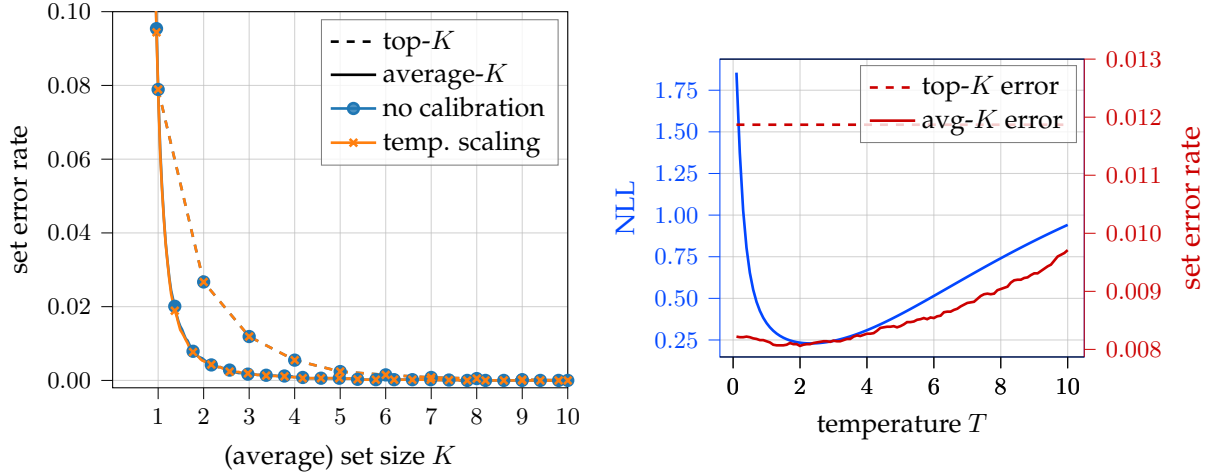
Although extremely simple, this last method was shown in Guo et al. (2017) to be very efficient.

We compare the different calibration methods on the CIFAR-10 dataset by calibrating on the validation set a single previously trained model. The results on the test set is shown in Table 4.2 and Figure 4.10a. These values can be compared with those of ensembling shown in Figure 4.9. Although these calibration methods do reduce the negative log likelihood, which is what they were designed for, their impact on the top- K and average- K error rates is very small.

In Figure 4.10b, we study the influence of temperature scaling on the model calibration in detail. NLL is reduced by using a temperature of $T \approx 2.3$, which is bigger than 1, the default

Calibration method	NLL	mean top- K error	mean avg- K error
No calibration	0.3792	0.0128	0.0088
Temperature scaling	0.2444	0.0128	0.0087
Vector scaling	0.2442	0.0129	0.0088
Full matrix	0.2394	0.0126	0.0086

Table 4.2: The impact of the probability calibration methods on top- K and average- K metrics computed on CIFAR-10 is small.



(a) Error rate vs. K (set size) curve.

(b) Effect of temperature scaling on NLL and top- K and average- K error rates.

Figure 4.10: Impact of probability calibration methods on top- K and average- K classifiers computed on CIFAR-10.

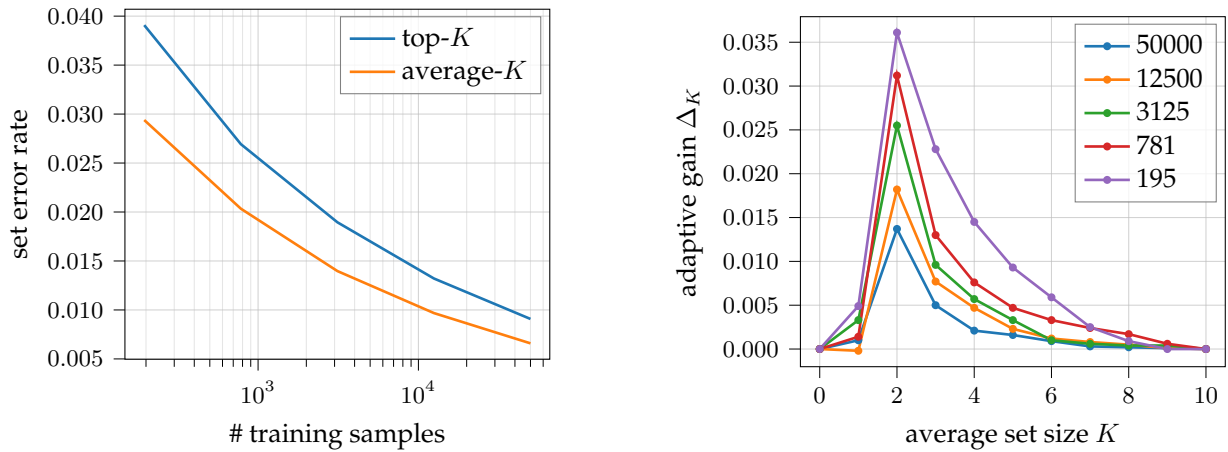
value without calibration. The model was thus overconfident before the calibration: the scaling reduces the values of probabilities outputted by the model. Temperature scaling has no influence on the top- K error rate as it preserves the order of the class for each sample taken individually. By contrast, we can see that the temperature has an effect on the average- K error but that it is still much lower than the top- K error as long as T is not too high. Moreover, the standard temperature $T = 1$ is very close to optimal.

These results suggest that, as noted in the previous section, the scores learned by a standard training of neural networks work well for top- K and average- K . Thus, additional probability calibration is not necessary. We will therefore not use these methods in the rest of the experiments.

When training data is scarce, average- K is still better

When the volume of training data is limited, one could expect top- K classifiers to be easier to estimate than average- K classifiers. In problems for which optimal average- K classifier would provide only a minor advantage over optimal top- K classifier, we would expect the estimated top- K to have a lower error rate than the estimated average- K . During our experiments, surprisingly, this expected phenomenon never occurred. The estimated average- K classifier was still better than the top- K one.

To highlight this, we use the Fashion-MNIST dataset (Xiao et al., 2017), which has a low



(a) Mean error rate vs. number of training samples.

(b) Adaptive gain evolution when decreasing the number of training samples.

Figure 4.11: Impact of training set size on top- K and average- K error rates measured on Fashion-MNIST's test set.

degree of ambiguity⁵. This dataset consists of 60,000 training images and 10,000 test images of clothes containing 10 different classes. We split the original training set into two parts: 50,000 for training purposes and 10,000 for validation of hyperparameters. We then subsample this training set by dividing it by four for every experiment resulting in set sizes of 50,000, 12,500, 3,125, 781, and 195. The results computed on the final test set are shown in Figure 4.11. Not only is average- K always better than top- K but the gap actually widens when given fewer samples. Even when we have a limited training set, average- K still improves upon top- K , so should be used.

4.6.2 Experiments on altered real data: potential of average- K classification

In this subsection, we consider experiments on image datasets in which we inject different types of noise to create ambiguity. In particular, we focus on (i) adding noise in the labels, and (ii) downsampling the images.

Label noise injection

We first consider modifying the labels in order to inject noise. We introduce a method to inject the noise in a principled way which will allow us to control its characteristics and to compute the quantities highlighted in previous sections.

To do so, we will start from a (nearly) un-ambiguous dataset, namely, MNIST (LeCun et al., 1998) for which the top-1 error rate is very low (less than 1%) even when using simple models. Note moreover that this dataset is balanced: every class has the same number of samples. We then group together the classes such that, if a sample's label is in a certain group, we regenerate a new label by randomly sampling a label of that group. This is equivalent to introducing a confusion matrix from which we regenerate the labels using $\mathbb{P}[Y_{\text{new}} = k \mid Y_{\text{old}} = k']$ with the values of the new confusion matrix.

⁵MNIST has even less ambiguity but we did not use it for the experiments here because the task is too simple for neural networks which achieved very low error rate even with very few training samples.

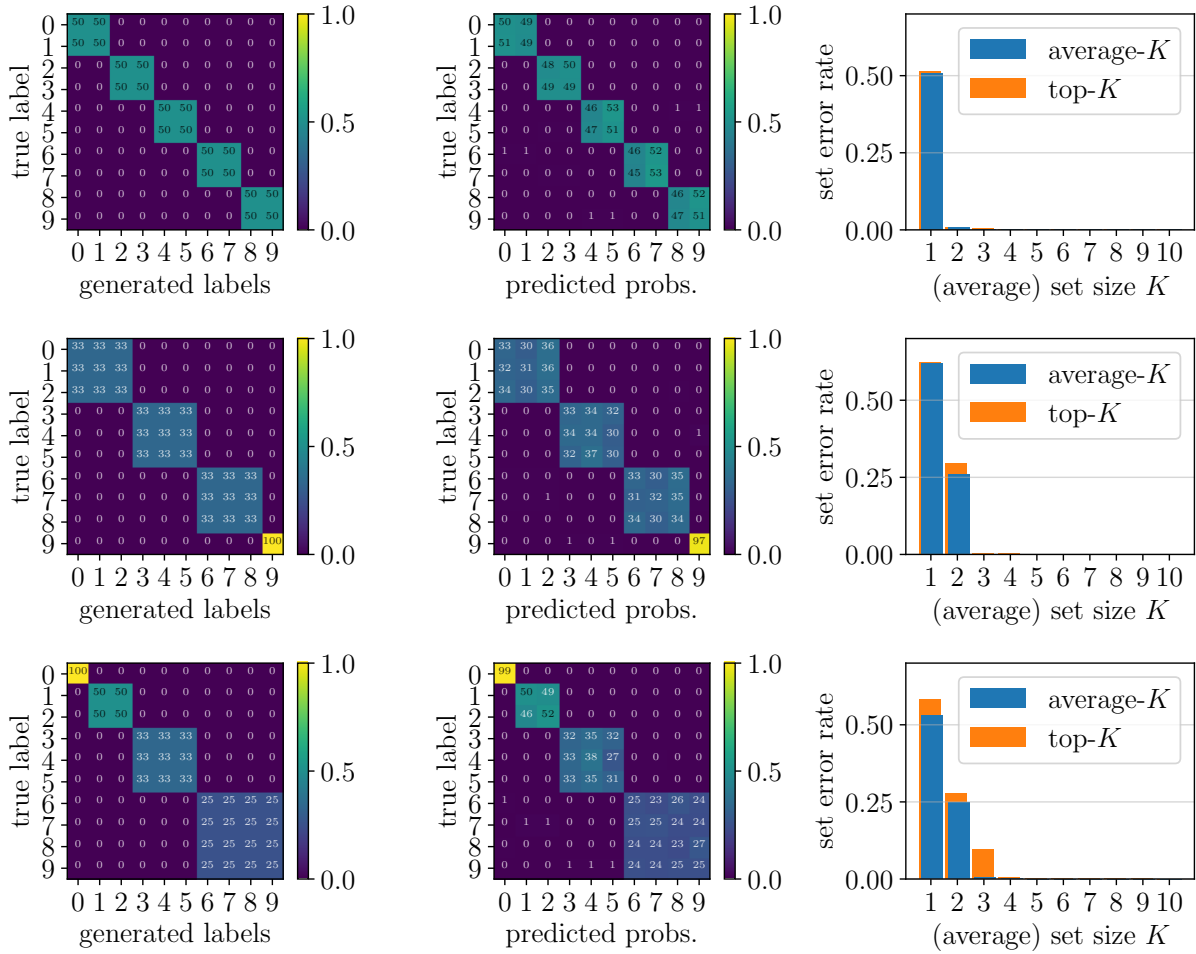


Figure 4.12: Influence of ambiguity injection via label noise. The first column shows the confusion matrix used to generate new labels: in the first two lines, neighbor classes are fused in groups of same size, respectively, of size 2 and 3, while in the last line, the groups have different sizes. The first two lines show homogeneous noise label injection while the last one shows heterogeneous label noise injection. The second column shows the predicted mean probabilities for samples of each class after the models are learned: the ambiguity is pretty well understood and estimated by the models. The last column shows the relative gain of average- K over top- K in each case. When the ambiguity is homogeneous, average- K has no advantage over top- K . But when it is heterogeneous, the gain can be substantial.

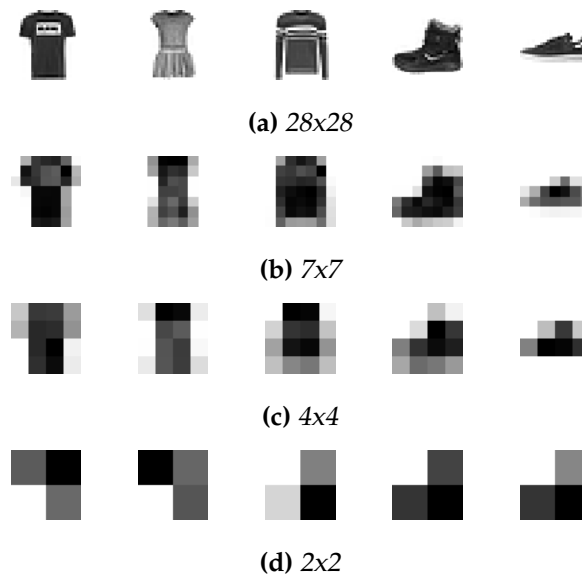


Figure 4.13: Fashion-MNIST with various amounts of downsampling. Each column corresponds to a single image taken from five different classes. From left to right: T-shirt, dress, pull-over, ankle boot and sneaker.

The results appear in Figure 4.12. In the first two lines, the classes are gathered into balanced groups of sizes 2 or 3. These cases are similar to the experiments carried out in Berrada et al. (2018). As one can see, the ambiguity introduced here is very homogeneous and the average- K classifier yields only a marginal gain over the top- K classifier.

On the other hand, in the last line, we study injecting heterogeneous label noise by gathering classes in different groups of different sizes. As one can see, in this case, we can have a noticeable gap between top- K and average- K classifiers. In particular, for $K = 3$, the error rate of the average-3 strategy is zero whereas the top-3 classifier has an error rate around 10%.

These experiments show that, in general, when the confusion among classes is homogeneous, as measured by confusion matrices, top- K is optimal or very close to average- K . When the confusion is heterogeneous, average- K yields a much lower error rate than top- K . In any case, average- K is always better than top- K .

Input image degradation analysis

We now consider experiments where noise is added to the input images. In this case, it is less clear what type of or how much ambiguity we are introducing. The aim of this section is to measure what happens in this case.

Adding noise on MNIST did not introduce enough ambiguity, the task being too easy. These experiments are thus carried out on Fashion-MNIST (Xiao et al., 2017). Here, we use an ensemble of simple convolutional neural networks whose error rate on the original dataset is around 6%. The original ambiguity is thus higher than on MNIST but still not very high. In order to inject ambiguity, we downsampled by the same amount all the images from all classes. Some resulting sample images are shown in Figure 4.13.

As expected, the greater the noise, the more ambiguous are the images as shown in the global increase of error rates displayed in Figure 4.14a. One might expect that the introduced ambiguity

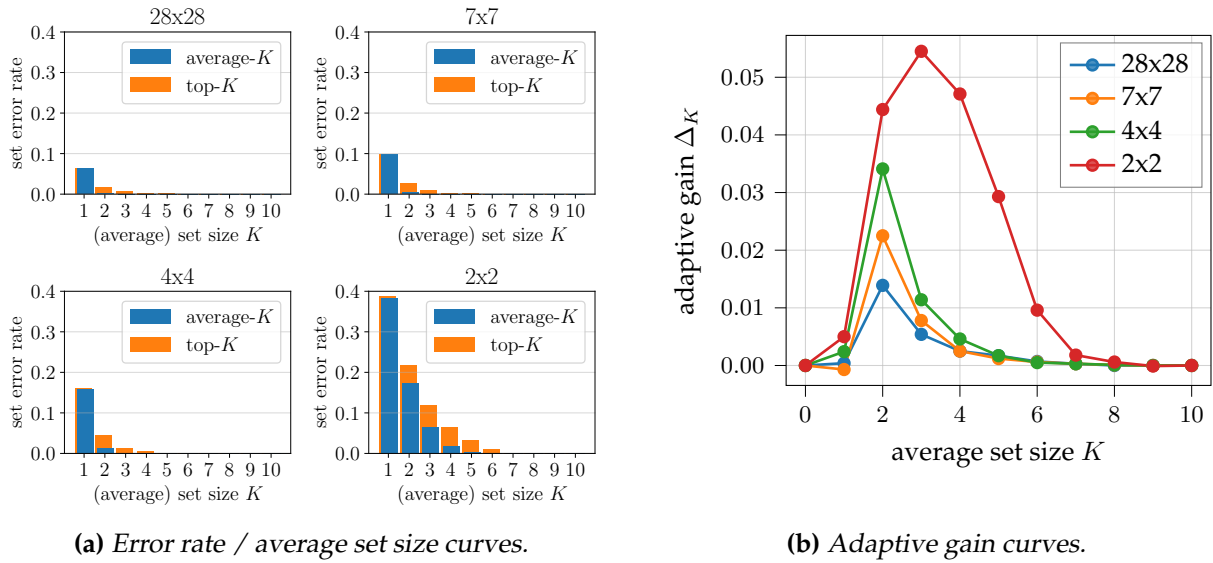


Figure 4.14: Impact of image downsampling in Fashion-MNIST on top- K and average- K classifiers.

is homogeneous because the noise is applied uniformly on all the images. However, Figure 4.14b shows that the gap between top- K and average- K strategies actually widens. This implies that the ambiguity is in fact heterogeneous. As can be seen for the samples from Figure 4.13, some classes are actually still recognizable even though the noise level is high, e.g. T-shirts (first column) and sneakers (last column), still are not confused. On the other hand, some classes become harder to distinguish, such as different classes of shoes, e.g. boots and sneakers in the two last columns.

Thus, in general, adding noise to the input will render the task more heterogeneously ambiguous. This experiment highlights the benefits of average- K in the presence of input noise and with increased ambiguity. Figure 4.14a shows how much average- K can reduce the error rate.

4.6.3 Experiments on unaltered real-world datasets

We now perform experiments on real datasets in order to see if the previous comments on the usefulness of average- K apply to practical tasks.

ImageNet

ImageNet (Russakovsky et al., 2015) is a natural choice of dataset for our demonstration. Indeed, as explained in the previously cited article, ImageNet is known to have several objects per image as it is used for different learning tasks including classification, object detection and localization. The class retained for an image is not necessarily the biggest object in the scene nor the most numerous. It only has to appear in the scene. We refer the reader to Russakovsky et al. (2015) for more details on the way the dataset was collected and annotated. For these reasons, in the classification task, the main metric considered is top-5 error rate. However, it is known that images do not contain exactly the same number of objects: ImageNet is thus a natural candidate

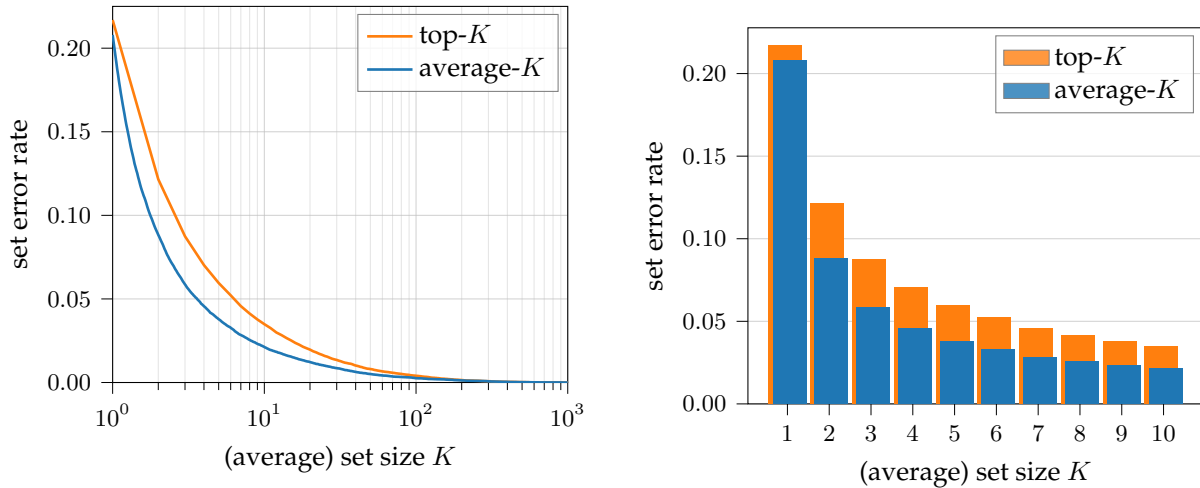


Figure 4.15: ImageNet top- K and average- K error rates for all values of K computed on the official validation set. ImageNet’s main metric is the top-5 error rate. For $K = 5$, average-5 reduces the error rate by ($\approx 37\%$) compared with top-5, as indicated by the additional error shown in orange.

for average- K .⁶

The results on the official validation set (the test set is private) are shown in Figure 4.15. Average- K greatly reduces the error rate compared to top- K . In particular, for $K = 5$, the official performance metric, the relative reduction is 37%. Average- K is thus particularly well suited for this scenario. To further analyze the difference between the two predictors on this dataset, the distribution of the sizes of sets predicted by average- K are given in Figure 4.16. Interestingly, the distribution does not have a mode around 5 but is strictly decreasing. Most of the distribution is concentrated at values below 5: 75% of the sets have a size lower than 5. Note that the distribution has a rather long tail: 5% of the sets have a size larger than 20 and the maximum set size is 78.

Fine-grained visual classification datasets

In this section, we study the performance of average- K strategy on fine-grained visual classification (FGVC) tasks. While top- K is a classic metric in this context, such tasks are likely to benefit more from average- K . Unlike ImageNet where the ambiguity arises from the fact that there are several objects in the image, in FGVC datasets, usually, each image has only a single object but the classes can be very close visually, so it can be hard to distinguish them from the given image. We carry out experiments on three large-scale life science datasets:

- PlantCLEF2015 (Göeau et al., 2015): plant species recognition,
- FGVC5-Fungi⁷: mushrooms species recognition, and
- FGVC6-ButterfliesMoths⁸: butterfly and moth species recognition.

Table 4.3 shows the characteristics of these datasets. In particular, they involve being able to discriminate a lot of different species (from 1K to over 5K). Note that the official metric of both

⁶Note that the choice of $K = 5$ is not discussed in the previous paper, in particular, the average number of objects per image is not given and is likely to be different from 5.

⁷<https://sites.google.com/view/fgvc5/competitions/fgvcx/fungi>

⁸<https://sites.google.com/view/fgvc6/competitions/butterflies-moths-2019>

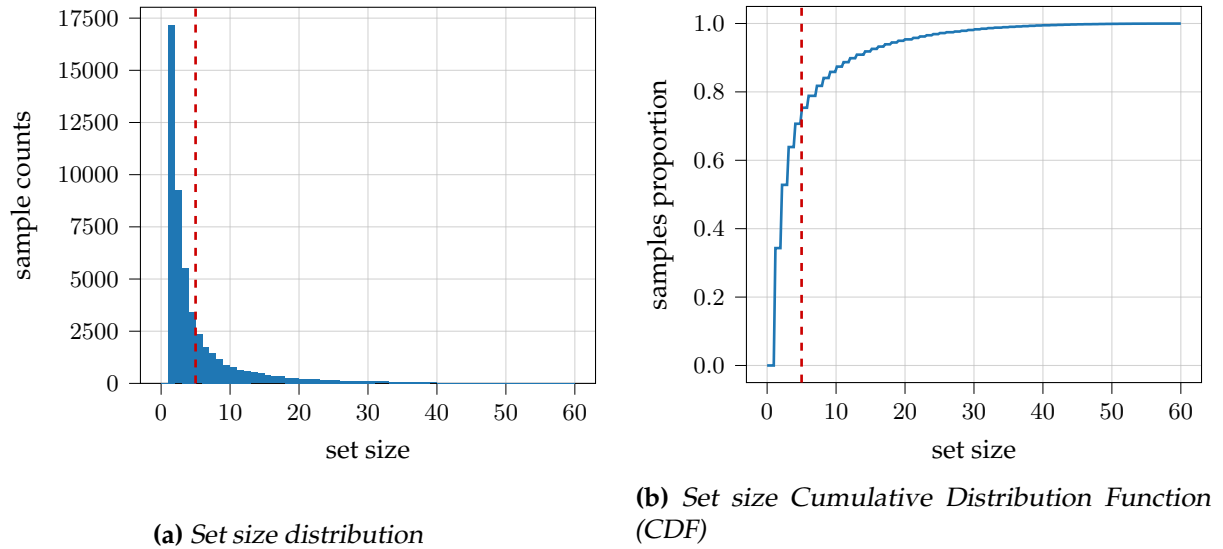


Figure 4.16: ImageNet set size distribution for the estimated adaptive top-5 classifier (on the validation set). As shown, the distribution of set sizes for average-5 does not peak at 5 (red dashed vertical line) but rather exploits the wide range of different set sizes. Most images guess only a small set of classes but some samples need a large set size.

FGVC5-Fungi and FGVC6-ButterfliesMoths is the top-3 error rate.

Name	# samples	# classes
PlantCLEF2015	113,205	1000
FGVC5-Fungi	85,578	1,394
FGVC6-ButterfliesMoths	473,438	5,419

Table 4.3: Fine-grained visual classification (FGVC) dataset characteristics.

The results are displayed in Figure 4.17. The relative improvement of average- K depends on the dataset. In particular, the gain is high for FGVC6-ButterfliesMoths, yielding around a 41% error rate reduction for $K = 3$, while being lower but still noticeable for PlantCLEF2015 and FGVC5-Fungi, both around 16% for $K = 3$. Although the improvement is dependent on the dataset, in all cases, using average- K reduces significantly the error rate compared to top- K .

4.7 Conclusion

In this chapter, we have studied how average- K , an adaptive version of top- K , compares to this latter. We have first studied theoretically what are the properties of the intrinsic ambiguity of the classification task which renders average- K most useful compared to top- K in the infinite sample regime. In particular, we showed what role played the heterogeneity of this ambiguity. We then provided consistent procedures to estimate top- K and average- K classifiers. Finally, we have carried out experiments on real-world datasets to highlight the usefulness of average- K in practice, even when the training data is scarce.

We have focused our theoretical analysis on the infinite sample regime in order to study how average- K classifiers handle task ambiguity. The natural next step would be to extend this analysis to the finite sample regime to understand how such set-valued classifiers behave in presence of model uncertainty. Indeed, even in absence of task ambiguity, in which case

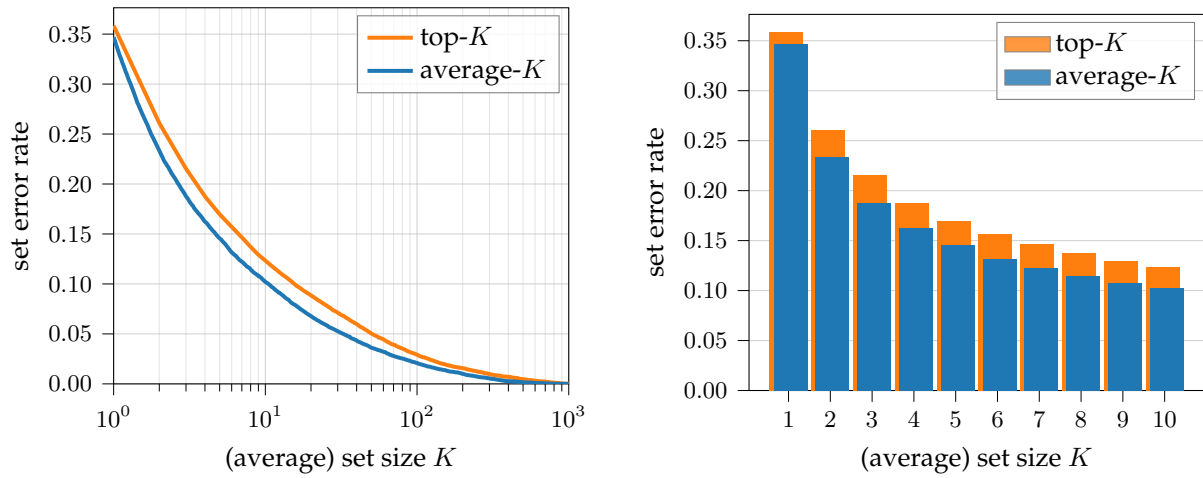
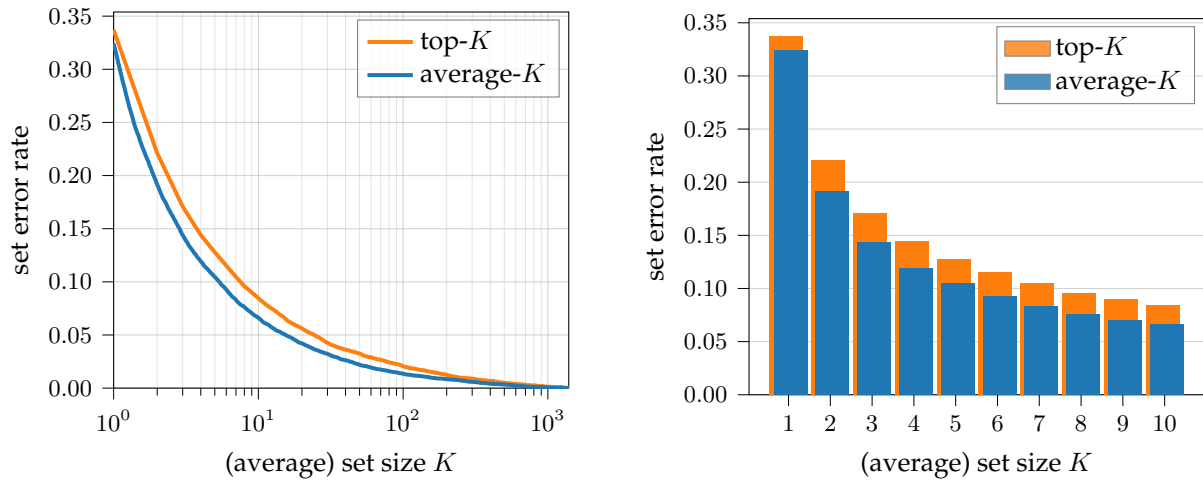
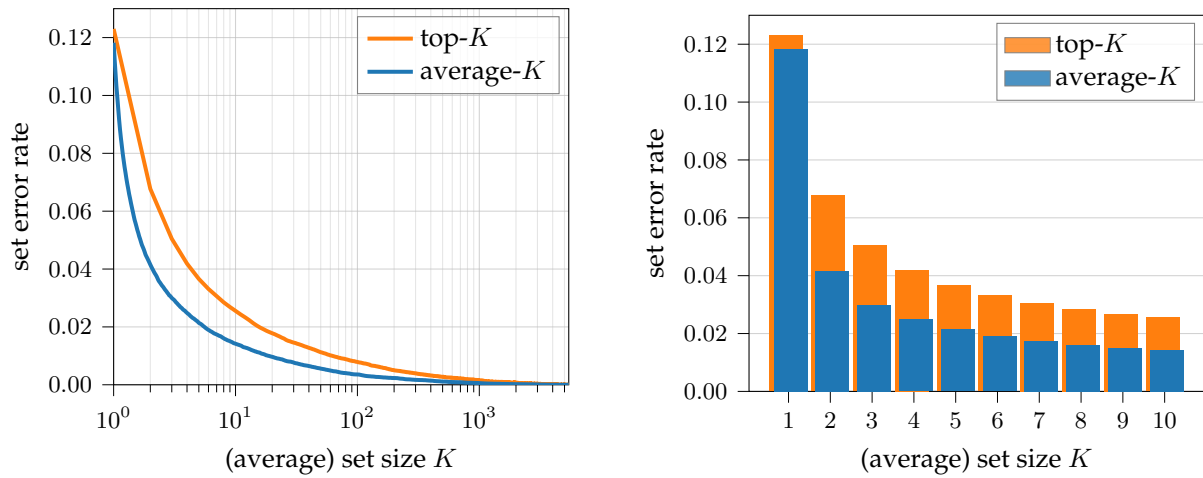
(a) *PlantCLEF2015*(b) *FGVC5-Fungi*(c) *FGVC6-ButterfliesMoths*

Figure 4.17: For all fine-grained visual classification datasets, top- K suffers a greater error rate than average- K , as shown in orange.

average- K classification is asymptotically useless, using set-valued classification approaches might still be relevant to lower the error rate when training data is limited.

Another related research direction is to derive specialized losses which target explicitly average- K classification. Such losses would allow to learn better average- K classifiers with fewer data. In particular, it would be interesting to see if the gaps observed between average- K and top- K in the experiments of [Section 4.6](#) could be broadened using such losses.

Finally, we have focused on a specific formulation of set-valued classification. However, other formulations have been proposed in the literature. Extending this analysis to them would allow to better understand when to use which set-valued classification framework. The next chapter is a step in that direction.

V

A unified framework for set-valued classification

Contents

5.1	A general framework for set-valued classifiers	84
5.1.1	From single-output to set-valued classifiers	84
5.1.2	Set-valued classification as a constrained optimization problem	85
5.2	Main set-valued classification frameworks	87
5.2.1	Point-wise control – almost sure type constraints	88
5.2.2	Average control – constraints in expectation	91
5.3	Empirical comparison of the frameworks	94
5.3.1	Estimation in practice with neural networks	94
5.3.2	Constraint satisfiability in practice	95
5.3.3	Comparison on real-world datasets	97
5.4	Summary of the set-valued classification formulations	101
5.4.1	Strengths and weaknesses of the frameworks	101
5.4.2	Hybrid set-valued classifiers	102
5.5	Conclusion	104

Set-valued classifiers generalize classic multi-class classifiers by outputting a set of class candidates rather than a single one. These classifiers are interesting in the presence of intrinsic ambiguity in the dataset as they allow to provide a decision close to that of a human. Unlike the usual classification setup, there is not a single way to define the set-valued classifier of interest and, in fact, several formulations have been proposed in the literature. One of the goals of this chapter is to unify most of these formulations in a common framework in order to analyze them jointly. We highlight similarities shared by all the formulations and emphasize their differences. In our exposition, we follow a unified statistical setting where the description of the frameworks, methods, and underlying trade-offs become more transparent.

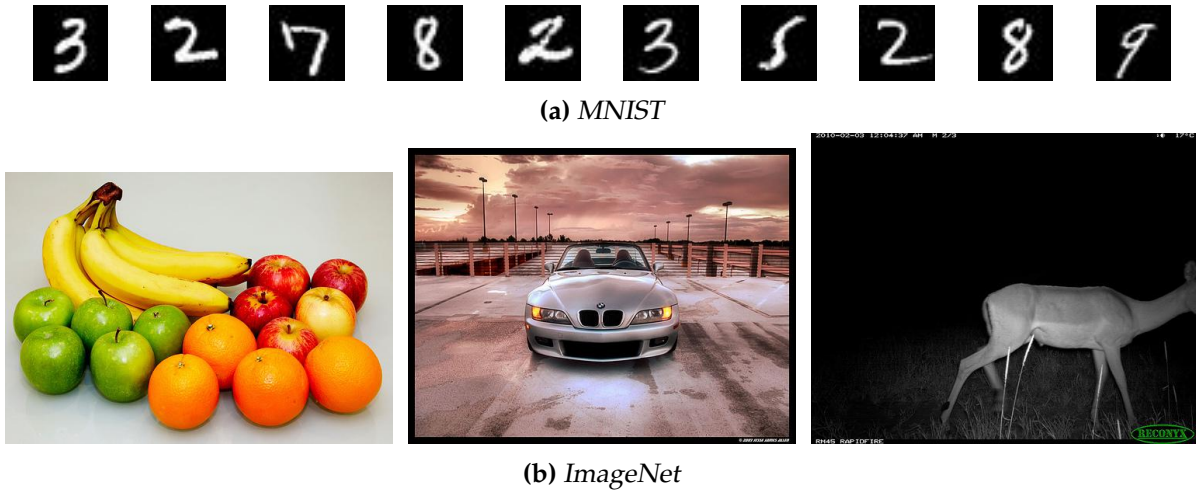


Figure 5.1: Examples from MNIST and ImageNet multi-class classification tasks.

5.1 A general framework for set-valued classifiers

In this section, we recall what are set-valued classifiers before introducing our unifying framework used to analyze set-valued classification frameworks.

5.1.1 From single-output to set-valued classifiers

Classically, multi-class classification considers the problem of learning a *single-output classifier* $h : \mathcal{X} \rightarrow \mathcal{Y}$, where $\mathcal{Y} := \{1, \dots, C\}$, based on data $(x_1, y_1), \dots, (x_n, y_n) \in \mathcal{X} \times \mathcal{Y}$ which minimizes the error rate $\mathbb{P}_{\mathcal{X}, \mathcal{Y}} [Y \neq h(X)]$ on unseen data.

For datasets such as MNIST (Figure 5.1a), single-output classifiers have been very successful and we are now able to learn models with a very low error rate, i.e. less than 1% (Ciregan et al., 2012). However, nowadays, common datasets are much more complex leading to higher classification error rates. For instance, on ImageNet dataset (Figure 5.1b) it is around 20% using state-of-the-art models (Xie et al., 2017). Such a high error rate is not solely due to the difficulty to construct a good classifier, but it is rather due to the ambiguity of the classification task. As shown in Figure 5.1b, some images are intrinsically ambiguous: in some images, several objects are present; in others, due to occlusion or noise in the image, it is not clear – even to a human expert – what class should be predicted. Forcing a classifier to predict a single class will thus mechanically increase its error rate as in such cases several answers can be considered as correct.

A way to deal with this ambiguity is to allow the classifier to predict a set of candidate classes rather than a single one (Grycko, 1993). These types of classifiers are called *set-valued classifiers*. They are defined as a mapping from the input space \mathcal{X} to the set of all subsets of \mathcal{Y} , i.e. the power set of \mathcal{Y} denoted by $\mathcal{P}(\mathcal{Y})$:

$$\Gamma : \mathcal{X} \rightarrow \mathcal{P}(\mathcal{Y}) .$$

Note that by considering set-valued classifiers we do not alter the underlying multi-class data-generating process, that is, the observed data $(x_1, y_1), \dots, (x_n, y_n) \in \mathcal{X} \times \mathcal{Y}$ is still that of multi-class – each instance x_i is associated to a unique label y_i .

The easiest way to build such a classifier is to always predict a fixed number of classes such as the top-5 most probable classes. This approach is in fact the one chosen in the official

ImageNet classification challenge (Russakovsky et al., 2015). However, as shown in the previous images, in general, there is no reason to predict exactly 5 or any other fixed number of classes all the time. As detailed in the previous chapter, Chapter 4, this observation generalizes to most recent datasets. In such cases, it is possible to provide a more informative prediction by relaxing this hard constraint on the set size.

In general, it is thus necessary to develop other formulations of set-valued classification strategies that could be more suited for the problem at hand. Most of the known formulations can be seen as different ways to choose a trade-off between two complementary quantities: the *error rate* and the *set size* (see Section 5.1.2 for its formal definition). The former quantifies the accuracy of a set-valued classifier Γ – how likely $\Gamma(\mathbf{X})$ contains the underlying real label Y – while the latter quantifies its informativeness – $\Gamma(\mathbf{X})$ should be as small as possible. For instance, top-5 classification strategy consists in minimizing the average error rate $\mathbb{P}_{\mathbf{X},Y} [Y \notin \Gamma(\mathbf{X})]$ under the constraint that the set size $|\Gamma(\mathbf{x})|$ is less than or equals to 5 for every input $\mathbf{x} \in \mathcal{X}$.

Note that there is no formulation that is always better than the others or one which is incorrect. On the contrary, each formulation is complementary to one another: they are suited to different contexts and applications. The goal of this chapter is to provide a comparison of the different frameworks with a concise overview of the literature and to fill gaps that were not previously addressed.

5.1.2 Set-valued classification as a constrained optimization problem

Formalizing the intuitive idea behind set-valued classifiers not only necessitate a rigorous definition of size and error rate of a set-valued classifier $\Gamma : \mathcal{X} \rightarrow \mathcal{P}(\mathcal{Y})$. It also requires, in the spirit of decision theory, a principled way to choose what is the optimal set-valued classifier Γ^* . Unlike single-output prediction where it is natural to target the classifier $h^* : \mathcal{X} \rightarrow \mathcal{Y}$ which maximizes the accuracy, in the set-valued classification framework, such a natural choice to define the objective does not exist. Indeed, as mentioned before, a good set-valued prediction always aims for a trade-off between two quantities: a measure of the accuracy of the predicted sets and a measure of their informativeness. In this paper, we will focus on a particular choice of such measures, respectively, *error rate* and *set size*. These quantities can be defined either point-wise or on average.

Error rate The error rate can be defined point-wise or on average as

$$\begin{aligned} \mathcal{E}(\Gamma; \mathbf{x}) &:= \mathbb{P}_{\mathbf{X},Y} [Y \notin \Gamma(\mathbf{X}) \mid \mathbf{X} = \mathbf{x}], & (\text{point-wise error}) \\ \mathcal{E}(\Gamma) &:= \mathbb{P}_{\mathbf{X},Y} [Y \notin \Gamma(\mathbf{X})]. & (\text{average error}) \end{aligned}$$

Naturally, the average error is related to point-wise error by $\mathcal{E}(\Gamma) = \mathbb{E}_{\mathbf{X}} [\mathcal{E}(\Gamma; \mathbf{X})]$. The error rate is associated to the accuracy of the set-valued classifier Γ . It is often named *coverage*, *recall*, or *risk*.

Set size Analogously, set size can also be viewed in a point-wise manner or on average,

$$\begin{aligned} \mathcal{I}(\Gamma; \mathbf{x}) &:= |\Gamma(\mathbf{x})|, & (\text{point-wise size}) \\ \mathcal{I}(\Gamma) &:= \mathbb{E}_{\mathbf{X}} [|\Gamma(\mathbf{X})|]. & (\text{average size}) \end{aligned}$$

Similarly to the error rate, the two definitions above admit the following relationship $\mathcal{I}(\Gamma) = \mathbb{E}_{\mathbf{X}} [\mathcal{I}(\Gamma; \mathbf{X})]$. The smaller the set size the more informative the set-valued classifier Γ is. In the literature, the set size can also be called *expected set size*, *adaptive set size* or *information*.

NAME	OBJECTIVE $\mathcal{F}_{\mathbb{P}}(\Gamma)$	CONSTRAINT $\mathcal{C}_{\mathbb{P}}$	OPTIMAL $\Gamma_{\mathcal{F}_{\mathbb{P}}, \mathcal{C}_{\mathbb{P}}}^*$	SECTION
Penalized	$\mathcal{E}(\Gamma) + \lambda \mathcal{I}(\Gamma)$	N/A	Γ_{λ}^* : threshold	Section 5.2.2
Point-wise size	$\mathcal{E}(\Gamma)$	$\mathcal{I}(\Gamma; \mathbf{x}) \leq k$	Γ_k^* : top- k	Section 5.2.1
Average size	$\mathcal{E}(\Gamma)$	$\mathcal{I}(\Gamma) \leq \bar{k}$	$\Gamma_{\bar{k}}^*$: threshold	Section 5.2.2
Point-wise error	$\mathcal{I}(\Gamma)$	$\mathcal{E}(\Gamma; \mathbf{x}) \leq \epsilon$	Γ_{ϵ}^* : top- $k(\mathbf{x})$	Section 5.2.1
Average error	$\mathcal{I}(\Gamma)$	$\mathcal{E}(\Gamma) \leq \bar{\epsilon}$	$\Gamma_{\bar{\epsilon}}^*$: threshold	Section 5.2.2
Hybrid size	$\mathcal{E}(\Gamma)$	$\mathcal{I}(\Gamma) \leq \bar{k},$ $\mathcal{I}(\Gamma; \mathbf{x}) \leq k$	hybrid	Section 5.4.2
Hybrid error	$\mathcal{I}(\Gamma)$	$\mathcal{E}(\Gamma) \leq \bar{\epsilon},$ $\mathcal{E}(\Gamma; \mathbf{x}) \leq \epsilon$	hybrid	Section 5.4.2

Table 5.1: Nomenclature and description based on the formulation (P). We recall the notation: average error rate $\mathcal{E}(\Gamma) = \mathbb{P}_{\mathbf{X}, Y} [Y \notin \Gamma(\mathbf{X})]$; point-wise error rate $\mathcal{E}(\Gamma; \mathbf{x}) = \mathbb{P} [Y \notin \Gamma(\mathbf{X}) \mid \mathbf{X} = \mathbf{x}]$; average set size $\mathcal{I}(\Gamma) = \mathbb{E}_{\mathbf{X}} [|\Gamma(\mathbf{X})|]$; point-wise set size $\mathcal{I}(\Gamma; \mathbf{x}) = |\Gamma(\mathbf{x})|$.

The trade-off between the set size and the error rate can be set in various ways and is dependent on the application at hand. Nevertheless, most of the optimal set-valued classifiers of interest can be formalized via a minimization problem under distribution dependent constraints of the form:

$$\begin{aligned} \Gamma_{\mathcal{F}_{\mathbb{P}}, \mathcal{C}_{\mathbb{P}}}^* \in \arg \min_{\Gamma: \mathcal{X} \rightarrow \mathcal{P}(\mathcal{Y})} \mathcal{F}_{\mathbb{P}}(\Gamma) \\ \text{s.t. } \Gamma \in \mathcal{C}_{\mathbb{P}} \end{aligned} \quad (\mathcal{P})$$

where $\mathbb{P}_{\mathbf{X}, Y}$ is denoted \mathbb{P} to simply notations, $\mathcal{F}_{\mathbb{P}}$ is a real-valued *objective functional* on set-valued classifiers, and $\mathcal{C}_{\mathbb{P}}$, the *feasible set*, is a distribution dependent family of set-valued classifiers satisfying the desired constraints. Table 5.1 provides a quick overview of the formulations that we describe in this chapter. They correspond to particular choices of the objective $\mathcal{F}_{\mathbb{P}}$ and the constraint $\mathcal{C}_{\mathbb{P}}$.

Within this unified framework, the optimal set-valued classifier $\Gamma_{\mathcal{F}_{\mathbb{P}}, \mathcal{C}_{\mathbb{P}}}^*$ is considered as the gold standard. Consequently, the goal of the practitioner is to build a data-driven set-valued classifier $\hat{\Gamma}$ using the sample $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)$ which preserves, as much as possible, all the desirable features (e.g. error rate or size) of the optimal set-valued classifier $\Gamma_{\mathcal{F}_{\mathbb{P}}, \mathcal{C}_{\mathbb{P}}}^*$. Despite the large variety of pairs $(\mathcal{F}_{\mathbb{P}}, \mathcal{C}_{\mathbb{P}})$ that can be considered, most of the frameworks admit a general plug-in driven approach for the construction of $\hat{\Gamma}$. It consists of two principal steps: first, derive the closed-form solution of the problem (P) (see Section 5.2 for details); second, estimate all the unknown quantities in the expression for $\Gamma_{\mathcal{F}_{\mathbb{P}}, \mathcal{C}_{\mathbb{P}}}^*$.

Assuming the distribution $\mathbb{P}_{\mathbf{X}, Y}$ is known, the optimal set-valued classifier $\Gamma_{\mathcal{F}_{\mathbb{P}}, \mathcal{C}_{\mathbb{P}}}^*$ can be derived explicitly. Its expression typically depends on the marginal distribution $\mathbb{P}_{\mathbf{X}}$ and the conditional distribution η of the labels given an instance \mathbf{x} defined as

$$\eta(\mathbf{x}) := \mathbb{P} [Y = l \mid \mathbf{X} = \mathbf{x}].$$

The former tells us how often an observation \mathbf{x} (or its neighborhood) can be observed, while the latter measures the relevance of label l for the instance \mathbf{x} . The explicit expression for the optimal set-valued classifier $\Gamma_{\mathcal{F}_{\mathbb{P}}, \mathcal{C}_{\mathbb{P}}}^*$ in all the considered frameworks reduces to one of the following three cases:

- **Thresholding:** there exists $\lambda = \lambda_{\mathbb{P}} \in [0, 1]$, such that for all $\mathbf{x} \in \mathcal{X}$ the optimal set-valued

classifier can be defined as

$$\Gamma_{\mathcal{F}_{\mathbb{P}}, \mathcal{C}_{\mathbb{P}}}^*(\mathbf{x}) = \{l \in \mathcal{Y} : \eta_l(\mathbf{x}) \geq \lambda\}.$$

In this case, for a given instance \mathbf{x} , the classifier $\Gamma_{\mathcal{F}_{\mathbb{P}}, \mathcal{C}_{\mathbb{P}}}^*$ predicts the classes which probability $\eta_l(\mathbf{x})$ is over a certain level λ .

Note that $\lambda = \lambda_{\mathbb{P}}$ can depend on the distribution $\mathbb{P}_{\mathcal{X}, \mathcal{Y}}$, which is unknown in practice.

- **Point-wise (adaptive) top-k:** for all $\mathbf{x} \in \mathcal{X}$, there exists $k = k_{\mathbb{P}}(\mathbf{x}) \in \{1, \dots, C\}$, such that the optimal set-valued classifier can be defined as

$$\Gamma_{\mathcal{F}_{\mathbb{P}}, \mathcal{C}_{\mathbb{P}}}^*(\mathbf{x}) = \text{top}_{\eta}(\mathbf{x}, k),$$

where the operator $\text{top}_{\eta}(\mathbf{x}, k)$ outputs the k labels with the largest values of conditional probability $\eta_l(\mathbf{x})$. Following this strategy implies that the point-wise size of Γ satisfies $\mathcal{I}(\Gamma; \mathbf{x}) = k_{\mathbb{P}}(\mathbf{x})$.

Note that $k = k_{\mathbb{P}}(\mathbf{x})$ can be adaptive, i.e. it is not necessarily the same for all the instances $\mathbf{x} \in \mathcal{X}$. Moreover, it can also depend on the distribution $\mathbb{P}_{\mathcal{X}, \mathcal{Y}}$, which is unknown in practice.

- **Hybrid:** the hybrid strategy can be obtained by a point-wise intersection of the thresholding rule and the point-wise (adaptive) top- k rule defined above. We describe several examples of such strategy in [Section 5.4.2](#), see [Table 5.1](#) for some examples. We are not aware of previous works that consider hybrid frameworks.

As said before, the value of λ , for the thresholding rule, and the value of k , for the top- k rule, can depend on the unknown distribution $\mathbb{P}_{\mathcal{X}, \mathcal{Y}}$. In such cases, they ought to be estimated and it can happen that the constraint $\mathcal{C}_{\mathbb{P}}$ is not guaranteed to be satisfied in practice.

5.2 Main set-valued classification frameworks

Several set-valued classifiers fall within the formulation [\(P\)](#). This section aims to review four popular and intuitive set-valued classification frameworks that fit in our unifying formulation. In these four formulations, the feasible set $\mathcal{C}_{\mathbb{P}}$ in [\(P\)](#) constrains either the error rate or the size of the set-valued classifier while minimizing the averaged version of the other quantity. For each of them, we describe the optimal set-valued classification rules, discussing their advantages and disadvantages, and provide a brief literature review. Because there have been contributions from different fields and there is no standardized naming for the different formulations, providing an exhaustive overview of the literature is impossible and some contributions might be missing.

As we will see, the closed-forms of the optimal set-valued classifiers are all expressed in terms of the conditional probability $\eta(\mathbf{x})$. This suggests a natural first approach to build data-driven estimators based on the plug-in principle: we first build an estimator $\hat{\eta}(\mathbf{x})$ of this conditional probability, then, we plug it into the optimal rule. However, as we will see, for some formulations, in order to enforce its constraint, we might need additional data. The estimation procedures can be categorized into three types depending if they require *no additional* data, *unlabeled* additional data, or *labeled* additional data.

Note that in the case where additional unlabeled data is required, if such data was already available beforehand, i.e. in the semi-supervised setting, then it can be leveraged directly by the estimation procedure. However, as we will see, in most cases, we do not require a lot of

additional data, less than the training data, unlike most semi-supervised cases where unlabeled data is abundant, more than training data. If, on the other hand, only labeled data is available, then the estimation procedure might require to split this data in two and to discard some labels.

As in the previous chapter, [Chapter 4](#), we make the following assumption.

Assumption C (Continuity assumption). *For all $k \in \{1, \dots, C\}$, the cumulative distribution function (CDF) of η_k denoted F_{η_k} , defined as $F_{\eta_k}(t) = \mathbb{P}_{\mathbf{X}} [\eta_k(\mathbf{X}) \leq t]$, is continuous.*

We refer to [Chapter 4](#) for a discussion on the implications of this assumption. In particular, this assumption provides a sufficient condition under which several set-valued frameworks are well defined in the sense that their optimal set-valued classifier $\Gamma_{\mathcal{F}_{\mathbb{P}}, \mathcal{C}_{\mathbb{P}}}^*$ exists, is unique, and is deterministic. We will also use the following generalized inverse as in the previous chapter to define the appropriate thresholds for some set-valued classifiers.

Definition 5.2.1 (Generalized inverse). *Let f be a right-continuous non-increasing function. Its generalized inverse denoted f^{-1} is defined as*

$$f^{-1}(t) := \inf \{x \in \mathbb{R} : f(x) \leq t\}.$$

5.2.1 Point-wise control – almost sure type constraints

In this section, we describe the frameworks for which the constraint $\mathcal{C}_{\mathbb{P}}$ is defined by a *point-wise* type constraint on the classifier Γ while the objective $\mathcal{F}_{\mathbb{P}}$ is expressed in terms of an average. This type of constraint is also referred to as *almost sure*, abbreviated as a.s., as it can be violated but only on a set of instances of null measure.

Point-wise size control: top- k classifier

In classification tasks, it is common to replace the top-1 error rate with the top- k error rate. This first natural approach to set-valued prediction simply consists in predicting sets of same size k for every input, i.e.

$$|\Gamma(\mathbf{x})| = k,$$

which is the most straightforward generalization of top-1 prediction. In this case, the problem can be formulated as finding the optimal classifier Γ_k^* solving the following problem,

$$\begin{aligned} \Gamma_k^* &\in \arg \min_{\Gamma} \mathbb{P}_{\mathbf{X}, Y} [Y \notin \Gamma(\mathbf{X})] \\ \text{s.t. } &|\Gamma(\mathbf{X})| \leq k \quad \text{a.s.} \end{aligned} \tag{point-wise size control}$$

Note that, for this formulation, the feasible set is equal to $\mathcal{C}_{\mathbb{P}} = \{\Gamma : |\Gamma(\mathbf{X})| \leq k \text{ a.s.}\}$. The attractive feature of such a formulation is that the optimal classifier Γ_k^* admits a simple and intuitive closed-form solution – at every point \mathbf{x} it outputs k most probable candidates.

Proposition 5.2.1. *For all $k \in \{1, \dots, C\}$, the following classifier Γ_k^* is an optimal solution of the [\(point-wise size control\)](#) problem:*

$$\forall \mathbf{x} \in \mathcal{X}, \quad \Gamma_k^*(\mathbf{x}) = \text{top}_{\eta}(\mathbf{x}, k),$$

where the operator $\text{top}_{\eta}(\mathbf{x}, k)$ outputs the k labels with the largest conditional probability $\eta_l(\mathbf{x})$.

Proof. The minimization can be done for every input $\mathbf{x} \in \mathcal{X}$ independently by looking at the point-wise error rate of a set-valued classifier Γ which can be written as

$$\mathbb{P}[Y \notin \Gamma(\mathbf{X}) \mid \mathbf{X} = \mathbf{x}] = 1 - \sum_{l \in \Gamma(\mathbf{x})} \eta_l(\mathbf{x}).$$

It is clear that, among all set-valued classifiers predicting at most k labels, this quantity is minimized by the one predicting the k classes with the highest conditional probability $\eta_l(\mathbf{x})$. \square

Pros and cons: Top- k classifier is the most straightforward approach to set-valued classification and provides an exact control on the point-wise set size. This simplicity is at the price of a lack of adaptation to the heterogeneity of the problem. Indeed, top- k outputs k labels for every $\mathbf{x} \in \mathcal{X}$ independently of the level of ambiguity of the samples. Furthermore, the exact value of k to use might not always be clear beforehand and might require a posteriori refinement. Finally, this framework does not provide any control over the error rate.

Estimation: To estimate this set-valued classifier, a single labeled training dataset is required, no additional data is necessary. In particular, the plug-in top- k classifier is defined element-wise by

$$\forall \mathbf{x} \in \mathcal{X}, \quad \hat{\Gamma}_k(\mathbf{x}) = \text{top}_{\hat{\eta}}(\mathbf{x}, k).$$

Note that in fact this estimator does not require to estimate exactly the conditional probability $\eta(\mathbf{x})$ as any scoring function preserving the ranking of $\eta(\mathbf{x})$ will give exactly the same classifier.

Bibliographic references: This formulation has been used for a long time as a performance metric known as the *top- k error rate*. For instance, the top-5 error rate is the official metric of ImageNet (Russakovsky et al., 2015). Lapin et al. (2015) were probably the firsts to try to derive losses explicitly built to learn top- k classifiers. To this end, they proposed a surrogate hinge loss and showed the superiority of their specialized procedure compared to one-versus-all SVMs used for standard classification. Several works followed this approach and proposed convex surrogates losses for the (point-wise size control) formulation. Lapin et al. (2016) proposed a new surrogate hinge loss as well as a modified cross-entropy loss. They also defined the notion of *top- k calibration*, a property that is satisfied by their introduced losses functions, while Yang and Koyejo (2020) extended this analysis to study the consistency of such surrogate losses. Berrada et al. (2018) modified the loss proposed by Lapin et al. (2015) to build a smooth loss function adapted for neural networks and showed that the resulting method improves the state-of-the-art performance on classical image datasets. Other related approaches were also proposed (Cuturi et al., 2019; Blondel et al., 2020).

Point-wise error control

The other point-wise control formulation, dual to the top- k one, consists in minimizing the average set size under a point-wise error rate constraint:

$$\begin{aligned} \Gamma_\epsilon^* \in \arg \min_{\Gamma} \mathbb{E}_{\mathbf{X}} [|\Gamma(\mathbf{X})|] \\ \text{s.t. } \mathbb{P}_{\mathbf{X}, Y} [Y \notin \Gamma(\mathbf{X}) \mid \mathbf{X} = \mathbf{x}] \leq \epsilon \quad \text{a.s.} \end{aligned} \quad (\text{point-wise error control})$$

In this case, the feasible set is equal to $\mathcal{C}_{\mathbb{P}} = \{\Gamma : \mathbb{P}_{\mathbf{X}, Y} [Y \notin \Gamma(\mathbf{X}) \mid \mathbf{X} = \mathbf{x}] \leq \epsilon \text{ a.s.}\}$. The closed-form expression of the optimal set-valued classifier Γ_ϵ^* can be expressed as a top- $k(\mathbf{x})$ rule. It is an adaptive version of the top- k strategy where the number of predicted labels depends on the instance \mathbf{x} . It consists in predicting, for each point \mathbf{x} , the labels corresponding to top conditional probabilities such that their cumulative sum exceeds $1 - \epsilon$.

Proposition 5.2.2. For every $\epsilon \in [0, 1]$, the following classifier Γ_ϵ^* is optimal for the (point-wise error control) problem:

$$\forall \mathbf{x} \in \mathcal{X}, \quad \Gamma_\epsilon^*(\mathbf{x}) = \text{top}_\eta(\mathbf{x}, k_\epsilon(\mathbf{x})),$$

where

$$k_\epsilon(\mathbf{x}) = \min \left\{ k \in \{1, \dots, C\} : \sum_{l=1}^k \tilde{\eta}_l(\mathbf{x}) \geq 1 - \epsilon \right\},$$

$\tilde{\eta}_l(\mathbf{x})$ being a reordering of $\eta_l(\mathbf{x})$ in decreasing order, i.e. $\tilde{\eta}_1(\mathbf{x}) \geq \tilde{\eta}_2(\mathbf{x}) \geq \dots \geq \tilde{\eta}_C(\mathbf{x})$.

Proof. Similarly to the previous formulation, the minimization can be done point-wise. For every input $\mathbf{x} \in \mathcal{X}$, we thus look for the smallest set of labels $\Gamma(\mathbf{x})$ satisfying

$$\mathbb{P}_{\mathbf{X}, Y} [Y \notin \Gamma(\mathbf{X}) \mid \mathbf{X} = \mathbf{x}] \leq \epsilon \quad \Leftrightarrow \quad \sum_{l \in \Gamma(\mathbf{x})} \eta_l(\mathbf{x}) \geq 1 - \epsilon.$$

For any $\Gamma(\mathbf{x})$ satisfying this constraint, denote $k(\mathbf{x})$ the size of this set, i.e. $k(\mathbf{x}) = |\Gamma(\mathbf{x})|$. Clearly, the set $\Gamma'(\mathbf{x}) = \text{top}_\eta(\mathbf{x}, k(\mathbf{x}))$ also satisfies the constraint and has the same size. Thus, we can reduce the search of the optimal set to sets predicting the $k(\mathbf{x})$ largest values of the conditional probability. By definition, $k_\epsilon(\mathbf{x})$ is the smallest value of $k(\mathbf{x})$ for which such a set satisfies the point-wise error constraint which concludes the proof. \square

Pros and cons: Theoretically, this formulation provides strong guarantees on the point-wise error rate. It is thus very relevant for scenarios where the error rate must be controlled precisely. One major drawback of this formulation is that it does not provide any control over the set size and can thus predict large sets which can then be uninformative. A practical limitation is that it is hard to measure the point-wise error rate and thus the theoretical guarantees are often violated in practice.

Estimation: To estimate this set-valued classifier, a single labeled training dataset is required, no additional data is necessary. In particular, the plug-in set-valued classifier is defined element-wise by

$$\forall \mathbf{x} \in \mathcal{X}, \quad \hat{\Gamma}_k(\mathbf{x}) = \text{top}_{\hat{\eta}}(\mathbf{x}, \hat{k}_\epsilon(\mathbf{x})).$$

with

$$\hat{k}_\epsilon(\mathbf{x}) = \min \left\{ k \in \{1, \dots, C\} : \sum_{l=1}^k \tilde{\hat{\eta}}_l(\mathbf{x}) \geq 1 - \epsilon \right\}$$

where $\tilde{\hat{\eta}}_l(\mathbf{x})$ is a reordering of $\hat{\eta}_l(\mathbf{x})$ in decreasing order, i.e. $\tilde{\hat{\eta}}_1(\mathbf{x}) \geq \tilde{\hat{\eta}}_2(\mathbf{x}) \geq \dots \geq \tilde{\hat{\eta}}_C(\mathbf{x})$. Note that this estimator requires to estimate exactly the conditional probability $\eta(\mathbf{x})$ as they serve to estimate the point-wise error rate on which we set our constraint. The model must thus output calibrated probabilities which requires additional care during training or an extra post-processing step.

Bibliographic references: Until very recently, most references dealing with the point-wise error rate constraint (most of the time referred to as *conditional validity*) is related to the regression setting and not classification. For instance, Cai et al. (2014); Lei and Wasserman (2014); Lei et al. (2018) proposed an asymptotic study of such predictors under smoothness conditions on the regression function. A classical result in the finite sample regime due to Lei and Wasserman (2014, Lemma 1) states that no control on the point-wise error rate can be achieved in the

distribution-free setting (unless using trivial set-valued predictors). To overcome this difficulty, Barber et al. (2019) studied relaxations of this point-wise constraint which do not require to make assumptions on the underlying distribution. In the classification setting, the same limiting result in the distribution-free finite sample regime holds (Vovk, 2013). Thus methods were proposed to obtain approximate conditional error rate guarantees such as works by Vovk (2013); Romano et al. (2020); Cauchois et al. (2020).

5.2.2 Average control – constraints in expectation

In general, we do not always require a precise control on the point-wise set size or error rate but we are rather interested in their averaged counterpart. In this case, the control is done by *average* constraints.

Penalized risk

As both the objective function and the constraints are defined in terms of averages, these type of formulations amounts in balancing average set size and average error rate. An intuitive way to formalize this balancing is to minimize the following penalized risk setting a trade-off with parameter $\lambda > 0$:

$$\Gamma_\lambda^* \in \arg \min_{\Gamma} \mathbb{P}_{\mathbf{X}, Y} [Y \notin \Gamma(\mathbf{X})] + \lambda \mathbb{E}_{\mathbf{X}} [|\Gamma(\mathbf{X})|]. \quad (\text{penalized risk})$$

Here, λ can be understood as the penalty for predicting an additional label in the set for a given instance \mathbf{x} . Notably, the optimal set-valued prediction Γ_λ^* admits a closed-form solution written as a simple thresholding rule. This thus gives another possible interpretation of λ as the minimum value of the conditional probability above which the labels should be predicted.

Proposition 5.2.3. *For all $\lambda \geq 0$, the following classifier Γ_λ^* is optimal for the (penalized risk) formulation:*

$$\forall \mathbf{x} \in \mathcal{X}, \quad \Gamma_\lambda^*(\mathbf{x}) = \{l \in \mathcal{Y} : \eta_l(\mathbf{x}) \geq \lambda\}.$$

Proof. Denote the penalized risk $R_\lambda(\Gamma)$, it can be rewritten as

$$\begin{aligned} R_\lambda(\Gamma) &= \mathbb{P}_{\mathbf{X}, Y} [Y \notin \Gamma(\mathbf{X})] + \lambda \mathbb{E}_{\mathbf{X}} [|\Gamma(\mathbf{X})|] \\ &= \mathbb{E}_{\mathbf{X}} \left[1 - \sum_{l \in \Gamma(\mathbf{X})} \eta_l(\mathbf{X}) \right] + \mathbb{E}_{\mathbf{X}} \left[\sum_{l \in \Gamma(\mathbf{X})} \lambda \right] \\ &= 1 + \mathbb{E}_{\mathbf{X}} \left[\sum_{l \in \Gamma(\mathbf{X})} (\lambda - \eta_l(\mathbf{X})) \right]. \end{aligned}$$

This risk can be minimized point-wise for every $\mathbf{x} \in \mathcal{X}$ which results in the thresholding rule of the proposition. \square

Pros and cons: When the cost/threshold λ is fixed, this formulation is very simple. However, choosing the appropriate value for λ beforehand can be hard in practice. In particular, the way this parameter controls the error rate and the set size is not explicit. To overcome this difficulty, the next formulations control these quantities explicitly.

Estimation: To estimate this set-valued classifier, a single labeled training dataset is required, no additional data is necessary. In particular, the plug-in set-valued classifier is defined element-wise by

$$\forall \mathbf{x} \in \mathcal{X}, \quad \hat{\Gamma}_\lambda(\mathbf{x}) = \{l \in \mathcal{Y} : \hat{\eta}_l(\mathbf{x}) \geq \lambda\}.$$

Note that this plug-in estimator requires to estimate exactly the conditional probability $\eta(\mathbf{x})$ to apply the correct threshold λ . However, in general, as this formulation is expressed as a risk minimization problem, the decision rule can be learned directly using empirical risk minimization approaches removing the need to estimate the conditional probability explicitly, much alike traditional classification methods.

Bibliographic references: This framework is among the first to be proposed and analyzed in the literature by Ha (1996, 1997a,b). It is also known as *class-selective rejection* (Ha, 1996) or *class-selection* (Le Capitaine, 2014). It can be seen as a generalization of classification with reject option to the multi-class setting. To the best of our knowledge, from a statistical learning perspective, this framework has not been much studied. In particular, we have not found any study of empirical risk minimization procedures, nor proposal of surrogate losses for this risk.

Average set size control: average- \bar{k} classifier

Instead of the above penalized version of the problem, one can consider its constrained counterpart minimizing the error rate given a constraint on the average set size:

$$\begin{aligned} \Gamma_{\bar{k}}^* \in \arg \min_{\Gamma} \mathbb{P}_{\mathbf{X}, \mathcal{Y}} [Y \notin \Gamma(\mathbf{X})] \\ \text{s.t. } \mathbb{E}_{\mathbf{X}} [|\Gamma(\mathbf{X})|] \leq \bar{k}. \end{aligned} \quad (\text{average size control})$$

In this case, the feasible set is equal to $\mathcal{C}_{\mathbb{P}} = \{\Gamma : \mathbb{E}_{\mathbf{X}} [|\Gamma(\mathbf{X})|] \leq \bar{k}\}$. Note how, unlike the previous point-wise control formulations, the set $\mathcal{C}_{\mathbb{P}}$ unavoidably depends on the marginal distribution $\mathbb{P}_{\mathbf{X}}$ of \mathbf{X} . Interestingly, the optimal classifier $\Gamma_{\bar{k}}^*$ is also a thresholding rule as shown in the next proposition.

Proposition 5.2.4 (Denis and Hebiri (2017), Proposition 4). *Define*

$$\forall t \in [0, 1], \quad G(t) = \sum_{l=1}^C \mathbb{P}_{\mathbf{X}} [\eta_l(\mathbf{X}) \geq t].$$

Under Assumption C, for all $\bar{k} \in [0, C]$, the following classifier $\Gamma_{\bar{k}}^*$ is optimal for the (average size control) formulation:

$$\forall \mathbf{x} \in \mathcal{X}, \quad \Gamma_{\bar{k}}^*(\mathbf{x}) = \{l \in \mathcal{Y} : \eta_l(\mathbf{x}) \geq G^{-1}(\bar{k})\},$$

where G^{-1} is the generalized inverse of G (see Definition 5.2.1).

Note that, if we pick $\lambda = G^{-1}(\bar{k})$ as penalization parameter, then the (penalized risk) formulation yields the same optimal set-valued classifier as the one of the present formulation.

Pros and cons: This formulation can be seen as an adaptive version of the (point-wise size control) formulation which can handle the heterogeneity of the task ambiguity as studied in Chapter 4. As a consequence, the present optimal set-valued classifier yields a smaller error rate than the top- k classifier (since this last method is feasible for the (average size control)

formulation with $k = \bar{k}$). On the other hand, the threshold $G^{-1}(\bar{k})$ is distribution dependent and thus relies on the marginal distribution \mathbb{P}_X . This formulation does not provide any control over the error rate which can be potentially large for individual samples.

Estimation: To estimate this set-valued classifier, an additional unlabeled dataset is required to fit the threshold. Indeed, the constraint involved in this set-valued classifier relies on the marginal distribution \mathbb{P}_X . In particular, denote $(x'_i)_{i \in \{1, \dots, n'\}}$ this unlabeled dataset of size n' sampled from \mathbb{P}_X and define the following function

$$\forall t \in [0, 1], \quad \hat{G}(t) = \frac{1}{n'} \sum_{i=1}^{n'} \sum_{l=1}^C \mathbb{1}_{\hat{\eta}_l(x'_i) \geq t}.$$

This function is essentially the empirical counterpart of the function G of [Proposition 5.2.4](#) computed using the unlabeled dataset and for the estimator $\hat{\eta}$. The plug-in set-valued classifier is then defined element-wise by

$$\forall \mathbf{x} \in \mathcal{X}, \quad \hat{\Gamma}_{\bar{k}}(\mathbf{x}) = \left\{ l \in \mathcal{Y} : \hat{\eta}_l(\mathbf{x}) \geq \hat{G}^{-1}(\bar{k}) \right\},$$

where \hat{G}^{-1} is the generalized inverse of \hat{G} (see [Definition 5.2.1](#)).

Bibliographic references: The present framework was introduced by [Denis and Hebiri \(2017\)](#), where the authors proposed a semi-supervised procedure based on empirical risk minimization. They derived rates of convergence under smoothness conditions on the conditional probabilities. Later, [Chzhen et al. \(2019\)](#) developed a minimax analysis of this framework and derived optimal rates of convergence for a semi-supervised approach based on plug-in under smoothness conditions. They showed the superiority of semi-supervised approaches over their supervised counterparts in certain situations.

Average error rate control

Alternatively, we minimize the average set size given a constraint on the average error rate as follows

$$\begin{aligned} \Gamma_{\bar{\epsilon}}^* &\in \arg \min_{\Gamma} \mathbb{E}_X [|\Gamma(\mathbf{X})|] \\ \text{s.t. } &\mathbb{P}_{X,Y} [Y \notin \Gamma(\mathbf{X})] \leq \bar{\epsilon}. \end{aligned} \quad (\text{average error control})$$

For this framework, the feasible set is equal to $\mathcal{C}_{\mathbb{P}} = \{\Gamma : \mathbb{P}_{X,Y} [Y \notin \Gamma(\mathbf{X})] \leq \bar{\epsilon}\}$ and unavoidably depends on the joint distribution $\mathbb{P}_{X,Y}$. Again, this constrained formulation is closely tied to the penalized version and the optimal set-valued classifier $\Gamma_{\bar{\epsilon}}^*$ is a thresholding rule.

Proposition 5.2.5 ([Sadinle et al. \(2019\)](#), Theorem 1). *Define*

$$\forall t \in [0, 1], \quad H(t) = \mathbb{P}_{X,Y} [\eta_Y(\mathbf{X}) \geq t].$$

Under [Assumption C](#), for all $\bar{\epsilon} \in [0, 1]$, the following classifier $\Gamma_{\bar{\epsilon}}^$ is optimal for the (average error control) formulation:*

$$\forall \mathbf{x} \in \mathcal{X}, \quad \Gamma_{\bar{\epsilon}}^*(\mathbf{x}) = \left\{ l \in \mathcal{Y} : \eta_l(\mathbf{x}) \geq H^{-1}(1 - \bar{\epsilon}) \right\},$$

where H^{-1} is the generalized inverse of H (see [Definition 5.2.1](#)).

Again, note that, if we pick $\lambda = H^{-1}(1 - \bar{\epsilon})$ as penalization parameter, then the (penalized risk) formulation yields the same optimal set-valued classifier as the one of the present formulation.

Pros and cons: The present framework provides guarantees over the error rate. These guarantees are weaker theoretically than those of (point-wise error control) as they can result in samples with a high error rate. However, they are easier to enforce in practice as the average error rate is easy to measure on a labeled dataset. We additionally point out that the threshold $H^{-1}(1 - \bar{\epsilon})$ in the present formulation relies on the joint distribution $\mathbb{P}_{X,Y}$, unlike to (average size control) which threshold relied on the marginal distribution \mathbb{P}_X .

Estimation: To estimate this set-valued classifier, an additional labeled dataset is required to fit the threshold. Indeed, the constraint here depends on the joint distribution $\mathbb{P}_{X,Y}$. In particular, denote $(x'_i, y'_i)_{i \in \{1, \dots, n'\}}$ this labeled dataset of size n' sampled from $\mathbb{P}_{X,Y}$ and define the following function

$$\forall t \in [0, 1], \quad \hat{H}(t) = \frac{1}{n'} \sum_{i=1}^{n'} \sum_{l=1}^C \mathbb{1}_{y'_i=l} \mathbb{1}_{\hat{\eta}_l(x'_i) \geq t}.$$

This function is essentially the empirical counterpart of the function H of Proposition 5.2.5 computed using the labeled dataset and for the estimator $\hat{\eta}$. The plug-in set-valued classifier is then defined element-wise by

$$\forall x \in \mathcal{X}, \quad \hat{\Gamma}_{\bar{\epsilon}}(x) = \left\{ l \in \mathcal{Y} : \hat{\eta}_l(x) \geq \hat{H}^{-1}(1 - \bar{\epsilon}) \right\},$$

where \hat{H}^{-1} is the generalized inverse of \hat{H} (see Definition 5.2.1).

Bibliographic references: Lei (2014) studied a similar framework in the context of binary classification with reject option. This framework was later extended to the multi-class classification framework by Sadinle et al. (2019) which provided the expression of the optimal set-valued classifier. They also studied the asymptotic properties of the plug-in set-valued classifier under some smoothness assumptions on $\mathbb{P}_{X,Y}$. This formulation shares some ties with conformal prediction (Vovk et al., 2005). As explained in Chapter 2, the setting in conformal prediction is different as it is focused on online transductive learning whereas, here, we set ourselves in the offline inductive setting.

5.3 Empirical comparison of the frameworks

In this section, we carry out experiments on real-world datasets to analyze and compare the properties of the previously described formulations.

5.3.1 Estimation in practice with neural networks

In this section, we briefly recall results from Chapter 4 on how to build an estimator of the conditional probability $\eta(x)$ using neural networks. Essentially, it was shown in the previous chapter that optimizing for the negative log-likelihood allowed to build such an estimator. Indeed, it was shown in the previous chapter that, due to the fact that this loss is strongly proper, we have a direct upper bound on the estimation error of $\hat{\eta}(x)$ of the form,

$$\mathbb{E}_X [\|\hat{\eta}(X) - \eta(X)\|_1] \leq \sqrt{2\text{Reg}_{\log}(\hat{\eta})}.$$

In practice, when training neural networks, the negative log-likelihood overfits on the training data and usually generalizes poorly on validation data. This means that the scores

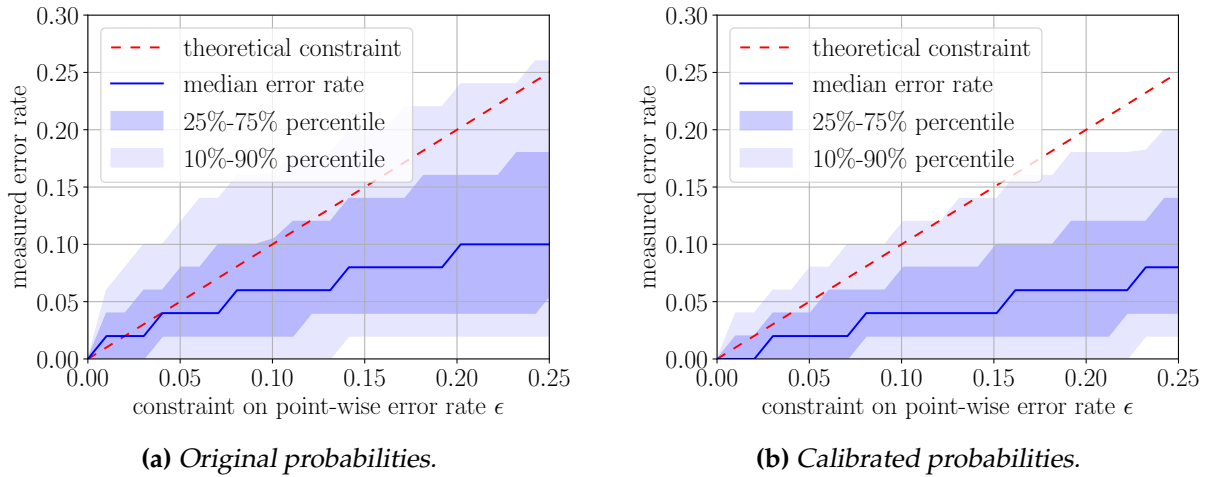


Figure 5.2: Plots assessing if the point-wise error rate constraint is violated on ImageNet. The distributions of the class error rates are computed for different values of point-wise error rate constraints. The median and the 10%, 25%, 75%, and 90% percentiles of the distribution of the class error rates are shown. If the constraint would have been satisfied, these curves would be below the red curve. These plots show that probability calibration is important for the (point-wise error control) formulation, however, it does not guarantee that the constraint will be satisfied.

outputted by the model are not correctly calibrated probabilities. However, in the previous chapter, these scores were shown to be calibrated for top- k and average- k classifiers estimation (i.e. for (point-wise size control) and (average size control) formulations). We thus use them to build the set-value classifiers.

5.3.2 Constraint satisfiability in practice

In this section, we analyze if the constraints defined in each framework are satisfied in practice. Indeed, for all frameworks, except point-wise size control (top- k), we can not guarantee that the constraint will be exactly satisfied. In this section, we study what influences these constraints and what are the good practices to accommodate to this limitation.

For these experiments, we use ImageNet (Russakovsky et al., 2015) which contains 1.3 million images from 1,000 classes. It is split into a training set of 1.2 million images, a validation set of 50,000 images, and a test set of 100,000 images. The labels of the official test set are not publicly available, it is thus common practice to use the official validation set as test set for experiments. This validation set contains 50,000 images of the 1,000 classes. It is balanced and thus contains 50 images per class. The proposed estimation procedures being built on top of any model, we use a pre-trained ResNet-152 neural network (He et al., 2016) from PyTorch model zoo¹. We then use the predictions made on the validation set to carry out the experiments in this section.

Satisfiability of the point-wise error rate constraint

We start with the (point-wise error control) framework. In this case, even on the training dataset, it is not possible to enforce this constraint. Worst, we can not explicitly compute the point-wise error rate for every data sample to check if the constraint is satisfied for all of them. To bypass

¹<https://pytorch.org/docs/stable/torchvision/models.html>

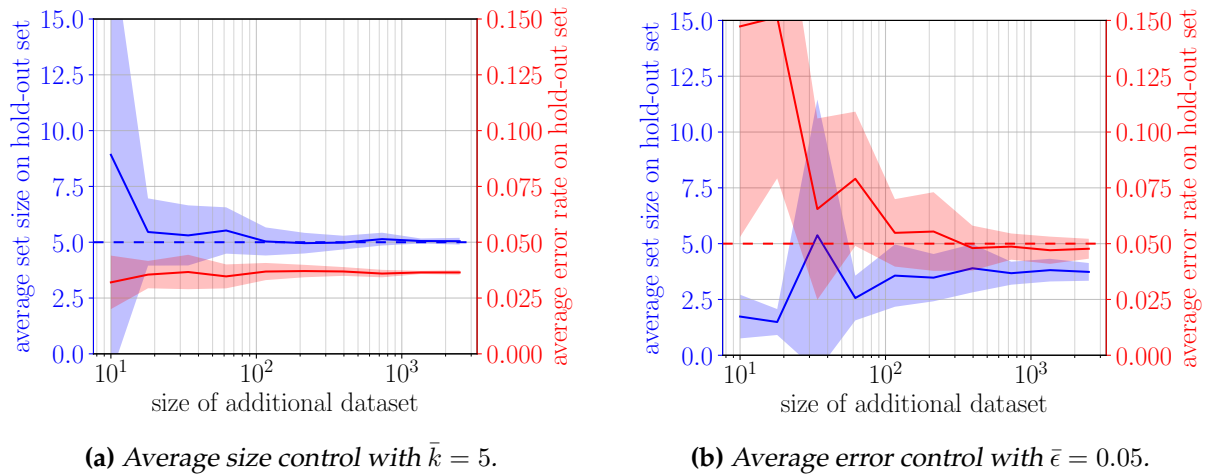


Figure 5.3: Estimation plots depending on the size of the additional dataset of the thresholds of the average size control and average error control formulations on ImageNet. The mean values are shown in full line while the shaded areas correspond to the standard deviation. The dashed line represents the constraint to estimate. Average size control requires less additional data than average error control to estimate a good threshold.

this difficulty, instead of computing point-wise error rates, we compute error rates per class. As said previously, ImageNet’s validation set is balanced and contains 50 images per class. This thus gives 1,000 error rate estimates which are equally reliable. Although not exactly point-wise error rates, they allow us to estimate the distribution of these error rates.

In Figure 5.2, the value of the constraint on the point-wise error rate is plotted against the distribution of the measured class error rates. In particular, for each constraint value, we display the 10%, 25%, 50%, 75% and 90 % percentiles. If these percentiles are above the $y = x$ line shown in red, the constraint is violated. As shown in the first plot, the original probabilities outputted by the neural network are not good enough to satisfy the constraint. We thus apply a probability calibration strategy for the second plot. This requires using the additional dataset to calibrate the predicted probabilities. We use a simple temperature scaling which is known to work well with neural networks (Guo et al., 2017). It consists in learning a temperature parameter T which is used to scale the predictions in the logit space before applying the softmax (see Chapter 4). As shown in the figure, this helps a lot to improve the satisfiability of the constraint. Thus, contrary to the other frameworks, having properly calibrated probabilities is important for the (point-wise error control) formulation. However, it is not sufficient to guarantee that the point-wise constraint will be satisfied.

This experiment shows that this point-wise error constraint is not easy to satisfy in practice. Moreover, although theoretically, we do not need an extra dataset to learn this constraint as shown in Section 5.2.1, in practice using this additional data to calibrate the probabilities is very helpful and worth doing. In the rest of the experiments, we will thus use temperature scaling to calibrate probabilities when studying this formulation.

Satisfiability of average constraints

We now focus on average control frameworks of Section 5.2.2. For these two frameworks, we need to compute a threshold. Unlike for (point-wise error control), this threshold can be estimated to exactly satisfy the constraint on a given dataset. However, if we do not have access to the data on which we want to make our predictions (e.g. in a classification model in

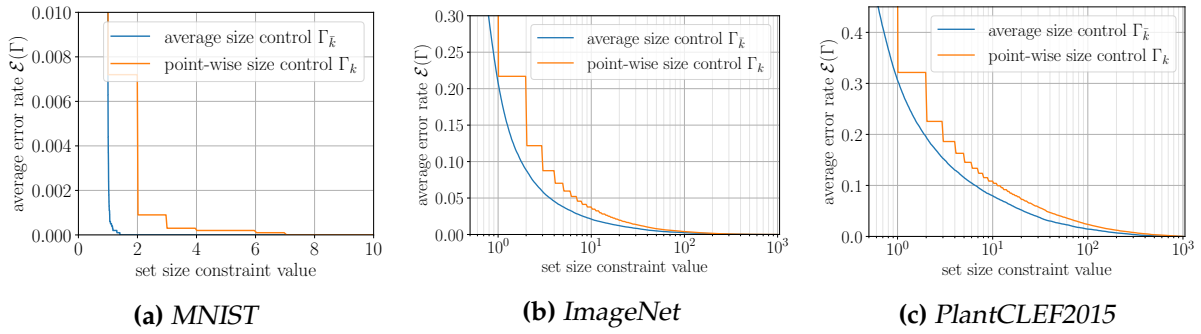


Figure 5.4: Comparison of (average size control) and (point-wise size control) formulations, i.e. average error rate minimization under set size constraint formulations, on different datasets. The average error rate is plotted against the value of the constraint for both formulations, i.e. \bar{k} for the average control and k for the point-wise control.

production), the threshold needs to be estimated beforehand on an additional dataset and will thus not exactly match the constraint. We study this case here.

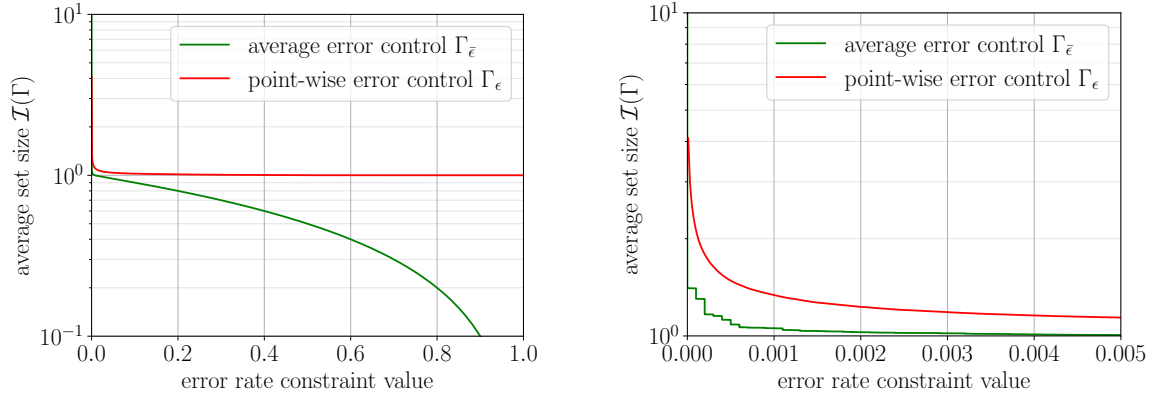
To do so, we analyze how the estimation quality of this threshold depends on the size of the additional dataset used. The validation set is split into two sets of equal sizes. The first one is used to sample this additional dataset while the second one serves as a hold-out set to measure the average set size and average error rate. The sampling of the additional dataset is performed 10 times to measure the standard deviation. The results are shown in Figure 5.3.

For (average size control), the average size of the predicted sets is contained in an interval around the fixed constraint and very quickly converges to it. The average error rate of those sets is also stable and quickly converges. On the other hand, for (average error control), the average error rate is much slower to converge to the fixed constraint. Moreover, it seems that it is biased towards higher error rates when using a small additional dataset. The average set size in this case is also less stable than for the (average size control). The main conclusion drawn from these plots is thus that the calibration of threshold requires fewer samples for (average size control) than for (average error control).

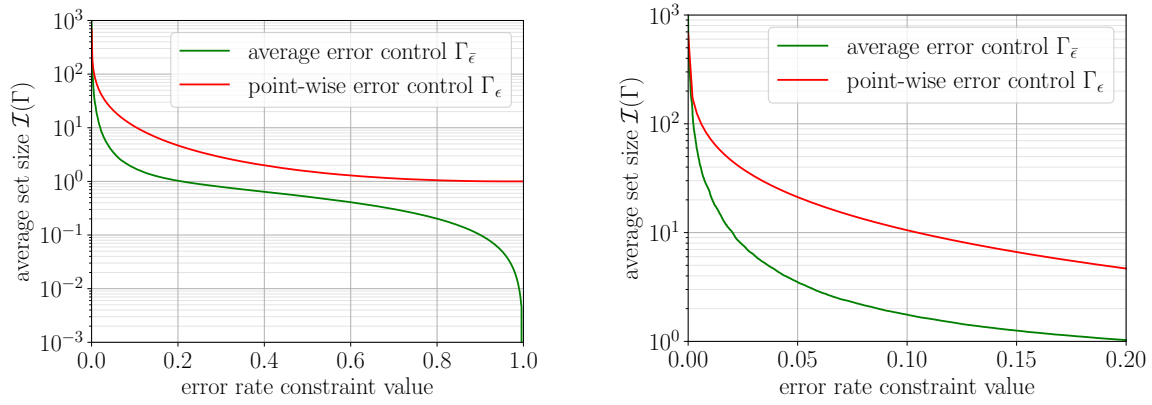
5.3.3 Comparison on real-world datasets

We now study the different formulations on different real-world datasets and analyze their properties. We focus on three image classification datasets: MNIST (LeCun et al., 1998), ImageNet (Russakovsky et al., 2015) and PlantCLEF2015 (Göeau et al., 2015). We use the same experimental setup as in Chapter 4 with the additional temperature scaling presented above.

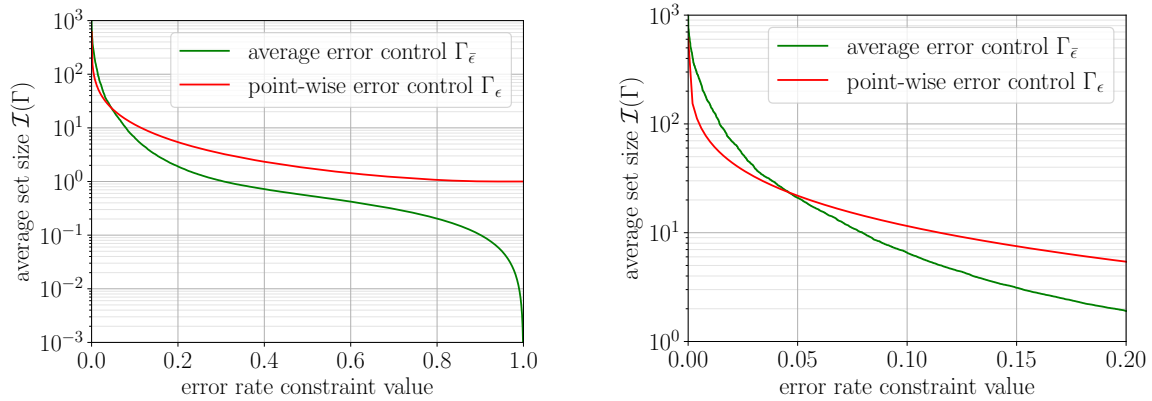
We first compare the formulations sharing the same objective but with a different constraint. In Figure 5.4, we compare (average size control) and (point-wise size control) formulations by plotting the achieved average error rate against the value of the constraint on the set size. Note that the curve of (point-wise size control) is a piecewise constant function as the point-wise set size can only take discrete values. The average set size does not have this constraint and thus the curve of (average size control) is continuous. As can be seen, the curve of (average size control) is always below the one of (point-wise size control). From left to right, those two curves are more or less steep. This suggests that those datasets have a different amount of ambiguity: MNIST is far less ambiguous than PlantCLEF2015. Moreover, the wideness of the gap between the two curves depends on the dataset. As studied in Chapter 4, this means that the heterogeneity of the ambiguity in those datasets are different.



(a) MNIST



(b) ImageNet



(c) PlantCLEF2015

Figure 5.5: Comparison of (average error control) and (point-wise error control) formulations, i.e. average set size minimization under error rate constraint formulations, on different datasets. The average set size is plotted against the value of the constraint for both formulations, i.e. $\bar{\epsilon}$ for the average control and ϵ for the point-wise control. The complete plots are displayed on the left column while the right one shows a zoom on a subpart of the plot.

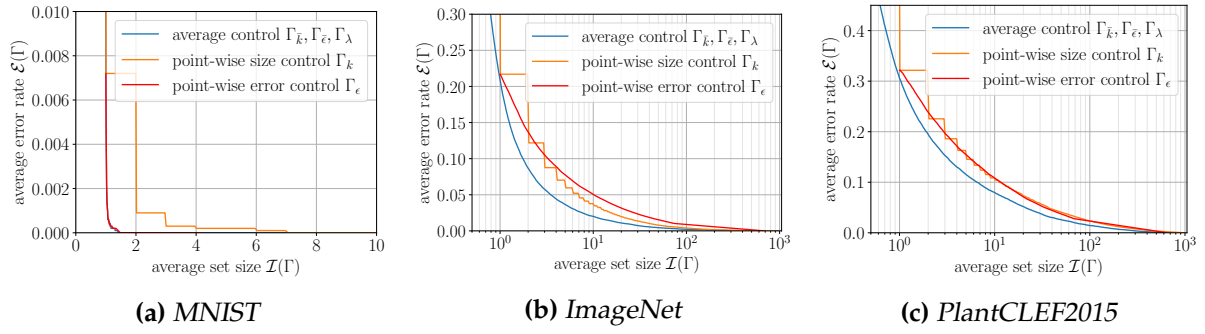


Figure 5.6: Comparison of the different frameworks in terms of their average set size / average error rate curves. In this case, all average control frameworks of Section 5.2.2 share the same optimal curve. The curves of the other frameworks are necessarily above this curve.

Similarly, in Figure 5.5, we compare (average error control) and (point-wise error control) formulations by plotting the achieved average set size against the value of the constraint on the error rate. As can be noticed, (average error control) allows to reach average set sizes below one whereas (point-wise error control) necessarily predicts sets of size greater or equal to one. Theoretically, if the constraints were properly enforced, the curve of (point-wise error control) should always be above the curve of (average error control). However, as we discussed in the previous section, the point-wise constraint is hard to guarantee in practice and thus this property can be violated as is the case on PlantCLEF2015. This suggests that the models on this dataset are unable to properly estimate η . As with the previous figure, these plots also give us insight into the amount of ambiguity in the datasets. In particular, we can see that it is easy to achieve a very low point-wise error rate on MNIST while keeping an average set size below two. Such (point-wise error control) formulation can thus be useful for MNIST if one wants a strong guarantee on the error rate. On the other hand, for ImageNet, enforcing a low point-wise error rate will imply predicting sets of size far bigger than 10 on average which will not be informative in most cases. Constraining the average error rate, however, allows predicting sets far smaller. Thus, for ImageNet, the (average error control) formulation can be interesting.

In Figure 5.6, we compare all the frameworks together by plotting their achieved average error rate against their measured average set size when varying their constraints. On this plot, the average formulations share the same curve which is also the optimal curve. Interestingly, for the different datasets, the two point-wise control formulations behave differently. On MNIST, the (point-wise error control) is very close to the optimal curve, while on ImageNet, it is above the (point-wise size control) curve, and on PlantCLEF2015, the curves of the point-wise control formulation coincide. This is in line with our previous comment on the usefulness of this formulation for MNIST and its limitations for ImageNet.

Finally, in Figure 5.7, we compare the distribution of the point-wise set sizes and point-wise error rates for the different formulations. These distributions are measured on ImageNet in a similar fashion than in the previous subsection: the error rates and set sizes are computed for each class and serve as proxies for the point-wise error rate and set size. These distributions are plotted using 2D histograms for various values of the constraints. To compare them, all the plots share the same color scale. Compared to the other frameworks, (point-wise size control) have a particular behavior: it allows a strict control on the point-wise set size with all the samples having exactly the same set size. The class error rates are thus always distributed along a line for this framework. Comparing with (average size control), we can see how the relaxation of the point-wise constraint allows reducing the error rate by predicting bigger sets for samples of higher error rate and smaller ones for samples with lower error rate. This remark is true for all formulations (except (point-wise size control)). Indeed, when transitioning from a weak to

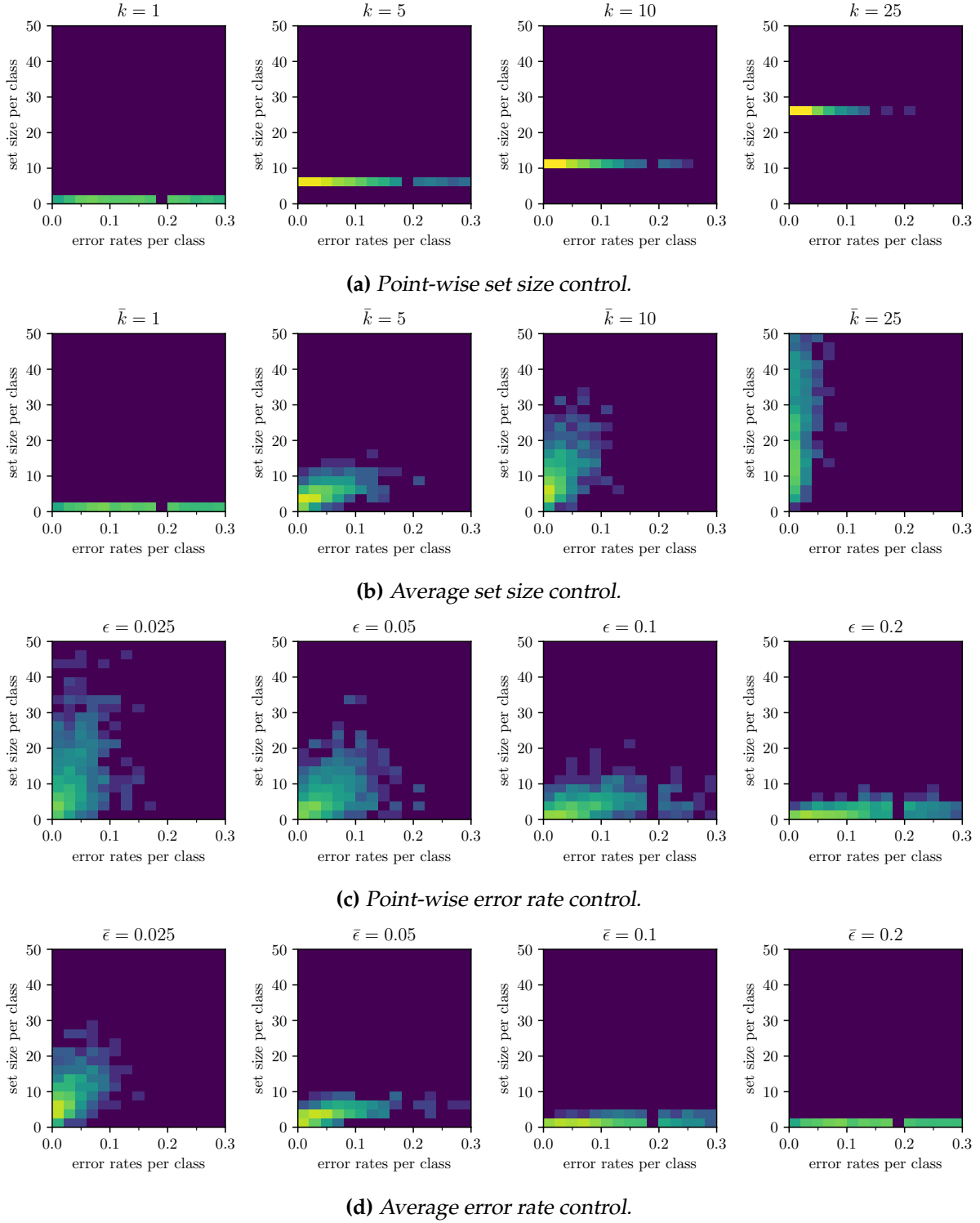


Figure 5.7: Comparison of the different formulations by looking at the distribution of the error rates and set sizes per class (on ImageNet). The plots share the same color scale.

a strong constraint, we can see that the point-wise error rates and the point-wise set size are correlated: the higher the error rate, the more labels are predicted, and vice-versa. To conclude, by focusing on the error control frameworks, we can notice that the set sizes are more dispersed for (point-wise error control) than for (average error control) at equal constraint values.

These different experiments highlight the differences between the proposed set-valued classification frameworks. We now summarize these different strengths and weaknesses.

5.4 Summary of the set-valued classification formulations

5.4.1 Strengths and weaknesses of the frameworks

In this section, we summarize what we believe are the strengths and weaknesses of each set-valued classification framework. We distinguish the pros and cons of each method related to the formulation itself from the ones related to the estimation of the associated set-valued classifiers on a real dataset.

Point-wise size control

- Formulation:
 - (+) Exact control on the set size.
 - (−) No control on the error rate which can be large.
 - (−) Does not take into account the heterogeneity of the task ambiguity.
 - (−) Setting a value beforehand for the set size k can be hard.
- Estimation:
 - (+) Relies only on the ordering of the conditional probabilities for each individual sample.
 - (+) Does not require any additional dataset for calibration.

Point-wise error control

- Formulation:
 - (+) Strong ideal control on the error rate.
 - (−) No control on the set size which can lead to huge (less uninformative) sets.
- Estimation:
 - (+) Does not require any additional dataset for calibration.
 - (−) Requires the exact values of conditional probabilities which might necessitate additional probability calibration in practice.
 - (−) Constraint often violated in practice, detecting and quantifying this violation is hard in general.

Average control – penalized risk

- Formulation:
 - (+) Simple and convenient formulation as a risk minimization problem.
 - (+) Simple optimal classifier expressed as a thresholding rule.
 - (−) No explicit control on the set size nor the error rate.
 - (−) Setting the parameter λ beforehand can be unnatural and hard.
- Estimation:
 - (+) Does not require any additional dataset for calibration.
 - (+) Formulation as a risk allows to naturally derive practical algorithms.

Average size control

- Formulation:
 - (+) Explicit control of the set size.
 - (+) Compared to point-wise size control, takes into account heterogeneity of the problem.
 - (−) No control on the error rate which can potentially be large.
 - (−) Only average control of the set size which can be large for some samples.
 - (−) Requires the knowledge of the marginal distribution \mathbb{P}_X to fix the threshold.
- Estimation:
 - (+) Satisfiability of the constraint can be checked on a dataset.
 - (·) A (small) additional unlabeled dataset can be required.

Average error control

- Formulation:
 - (+) Explicit control on the error rate.
 - (−) No control on the set size which can lead to huge (less uninformative) sets.
 - (−) Only average control of the error rate which can be large for some samples.
 - (−) Requires the knowledge of the joint distribution $\mathbb{P}_{X,Y}$ to fix the threshold.
- Estimation:
 - (−) Requires the exact values of conditional probabilities (which might necessitate additional calibration in practice).
 - (−) A (potentially large) additional labeled dataset can be required.

In the next subsection, we propose to look at some hybrid frameworks which attempt to mitigate the limitations of the previous formulations.

5.4.2 Hybrid set-valued classifiers

In the previous sections, we focused on formulations with a single, either point-wise or on average, constraint. In what follows, we describe some possible ways to combine both types of constraints into one formulation. In particular, we propose two simple hybrid formulations: hybrid size control and hybrid error control. Other hybrid formulations are possible but they

require more care as, depending on the values of the constraints, they might not have any solution. We thus propose to restrict ourselves to these two frameworks.

Hybrid size control

This formulation consists in minimizing the average error given constraints on both the point-wise and average set sizes. Given $k \in \mathbb{N}$ and $\bar{k} \in \mathbb{R}$ such that $0 < \bar{k} < k \leq C$, it consists in solving

$$\begin{aligned} \Gamma_{\bar{k},k}^* &\in \arg \min_{\Gamma} \mathbb{P}_{\mathbf{X},Y} [Y \notin \Gamma(\mathbf{X})] \\ \text{s.t. } \mathbb{E}_{\mathbf{X}} [|\Gamma(\mathbf{X})|] &\leq \bar{k} \\ |\Gamma(\mathbf{X})| &\leq k \quad \text{a.s.} \end{aligned} \quad (\text{hybrid size control})$$

This framework aims to have stronger control over the maximum set size than [\(average size control\)](#) while preserving its adaptivity compared to [\(point-wise size control\)](#). In particular, it allows to have more informative sets by forbidding huge sets. However, this added informativeness is traded against error rate as this formulation will typically have a higher error rate than [\(average size control\)](#).

The optimal set-valued classifier is then the intersection of a thresholding rule and of a top- k rule. It is given in the following proposition.

Proposition 5.4.1. *For all $k \in \{1, \dots, C\}$, define*

$$\forall t \in [0, 1], \quad G_k(t) = \sum_{l=1}^k \mathbb{P}_{\mathbf{X}} [\tilde{\eta}_l(\mathbf{X}) \geq t].$$

Under [Assumption C](#), for any $k \in \{1, \dots, C\}$ and $\bar{k} \in (0, k)$, assuming that $G_k^{-1}(\bar{k}) \neq 0$, the following set-valued classifier $\Gamma_{\bar{k},k}^$ is optimal for the [\(hybrid size control\)](#) formulation,*

$$\forall \mathbf{x} \in \mathcal{X}, \quad \Gamma_{\bar{k},k}^*(\mathbf{x}) = \left\{ l \in \mathcal{Y} : \eta_l(\mathbf{x}) \geq G_k^{-1}(\bar{k}) \right\} \cap \text{top}_{\eta}(\mathbf{x}, k),$$

where G_k^{-1} is the generalized inverse of G_k .

This classifier can then be estimated in a similar fashion than [\(average size control\)](#) but restricted to the top- k predictions.

Hybrid error control

This formulation consists in minimizing the average set size given constraints on both the point-wise and average error rates. Given $0 \leq \bar{\epsilon} < \epsilon \leq 1$, it consists in solving

$$\begin{aligned} \Gamma_{\bar{\epsilon},\epsilon}^* &\in \arg \min_{\Gamma} \mathbb{E}_{\mathbf{X}} [|\Gamma(\mathbf{X})|] \\ \text{s.t. } \mathbb{P}_{\mathbf{X},Y} [Y \notin \Gamma(\mathbf{X})] &\leq \bar{\epsilon} \\ \mathbb{P}_{\mathbf{X},Y} [Y \notin \Gamma(\mathbf{X}) \mid \mathbf{X} = \mathbf{x}] &\leq \epsilon \quad \text{a.s.} \end{aligned} \quad (\text{hybrid error control})$$

This framework can be seen as a relaxed version of [\(point-wise error control\)](#) where the constraint on the point-wise error rate can be loosened but is compensated by adding a constraint

on the average error rate. It will typically produce sets of size smaller than those of (point-wise error control) but bigger than (average error control). The optimal set-valued classifier in this case is a thresholding rule given in the following proposition.

Proposition 5.4.2. *For all $\epsilon \in [0, 1]$, define $k_\epsilon(\mathbf{X})$ as in Proposition 5.2.2, i.e.*

$$k_\epsilon(\mathbf{x}) = \min \left\{ k \in \{1, \dots, C\} : \sum_{l=1}^k \tilde{\eta}_l(\mathbf{x}) \geq 1 - \epsilon \right\},$$

and define

$$\forall t \in [0, 1], \quad H_\epsilon(t) = \mathbb{E}_{\mathbf{X}} \left[\sum_{l=1}^{k_\epsilon(\mathbf{X})} \tilde{\eta}_l(\mathbf{X}) \mathbb{1}_{\tilde{\eta}_l(\mathbf{X}) \geq t} \right].$$

Under Assumption C, for any $\epsilon \in [0, 1]$ and $\bar{\epsilon} \in (0, \epsilon)$, assuming $H_\epsilon(1 - \bar{\epsilon}) \neq 0$, the following set-valued classifier $\Gamma_{\bar{\epsilon}, \epsilon}^*$ is optimal for the (hybrid error control) formulation,

$$\forall \mathbf{x} \in \mathcal{X}, \quad \Gamma_{\bar{\epsilon}, \epsilon}^*(\mathbf{x}) = \left\{ l \in \mathcal{Y} : \eta_l(\mathbf{x}) \geq H_\epsilon^{-1}(1 - \bar{\epsilon}) \right\},$$

where H_ϵ^{-1} is the generalized inverse of H_ϵ .

This set-valued classifier is much harder to estimate in practice than those of the other formulations as the computation threshold requires to estimate H_ϵ which depends on $k_\epsilon(\mathbf{x})$. The point-wise optimal set size $k_\epsilon(\mathbf{x})$ is hard to estimate as shown previously in Section 5.3.

This set-valued classification framework, although easy to express as a minimization under constraints problem, is not exploitable in practice. Finding other, less direct, relaxations of the (point-wise error control) framework would be of practical interest.

5.5 Conclusion

In this chapter, we have proposed a general framework to study the most common set-valued classification formulations by providing a principled way to set the trade-off between error rate and set size. For each of these formulations, we provided the resulting optimal set-valued classifier and described how to estimate them. We have studied how the constraints defined in the framework are satisfied in practice and showed how these formulations compared to one another on real-world datasets. Finally, we have summarized the strengths and weaknesses of these frameworks and introduced some hybrid frameworks which attempt to mitigate some of their limitations.

Future research directions would be to develop the comparison between the different frameworks in a similar fashion than done in Chapter 4. Moreover, finer analysis of the hybrid frameworks and finding other ways to relax the point-wise error control framework would be of interest. Indeed, this last framework, by providing strong error control, seems natural in a lot of scenarios. Unfortunately, this power comes at the cost of estimation issues. Finding an appropriate framework trading some of this control against estimation guarantees in practice would be highly interesting.

More generally, although we have tried to incorporate most of the proposed formulations in our general framework, some set-valued classifiers do not fit in it. For instance, Del Coz et al.

(2009) propose to balance error rate and set size using an instance-level F_β -measure defined as

$$F_\beta(\Gamma) = \mathbb{E}_{\mathbf{X}, Y} \left[\frac{(1 + \beta^2) \mathbb{1}_{Y \in \Gamma(\mathbf{X})}}{\beta^2 + |\Gamma(\mathbf{X})|} \right],$$

where β controls the trade-off. Another example is the frameworks studied by [Mortier et al. \(2019\)](#). These are cast as a maximization of the expected value of utility functions of the form

$$U(\Gamma) = \mathbb{E}_{\mathbf{X}, Y} \left[g(|\Gamma(\mathbf{X})|) \mathbb{1}_{Y \in \Gamma(\mathbf{X})} \right],$$

where $g : \mathbb{N} \rightarrow [0, 1]$ is a function penalizing large sets. Fitting such set-valued formulations in our general framework can not be done directly. It would thus be interesting to understand what are the differences between our framework and these ones and how different are the sets predicted by each other.

VI

Conclusion and perspectives

6.1 Synthesis of results and general perspectives

In this thesis, we first studied, in [Chapter 3](#), how uncertainty information can be exploited to reject some of the predictions made by a classifier which incur a high error rate. To this end, we proposed to study a particular instance of classification with reject option framework: learning a rejector when the classifier is already trained and considered fixed. We showed that this amounts to learning an uncertainty score that preserves the ordering of the point-wise error rate of the classifier. We then focused on the case where the rejector is a hand-crafted criterion because it provides a way to analyze how we could estimate the point-wise error rate directly and in particular how to exploit uncertainty information for rejection purposes. By analyzing the point-wise error rate, we showed that both task ambiguity / aleatoric uncertainty and model / epistemic uncertainty are important. We empirically showed that rejecting based on the highest probability predicted by a neural network performs consistently well, independently of the amount of aleatoric and epistemic uncertainty. This suggests that it captures both uncertainties at the same time. We attempted to build criteria that separated both uncertainties and recombined them properly however experiments showed that this is more difficult than expected.

Based on this observation, in [Chapter 4](#), we focused on the first form of uncertainty, task ambiguity, and study natural frameworks to handle it: set-valued classification frameworks. The most naive approach to build sets is to predict the K most probable classes. However, this assumes that all the samples have the same level of ambiguity which is known to be wrong in most cases. Instead, we proposed to use average- K : the predictor can output sets of different sizes but on average their size must be equal to K . We studied when average- K is most beneficial compared to top- K . To this end, we characterized and described which was the correct notion of heterogeneity of ambiguity captured by average- K . We then showed that natural estimation procedures for top- K and average- K are in fact consistent and can thus be used to carry out the experiments. Finally, empirical studies on real-world image datasets showed that average- K always gave better results than top- K , proving its usefulness in practical settings.

In [Chapter 5](#), we generalized to other adaptive set-valued classification approaches and proposed a framework unifying most of them. These approaches are characterized by two quantities: error rate and set size, measured either point-wisely or on average. We proposed to set

the trade-off between these quantities by formulating set-valued classification as a constrained optimization problem enforcing constraints on one of this quantity and minimizing the other. For each of these formulations, we gave their optimal set-valued predictor which can be expressed either as a thresholding strategy on the conditional probability or as a (potentially) adaptive top- K . We showed how to estimate them and analyzed empirically how their constraints are satisfied in practice. Then, we studied their relative advantages and weaknesses on real-world data. To conclude, we introduced hybrid approaches mixing several of the previous constraints in order to mitigate the weaknesses of the different formulations.

These set-valued classification approaches describe natural ways to accommodate with task ambiguity. Carrying out a similar analysis than the one of [Chapter 4](#) could help understand if they capture a similar notion of heterogeneity of ambiguity or not. If not, this could help decide which formulation one should use on a given dataset based on the nature of its intrinsic ambiguity. A more qualitative analysis of the sets predicted by the different formulations could also help in that aspect.

For now, we have mainly focused on the infinite sample regime and left apart the model uncertainty present in practice. It would be interesting to understand how these approaches deal with such uncertainty. For instance, even in absence of task ambiguity, predicting a set could help reduce the error rate due to misclassification of the top-1 classifier. Furthermore, we only provided general estimation procedures that work for all formulation. Deriving more specialized procedures could help to build better estimators when only given a finite training dataset. It would be interesting to also understand the convergence rate of such procedures and to compare them across the different formulations: are some of these formulations easier to estimate than others?

Another interesting point to explore is to understand how we could create a *set-valued classification with reject option* framework. In particular, it would be interesting to use the set-valued classifier to handle properly the task ambiguity while the reject option would be used to detect the presence of model uncertainty. Such a framework would provide a nice way to force the model to disentangle both uncertainties. This could be helpful for a lot of practical scenarios. This could also serve as a proxy to measure the quality of both aleatoric and epistemic uncertainty estimators and to measure if we were able to properly separate them.

6.2 Applicative perspectives

We conclude this manuscript by highlighting some applied work done during this PhD which were not presented here. They illustrate some of the potential concrete applications of this work and provides some directions for future work.

6.2.1 Herbarium phenology annotation

In [Lorieul et al. \(2019\)](#), deep learning approaches were applied to show the performance of automated phenological stage annotation of herbariums specimens. Herbarium specimens are dried pressed plants or parts of plants that have been mounted on archival paper, see [Figure 6.1](#) for examples. These specimens are labeled with information about the plant species, the collection locality, date, etc. This type of data is important as those specimens have been collected over centuries and thus provide some historical information about the evolution of biodiversity through time and about the impact of human activity. In particular, phenological



Figure 6.1: Three herbarium specimens belonging to the same species (*Tilia americana*) with different phenological stage advancement. From left to right: non-fertile specimen, specimen with open flowers, and specimen with ripe fruits.



Figure 6.2: Illustration of the nine different phenophases to recognize.

annotation – such as the presence/absence of reproductive structures – is of interest as it serves as a proxy to study the impact of climate change on the reproduction of plants (Menzel et al., 2006). Deep learning approaches could be useful to automatically annotate phenological traits which would enable to leverage the vast amount of herbarium specimens digitized recently. Interestingly, this annotation task does contain a fair amount of ambiguity.

The paper first studied the classification of herbarium specimens into fertile/non-fertile categories. To assess the performance of the model, one of the co-author – an expert botanist – re-annotated a fraction of the dataset. Interestingly, he did not recover 100% of the original annotations, however, except for one, all of the mismatches were due to an ambiguous situation. This happened for different reasons:

- the reproductive structure is absent: it was present on the specimen on the field but not on the collected part, or it was lost during previous manipulations;
- the reproductive structure is hidden: it is present on the sheet but not visible on the scan, e.g. in an envelop, obscured by leaves, etc.;
- the exact definition of fertility is ambiguous: in particular, is a closed bud a sign of fertility?

To add up to the confusion, the text label on the sheet can state the specimen as being fertile but no reproductive material is visually evident on the scan.

This example shows how it could be useful to detect automatically these ambiguous cases and to separate them from the cases where the model is uncertain. Indeed, the action to be taken would differ:

- when the prediction of the model can not be trusted, a human could be asked to manually annotate the scan;
- in presence of intrinsic ambiguity, on the other hand, the scan is not sufficient to remove this uncertainty and it might be required to look at the physical herbarium sheet.

Correctly separating these two cases related to the two forms of uncertainty is thus important to take the correct action and to avoid wasting expert time.

Another goal of the paper was to push this analysis further and to automatically annotate more finely the phenological stage of the specimens. One co-author proposed using nine phenophases to classify the continuum between specimens with unopened flowers to specimens in fruits (Figure 6.2). The proposed division in nine stages was chosen based on an intuition that arose when the co-author looked at the data. These phenophases are defined using the relative proportions of closed buds, flowers, and fruits (see the paper for more details). This annotation methodology and the dataset used for our study were presented and published previously in Pearson (2019). When trained on this data, the neural network was able to recover the exact phenophase only 43% of the time. When it did not match the annotation, the predicted phenophase was however close to it. Manual re-annotation of the data showed that even the original annotator was not able to perform better than the neural network. This suggests that the annotation methodology chosen was not fully reproducible and introduced a fair amount of ambiguity in the annotations.

Using set-valued classification methods, it could be possible to confirm this hypothesis by measuring more precisely the amount and the form of ambiguity. In particular, it could be possible to measure if the neighborhood phenophases are often predicted together and thus ambiguous. Moreover, such methods could help decide which phenophases should be fused



Figure 6.3: Occurrences distribution over France and US. Blue dots represent training data while red dots represent test data.

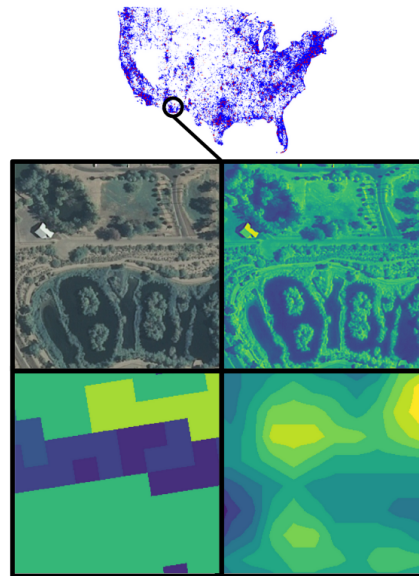


Figure 6.4: Each observation is associated with high-resolution covariates (clockwise from top left: RGB imagery, IR imagery, altitude, land cover).

together and which ones should not. In this example, set-valued classification approaches, used as proxies to measure the intrinsic ambiguity, could provide a way to analyze and understand more finely the dataset and its annotations.

6.2.2 GeoLifeCLEF: location-based species prediction

The GeoLifeCLEF2020 dataset (Cole et al., 2020) is a large-scale dataset consisting in 1.9 million species observations sampled over France and US (Figure 6.3). Each of these observations is coupled with high-resolution covariates (Figure 6.4) as well as climate and soil variables. The aim is to predict the species present at a certain location based on these variables. Because the number of species greatly varies depending on the position, average- K accuracy is a natural choice of metric for this dataset. Based on an estimation of the average number of species per location, average-30 accuracy was set as the official metric. Top-30 accuracy is kept as a secondary metric to compare to. A challenge using this data was organized as part of LifeCLEF2020 (Joly et al., 2020b). The results are shown in Table 6.1 – more details can be found in the overview of the challenge (Deneu et al., 2020).

Participant	Submission #	Average-30 accuracy	Top-30 accuracy
LIRMM	Submission 3	23.3%	23.5%
LIRMM	Submission 1	21.3%	20.4%
Stanford	Submission 3	4.8%	4.8%

Table 6.1: *GeoLifeCLEF 2020 main results: average-30 accuracy and top-30 accuracy per submission (sorted by decreasing average-30 accuracy).*

In the results provided by the participants, there is no significant difference between average-30 and top-30 accuracy. As the heterogeneity of the ambiguity is obvious in this task, the fact that the gap is narrow is likely due to an estimation issue. The winning submission consisted of a neural network with an architecture adapted to the structure of the data trained using a traditional procedure, i.e. using negative log-likelihood in a similar manner than proposed in [Chapter 4](#). However, when contacted, the participants notified us that the negative log-likelihood only very marginally decreased on the validation through the training process, staying very close to its initial value. This observation might explain the absence of gap between average-30 and top-30 accuracy. It must however be noted that average-30 accuracy was still as high as top-30 accuracy, and never worst.

This “failure” case is interesting: analyzing it more carefully might shed light on some limitations of average- K and could give insights on how to overcome them. This example suggests that the specificities of the data and/or the modification of the architecture could generate different training dynamics than usually observed when training neural networks for image classification tasks. This task might thus benefit from using a specialized training procedure targetted specifically for average- K classification.

6.2.3 Pl@ntNet project

Understanding and improving the returned species list

The list of candidate species returned by the Pl@ntNet app to the users is generated using a heuristic which was found by trial and error. It corresponds to the intersection of the thresholding and the cumulative sum strategies:

$$\Gamma(\mathbf{x}) = \{k \in \mathcal{Y} \mid \eta_k(\mathbf{x}) \geq \lambda\} \cap \text{top}_\eta(\mathbf{x}, k_\epsilon(\mathbf{x})).$$

The values of λ and ϵ were chosen empirically and set to $\lambda = 0.005$ and $\epsilon = 0.01$.

This hybrid set-valued classification formulation corresponds to a relaxation of the point-wise error control framework detailed in [Chapter 5](#). Here the point-wise error rate constraint can be violated locally if it requires predicting classes with a very low probability which results in a big increase of the average set size. It thus mitigates one of the limitations of point-wise error control and was found to be a good balance in practice.

Further analysis of this formulation would be interesting in order to understand what are the exact connections between the parameters λ and ϵ and the size and error rate of predicted sets. This could allow to automatically set the values of these parameters according to explicit constraints on those quantities. Moreover, a comparison with other frameworks presented in [Chapter 5](#) would allow to understand if this formulation is indeed the best suited for the Pl@ntNet app. Further analysis on how to improve the estimation of this set-valued classifier would also have a big impact on the quality of the returned results. In particular, it would

be interesting to see if improving the calibration of the scores outputted by the models would enhance these results. Furthermore, it might be relevant to design specialized losses targetted for this set-valued formulation.

Dataset querying by stakeholders

The previous discussion is focused on answering the queries of users of the Pl@ntNet app. At the other end of the pipeline, stakeholders want to extract data adapted to their needs from the Pl@ntNet database. Typically, their requirements are very different from the ones of the users of the app. Moreover, among the different stakeholders, these requirements also differ. Using a global approach saving the scores predicted by the models from which we could apply any set-valued formulation depending on the query is thus the most adapted.

This leaves a last very important question: what is the influence of the set-valued classification strategy and of the filtering process on the studies and the decisions made by the stakeholders? An analysis of the potential biases introduced by them and a study of the sensitivity of the conclusions drawn from data extracted this way is important to understand how they should be used in practice.

Bibliography

- Abe, N. (1998). Query learning strategies using boosting and bagging. *International Conference on Machine Learning*, pages 1–9.
- Affouard, A., Goëau, H., Bonnet, P., Lombardo, J.-C., and Joly, A. (2017). Pl@ntnet app in the era of deep learning. In *ICLR (Workshop)*.
- Agarwal, S. (2014). Surrogate regret bounds for bipartite ranking via strongly proper losses. *The Journal of Machine Learning Research*, 15(1):1653–1674.
- Angermueller, C., Pärnamaa, T., Parts, L., and Stegle, O. (2016). Deep learning for computational biology. *Molecular systems biology*, 12(7):878.
- Ashukha, A., Lyzhov, A., Molchanov, D., and Vetrov, D. (2020). Pitfalls of in-domain uncertainty estimation and ensembling in deep learning. In *International Conference on Learning Representations*.
- Baldi, P., Sadowski, P., and Whiteson, D. (2014). Searching for exotic particles in high-energy physics with deep learning. *Nature communications*, 5(1):1–9.
- Barber, R. F., Candes, E. J., Ramdas, A., and Tibshirani, R. J. (2019). The limits of distribution-free conditional predictive inference. *Information and Inference: A Journal of the IMA*.
- Bartlett, P. L. (1997). For valid generalization the size of the weights is more important than the size of the network. In *Advances in neural information processing systems*, pages 134–140.
- Bartlett, P. L., Bousquet, O., and Mendelson, S. (2005). Local Rademacher complexities. *The Annals of Statistics*, 33(4):1497–1537.
- Bartlett, P. L., Foster, D. J., and Telgarsky, M. J. (2017). Spectrally-normalized margin bounds for neural networks. In *Advances in Neural Information Processing Systems*, pages 6240–6249.
- Bartlett, P. L., Harvey, N., Liaw, C., and Mehrabian, A. (2019). Nearly-tight VC-dimension and pseudodimension bounds for piecewise linear neural networks. *J. Mach. Learn. Res.*, 20:63–1.
- Bartlett, P. L., Jordan, M. I., and McAuliffe, J. D. (2006). Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156.
- Bartlett, P. L. and Mendelson, S. (2002). Rademacher and Gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482.
- Bartlett, P. L. and Wegkamp, M. H. (2008). Classification with a reject option using a hinge loss. *Journal of Machine Learning Research*, 9(Aug):1823–1840.
- Bendale, A. and Boulton, T. E. (2016). Towards open set deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1563–1572.

- Bengio, Y., Lamblin, P., Popovici, D., and Larochelle, H. (2007). Greedy layer-wise training of deep networks. In *Advances in neural information processing systems*, pages 153–160.
- Bergen, K. J., Johnson, P. A., Maarten, V., and Beroza, G. C. (2019). Machine learning for data-driven discovery in solid earth geoscience. *Science*, 363(6433).
- Berrada, L., Zisserman, A., and Kumar, M. P. (2018). Smooth loss functions for deep top-k classification. In *International Conference on Learning Representations*.
- Blondel, M., Teboul, O., Berthet, Q., and Djolonga, J. (2020). Fast differentiable sorting and ranking. In *International Conference on Machine Learning*.
- Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. (2015). Weight uncertainty in neural networks. *arXiv preprint arXiv:1505.05424*.
- Bonnet, P., Joly, A., Faton, J.-M., Brown, S., Kimiti, D., Deneu, B., Servajean, M., Affouard, A., Lombardo, J.-C., Mary, L., et al. (2020). How citizen scientists contribute to monitor protected areas thanks to automatic plant identification tools. *Ecological Solutions and Evidence*, 1(2):e12023.
- Botella, C., Joly, A., Bonnet, P., Monestiez, P., and Munoz, F. (2018). Species distribution modeling based on the automated identification of citizen observations. *Applications in Plant Sciences*, 6(2):e1029.
- Bottou, L., Curtis, F. E., and Nocedal, J. (2018). Optimization methods for large-scale machine learning. *Siam Review*, 60(2):223–311.
- Boucheron, S., Bousquet, O., and Lugosi, G. (2005). Theory of classification: A survey of some recent advances. *ESAIM: probability and statistics*, 9:323–375.
- Boucheron, S., Lugosi, G., and Massart, P. (2013). *Concentration inequalities: A nonasymptotic theory of independence*. Oxford university press.
- Bulusu, S., Kailkhura, B., Li, B., Varshney, P. K., and Song, D. (2020). Anomalous example detection in deep learning: A survey. *IEEE Access*, 8:132330–132347.
- Cai, T. T., Low, M., and Ma, Z. (2014). Adaptive Confidence Bands for Nonparametric Regression Functions. *Journal of the American Statistical Association*, 109(507):1054–1070.
- Cardon, D., Cointet, J.-P., Mazières, A., and Libbrecht, E. (2018). Neurons spike back. *Réseaux*.
- Cauchois, M., Gupta, S., and Duchi, J. (2020). Knowing what you know: valid confidence sets in multiclass and multilabel prediction. *arXiv preprint arXiv:2004.10181*.
- Chalapathy, R. and Chawla, S. (2019). Deep learning for anomaly detection: A survey. *arXiv preprint arXiv:1901.03407*.
- Chandola, V., Banerjee, A., and Kumar, V. (2009). Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):1–58.
- Chaudhari, P., Choromanska, A., Soatto, S., LeCun, Y., Baldassi, C., Borgs, C., Chayes, J., Sagun, L., and Zecchina, R. (2019). Entropy-SGD: Biasing gradient descent into wide valleys. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12):124018.
- Chow, C. (1970). On optimum recognition error and reject tradeoff. *IEEE Transactions on information theory*, 16(1):41–46.

- Chow, C.-K. (1957). An optimum character recognition system using decision functions. *IRE Transactions on Electronic Computers*, pages 247–254.
- Christin, S., Hervet, É., and Lecomte, N. (2019). Applications for deep learning in ecology. *Methods in Ecology and Evolution*, 10(10):1632–1644.
- Chzhen, E., Denis, C., and Hebiri, M. (2019). Minimax semi-supervised confidence sets for multi-class classification. *arXiv preprint arXiv:1904.12527*.
- Chzhen, E., Denis, C., Hebiri, M., and Lorieul, T. (2020). Set-valued classification—overview via a unified framework.
- Ciregan, D., Meier, U., and Schmidhuber, J. (2012). Multi-column deep neural networks for image classification. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3642–3649. IEEE.
- Cléménçon, S., Lugosi, G., Vayatis, N., et al. (2008). Ranking and empirical minimization of u-statistics. *The Annals of Statistics*, 36(2):844–874.
- Cohn, D., Atlas, L., and Ladner, R. (1994). Improving generalization with active learning. *Machine learning*, 15(2):201–221.
- Cole, E., Deneu, B., Lorieul, T., Servajean, M., Botella, C., Morris, D., Jojic, N., Bonnet, P., and Joly, A. (2020). The GeoLifeCLEF 2020 dataset. *arXiv preprint arXiv:2004.04192*.
- Cortes, C., DeSalvo, G., and Mohri, M. (2016). Learning with rejection. In *International Conference on Algorithmic Learning Theory*, pages 67–82. Springer.
- Cristianini, N., Shawe-Taylor, J., et al. (2000). *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press.
- Csurka, G., Dance, C., Fan, L., Willamowski, J., and Bray, C. (2004). Visual categorization with bags of keypoints. In *Workshop on statistical learning in computer vision, ECCV*, volume 1, pages 1–2. Prague.
- Cuturi, M., Teboul, O., and Vert, J.-P. (2019). Differentiable ranking and sorting using optimal transport. In *Advances in Neural Information Processing Systems*.
- Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Conference on computer vision and pattern recognition*, volume 1, pages 886–893. IEEE.
- de Bezenac, E., Pajot, A., and Gallinari, P. (2019). Deep learning for physical processes: Incorporating prior scientific knowledge. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12):124009.
- Del Coz, J. J., Díez, J., and Bahamonde, A. (2009). Learning nondeterministic classifiers. *Journal of Machine Learning Research*, 10(10).
- Dembczyński, K., Waegeman, W., Cheng, W., and Hüllermeier, E. (2012). On label dependence and loss minimization in multi-label classification. *Machine Learning*, 88(1-2):5–45.
- Deneu, B., Lorieul, T., Cole, E., Servajean, M., Botella, C., Morris, D., Jojic, N., Bonnet, P., and Joly, A. (2020). Overview of LifeCLEF location-based species prediction task 2020 (GeoLifeCLEF). *CLEF task overview*.
- Denis, C. and Hebiri, M. (2017). Confidence sets with expected sizes for multiclass classification. *Journal of Machine Learning Research*, 18(102):1–28.

- Denis, C. and Hebiri, M. (2020). Consistency of plug-in confidence sets for classification in semi-supervised learning. *Journal of Nonparametric Statistics*, 32(1):42–72.
- Der Kiureghian, A. and Ditlevsen, O. (2009). Aleatory or epistemic? does it matter? *Structural safety*, 31(2):105–112.
- DeVries, T. and Taylor, G. W. (2018). Learning confidence for out-of-distribution detection in neural networks. *arXiv preprint arXiv:1802.04865*.
- Dinh, L., Pascanu, R., Bengio, S., and Bengio, Y. (2017). Sharp minima can generalize for deep nets. In *International Conference on Machine Learning*, pages 1019–1028.
- Duchi, J., Hazan, E., and Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7).
- El-Yaniv, R. and Wiener, Y. (2010). On the foundations of noise-free selective classification. *Journal of Machine Learning Research*, 11(May):1605–1641.
- El-Yaniv, R. and Wiener, Y. (2012). Active learning via perfect selective classification. *The Journal of Machine Learning Research*, 13(1):255–279.
- Elkan, C. (2001). The foundations of cost-sensitive learning. In *International joint conference on artificial intelligence*.
- Flach, P. A. (2017). *Classifier Calibration*, pages 210–217. Springer.
- Frénay, B. and Verleysen, M. (2013). Classification in the presence of label noise: a survey. *IEEE transactions on neural networks and learning systems*, 25(5):845–869.
- Freund, Y., Mansour, Y., Schapire, R. E., et al. (2004). Generalization bounds for averaged classifiers. *The annals of statistics*, 32(4):1698–1722.
- Freund, Y., Seung, H. S., Shamir, E., and Tishby, N. (1997). Selective sampling using the query by committee algorithm. *Machine learning*, 28(2-3):133–168.
- Gal, Y. (2016). *Uncertainty in deep learning*. PhD thesis, University of Cambridge.
- Gal, Y. and Ghahramani, Z. (2016). Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning*, pages 1050–1059.
- Gelbart, R. and El-Yaniv, R. (2019). The relationship between agnostic selective classification, active learning and the disagreement coefficient. *J. Mach. Learn. Res.*, 20:33–1.
- Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, A., and Rubin, D. B. (2013). *Bayesian data analysis*. CRC press.
- Geng, C., Huang, S.-j., and Chen, S. (2020). Recent advances in open set recognition: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
- Glorot, X. and Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In *International Conference on Artificial Intelligence and Statistics*, pages 249–256.
- Glorot, X., Bordes, A., and Bengio, Y. (2011). Deep sparse rectifier neural networks. In *International Conference on Artificial Intelligence and Statistics*, pages 315–323.
- Gneiting, T. and Raftery, A. E. (2007). Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378.

- Göeau, H., Joly, A., and Bonnet, P. (2015). LifeCLEF plant identification task 2015. *CLEF working notes*, 2015.
- Goh, G. B., Hodas, N. O., and Vishnu, A. (2017). Deep learning for computational chemistry. *Journal of computational chemistry*, 38(16):1291–1307.
- Golowich, N., Rakhlin, A., and Shamir, O. (2019). Size-independent sample complexity of neural networks. *arXiv preprint arXiv:1712.06541*.
- Goodfellow, I., Bengio, Y., Courville, A., and Bengio, Y. (2016). *Deep learning*, volume 1. MIT press Cambridge.
- Gordon, D. F. and Desjardins, M. (1995). Evaluation and selection of biases in machine learning. *Machine learning*, 20(1-2):5–22.
- Greenspan, H., Van Ginneken, B., and Summers, R. M. (2016). Guest editorial deep learning in medical imaging: Overview and future promise of an exciting new technique. *IEEE Transactions on Medical Imaging*, 35(5):1153–1159.
- Gribonval, R., Kutyniok, G., Nielsen, M., and Voigtlaender, F. (2019). Approximation spaces of deep neural networks. *arXiv preprint arXiv:1905.01208*.
- Grycko, E. (1993). Classification with set-valued decision functions. In Opitz, O., Lausen, B., and Klar, R., editors, *Information and Classification*, pages 218–224, Berlin, Heidelberg. Springer Berlin Heidelberg.
- Gühring, I., Raslan, M., and Kutyniok, G. (2020). Expressivity of deep neural networks. *arXiv preprint arXiv:2007.04759*.
- Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. (2017). On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1321–1330. JMLR. org.
- Ha, T. M. (1996). An optimum class-selective rejection rule for pattern recognition. In *Proceedings of 13th International Conference on Pattern Recognition*, volume 2, pages 75–80. IEEE.
- Ha, T. M. (1997a). The optimum class-selective rejection rule. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(6):608–615.
- Ha, T. M. (1997b). Optimum tradeoff between class-selective rejection error and average number of classes. *Engineering Applications of Artificial Intelligence*, 10(6):525–529.
- Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media.
- He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.
- Hendrycks, D. and Gimpel, K. (2017). A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *International Conference on Learning Representations*.
- Herbei, R. and Wegkamp, M. H. (2006). Classification with reject option. *Canadian Journal of Statistics*, 34(4):709–721.
- Hochreiter, S. and Schmidhuber, J. (1997). Flat minima. *Neural Computation*, 9(1):1–42.

- Hoffer, E., Hubara, I., and Soudry, D. (2017). Train longer, generalize better: closing the generalization gap in large batch training of neural networks. In *Advances in Neural Information Processing Systems*, pages 1731–1741.
- Hornik, K., Stinchcombe, M., and White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366.
- Hüllermeier, E. and Waegeman, W. (2019). Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods.
- Joly, A., Bonnet, P., Goëau, H., Barbe, J., Selmi, S., Champ, J., Dufour-Kowalski, S., Affouard, A., Carré, J., Molino, J.-F., et al. (2016). A look inside the Pl@ntNet experience. *Multimedia Systems*, 22(6):751–766.
- Joly, A., Goëau, H., Kahl, S., Botella, C., De Castaneda, R. R., Glotin, H., Cole, E., Champ, J., Deneu, B., Servajean, M., Lorieul, T., Vellinga, W.-P., Stöter, F.-R., Durso, A., Bonnet, P., and Müller, H. (2020a). LifeCLEF 2020 teaser: Biodiversity identification and prediction challenges. In *European Conference on Information Retrieval*, pages 542–549. Springer.
- Joly, A., Goëau, H., Kahl, S., Deneu, B., Servajean, M., Cole, E., Picek, L., Ruiz de Castañeda, R., Bolon, I., Durso, A., Lorieul, T., Botella, C., Glotin, H., Champ, J., Eggel, I., Vellinga, W.-P., Bonnet, P., and Müller, H. (2020b). Overview of LifeCLEF 2020: A system-oriented evaluation of automated species identification and species distribution prediction. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction*, pages 342–363. Springer International Publishing.
- Jordan, M. I. and Mitchell, T. M. (2015). Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245):255–260.
- Kawaguchi, K. (2016). Deep learning without poor local minima. In *Advances in neural information processing systems*, pages 586–594.
- Kendall, A. and Gal, Y. (2017). What uncertainties do we need in Bayesian deep learning for computer vision? In *Advances in neural information processing systems*, pages 5574–5584.
- Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M., and Tang, P. T. P. (2017). On large-batch training for deep learning: Generalization gap and sharp minima. In *International conference on machine learning*.
- Kingma, D. P. and Welling, M. (2014). Auto-encoding variational Bayes. In *International Conference on Learning Representations, ICLR*.
- Koltchinskii, V. (2006). Local Rademacher complexities and oracle inequalities in risk minimization. *The Annals of Statistics*, 34(6):2593–2656.
- Koltchinskii, V. and Panchenko, D. (2002). Empirical margin distributions and bounding the generalization error of combined classifiers. *The Annals of Statistics*, 30(1):1–50.
- Kononenko, I. (2001). Machine learning for medical diagnosis: history, state of the art and perspective. *Artificial Intelligence in Medicine*, 23(1):89 – 109.
- Krizhevsky, A. and Hinton, G. (2009). Learning multiple layers of features from tiny images. Technical report, Citeseer.
- Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

- Lapin, M., Hein, M., and Schiele, B. (2015). Top-k multiclass svm. In *Advances in Neural Information Processing Systems*, pages 325–333.
- Lapin, M., Hein, M., and Schiele, B. (2016). Loss functions for top-k error: Analysis and insights. In *CVPR*.
- Lapin, M., Hein, M., and Schiele, B. (2017). Analysis and optimization of loss functions for multiclass, top-k, and multilabel classification. *IEEE transactions on pattern analysis and machine intelligence*, 40(7):1533–1554.
- Le Capitaine, H. (2014). A unified view of class-selection with probabilistic classifiers. *Pattern recognition*, 47(2):843–853.
- LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *nature*, 521(7553):436–444.
- LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.
- Lei, J. (2014). Classification with confidence. *Biometrika*.
- Lei, J., G’Sell, M., Rinaldo, A., Tibshirani, R., and Wasserman, L. (2018). Distribution-free predictive inference for regression. *J. Amer. Statist. Assoc.*, 113(523):1094–1111.
- Lei, J. and Wasserman, L. (2014). Distribution-free prediction bands for non-parametric regression. *Journal of the Royal Statistical Society Series B*, 76(1):71–96.
- Li, H., Xu, Z., Taylor, G., Studer, C., and Goldstein, T. (2018). Visualizing the loss landscape of neural nets. In *Advances in Neural Information Processing Systems*, pages 6389–6399.
- Liang, S., Li, Y., and Srikant, R. (2018). Enhancing the reliability of out-of-distribution image detection in neural networks. In *International Conference on Learning Representations*.
- Liang, T., Poggio, T., Rakhlin, A., and Stokes, J. (2019). Fisher-Rao metric, geometry, and complexity of neural networks. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 888–896.
- Lloyd, S. (1982). Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137.
- Lorieul, T. and Joly, A. (2019). Vers un désenchevêtrement de l’ambiguïté de la tâche et de l’incertitude du modèle pour la classification avec option de rejet à l’aide de réseaux neuronaux. In *Conférence sur l’Apprentissage automatique (CAp)*.
- Lorieul, T. and Joly, A. (2020). Leveraging uncertainty information to reject predictions made by a previously trained classifier.
- Lorieul, T., Joly, A., and Shasha, D. (2020). Average-K classification: when and how to predict adaptive confidence sets rather than top-K.
- Lorieul, T., Pearson, K. D., Ellwood, E. R., Goëau, H., Molino, J.-F., Sweeney, P. W., Yost, J. M., Sachs, J., Mata-Montero, E., Nelson, G., Soltis, P. S., Bonnet, P., and Joly, A. (2019). Toward a large-scale and deep phenological stage annotation of herbarium specimens: Case studies from temperate, tropical, and equatorial floras. *Applications in Plant Sciences*, 7(3):e01233.
- Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *Proceedings of the international conference on computer vision*, volume 2, pages 1150–1157. Ieee.

- Mac Aodha, O., Cole, E., and Perona, P. (2019). Presence-only geographical priors for fine-grained image classification. In *International Conference on Computer Vision*, pages 9596–9606.
- MacKay, D. J. (1992). A practical Bayesian framework for backpropagation networks. *Neural computation*, 4(3):448–472.
- Mater, A. C. and Coote, M. L. (2019). Deep learning in chemistry. *Journal of chemical information and modeling*, 59(6):2545–2559.
- Menon, A. K. and Williamson, R. C. (2016). Bipartite ranking: a risk-theoretic perspective. *The Journal of Machine Learning Research*, 17(1):6766–6867.
- Menzel, A., Sparks, T. H., Estrella, N., Koch, E., Aasa, A., Ahas, R., Alm-Kübler, K., Bissolli, P., Braslavská, O., Briede, A., Chmielewski, F. M., Crepinsek, Z., Curnel, Y., Dahl, A., Defila, C., Donnelly, A., Filella, Y., Jatzcak, K., Mage, F., Mestre, A., Nordli, O., Peñuelas, J., Pirinen, P., Remišová, V., Scheifinger, H., Striz, M., Susnik, A., Van Vliet, A. J. H., Wielgolaski, F.-E., Zach, S., and Zust, A. (2006). European phenological response to climate change matches the warming pattern. *Global Change Biology*, 12(10):1969–1976.
- Minsky, M. and Papert, S. A. (1969). *Perceptrons: An introduction to computational geometry*. MIT Press.
- Mitchell, T. M. (1980). The need for biases in learning generalizations. Technical report, Rutgers University, Computer Science Department.
- Montufar, G. F., Pascanu, R., Cho, K., and Bengio, Y. (2014). On the number of linear regions of deep neural networks. In *Advances in neural information processing systems*, pages 2924–2932.
- Mortier, T., Wydmuch, M., Hüllermeier, E., Dembczyński, K., and Waegeman, W. (2019). Efficient algorithms for set-valued prediction in multi-class classification. *arXiv preprint arXiv:1906.08129*.
- Nadeem, M. S. A., Zucker, J.-D., and Hanczar, B. (2009). Accuracy-rejection curves (ARCs) for comparing classification methods with a reject option. In *Machine Learning in Systems Biology*, pages 65–81.
- Neal, R. M. (1996). *Bayesian learning for neural networks*. Springer Science & Business Media.
- Neyshabur, B., Salakhutdinov, R. R., and Srebro, N. (2015a). Path-SGD: Path-normalized optimization in deep neural networks. In *Advances in Neural Information Processing Systems*, pages 2422–2430.
- Neyshabur, B., Tomioka, R., and Srebro, N. (2015b). Norm-based capacity control in neural networks. In *Conference on Learning Theory*, pages 1376–1401.
- Ng, A. Y. (2004). Feature selection, l_1 vs. l_2 regularization, and rotational invariance. In *Proceedings of the twenty-first international conference on Machine learning*, page 78.
- Ni, C., Charoenphakdee, N., Honda, J., and Sugiyama, M. (2019). On the calibration of multiclass classification with rejection. In *Advances in Neural Information Processing Systems*, pages 2586–2596.
- Ojala, T., Pietikainen, M., and Harwood, D. (1994). Performance evaluation of texture measures with classification based on kullback discrimination of distributions. In *Proceedings of 12th International Conference on Pattern Recognition*, volume 1, pages 582–585. IEEE.
- Pearson, K. D. (2019). A new method and insights for estimating phenological events from herbarium specimens. *Applications in Plant Sciences*, 7(3):e01224.

- Pearson, K. D., Nelson, G., Aronson, M. F. J., Bonnet, P., Brenskelle, L., Davis, C. C., Denny, E. G., Ellwood, E. R., Goëau, H., Heberling, J. M., Joly, A., Lorieul, T., Mazer, S. J., Meineke, E. K., Stucky, B. J., Sweeney, P., White, A. E., and Soltis, P. S. (2020). Machine learning using digitized herbarium specimens to advance phenological research. *BioScience*, 70(6):610–620.
- Peterson, J. C., Battleday, R. M., Griffiths, T. L., and Russakovsky, O. (2019). Human uncertainty makes classification more robust. In *International Conference on Computer Vision (ICCV)*.
- Pimentel, M. A., Clifton, D. A., Clifton, L., and Tarassenko, L. (2014). A review of novelty detection. *Signal Processing*, 99:215–249.
- Platt, J. et al. (1999). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74.
- Poggio, T., Mhaskar, H., Rosasco, L., Miranda, B., and Liao, Q. (2017). Why and when can deep-but not shallow-networks avoid the curse of dimensionality: a review. *International Journal of Automation and Computing*, 14(5):503–519.
- Ramaswamy, H. G., Tewari, A., Agarwal, S., et al. (2018). Consistent algorithms for multiclass classification with an abstain option. *Electronic Journal of Statistics*, 12(1):530–554.
- Reichstein, M., Camps-Valls, G., Stevens, B., Jung, M., Denzler, J., Carvalhais, N., et al. (2019). Deep learning and process understanding for data-driven earth system science. *Nature*, 566(7743):195–204.
- Reid, M. D. and Williamson, R. C. (2009). Surrogate regret bounds for proper losses. In *International Conference on Machine Learning*, pages 897–904.
- Reiss, R.-D. (1989). *Approximate distributions of order statistics: with applications to nonparametric statistics*. Springer science & Business media.
- Rezende, D. J., Mohamed, S., and Wierstra, D. (2014). Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, pages 1278–1286.
- Robbins, H. and Monro, S. (1951). A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407.
- Romano, Y., Sesia, M., and Candès, E. (2020). Classification with valid and adaptive coverage. *arXiv preprint arXiv:2006.02544*.
- Rosenblatt, F. (1957). *The perceptron, a perceiving and recognizing automaton Project Para*. Cornell Aeronautical Laboratory.
- Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. (2015). ImageNet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252.
- Sadinle, M., Lei, J., and Wasserman, L. (2019). Least ambiguous set-valued classifiers with bounded error levels. *Journal of the American Statistical Association*, 114(525):223–234.
- Sánchez, J., Perronnin, F., Mensink, T., and Verbeek, J. (2013). Image classification with the Fisher vector: Theory and practice. *International journal of computer vision*, 105(3):222–245.
- Scheirer, W. J., de Rezende Rocha, A., Sapkota, A., and Boulton, T. E. (2012). Toward open set recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(7):1757–1772.

- Servajean, M., Joly, A., Shasha, D., Champ, J., and Pacitti, E. (2016). ThePlantGame: Actively training human annotators for domain-specific crowdsourcing. In *Proceedings of the 24th ACM international conference on Multimedia*, pages 720–721.
- Settles, B. (2012). *Active learning*. Morgan & Claypool.
- Seung, H. S., Oppor, M., and Sompolinsky, H. (1992). Query by committee. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 287–294.
- Shalev-Shwartz, S. and Ben-David, S. (2014). *Understanding machine learning: From theory to algorithms*. Cambridge university press.
- Shen, D., Wu, G., and Suk, H.-I. (2017). Deep learning in medical image analysis. *Annual review of biomedical engineering*, 19:221–248.
- Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations, ICLR 2015*.
- Soudry, D. and Carmon, Y. (2016). No bad local minima: Data independent training error guarantees for multilayer neural networks. *arXiv preprint arXiv:1605.08361*.
- Sutskever, I., Martens, J., Dahl, G., and Hinton, G. (2013). On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147.
- Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.
- Tewari, A. and Bartlett, P. L. (2007). On the consistency of multiclass classification methods. *Journal of Machine Learning Research*, 8(May):1007–1025.
- Tieleman, T. and Hinton, G. (2012). Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning.
- Tikhonov, A. N. (1943). On the stability of inverse problems. In *Dokl. Akad. Nauk SSSR*, volume 39, pages 195–198.
- Tsybakov, A. B. et al. (2004). Optimal aggregation of classifiers in statistical learning. *The Annals of Statistics*, 32(1):135–166.
- Vapnik, V. N. and Chervonenkis, A. Y. (1971). On the uniform convergence of relative frequencies of events to their probabilities. In *Measures of complexity*, pages 11–30. Springer.
- Vernet, E., Reid, M. D., and Williamson, R. C. (2011). Composite multiclass losses. In *Advances in Neural Information Processing Systems*, pages 1224–1232.
- Vilalta, R. and Drissi, Y. (2002). A perspective view and survey of meta-learning. *Artificial intelligence review*, 18(2):77–95.
- Vovk, V. (2013). Conditional validity of inductive conformal predictors. *Mach. Learn.*, 92(2-3):349–376.
- Vovk, V., Gammerman, A., and Shafer, G. (2005). *Algorithmic learning in a random world*. Springer Science & Business Media.
- Wenzel, F., Roth, K., Veeling, B. S., Świątkowski, J., Tran, L., Mandt, S., Snoek, J., Salimans, T., Jenatton, R., and Nowozin, S. (2020). How good is the Bayes posterior in deep neural networks really? *arXiv preprint arXiv:2002.02405*.

- Werner, T. (2019). A review on ranking problems in statistical learning. *arXiv preprint arXiv:1909.02998*.
- Wiener, Y. and El-Yaniv, R. (2011). Agnostic selective classification. In *Advances in neural information processing systems*, pages 1665–1673.
- Wiener, Y. and El-Yaniv, R. (2015). Agnostic pointwise-competitive selective classification. *Journal of Artificial Intelligence Research*, 52:171–201.
- Wilson, A. C., Roelofs, R., Stern, M., Srebro, N., and Recht, B. (2017). The marginal value of adaptive gradient methods in machine learning. In *Advances in neural information processing systems*, pages 4148–4158.
- Wolpert, D. H. (1996). The lack of a priori distinctions between learning algorithms. *Neural computation*, 8(7):1341–1390.
- Xiao, H., Rasul, K., and Vollgraf, R. (2017). Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*.
- Xie, S., Girshick, R., Dollár, P., Tu, Z., and He, K. (2017). Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500.
- Yang, F. and Koyejo, S. (2020). On the consistency of top-k surrogate losses. In *International Conference on Machine Learning*.
- Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. (2017). Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations (ICLR)*.
- Zhang, M.-L. and Zhou, Z.-H. (2013). A review on multi-label learning algorithms. *IEEE transactions on knowledge and data engineering*, 26(8):1819–1837.