Learning with Partially Labeled Data for Multi-class Classification and Feature Selection

Vasilii Feofanov

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITE GRENOBLE ALPES

Spécialité : **Mathématiques Appliquées**

Arrêté ministériel : 25 mai 2016

Présentée par

Vasilii FEOFANOV

Thèse dirigée par **Massih-Reza AMINI**, Professeur, Université Grenoble Alpes
et co-encadrée par **Emilie DEVIJVER**, CR, Université Grenoble Alpes

préparée au sein du **Laboratoire d'Informatique de Grenoble**
dans **l'École Doctorale Mathématiques, Sciences et technologies de l'information, Informatique**

Classification Multi-classe et Sélection de Variables avec des Données Partiellement Étiquetées

Learning with Partially Labeled Data for Multi-class Classification and Feature Selection

Thèse soutenue publiquement le **29 septembre 2021**,
devant le jury composé de :

Madame Florence d'Alché-Buc
Professeure, Télécom Paris, Rapporteure
Monsieur Massih-Reza Amini
Professeur, Université Grenoble Alpes, Directeur de thèse
Monsieur Laurent Besacier
Chercheur HDR, NAVER LABS Europe, Examinateur
Madame Emilie Devijver
Chercheure, CNRS, Co-encadrante de thèse
Madame Mélina Gallopin
Maître de conférence, Université Paris Sud, Examinatrice
Monsieur Pascal Germain
Professeur adjoint, Université Laval à Québec, Rapporteur
Monsieur Anatoli Iouditski
Professeur, Université Grenoble Alpes, Président

Abstract

Learning with partially labeled data, known as semi-supervised learning, deals with problems where few training examples are labeled while available unlabeled data are abundant and valuable for training. In this thesis, we study this framework in the multi-class classification case with a focus on self-learning and feature selection. Self-learning is a classical approach that iteratively assigns pseudo-labels to unlabeled training examples with a confidence score above a predetermined threshold. This pseudo-labeling technique is prone to error and runs the risk of adding noisy labels into unlabeled training data. Our first contribution is to propose a theoretical framework for analyzing self-learning in the multi-class case. We derive a transductive bound over the risk of the multi-class majority vote classifier and propose to use this bound for automatically choosing the pseudo-labeling threshold. Then, we introduce a mislabeling error model to analyze the error of the majority vote classifier in the case of the pseudo-labeled data. We derive a probabilistic second-order bound over the majority vote error given an imperfect label. Our second contribution is an extension of the self-learning strategy to the case where some unlabeled examples come from classes not previously seen. The new approach is applied for classification of real biological data, and it is based on assuming the existence of clusters in unlabeled data. Finally, we propose an approach for semi-supervised feature selection that utilizes self-learning to increase the variety of training data and a new modification of the genetic algorithm to perform a feature subset search. The proposed genetic algorithm produces both a sparse and accurate solution by considering feature weights during its evolutionary process and iteratively removing irrelevant features.

Résumé

L'apprentissage avec des données partiellement étiquetées, connu sous le nom d'apprentissage semi-supervisé, traite des problèmes où peu d'exemples d'entraînement sont étiquetés alors que les données disponibles non étiquetées sont abondantes et précieuses pour l'apprentissage. Dans cette thèse, nous étudions ce cadre dans le cas de la classification multi-classes en mettant l'accent sur l'auto-apprentissage et la sélection de variables. L'auto-apprentissage est une approche classique qui attribue de manière itérative des pseudo-étiquettes à des exemples d'entraînement non étiquetés avec un score de confiance supérieur à un seuil prédéterminé. Cette technique de pseudo-étiquetage est sujette aux erreurs et risque d'ajouter des étiquettes bruitées dans des données d'apprentissage non étiquetées. Notre première contribution est de proposer un cadre théorique d'analyse de l'auto-apprentissage dans le cas multi-classes. Nous dérivons une borne transductive sur le risque du classificateur de vote majoritaire multi-classes et proposons d'utiliser cette borne pour choisir automatiquement le seuil de pseudo-étiquetage. Ensuite, nous introduisons un modèle d'erreur d'étiquetage pour analyser l'erreur du classificateur de vote majoritaire dans le cas des données pseudo-étiquetées. Nous dérivons une borne probabiliste de second ordre sur l'erreur de vote majoritaire étant donné une étiquette imparfaite. Notre deuxième contribution est une extension de la stratégie d'auto-apprentissage au cas où certains exemples non étiquetés proviennent de classes jamais vues auparavant. La nouvelle approche est appliquée pour la classification de données biologiques réelles, et elle est basée sur l'hypothèse de l'existence de clusters dans des données non étiquetées. Enfin, nous proposons une approche de sélection de variables semi-supervisée qui utilise l'auto-apprentissage pour augmenter la variété des données d'entraînement et une nouvelle modification de l'algorithme génétique pour effectuer une recherche de sous-ensembles de variables. L'algorithme génétique proposé produit à la fois une solution clairsemée et précise en tenant compte des pondérations des variables au cours de son processus évolutif et en supprimant de manière itérative les variables non pertinentes.

Contents

Introduction **1**

1 Background **5**
1.1 Supervised Learning 5
1.1.1 Single Hypothesis Learning 7
1.1.2 Ensemble Learning 8
1.1.3 Analysis of Majority Vote Classifiers 11
1.2 Semi-supervised Learning 14
1.2.1 Existing Approaches 15
1.2.2 Transductive Bounds for the Majority Vote 19

2 Probabilistic Bounds for the Multi-class Majority Vote Classifier **23**
2.1 Introduction 23
2.2 Framework and Definitions 24
2.3 Probabilistic Transductive Bounds 26
2.3.1 Transductive conditional risk 26
2.3.2 Transductive confusion matrix and transductive error rate . . 29
2.3.3 Tightness Guarantees 31
2.4 Probabilistic C-Bound with Imperfect Labels 33
2.4.1 Ordinary C-Bound 34
2.4.2 Mislabeling Error Model 35
2.4.3 C-Bounds with Imperfect Labels 38
2.4.4 PAC-Bayesian Theorem for C-Bound Estimation 40
2.4.5 Empirical Illustration of (CBIL) 42
2.5 Conclusion and Perspectives 44
2.6 Appendix 45
2.6.1 Mathematical Tools for Section 2.3 45
2.6.2 Mathematical Tools for Section 2.4 49

3 Self-learning and Application **55**
3.1 Multi-class Self-learning Algorithm 55
3.1.1 The Proposed Approach 55
3.1.2 Experimental Setup 57

3.1.3 Experimental Results 60
3.1.4 Complexity Analysis 62
3.1.5 Approximation of Posterior Probabilities for Self-learning . . . 64
3.1.6 Discussion on Confirmation Bias 65
3.2 Self-learning for a Biological Application 66
3.2.1 Introduction 66
3.2.2 Framework 67
3.2.3 Open-World Self-learning Algorithm 68
3.2.4 CRISPR-Cas Subtype Prediction 71
3.3 Conclusion and Perspectives 76

4 Semi-supervised Feature Selection **79**
4.1 Introduction 79
4.2 Related Work 81
4.3 New Semi-supervised Wrapper: `MSLA-FSGA` 82
4.3.1 `FSGA`: Feature Selection Genetic Algorithm 83
4.3.2 Time Complexity 85
4.4 Experimental Results 86
4.4.1 Validation of the Feature Selection Genetic Algorithm 87
4.4.2 Improvement from Pseudo-labeling Unlabeled Data 89
4.4.3 Comparison with the State-of-the-Art 92
4.4.4 Additional Ablative Study 95
4.5 Conclusion and Perspectives 96

Conclusion and Perspectives **97**

Introduction

Artificial Intelligence (AI) is increasingly present in people's lives, capturing the attention of more and more scientists from various fields of study. One of the traditional problems of AI is Machine Learning, which is concerned with building algorithms that can learn and take actions from experience. Traditionally, experience is represented by training data that is a collection of inputs (usually called examples or observations) described by some characteristics (called features). Machine learning can be broadly divided into multiple categories that represent fundamentally different tasks (Bishop, 2006; Hastie et al., 2009). We focus on the three following ones that differ from each other based on the availability of a target variable: unsupervised, supervised and semi-supervised learning.

Unsupervised learning considers training data that are not supplied with any target values. Unsupervised approaches aim for analyzing intrinsic regularities in data in order to find clusters of similar examples, project data into a low dimension or estimate density. Supervised learning considers applications where every available training example comes with a label, i.e. with a target value. In this case, the goal is to learn (approximate) the target variable based on the available training data in order to be able to predict the target value for a potentially new example. Supervised problems can also categorized depending on the type of the target variable. For example, regression problems deal with a continuous target variable, while classification problems divide data into categories, i.e. the target is a categorical variable. Further, we focus on classification problems.

Finally, semi-supervised learning (Chapelle et al., 2010) considers problems where data are partially-labeled, i.e., labels are available only for some of the available training examples. This is inherent for many real-life applications, where the labeling of all training examples is costly and sometimes even not realistic. For example, in medical diagnosis or biological data analysis, labeling data may require very expensive tests so that only small labeled data sets are generally available. In many other cases, like web-oriented applications, a huge number of unlabeled observations arrive sequentially, and there is not enough time to manually label them all. The semi-supervised learning lies inherently between the supervised and unsupervised learning, and from the algorithmic perspective it combines the best of the two worlds. Typically, it is assumed that the number of labeled examples is small leading to inefficiency of supervised methods, while unlabeled examples con-

tain valuable information about the problem, thus their exploitation improves the learning quality.

In this thesis, we study semi-supervised learning in the multi-class classification case, tackling this problem from a theoretical, algorithmic, and applied point of view, with a particular focus on the self-learning algorithm (Tür et al., 2005; Amini and Usunier, 2015). Self-learning is a classical approach to classify partially-labeled data in a supervised fashion, where the training set is augmented by iteratively assigning pseudo-labels to unlabeled examples with the confidence score above a certain threshold. While the algorithm is widely used in practice, a little attention is given to its theoretical analysis. Moreover, its application raises several practical questions. At every iteration, the self-learning algorithm injects some noise in labeling, so the first question is how to optimally choose the confidence threshold to minimize the mislabeling probability. The second question is how the measure of confidence used in self-learning is adequate on real-life data, where the unlabeled set may contain out-of-distribution examples. The third question is to what extent pseudo-labels generated by self-learning can be used for post-analysis, e.g., for some model selection applications.

Outline and Contributions

Below we present an outline and overview the proposed contributions of this thesis. Chapter 1 is dedicated to introduce the problem, define basic concepts and discuss relevant background work.

In Chapter 2, we present theoretical analysis of self-learning in the context of the majority vote classifier, which represents a broad class of learning methods including Random Forest (Breiman, 2001), AdaBoost (Freund and Schapire, 1997), SVM (Vapnik, 1982) and neural networks (Rumelhart et al., 1986). The essence of the approach is to linearly combine predictions of different voters and output the class that received the largest vote. Generalization guarantees of majority vote classifiers are well studied but primarily in the binary supervised case. Following the transductive setting (Vapnik, 1998, p. 339), where the aim is to correctly classify unlabeled training examples, we extend the work of Amini et al. (2008) and derive a bound for the multi-class majority vote classifier by analyzing distribution of the class vote and focusing on the class confusion matrix as an error indicator (Morvant et al., 2012). This bound is obtained by analytically solving a linear program and it comes out that in the case when the majority vote classifier makes most of its errors on examples with a low class vote, the obtained bound is tight. Then, we derive another bound that takes explicitly into account possible mislabeling produced by self-learning. Considering a mislabeling model of Chittineni (1980), we show the relationship between the true and the imperfect label in the misclassification of a particular example. Then, we derive a new probabilistic C-bound over the error of the multi-class majority vote classifier in the presence of imperfect la-

bels. This bound is based on the mean and the variance of the prediction margin (Lacasse et al., 2007), so it reflects both the individual strength of voters and their correlation in prediction.

In Chapter 3, we propose a new policy for multi-class self-learning, where the confidence threshold is automatically found based on minimization of the transductive bound proposed in Chapter 2. The proposed approach outperforms several semi-supervised state-of-the-art methods. Then, we apply self-learning for a real-life DNA sequence classification task, where unlabeled data may contain classes not previously seen. In the context of decision trees, we propose a modification of the self-learning algorithm that is able to discover regions of potentially new classes by clustering examples with low prediction confidence.

In Chapter 4, we study semi-supervised problems in a high-dimensional regime with the goal to select important characteristics from the original set of features (Guyon and Elisseeff, 2003). We propose a feature selection approach that is based on a new modification of the genetic algorithm that creates and evaluates candidate feature subsets during an evolutionary process, taking into account feature weights and eliminating irrelevant features. In order to increase diversity of data used for feature subset strength evaluation, the proposed self-learning algorithm is used. Empirical results on different data sets show the effectiveness of our framework compared to several state-of-the-art semi-supervised feature selection approaches. As self-learning may mislabel unlabeled data, we also demonstrate the prospects of using the proposed C-bound with imperfect labels as a feature selection criterion.

Corresponding Articles

The contribution of this thesis includes the following articles, prepared during the postgraduate studies.

- The contribution from (Feofanov et al., 2019) is presented in Chapter 2 and Chapter 3.
- The contribution from (Feofanov et al., 2021a) is presented in Chapter 2 and Chapter 3.
- The contribution from (Feofanov et al., 2021b) is presented in Chapter 4.
- In addition, Chapter 3 (Section 3.2) contains work in preparation for publication (Feofanov et al., 2021c).

Chapter 1

Background

In this chapter, we introduce the basic concepts of the supervised as well as semi-supervised learning, and present the background needed for the rest of this thesis. The chapter is organized as follows. In Section 1.1, we overview supervised learning of a single classifier as well as an ensemble of classifiers and present a framework for theoretical analysis of majority vote classifiers. In Section 1.2, we introduce semi-supervised learning, typical assumptions made by algorithms in this framework, and existing theoretical studies of majority vote classifiers in the transductive setting.

1.1 Supervised Learning

Consider a classification problem with an input space $\mathcal{X} \subset \mathbb{R}^d$ and an output space $\mathcal{Y}$, and we write $\mathcal{Y} = \{1, \ldots, K\}$ when the general multi-class $K \geq 2$ is considered, while $\mathcal{Y} = \{-1, +1\}$ is used in the binary case. We denote by $\mathbf{X} = (X_1, \ldots, X_d) \in \mathcal{X}$ (respectively $Y \in \mathcal{Y}$) an input (respectively output) random variable. In the supervised framework, we assume there is access to a set of labeled training examples $Z_{\mathcal{L}} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{l} \in (\mathcal{X} \times \mathcal{Y})^l$, identically and independently distributed (i.i.d.) with respect to a fixed yet unknown probability distribution $P(\mathbf{X}, Y)$ over $\mathcal{X} \times \mathcal{Y}$. Let $\mathcal{H}$ be a fixed set of classifiers $\mathcal{H} = \{h \,|\, h : \mathcal{X} \to \mathcal{Y}\}$, called the *hypothesis space*, is considered and defined without reference to the training set. In turn, every classifier $h \in \mathcal{H}$ is called *hypothesis* to the true labeling mechanism. To measure the cost of errors made by a hypothesis $h \in \mathcal{H}$, a loss function ℓ is defined as:

$$\begin{aligned} \ell : \mathcal{Y} \times \mathcal{Y} &\to \mathbb{R} \\ (h(\mathbf{x}), y) &\mapsto \ell(h(\mathbf{x}), y). \end{aligned}$$

In this work, we focus on the 0/1 loss function $\ell_{0/1}(h(\mathbf{x}), y) := \mathbb{I}(h(\mathbf{x}) \neq y)$, where $\mathbb{I}(\pi)$ is the indicator function, which is equal to 1 if the predicate π is true and to 0 otherwise.

We evaluate the quality of a learning hypothesis h by defining the generalization

risk or *error rate* as the expected loss:

$$R(h) := \mathbb{E}_{\mathbf{X},Y}\mathbb{I}(h(\mathbf{x}) \neq y) = \int_{\mathcal{X}\times\mathcal{Y}} \mathbb{I}(h(\mathbf{x}) \neq y)\mathrm{d}P(\mathbf{x}, y).$$

Depending on how the true label is generated, two cases can be distinguished. In the *deterministic* setting, class labels are generated according to a deterministic function $y : \mathcal{X} \to \mathcal{Y}$, i.e., for every $\mathbf{x} \in \mathcal{X}$, there exists only one possible label, so the risk can be written as:

$$R(h) = \mathbb{E}_{\mathbf{X}}\mathbb{I}(h(\mathbf{x}) \neq y(\mathbf{x})).$$

In contrast, the more general *probabilistic* setting assumes a possibility of multiple outcomes for each example:

$$R(h) := \int_{\mathcal{X}} \sum_{\substack{c\in\{1,\ldots,K\} \\ c\neq h(\mathbf{x})}} P(Y = c|\mathbf{X} = \mathbf{x})\mathrm{d}P(\mathbf{X} = \mathbf{x}) \tag{1.1}$$

$$:= \mathbb{E}_{\mathbf{X}} \sum_{\substack{c\in\{1,\ldots,K\} \\ c\neq h(\mathbf{x})}} P(Y = c|\mathbf{X} = \mathbf{x}) = \mathbb{E}_{\mathbf{X}}[1 - P(Y = h(\mathbf{x})|\mathbf{X} = \mathbf{x})]. \tag{1.2}$$

Further, we will consider the probabilistic setting by default until the opposite is said.

The goal of learning a single classifier is formulated as to choose a hypothesis h^* that minimizes the risk:

$$h^* = \operatorname*{argmin}_{h\in\mathcal{H}} R(h).$$

It is straightforward to show that the best possible classifier we can design is to select the class according to maximum a posteri:

$$O(\mathbf{x}) := \operatorname*{argmax}_{c\in\mathcal{Y}} P(Y = c|\mathbf{X} = \mathbf{x}). \tag{1.3}$$

We further call this rule as the oracle classifier[1] whose risk $R(O)$ is the minimal possible error, i.e., irreducible error. However, in practice, we have access only to the training data $Z_{\mathcal{L}}$, and the joint probability $P(\mathbf{x}, y)$ is unknown. The same applies for evaluation of $R(h)$, so learning a classifier is based on minimization of the *empirical risk* defined as follows:

$$R_{\mathcal{L}}(h) = \frac{1}{l}\sum_{i=1}^{l} \mathbb{I}(h(\mathbf{x}_i) \neq y_i).$$

[1]The classifier is often called the Bayes classifier, but we do not use this name to avoid confusion with another Bayesian approach defined below.

Below, we briefly overview some existing approaches for supervised classification, when one hypothesis is learned (Section 1.1.1) and when multiple hypotheses are combined (Section 1.1.2). Finally, we give more details about theoretical analysis of a class of majority vote classifiers.

1.1.1 Single Hypothesis Learning

There exists plenty of supervised classification algorithms based on different learning paradigms, we refer to Bishop (2006); Hastie et al. (2009) for a complete review of the most known results. Some approaches like the discriminant analysis and the naive Bayes classifier estimate the oracle classifier by making explicit assumptions on the data distribution. The k-nearest neighbors finds for a new example k closest training examples (neighbors) and outputs a class label that is the most represented among neighbors. Many approaches are based on optimization of some objective function including the likelihood, the hinge loss or the cross-entropy loss. Classifiers can be linear (e.g., in the binary case, it is written as $h(\mathbf{x}) = \text{sgn}(\mathbf{w}^\intercal \mathbf{x} + b)$ with a weight vector $\mathbf{w}$ of size d) like the logistic regression or the support vector machine, where the latter is extended to a non-linear case by a kernel trick. Other well known non-linear algorithms are neural networks and decision trees. In this paper, we focus on a decision tree algorithm, so we give more details about it below. Our choice is motivated by its versatility, since it can be applied for both numerical and categorical features being a scale-invariant learning algorithm. Also, it is well known that trees is a good choice for ensemble learning (Hastie et al., 2009).

Decision Tree Learning

Decision tree learning (Breiman et al., 1984; Quinlan, 1986) is a widely known approach for classification. There are various architectures to design it, we give details of the CART tree. The main principle of the approach consists in top-down recursive binary partitioning of the feature space into disjoint regions. Starting from the root with the whole training set, at every node, a feature and a split value are chosen to construct left and right child nodes. After the split, the left/right child node includes training examples with the feature value smaller/greater than the split value. As a rule, the feature and the split value are searched exhaustively over all feature values appeared in the training set with the goal to maximize the impurity decrease criterion based on the Gini index. Let a node contains the training examples from classes $1, \dots, K$ in proportions $p_1, \dots, p_K$. The Gini index for this node is defined as $G(\text{node}) := \sum_{k=1}^{K} p_k(1 - p_k)$. Then, the impurity decrease criterion $\mathcal{S}^{\text{sup}}$ with respect to a feature f and a split value s is defined as

$$\mathcal{S}^{\text{sup}}(f, s) := \frac{n_{\text{parent}}}{n} \left(G(\text{parent}) - \frac{n_{\text{l-child}}}{n_{\text{parent}}} G(\text{l-child}) - \frac{n_{\text{r-child}}}{n_{\text{parent}}} G(\text{r-child}) \right), \quad (1.4)$$

where $n_{\text{parent}}, n_{\text{l-child}}, n_{\text{r-child}}$ are respectively the number of examples in the parent, left child and right child nodes. The tree is constructed recursively while a stopping condition is satisfied. In order to predict a class label for a new example, the latter is passed through the tree until it falls to a leaf, i.e., to a certain region. The predicted label is the majority vote class considering all training examples that belong to this region.

1.1.2 Ensemble Learning

Before explaining the principle of ensemble learning, we would like to give insights on limitations of a single hypothesis learning. Let $h \in \mathcal{H}$ be our trained classifier. One can notice that the difference between the risk of the hypothesis h and the irreducible error $R(O)$ can be decomposed into two parts (Mohri et al., 2018):

$$R(h) - R(O) = \underbrace{R(h) - \inf_{h' \in \mathcal{H}} R(h')}_{\text{estimation error}} + \underbrace{\inf_{h' \in \mathcal{H}} R(h') - R(O)}_{\text{approximation error}}.$$

It can be seen that efficiency of learning h depends on how well the hypothesis space was chosen to approximate the oracle classifier (approximation error) and how much the error of h differs from the best hypothesis in the hypothesis space (estimation error). Since there is no access to the data distribution, a direct estimate of the approximation error requires strong assumptions. If the hypothesis space is of large capacity, the approximation error may be small, but the estimation error may increase due to the increase in the complexity of the search space. On the other hand, the approximation error may be large when the hypothesis space $\mathcal{H}$ has a lack of richness.

To overcome this, an ensemble of classifiers can be considered instead of learning a single hypothesis. As pointed out by Dietterich (2000), the ensemble primarily has three significant advantages. First of all, different hypotheses may have similar empirical errors on a given training set, which, however, does not mean that they all generalize well. By combining predictions of different hypotheses, we minimize our chances of overfitting the training data and having a large generalization error. Secondly, many learning algorithms have a large estimation error even with a plenty of training examples, since they may get into local optima. Hence, an aggregation of various hypotheses suggests to decrease the estimation error being less prone to output a suboptimal solution. Thirdly, with ensemble approaches we are not limited by the initial hypothesis space, since linear combinations of the hypotheses' predictions may output new hypotheses out of $\mathcal{H}$[2]. Thus, more hypotheses can be explored without increasing the complexity of $\mathcal{H}$. This is especially interesting because the approximation error can be reduced not only by choosing the hypothe-

[2]Note that a learning algorithm with a finite training set explores only a finite set of hypotheses, so the hypothesis space is practically finite.

sis space properly, but also by studying how hypotheses can be optimally combined (Kuncheva, 2014). Below, we give a overview of some important studies of ensemble learning.

Bayesian Reasoning

One of the earliest approaches to combine hypotheses is based on Bayesian reasoning. Let us define a random variable H that *a priori* distributed over $\mathcal{H}$ as $P(H = h)$, which can be chosen based on domain knowledge. Observing a training data $\mathrm{Z}_{\mathcal{L}} \sim P(\mathbf{X}, Y)^l$, we can apply the Bayes rule and express the *posterior* as follows:

$$P(H = h|\mathrm{Z}_{\mathcal{L}}) \propto P(H = h)P(\mathrm{Z}_{\mathcal{L}}|H = h), \tag{1.5}$$

where $P(\mathrm{Z}_{\mathcal{L}}|H = h)$ denotes the data likelihood given that h is a true labeling mechanism. Thus, we can identify the most probable hypothesis given a training set based on the prior knowledge about the hypotheses and their fitness to the data. Then, for every new observation $\mathbf{x} \in \mathcal{X}$, we output the majority vote class by combining the hypotheses' predictions weighted by the posterior probabilities. This approach is called the *Bayes optimal classifier* and defined in the following way:

$$\begin{aligned} B_{\text{opt}}(\mathbf{x}) &:= \operatorname*{argmax}_{c \in \mathcal{Y}} \mathbb{E}_{h|\mathrm{Z}_{\mathcal{L}}} \mathbb{I}(h(\mathbf{x}) = c) \\ &= \operatorname*{argmax}_{c \in \mathcal{Y}} \sum_{h \in \mathcal{H}} \mathbb{I}(h(\mathbf{x}) = c) P(H = h) P(\mathrm{Z}_{\mathcal{L}}|H = h). \end{aligned}$$

When the prior and the hypothesis space are fixed, this classifier is optimal in a sense that any other linear combination of the hypotheses would be worse on average (Mitchell, 1997). However, one can notice that estimation of $P(\mathrm{Z}_{\mathcal{L}}|H = h)$ for every hypothesis could be practically infeasible when the cardinality of $\mathcal{H}$ is high. Moreover, the approach depends on the choice of the prior, which is often not clear how to set.

In this connection, a research focus has been switched to a general *Bayes classifier* that weights the hypotheses according to a posterior distribution Q over $\mathcal{H}$ that is not necessarily a posterior distribution in the Bayesian sense[3]. The approach is commonly found in the literature under two names, the (Q-weighted) *majority vote classifier* and the Bayes classifier, and for all $\mathbf{x} \in \mathcal{X}$, it is defined as follows:

$$B_Q(\mathbf{x}) := \operatorname*{argmax}_{c \in \mathcal{Y}} \mathbb{E}_{h \sim Q} \mathbb{I}(h(\mathbf{x}) = c) = \operatorname*{argmax}_{c \in \mathcal{Y}} \sum_{h : h(\mathbf{x}) = c} Q(h).$$

In this case, the goal of learning can be formulated as to choose a posterior distribution Q over $\mathcal{H}$ such that the classifier B_Q will have the smallest possible error value.

[3]In this case, by posterior distribution we call any distribution over $\mathcal{H}$ that has been obtained after observing data.

It is worth to notice that a theoretical study of the Bayes classifier provides a tool for analyzing a wide class of ensemble methods, where the predictions of hypotheses are aggregated using the majority vote rule scheme. Moreover, as pointed out by Germain et al. (2015), the Support Vector Machine as well as the neural networks can also be regarded as the majority vote classifiers.

Bagging and Random Forests

A single decision tree is an unstable algorithm that tends to overfit the training data, resulting in poor generalization performance. To overcome this, a *bagging* procedure was proposed by Breiman (1996) that consists in parallel learning of multiple classifiers, where each classifier h_t, $t \in \{1, \ldots, T\}$ is trained on a bootstrap sample B_t (Efron, 1992) of size l drawn with replacement from the training set $\mathrm{Z}_{\mathcal{L}}$. Since the classifiers are learned on slightly different data sets (each bootstrap sample contains approximately 63.2% of the original training samples), their aggregation results in a stable algorithm. While the bagging ensemble can be constructed based on any learning algorithm, a common choice is the decision tree. A random forest (Breiman, 2001, denoted by `RF`) is a majority vote bagging ensemble of trees that for an example $\mathbf{x}$, predicts a label by combining predictions of trees h_t with equal weights:

$$B_{\mathrm{RF}}(\mathbf{x}) := \operatorname*{argmax}_{c \in \mathcal{Y}} \sum_{t=1}^{T} \mathbb{I}(h_t(\mathbf{x}) = c).$$

Since the trees $\{h_t\}_{t=1}^{T}$ are trained on different bootstrap samples, its generalization ability can be advantageously estimated by the out-of-bag error, which takes less time than the cross-validation (Stone, 1974). For every training example $\mathbf{x} \in \mathrm{Z}_{\mathcal{L}}$ and a class $c \in \mathcal{Y}$, the out-of-bag vote is evaluated as the proportion of trees that did not contain the example $\mathbf{x}$ in their respective bootstrap sample:

$$v_{OOB}(\mathbf{x}, c) = \frac{1}{|\{t : \mathbf{x} \notin \mathrm{B}_t\}|} \sum_{t:\ \mathbf{x} \notin \mathrm{B}_t} \mathbb{I}(h_t(\mathbf{x}) = c). \tag{1.6}$$

Then, the out-of-bag error is defined as follows:

$$\frac{1}{l} \sum_{i=1}^{l} \mathbb{I}(y_i \neq \operatorname*{argmax}_{c \in \mathcal{Y}} v_{OOB}(\mathbf{x}_i, c)). \tag{1.7}$$

The out-of-bag error has been shown to be an unbiased estimator of the generalization error (Breiman, 2001).

Boosting

In contrast to a bagging ensemble that learns classifiers in parallel, a *boosting* algorithm trains "weak" classifiers successively so that every next classifier improves

performance by focusing on weak points of its predecessors. For example, the *AdaBoost* (Freund and Schapire, 1997) at step $t = 2, \ldots,$, learns a classifier on the training examples that are weighted depending on where the $(t-1)$-th classifier made mistakes. The weights of correctly classified examples are decreased, while the weights are increased for the misclassified examples thereby drawing attention of the t-th classifier on these difficult examples. The output of the AdaBoost is the majority vote of the weak classifiers weighted based on their individual performance. There exists many other variants of boosting, particularly, Mason et al. (1999); Friedman (2001) proposed a gradient boosting, which allows to optimize any differentiable loss function.

Diversity

In ensemble learning, there exists another branch of study that has a particular focus on generating and/or combining classifiers based on some diversity criterion (Kuncheva, 2014). Indeed, it is widely known that classifiers should be not only reasonably accurate but also diverse in order to gain a strong ensemble (Hansen and Salamon, 1990). This fact is clearly illustrated in several bias/variance decomposition (Kohavi et al., 1996; Krogh and Vedelsby, 1995). Particularly, Krogh and Vedelsby (1995) decomposes the error[4] into two terms: the average error of hypotheses and the ambiguity term that expresses correlation between hypotheses. As pointed out by Brown and Wyatt (2003), optimization of the ambiguity term is practically employed in the negative correlation framework (Liu and Yao, 1999), where multiple neural networks are trained simultaneously and are forced to have negatively correlated errors with respect to each other. This allows to specialize all networks on different regions or subtasks, so their aggregation yields high performance. The benefit of negatively correlated errors was theoretically shown by Tumer and Ghosh (1996) in the multi-class case, but under very strong assumptions.

1.1.3 Analysis of Majority Vote Classifiers

In this section, we survey some theoretical results obtained for the majority vote classifier (Section 1.1.2) in the supervised case.

In the binary case, the output space can be described as $\mathcal{Y} = \{-1, +1\}$, so the Bayes classifier can be represented without the argmax operator as follows:

$$B_Q(\mathbf{x}) := \operatorname{sgn}(\mathbb{E}_{h \sim Q} h(\mathbf{x})), \qquad \forall \mathbf{x} \in \mathcal{X}, \tag{1.8}$$

where $\operatorname{sgn}(\pi)$ denotes the sign function, which is equal to 1 if the predicate π is positive and to -1 otherwise.

[4]It was shown for the regression task, but it also holds for the binary classification with the squared loss.

Following Germain et al. (2015), we define the prediction *margin* as

$$M_Q(\mathbf{x}, y) := \mathbb{E}_{h\sim Q}[y \cdot h(\mathbf{x})] = \mathbb{E}_{h\sim Q}\mathbb{I}(h(\mathbf{x}) = y) - \mathbb{E}_{h\sim Q}\mathbb{I}(h(\mathbf{x}) = -y).$$

The margin measures a gap between the vote of the true class y and the vote of the opposite class. If the value is strictly positive for an example $\mathbf{x}$, then y will be the output of the majority vote, so the example will be correctly classified.

Together with the Bayes classifier, we consider the related *Gibbs classifier*, which is a stochastic learning algorithm that for every $\mathbf{x} \in \mathcal{X}$, randomly chooses a hypothesis $h \in \mathcal{H}$ according to the distribution Q and then predicts a label as $h(\mathbf{x})$. Although the classifier is stochastic, we define its risk and empirical risk in expectation over all hypotheses in $\mathcal{H}$:

$$R(G_Q) := \mathbb{E}_{\mathbf{X},Y}\mathbb{E}_{h\sim Q}\mathbb{I}(h(\mathbf{x}) \neq y), \tag{1.9}$$
$$R_{\mathcal{L}}(G_Q) := \frac{1}{l}\sum_{i=1}^{l}\mathbb{E}_{h\sim Q}\mathbb{I}(h(\mathbf{x}_i) \neq y_i).$$

It can be noticed that the Gibbs risk represents the weighted average error rate of individual hypotheses.

Bounds on the Gibbs Risk

A common approach is to upper-bound the Bayes risk by twice the Gibbs risk (Langford and Shawe-Taylor, 2002, Lemma 4.1):

$$R(B_Q) \leq 2R(G_Q). \tag{1.10}$$

Since the data distribution is unknown, Eq. (1.10) can not be exactly evaluated, so the question is how to upper-bound the generalization error based on the available training set. A common choice is the PAC-Bayesian framework initiated by McAllester (1999, 2003) that provides Probably Approximately Correct (PAC) guarantees (a PAC bound depends on ϵ and holds with probability at least $1-\epsilon$) over the Gibbs risk. This approach considers a prior Q_0 and a posterior distribution Q over $\mathcal{H}$, and upper-bounds the Gibbs risk by the empirical Gibbs risk on the training data and the Kullback-Leibler divergence between Q and Q_0. The classical PAC-Bayesian theorem is formulated below.

Theorem 1.1.1 (Langford, 2005, Theorem 5.1)**.** *For all $P(\mathbf{X}, Y)$, for any choice of Q_0 over $\mathcal{H}$, for any $\epsilon \in (0, 1]$, with a probability at least $1-\epsilon$ over the choice of the sample $Z_{\mathcal{L}} \sim P(\mathbf{X}, Y)^l$ for every posterior distribution Q over $\mathcal{H}$ we have:*

$$kl\left(R_{\mathcal{L}}(G_Q)||R(G_Q)\right) \leq \frac{1}{l}\left[KL(Q \parallel Q_0) + \ln\frac{l+1}{\epsilon}\right],$$

where $KL(Q \parallel Q_0)$ is the Kullback-Leibler divergence between Q and Q_0:

$$KL(Q \parallel Q_0) := \mathbb{E}_{h\sim Q}\left[\ln \frac{dQ(h)}{dQ_0(h)}\right],$$

and $kl(q||p)$ is the Kullback-Leibler divergence between the two Bernoulli distributions with probability of success q and p respectively:

$$kl(q||p) := q \log \frac{q}{p} + (1-q) \log \frac{1-q}{1-p}.$$

Thus, Theorem 1.1.1 establishes the connection between the generalization risk and its empirical estimate in an implicit way. If we apply the Pinsker's inequality $2(R_{\mathcal{L}}(G_Q) - R(G_Q))^2 \leq kl\left(R_{\mathcal{L}}(G_Q)||R(G_Q)\right)$, we obtain an explicit bound on the generalization risk (McAllester, 2003):

$$R(G_Q) \leq R_{\mathcal{L}}(G_Q) + \sqrt{\frac{1}{2l}\left[KL(Q \parallel Q_0) + \ln \frac{l+1}{\epsilon}\right]} \tag{1.11}$$

One can see that the bound given by Eq. (1.11) consists of two terms: the empirical risk and the penalty term that with the growth of the sample size l is decreasing to 0. An advantage of the PAC-Bayes theorem is that this bound holds for any choice of Q_0 and Q, which means that the PAC-Bayesian bound can be used for optimizing Q. Roughly speaking, minimization of the bound implies minimization of the empirical Gibbs risk regularized by the Kullback-Leibler divergence. Thiemann et al. (2017) proposed a relaxation of the bound such that it is convex in Q and convex in a parameter that balances the Kullback-Leibler divergence in order to prevent excessive regularization by the divergence. However, the bound involves only the individual errors of hypotheses ignoring the fact that the hypotheses in combination may be much stronger than individually.

C-Bound

To overcome the issue discussed above, it was proposed to upper-bound the Bayes risk directly (Lacasse et al., 2007). The C-bound is based on the mean and variance of the prediction margin and is derived by applying the Cantelli-Chebyshev inequality. A similar result was obtained in a different context by Breiman (2001). We define the first and the second statistical margin moments respectively as:

$$\mu_1 := \mathbb{E}_{\mathbf{X},Y} M_Q(\mathbf{x}, y), \quad \mu_2 := \mathbb{E}_{\mathbf{X},Y}\left[M_Q(\mathbf{x}, y)\right]^2.$$

The C-bound is formulated in the following theorem.

Theorem 1.1.2 (Germain et al. (2015, Theorem 11))**.** *For any posterior Q over $\mathcal{H}$,*

any data distribution $P(\mathbf{X}, Y)$*, if* $\mu_1 > 0$*, then*

$$R(B_Q) \leq 1 - \frac{(\mu_1)^2}{\mu_2} = 1 - \frac{\left(1 - 2R(G_Q)\right)^2}{1 - 2D_Q},$$

where $D_Q := \mathbb{E}_{\mathbf{X}} \mathbb{E}_{h1\sim Q} \mathbb{E}_{h2\sim Q} \mathbb{I}(h_1(\mathbf{x}) \neq h_2(\mathbf{x}))$ *is the expected disagreement of hypotheses.*

From Theorem 1.1.2 we can see that the C-bound involves the Gibbs risk, i.e., the individual errors of hypotheses, and the disagreement of hypotheses' predictions. This result is consistent with other studies on ensemble learning, showing that the effectiveness of the majority vote depends on the individual performance of hypotheses and their diversity (Krogh and Vedelsby, 1995; Tumer and Ghosh, 1996). The minimization of the C-bound underlies several learning algorithms (Roy et al., 2016; Bauvin et al., 2020; Viallard et al., 2021). Note that the associated PAC-Bayesian C-bound can also be derived to bound the generalization error based on the available sample data (see Germain et al. (2015) for more details).

Multi-class Classification

Theoretical studies of the majority vote classifier have a focus mostly on the binary setting, while only few results exist in the multi-class setting. Morvant et al. (2012) considered a confusion matrix as an error indicator and derived generalization guarantees on the confusion matrix' norm of the Gibbs classifier. Laviolette et al. (2017) extended the C-bound to the multi-class case by considering the multi-class margin. Masegosa et al. (2020) derived a bound that is based on the second-order Markov's inequality, which can be regarded as a relaxation of the C-bound based on the Cantelli-Chebyshev inequality. In this work, we also consider the multi-class majority vote classifier, but with a focus on semi-supervised learning.

1.2 Semi-supervised Learning

Considering the semi-supervised framework (Chapelle et al., 2010), we assume that two sets are available: a set of labeled training examples $Z_{\mathcal{L}} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{l} \in (\mathcal{X} \times \mathcal{Y})^l$ i.i.d. with respect to a fixed yet unknown probability distribution $P(\mathbf{X}, Y)$ over $\mathcal{X} \times \mathcal{Y}$, and a set of unlabeled training examples $X_{\mathcal{U}} = \{\mathbf{x}_i\}_{i=l+1}^{l+u} \in \mathcal{X}^u$ that are supposed to be drawn i.i.d. from the marginal distribution $P(\mathbf{X})$ over the domain $\mathcal{X}$. It is additionally assumed that $l \ll u$, so that the supervised learning is inefficient.

Semi-supervised learning can be formulated following one of two settings: inductive or transductive. In Section 1.1 we formulated the learning objective as minimization of the generalization risk. This approach follows the *inductive* inference, i.e., we aim for a classifier that is learned on a finite number of training examples

and has the lowest error averaged over all possible examples with respect to the distribution $P(\mathbf{X}, Y)$. This approach is also relevant for semi-supervised learning and can be used to label the unlabeled data.

However, since the labeled data is assumed scarce, estimation of the data distribution from few training examples is a difficult problem. Moreover, in many applications it is required to accurately label the available unlabeled examples rather than to infer a general rule. As Vapnik (1982, 1998) pointed out, with a lack of available information, it would be better to focus on solving the task directly and avoid solving a general problem as an intermediate step. Thus, Vapnik (1998) proposed to consider the *transductive* inference that consists in reasoning from the given labeled set directly to the unlabeled one. Instead of searching a general rule, we focus on the unlabeled examples in order to label them as accurately as possible. Thus, the goal of learning can be formulated as to design a classifier h based on $\mathrm{Z}_{\mathcal{L}}$ and $\mathrm{X}_{\mathcal{U}}$ that minimizes the *transductive risk* defined in the deterministic and probabilistic cases respectively by:

$$R_{\mathcal{U}}(h) := \frac{1}{u} \sum_{i=l+1}^{l+u} \mathbb{I}(h(\mathbf{x}_i) \neq y_i) \tag{1.12}$$

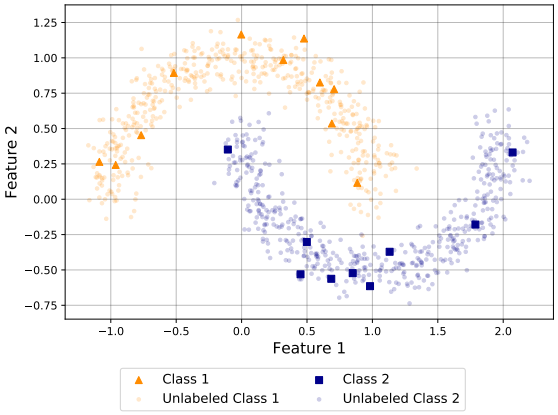
$$R_{\mathcal{U}}(h) := 1 - \frac{1}{u} \sum_{i=l+1}^{l+u} P(Y = h(\mathbf{x}_i) | \mathbf{X} = \mathbf{x}_i). \tag{1.13}$$

While minimization of the transductive objective is supposed to be an easier problem, in practice, both inductive and transductive learning face the same issue of integrating unlabeled data into the learning process. In Section 1.2.1, we give a brief introduction to some of the existing semi-supervised approaches and the assumptions they rely on. Then, Section 1.2.2 overviews existing studies of the majority vote classifier in the transductive setting.

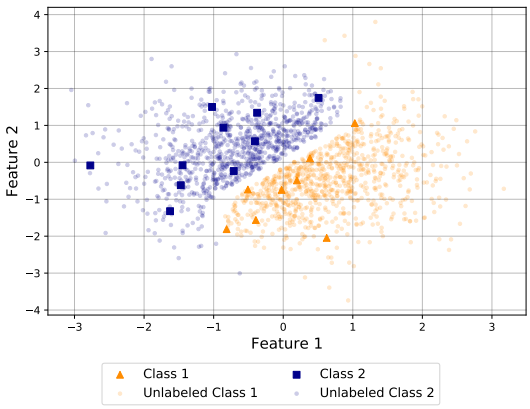
1.2.1 Existing Approaches

In semi-supervised learning, it is generally expected that unlabeled examples contain valuable information about the prediction problem, thus their exploitation may lead to an increase of performance. However, to a certain degree, it is not clear when the unlabeled data may be useful, since it depends on the nature of data, the number of the labeled and unlabeled examples, and many other factors. Almost all semi-supervised approaches rely on specific assumptions about how data is distributed. While meeting these assumptions we benefit from the unlabeled examples, the violation may lead to degradation with respect to supervised baselines (Cozman et al., 2002).

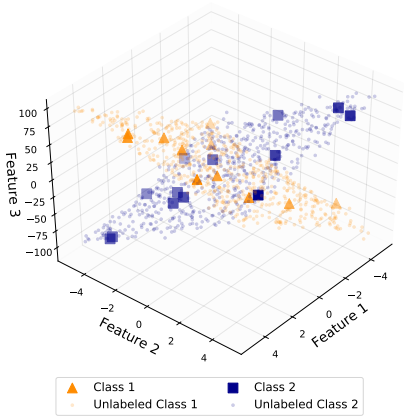
There are several common assumptions used to develop a semi-supervised algorithm. Most of them embody an idea of smoothness: if examples are "close" to each other, they belong to the same class. For example, some approaches assume



(a) Cluster Assumption



(b) Low Density Separation



(c) Manifold Assumption

Figure 1-1: Illustration of three assumptions typically made in semi-supervised learning: (a) Cluster Assumption, (b) Low Density Separation, (c) Manifold Assumption.

that the labeled and unlabeled examples can be divided into informative *clusters* (Figure 1-1a). This assumption is usually implemented by asserting that examples from the same cluster share the same class label (Rigollet, 2007), so the labeled examples propagate their labels to the unlabeled ones from the same cluster. Alternatively, Maximov et al. (2018) defines a cluster as a dense region with κ predominant classes, and use such clusters to regularize a supervised classifier. In practice, these approaches rely heavily on the underlying clustering algorithm, which may output completely different clusters depending on the values of its hyperparameters and the training data.

Another similar yet different assumption is the *low density separation* (Figure 1-1b). In this case, a semi-supervised classifier focuses on constructing a decision boundary that avoids passing through dense regions of unlabeled data (Chapelle and Zien, 2005). The idea is usually implemented by relying on prediction per-

formance of a base classifier and margin maximization for both the labeled and unlabeled examples (Joachims, 1999; d'Alché-Buc et al., 2001). Since margins for unlabeled data can be only approximated, it implies that a low density separation algorithm depends on the initial performance of its base classifier, which can be low due to the lack of labeled examples.

Algorithms based on the *manifold assumption* (Figure 1-1c) suggest that the training data lie on a low-dimensional manifold. If to additionally assume that two examples are likely share the same label when they are close to each other on a manifold, then the manifold assumption becomes similar to the cluster assumption. A common approach for capturing intrinsic geometry of data is to represent both the labeled and unlabeled observations as vertices of a graph and edge weights representing similarity between two vertices. The graph is then analyzed via the Laplacian matrix (Zhu et al., 2003; Zhou et al., 2004; Belkin and Niyogi, 2004). Although the graph-based approaches are widely spread for semi-supervised learning, their performance is highly sensitive to data topology.

Self-learning Algorithm

The self-learning algorithm[5] (`SLA`) is a widely known approach to learn on labeled and unlabeled data. The main idea is to iteratively re-learn a supervised base classifier by expanding the labeled set. At each iteration, it assigns pseudo-labels (i.e., predicted labels) to those unlabeled examples that have prediction confidence above a threshold. The pseudo-labeled examples are then included in the training set, and the classifier is retrained. The process is repeated until no examples for pseudo-labeling are left. The algorithm is illustrated in Figure 1-2.

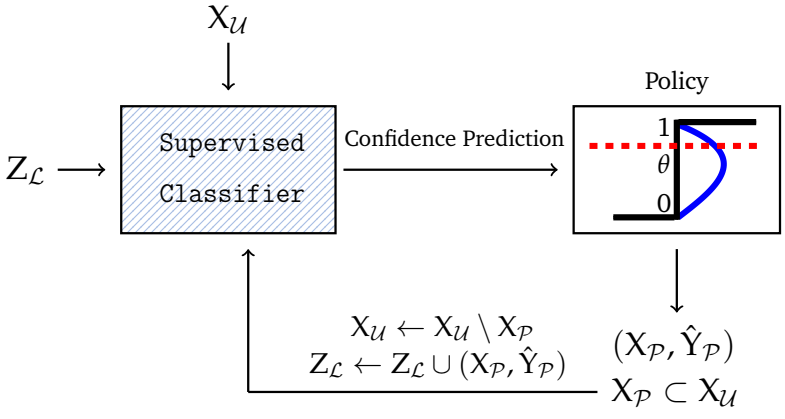


Figure 1-2: Self-Learning Algorithm `SLA`.

Generally speaking, the algorithm implicitly assumes that the confidence measure is well-calibrated, i.e., the supervised classifier makes its mistakes mostly on

[5]It is also known as self-training or self-labeling.

examples with low confidence (at least, at the initial step). Depending on which base classifier is chosen, this assumption can be interpreted in different ways. For example, in the case of margin maximization algorithms (e.g., SVM or decision trees that also fit this principle (Leistner et al., 2009)), the assumption corresponds to the low density separation, since by including pseudo-labeled unlabeled examples in the training set we push away the decision boundary from these examples, and the final classifier will pass through regions of low density.

A classical policy for pseudo-labeling is to fix the confidence threshold to a fixed value (Tür et al., 2005). In this case, it is not clear which threshold value should be taken, since it depends on the base classifier, the data set and the number of classes. In reality, when the threshold was not selected properly, the classification performance may degrade with respect to the base classifier learned on the labeled examples only. In the context of majority vote classifiers, we will show how this threshold can be selected dynamically based on transductive guarantees in the binary case (Section 1.2.2) and in the multi-class case (Section 3.1).

There exists many algorithms closely related to self-learning. The most known one is co-training (Blum and Mitchell, 1998), where two classifiers are trained on different data views, and instead of learning themselves, they co-learn each other, so that examples pseudo-labeled by one classifier are included in the training set of another classifier. The approach assumes that diversity of two classifiers is supplied by the existence of two data views (e.g., it can be two independent feature sets), which is not always possible to have. Note that the co-training algorithm also pseudo-labels based on a confidence threshold, hence it has similar limitations to self-learning.

Leistner et al. (2009) proposed a pseudo-labeling approach for the random forest. In contrast to self-learning, at each iteration it is trained on all labeled and unlabeled examples, where pseudo-labels for unlabeled data are generated independently for each tree according to posteriors $\{\hat{P}(Y|\mathbf{X} = \mathbf{x}_i)\}_{i=l+1}^{l+u}$. These posteriors are optimized via deterministic annealing, so pseudo-labels are re-computed at each iteration. The proposed approach is supposed to have less chances to degrade performance since it employs soft pseudo-labeling, which also can be regarded as a limitation compared to the classical pseudo-labeling. We will see this more clearly in Section 3.1.

Low Density Separation Split for Decision Trees

In the classical supervised setting, the training set is assumed to be sufficient for generalization. However, in the case of semi-supervised learning, some regions are sparse in training data, so splits in a decision tree are not performed accurately. This may lead to a paradox situation when some examples are misclassified with a very large prediction confidence. To overcome this problem, Kim (2016) have proposed to take into account the unlabeled data for searching feature splits such that will improve both the impurity decrease (labeled part) and split data in a place

of low density (unlabeled part). Let $\bar{x}^{(f)}_{\text{parent}}$, $\bar{x}^{(f)}_{\text{l-child}}$ and $\bar{x}^{(f)}_{\text{r-child}}$ be the mean value of a feature f respectively in the parent, left child and right child nodes. Then, the low density split criterion $\mathcal{S}^{\text{unsup}}$ can be defined as a ratio of the between-group variance and the total variance:

$$\mathcal{S}^{\text{unsup}}(f,s) := \frac{n_{\text{l-child}}(\bar{x}^{(f)}_{\text{l-child}} - \bar{x}^{(f)}_{\text{parent}})^2 + n_{\text{r-child}}(\bar{x}^{(f)}_{\text{r-child}} - \bar{x}^{(f)}_{\text{parent}})^2}{\sum_{x^{(f)} \in \text{parent}} (x^{(f)} - \bar{x}^{(f)}_{\text{parent}})^2}.$$

The higher the value of $\mathcal{S}^{\text{unsup}}(f,s)$, the lower density around the split value s. Thus, the semi-supervised split is defined as:

$$\mathcal{S}^{\text{ssl}}(f,s) := (1-\lambda)\mathcal{S}^{\text{sup}}(f,s) + \lambda \mathcal{S}^{\text{unsup}}(f,s), \tag{1.14}$$

where $\mathcal{S}^{\text{sup}}(f,s)$ is defined by Eq. (1.4), and λ is a parameter that balances between the supervised and the unsupervised splits criteria. This semi-supervised criterion is a promising tool to improve estimation of prediction probabilities for unlabeled data, and it can be used in combination with self-learning. However, the optimal value of λ may deviate from application to application, which is seen from the empirical experiments performed by Kim (2016).

1.2.2 Transductive Bounds for the Majority Vote

It is common to analyze theoretically semi-supervised algorithms in the transductive setting. Initiated by Vapnik (1982), there has been proposed various transductive bounds in terms of the Rademacher complexity (El-Yaniv and Pechyony, 2009), the algorithmic stability (El-Yaniv and Pechyony, 2006) or the PAC-Bayesian theorem (Derbeko et al., 2004; Bégin et al., 2014), where the latter is an extension of the supervised results presented in Section 1.1.3. Similarly to (1.12), we define the transductive Bayes and Gibbs risk as:

$$R_{\mathcal{U}}(B_Q) := \frac{1}{u} \sum_{i=l+1}^{l+u} \mathbb{I}(B_Q(\mathbf{x}_i) \neq y_i),$$

$$R_{\mathcal{U}}(G_Q) := \frac{1}{u} \sum_{i=l+1}^{l+u} \mathbb{E}_{h\sim Q} \mathbb{I}(h(\mathbf{x}_i) \neq y_i).$$

One can notice that in the transductive setting, the Bayes risk is also bounded by twice the Gibbs risk: $R_{\mathcal{U}}(B_Q) \leq 2R_{\mathcal{U}}(G_Q)$. A transductive PAC-Bayesian theorem was first derived by Derbeko et al. (2004) based on the work of Vapnik (1982). Then, this result was refined by Bégin et al. (2014). They assume that the training data are generated in the following way: first, $l+u$ unlabeled examples $\mathbf{x}_i$ are drawn i.i.d. from the marginal distribution $P(\mathbf{X})$; then, l of $l+u$ examples are randomly chosen, i.e., sampled without replacement, and are labeled according to

$P(Y|\mathbf{X})$. Thus, this data generation implies that the training examples are dependent. Below, we provide an explicit PAC-Bayesian bound of Bégin et al. (2014) (re-arranging some terms to get exactly the bound on $R_{\mathcal{U}}(G_Q)$).

Theorem 1.2.1 (Corollary 7 in Bégin et al. (2014)). *For any set $X_{\mathcal{L}} \cup X_{\mathcal{U}}$ of $l+u \geq 40$ examples, for any hypothesis space $\mathcal{H}$, for any $\delta \in (0,1]$, with probability at least $1-\delta$ over the choices $Z_{\mathcal{L}}$ ($l \in [20, n-20]$), we have:*

$$R_{\mathcal{U}}(G_Q) \leq R_{\mathcal{L}}(G_Q) + \frac{l+u}{u}\sqrt{\frac{u}{2l(l+u)}\left[KL(Q \parallel P) + \ln\frac{t(l,u)}{\delta}\right]},$$

where $t(l,u) := 3\ln(l)\sqrt{lu/(l+u)}$.

The main disadvantage of this bound is similar to the inductive case: the Bayes risk is bounded indirectly. In this connection, Amini et al. (2008) have derived a direct transductive bound on the Bayes risk in the binary case based on the margin distribution of unlabeled data. As the margin can not be evaluated for an unlabeled example, they consider the *unsigned margin* originally introduced by d'Alché-Buc et al. (2001) and defined in the following way:

$$m_Q(\mathbf{x}) := \left|\mathbb{E}_{h\sim Q}h(\mathbf{x})\right| = \left|\mathbb{E}_{h\sim Q}\mathbb{I}(h(\mathbf{x})=1) - \mathbb{E}_{h\sim Q}\mathbb{I}(h(\mathbf{x})=-1)\right|. \tag{1.15}$$

Thus, the unsigned margin represents the absolute value of the prediction margin and can be treated as an indicator of confidence. This is relevant if we additionally assume that the Bayes classifier makes its mistakes on low margin.

Amini et al. (2008) derives a bound for extended risk called the transductive *joint Bayes risk* defined for $\theta \in [0,1)$ as follows:

$$R_{\mathcal{U}\wedge\theta}(B_Q) := \frac{1}{u}\sum_{i=l+1}^{l+u}\mathbb{I}(B_Q(\mathbf{x}_i) \neq y_i \wedge m_Q(\mathbf{x}) > \theta), \tag{1.16}$$

which evaluates the transductive risk on those unlabeled examples that have margin above a threshold θ.

Theorem 1.2.2 (Theorem 1 in Amini et al. (2008)). *Suppose that an upper bound of the transductive Gibbs risk $R_u^\delta(G_Q)$ is given. Then for any Q and for all $\delta \in (0,1]$, $\forall\theta \geq 0$, with probability at least $1-\delta$, the following bound holds:*

$$R_{\mathcal{U}\wedge\theta}(B_Q) \leq \inf_{\gamma\in(\theta,1]}\left\{\frac{1}{u}\sum_{i=l+1}^{l+u}\mathbb{I}(\theta < m_Q(\mathbf{x}_i) < \gamma) + \frac{1}{\gamma}\left\lfloor K_u^\delta(Q) + M_Q^{\leq}(\theta) - M_Q^{\leq}(\gamma)\right\rfloor_+\right\},$$

where

- $K_u^\delta(Q) = R_u^\delta(G_Q) + \frac{1}{2u}\sum_{i=l+1}^{l+u} m_Q(\mathbf{x}_i) - \frac{1}{2}$,

- $M_Q^{\triangleleft}(t) = \frac{1}{u}\sum_{i=l+1}^{l+u} m_Q(\mathbf{x}_i)\mathbb{I}(m_Q(\mathbf{x}_i) \triangleleft t)$, $(\triangleleft \in \{<, \leq\})$,
- $\lfloor x \rfloor_+ = x \cdot \mathbb{I}(x > 0)$.

The bound is found from the solution of a linear program, where the connection with the Gibbs risk is used as a linear constraint. One can notice that $R_u^{\delta}(G_Q)$ can be evaluated using Theorem 1.2.1. The bound for $R_{\mathcal{U}}(B_Q)$ is derived from Theorem 1.2.2 by noticing that $R_{\mathcal{U}}(B_Q) \leq R_{\mathcal{U}\wedge 0}(B_Q) + \frac{1}{u}\sum_{i=l+1}^{l+u}\mathbb{I}(m_Q(\mathbf{x}_i) = 0)$.

Amini et al. (2008) proposed to apply the derived bound for semi-supervised learning. As it was previously discussed, the choice of a confidence threshold is a bottleneck of the self-learning algorithm. To overcome this, a criterion to select this threshold automatically is considered. More specifically, the unsigned margin is treated as an indicator of confidence, and we are looking for a threshold that minimizes the conditional Bayes error rate defined for a given $\theta > 0$ as:

$$R_{\mathcal{U}|\theta}(B_Q) := \frac{R_{\mathcal{U}\wedge\theta}(B_Q)}{\frac{1}{u}\sum_{i=l+1}^{l+u}\mathbb{I}(m_Q(\mathbf{x}_i) > \theta)}, \tag{1.17}$$

where $R_{\mathcal{U}\wedge\theta}(B_Q)$ is evaluated using Theorem 1.2.2. Thus, the threshold is selected by balancing between the number of pseudo-labeled examples (i.e., those with $m_Q(\mathbf{x}_i) > \theta$) and the bounded error evaluated on them.

Chapter 2

Probabilistic Bounds for the Multi-class Majority Vote Classifier

In this chapter, we present a probabilistic framework for analyzing the majority vote classifier in the multi-class classification scenario with partially labeled data. First, we derive a transductive bound on the multi-class majority vote classifier's error evaluated on the unlabeled examples with a prediction vote higher than a given threshold. The bound is obtained by considering the class confusion matrix as an error indicator and involving the distribution of the classifier's votes over each class. We prove that this bound is tight when it is assumed that the errors of the majority vote classifier are concentrated in a region of a low prediction vote. Then, we analyze the majority vote classifier in the case of imperfectly-labeled data. For this, we introduce a mislabeling error model and derive a connection between the true and imperfect labels. Based on this, we derive a new C-bound on the majority vote error when an imperfect label is given. Finally, the chapter also discusses possible applications of the two proposed bounds for semi-supervised learning and, in particular, for self-learning.

2.1 Introduction

Generalization guarantees of majority vote classifiers are well studied in the binary supervised setting. A common approach is to bound the majority vote risk by twice the Gibbs risk (Langford and Shawe-Taylor, 2002). Many works are focused on deriving tight PAC guarantees for the Gibbs classifier in the inductive case (McAllester, 2003; Maurer, 2004; Catoni, 2007) as well as in the transductive one (Derbeko et al., 2004; Bégin et al., 2014), and applying these results for optimization (Thiemann et al., 2017), linear classifiers (Germain et al., 2009), random forests (Lorenzen et al., 2019), neural networks (Letarte et al., 2019). While this bound can be tight, it reflects only the individual strength of voters, so using it as a minimization criterion often leads to an increase in the test error (Masegosa et al.,

2020). This motivates to opt for bounds that directly upper bound the majority vote error. Amini et al. (2008) derives a transductive bound based on how voters agree on every unlabeled example (reminded in Section 1.2.2), while Lacasse et al. (2007) upper bounds the generalization error by taking additionally into account the error correlation between voters (reminded in Section 1.1.3).

Only few results exist for the multi-class majority vote classifier. In the supervised setting, Morvant et al. (2012) derives generalization guarantees on the confusion matrix' norm based on the matrix concentration inequality proposed by Tropp (2012). Laviolette et al. (2017) extends the C-bound of Lacasse et al. (2007) to the multi-class case. Masegosa et al. (2020) studies tight estimations from data by deriving a relaxed version of Laviolette et al. (2017).

In this chapter, we derive two bounds on the error of the multi-class majority vote classifier in the semi-supervised setting. First, we extend the work of Amini et al. (2008) to the multi-class case by focusing on the class confusion matrix as an error indicator as proposed by Morvant et al. (2012). This bound is obtained by analytically solving a linear program and it comes out that in the case when the majority vote classifier makes most of its errors on examples with low class vote, the obtained bound is tight.

However, the proposed transductive bound can not be used to evaluate the error after learning a semi-supervised classifier. For example, self-learning results in learning a classifier on the labeled examples and the unlabeled examples with pseudo-labels that are potentially erroneous. In order to take explicitly into account possible mislabeling, we consider a mislabeling error model (Chittineni, 1980) and derive our second bound for the majority vote by extending the multi-class C-bound (Laviolette et al., 2017) to the case of imperfectly labeled data. This result is based on analysis of how the true and the imperfect label are connected. Chittineni (1980) derived this connection but only for the oracle classifier. We extend the latter result by deriving another bound that holds for an arbitrary classifier.

The rest of this chapter is organized as follows. In Section 2.2 we introduce the problem statement and the proposed framework. In Section 2.3 we present probabilistic bounds over the transductive risk of the multi-class majority vote classifier. Section 2.4 shows how to derive the C-bound in the probabilistic framework taking into account mislabeling errors. In Section 2.5 we summarize the outcome of this study and discuss the perspectives. Section 2.6 is the appendix of this chapter.

2.2 Framework and Definitions

In this chapter, we focus on the multi-class classification, i.e., $\mathcal{Y} = \{1, \ldots, K\}$, $K \geq 2$. We consider the semi-supervised setting, i.e., we assume an available set of labeled training examples $Z_{\mathcal{L}} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{l} \in (\mathcal{X} \times \mathcal{Y})^l$ and an available set of unlabeled training examples $X_{\mathcal{U}} = \{\mathbf{x}_i\}_{i=l+1}^{l+u} \in \mathcal{X}^u$. Further, we denote by $\mathbf{0}_K$ the zero vector of size K, $\mathbf{0}_{K,K}$ is the zero matrix of size $K \times K$ and $n := l + u$.

As before, we consider a fixed set of classifiers $\mathcal{H} = \{h | h : \mathcal{X} \to \mathcal{Y}\}$, called the hypothesis space. Over $\mathcal{H}$, the prior Q_0 and the posterior Q are defined respectively before and after observing the training set. We focus on two classifiers: the majority vote classifier (the Bayes classifier) defined for all $\mathbf{x} \in \mathcal{X}$ as:

$$B_Q(\mathbf{x}) := \underset{c \in \{1,\dots,K\}}{\operatorname{argmax}} \left[\mathbb{E}_{h \sim Q} \mathbb{I}(h(\mathbf{x}) = c) \right], \tag{2.1}$$

and, the stochastic *Gibbs classifier* G_Q that for every $\mathbf{x} \in \mathcal{X}$ predicts the label using a randomly chosen classifier $h \in \mathcal{H}$ according to Q.

The goal of learning is formulated as to choose a posterior distribution Q over $\mathcal{H}$ based on the training set $Z_{\mathcal{L}} \cup X_{\mathcal{U}}$ such that the classifier B_Q will have the smallest possible error value. In Section 2.3 our objective is transductive, while the inductive setting is considered in Section 2.4. As opposed to works of Derbeko et al. (2004); Bégin et al. (2014); Amini et al. (2008) who considered the deterministic case, we consider the more general *probabilistic* case assuming possibility of multiple outcomes for each example.

To measure confidence of the majority vote classifier in its prediction, the notions of class votes and margin are further considered. Given an observation $\mathbf{x}$, we define a vector of *class votes* $\mathbf{v}_{\mathbf{x}} = (v_Q(\mathbf{x}, c))_{c=1}^K$ where the c-th component corresponds to the total vote given to the class c:

$$v_Q(\mathbf{x}, c) := \mathbb{E}_{h \sim Q} \mathbb{I}(h(\mathbf{x}) = c) = \sum_{h : h(\mathbf{x}) = c} Q(h).$$

In practice, the vote $v_Q(\mathbf{x}, c)$ can be regarded as an estimation of the posterior probability $P(Y = c | \mathbf{X} = \mathbf{x})$; a large value indicates high confidence of the classifier that the true label of $\mathbf{x}$ is c.
Given an observation $\mathbf{x}$, we consider the multi-class *margin* that is defined in the following way:

$$M_Q(\mathbf{x}, y) := \mathbb{E}_{h \sim Q} \mathbb{I}(h(\mathbf{x}) = y) - \max_{\substack{c \in \mathcal{Y} \\ c \neq y}} \mathbb{E}_{h \sim Q} \mathbb{I}(h(\mathbf{x}) = c) = v_Q(\mathbf{x}, y) - \max_{\substack{c \in \mathcal{Y} \\ c \neq y}} v_Q(\mathbf{x}, c). \tag{2.2}$$

The margin measures a gap between the vote of the true class and the maximal vote among all other classes. Similarly to the binary case, a strictly positive margin for an example $\mathbf{x}$ implies its correct classification.

One can see that analysis of the majority vote becomes more challenging in the multi-class case. The definition of the Bayes classifier and its margin contain respectively argmax and max operators. Moreover, the class votes can be described just by one number in the binary case, since $v_Q(\mathbf{x}, -1) = 1 - v_Q(\mathbf{x}, 1)$, whereas in the multi-class case it becomes essential to consider the whole class vote vector. Also some properties of the binary majority vote do not hold in the multi-class case. For

example, in the binary case, $v_Q(\mathbf{x}, c) > 1/2$ is a necessary and sufficient condition that c is the majority vote class, whereas in the multi-class case is just a sufficient condition.

2.3 Probabilistic Transductive Bounds

In this section, we derive guarantees for the multi-class majority vote classifier in the transductive setting (Vapnik, 1982, 1998), i.e., when the error is evaluated on the unlabeled set $X_{\mathcal{U}}$ only. The proposed bounds assumes that the majority vote classifier makes mistake on low class votes and thereby use votes as indicators of confidence.

2.3.1 Transductive conditional risk

At first, we show how to upper bound the risk evaluated conditionally to the values of the true and the predicted class. Given a classifier h, for each class pair $(i, j) \in \{1, \ldots, K\}^2$ such that $i \neq j$, the *transductive conditional risk* is defined as follows:

$$R_{\mathcal{U}}(h, i, j) := \frac{1}{u_i} \sum_{\mathbf{x} \in X_{\mathcal{U}}} P(Y = i | X = \mathbf{x}) \mathbb{I}(h(\mathbf{x}) = j),$$

where $u_i = \sum_{\mathbf{x} \in X_{\mathcal{U}}} P(Y = i | X = \mathbf{x})$ is the expected number of unlabeled observations from the class $i \in \{1, \ldots, K\}$. The value of $R_{\mathcal{U}}(h, i, j)$ indicates the expected proportion of unlabeled examples that are classified to the class j being from the class i. We call $R_{\mathcal{U}}(B_Q, i, j)$ as the *transductive Bayes conditional risk*. The *transductive Gibbs conditional risk* is defined in expectation over Q for all $(i, j) \in \{1, \ldots, K\}^2$, $i \neq j$ by:

$$R_{\mathcal{U}}(G_Q, i, j) := \mathbb{E}_{h \sim Q} R_{\mathcal{U}}(h, i, j).$$

In addition, we define the transductive *joint* Bayes conditional risk for a threshold vector $\boldsymbol{\theta} \in [0, 1]^K$, for $(i, j) \in \{1, \ldots, K\}^2$, $i \neq j$, as follows:

$$R_{\mathcal{U} \wedge \boldsymbol{\theta}}(B_Q, i, j) := \frac{1}{u_i} \sum_{\mathbf{x} \in X_{\mathcal{U}}} P(Y = i | X = \mathbf{x}) \mathbb{I}(B_Q(\mathbf{x}) = j) \mathbb{I}(v_Q(\mathbf{x}, j) \geq \theta_j). \quad (2.3)$$

If the Bayes classifier makes mistakes, i.e., outputs the class j when the true class is i, on the examples with low values of $v_Q(\mathbf{x}, j)$, then the joint risk computes the probability to make the conditional error on confident observations when a large enough θ_j is set with respect to the distribution of $v_Q(\mathbf{x}, j)$.

The following Lemma 2.3.1 connects the conditional Gibbs risk and the joint Bayes conditional risk by considering a conditional Bayes error regarding a certain class vote.

Lemma 2.3.1. *For $c \in \{1, \dots, K\}$, let $\Gamma_c = \{\gamma_c \in [0,1] | \ \exists\, \mathbf{x} \in \mathbb{X}_{\mathcal{U}} : \gamma_c = v_Q(\mathbf{x}, c)\}$ be the set of unique votes for the unlabeled examples to the class c. Let enumerate its elements such that they form an ascending order:*

$$\gamma_c^{(1)} \leq \gamma_c^{(2)} \leq \cdots \leq \gamma_c^{(N_c)},$$

where $N_c := |\Gamma_c|$. Denote $b_{i,j}^{(t)} := \frac{1}{u_i} \sum_{\mathbf{x} \in \mathbb{X}_{\mathcal{U}}} P(Y = i | X = \mathbf{x}) \mathbb{I}(B_Q(\mathbf{x}) = j) \mathbb{I}(v_Q(\mathbf{x}, j) = \gamma_j^{(t)})$.
Then, for all $(i,j) \in \{1, \dots, K\}^2$:

$$R_{\mathcal{U}}(G_Q, i, j) \geq K_{i,j} := \sum_{t=1}^{N_j} b_{i,j}^{(t)} \gamma_j^{(t)}, \tag{2.4}$$

$$R_{\mathcal{U} \wedge \boldsymbol{\theta}}(B_Q, i, j) = \sum_{t=k_j+1}^{N_j} b_{i,j}^{(t)}, \tag{2.5}$$

where $k_j = \max\{t | \gamma_j^{(t)} < \theta_j\}$ with $\max(\emptyset) = 0$ by convention.

The proof is provided in Appendix 2.6.1. Following Lemma 2.3.1, we derive a bound on the Bayes conditional risk using the class vote distribution.

Theorem 2.3.2. *Let B_Q be the Q-weighted majority vote classifier defined by Eq.* (2.1). *Then for any Q, for all $\boldsymbol{\theta} \in [0,1]^K$, for all $(i,j) \in \{1, \dots, K\}^2$ we have:*

$$R_{\mathcal{U} \wedge \boldsymbol{\theta}}(B_Q, i, j) \leq \inf_{\gamma \in [\theta_j, 1]} \left\{ I_{i,j}^{(\leq, <)}(\theta_j, \gamma) + \frac{1}{\gamma} \left| K_{i,j} - M_{i,j}^{<}(\gamma) + M_{i,j}^{\leq}(\theta_j) \right|_+ \right\}, \quad (\mathrm{TB}_{i,j})$$

where

- $K_{i,j} = \frac{1}{u_i} \sum_{\mathbf{x} \in X_{\mathcal{U}}} P(Y = i | X = \mathbf{x}) v_Q(\mathbf{x}, j) \mathbb{I}(B_Q(\mathbf{x}) = j)$ *is the transductive Gibbs conditional risk evaluated on the examples for which the majority vote class is j,*
- $I_{i,j}^{(\leq, <)}(\theta_j, \gamma) = \frac{1}{u_i} \sum_{\mathbf{x} \in \mathbb{X}_{\mathcal{U}}} P(Y = i | X = \mathbf{x}) \mathbb{I}(\theta_j \leq v_Q(\mathbf{x}, j) < \gamma)$ *is the expected proportion of unlabeled examples from the class i with $v_Q(\mathbf{x}, j)$ in interval $[\theta_j, \gamma)$,*
- $M_{i,j}^{<}(s) = \frac{1}{u_i} \sum_{\mathbf{x} \in \mathbb{X}_{\mathcal{U}}} P(Y = i | X = \mathbf{x}) v_Q(\mathbf{x}, j) \mathbb{I}(v_Q(\mathbf{x}, j) < s)$, *is the average of j-votes in the class i that less than s.*

Proof. We would like to find an upper bound for the joint Bayes conditional risk. Hence, for all $(i,j) \in \{1, \dots, K\}^2$, for all $\boldsymbol{\theta} \in [0,1]^K$, we consider the case when the mistake is maximized. Then, using Lemma 2.3.1:

$$R_{\mathcal{U} \wedge \boldsymbol{\theta}}(B_Q, i, j) = \sum_{t=k_j+1}^{N_j} b_{i,j}^{(t)} \leq \max_{b_{i,j}^{(1)}, \dots, b_{i,j}^{(N_j)}} \sum_{t=k_j+1}^{N_j} b_{i,j}^{(t)}, \tag{2.6}$$

with $k_j = \max\{t|\gamma_j^{(t)} < \theta_j\}\mathbb{I}(\{t|\gamma_j^{(t)} < \theta_j\} \neq \emptyset)$.

Let $B_{i,j}^{(t)} = \sum_{\mathbf{x}\in\mathcal{X}_\mathcal{U}} P(Y=i|X=\mathbf{x})\mathbb{I}(v_Q(\mathbf{x},j) = \gamma_j^{(t)})/u_i$. Then, it can be noticed that $0 \leq b_{i,j}^{(t)} \leq B_{i,j}^{(t)}$. Remember that $K_{i,j}$ can also be written as $\sum_{t=1}^{N_j} b_{i,j}^{(t)}\gamma_j^{(t)}$. Hence the bound defined by Eq. (2.6) should satisfy the following linear program :

$$\max_{b_{i,j}^{(1)},\ldots,b_{i,j}^{(N_j)}} \sum_{t=k_j+1}^{N_j} b_{i,j}^{(t)} \tag{2.7}$$
$$\text{s.t.} \quad \forall t,\ 0 \leq b_{i,j}^{(t)} \leq B_{i,j}^{(t)} \text{ and } \sum_{t=1}^{N_j} b_{i,j}^{(t)}\gamma_j^{(t)} = K_{i,j}.$$

The solution of (2.7) can be solved analytically and it is attained for:

$$b_{i,j}^{(t)} = \min\left(B_{i,j}^{(t)}, \left\lfloor \frac{1}{\gamma_j^{(t)}}(K_{i,j} - \sum_{k<w<t} \gamma_j^{(w)} B_{i,j}^{(w)}) \right\rfloor_+\right)\mathbb{I}(t \leq k_j). \tag{2.8}$$

For the sake of a better presentation, the proof of this solution is deferred to Appendix 2.6, Lemma 2.6.1. Further, we can notice that, for all $(i,j) \in \{1,\ldots,K\}^2$,

$$\sum_{k_j<w<t} \gamma_j^{(w)} B_{i,j}^{(w)} = M_{i,j}^{<}(\gamma_j^{(t)}) - M_{i,j}^{<}(\theta_j).$$

Let $p = \max\{t|K_{i,j} - M_{i,j}^{<}(\gamma_j^{(t)}) + M_{i,j}^{<}(\theta_j) > 0\}$. Then, Eq. (2.8) can be re-written as follows:

$$b_{i,j}^{(t)} = \begin{cases} 0 & t \leq k_j \\ B_{i,j}^{(t)} & k_j + 1 \leq t < p \\ \frac{1}{\gamma_j^{(p)}}(K_{i,j} - M_{i,j}^{<}(\gamma_j^{(p)}) + M_{i,j}^{<}(\theta_j)) & t = p \\ 0 & t > p. \end{cases} \tag{2.9}$$

Notice that $\sum_{t=k_j+1}^{p-1} B_{i,j}^{(t)} = I_{i,j}^{(\leq,<)}(\theta_j, \gamma_j^{(p)})$. Using this fact as well as Eq. (2.9), we infer:

$$R_{\mathcal{U}\wedge\boldsymbol{\theta}}(B_Q, i, j) \leq I_{i,j}^{(\leq,<)}(\theta_j, \gamma_j^{(p)}) + \frac{1}{\gamma_j^{(p)}}(K_{i,j} - M_{i,j}^{<}(\gamma_j^{(p)}) + M_{i,j}^{<}(\theta_j)).$$

Consider the function

$$\gamma \mapsto U_{i,j}(\gamma) := I_{i,j}^{(\leq,<)}(\theta_j, \gamma) + \frac{1}{\gamma}\left\lfloor K_{i,j} - M_{i,j}^{<}(\gamma) + M_{i,j}^{<}(\theta_j)\right\rfloor_+.$$

To prove the theorem, it remains to verify that, for all $(i,j) \in \{1,\ldots,K\}^2$, for all $\gamma \in [\theta_j, 1]$, $U_{i,j}(\gamma_j^{(p)}) \leq U_{i,j}(\gamma)$. For this, consider $\gamma_j^{(w)}$ with $w \in \{1,\ldots,N_j\}$.

If $w > p$, then $U_{i,j}(\gamma_j^{(p)}) \leq I_{i,j}^{(\leq,\leq)}(\theta_j,\gamma) \leq U_{i,j}(\gamma_j^{(w)})$.

If $w < p$, then

$$\begin{aligned} U_{i,j}(\gamma_j^{(p)}) - U_{i,j}(\gamma_j^{(w)}) &= \sum_{t=w}^{p} b_{i,j}^{(t)} - \frac{1}{\gamma_j^{(w)}} \left(K_{i,j} - M_{i,j}^{\leq}(\gamma_j^{(w)}) + M_{i,j}^{\leq}(\theta_j) \right) \\ &= \sum_{t=w}^{p} b_{i,j}^{(t)} - \frac{1}{\gamma_j^{(w)}} \left(\sum_{t=k+1}^{p} b_{i,j}^{(t)} \gamma_j^{(t)} - \sum_{t=k+1}^{w-1} \gamma_j^{(t)} b_{i,j}^{(t)} \right) \\ &= \frac{1}{\gamma_j^{(w)}} \left(\sum_{t=w}^{p} b_{i,j}^{(t)} \gamma_j^{(w)} - \sum_{t=w}^{p} b_{i,j}^{(t)} \gamma_j^{(t)} \right) \leq 0. \end{aligned}$$

which completes the proof. □

Following this result, a transductive bound for the joint Bayes conditional risk can be found by arranging the class votes in an ascending order and considering the linear program (2.7), where the connection with the Gibbs classifier is used as a linear constraint. Furthermore, as the bound is the infimum of the function $U_{i,j}$ on the interval $[\theta_j, 1]$ it can be computed in practice without solving the linear program explicitly.

When $\theta_j = 0$, a bound over the transductive Bayes conditional risk is directly obtained from ($\mathrm{TB}_{i,j}$) by noticing that $M_{i,j}^{\leq}(0) = 0$ in this case:

$$R_{\mathcal{U}}(B_Q, i, j) \leq \inf_{\gamma \in [0,1]} \left\{ I_{i,j}^{(\leq,<)}(0,\gamma) + \frac{1}{\gamma} \left| K_{i,j} - M_{i,j}^{\leq}(\gamma) \right|_+ \right\}. \tag{2.10}$$

We note that in the binary case (Amini et al., 2008), the transductive Gibbs risk used inside the linear program can be bounded either by the PAC-Bayesian bound (Derbeko et al., 2004; Bégin et al., 2014) or by 1/2 (the worst possible error of the binary classifier), which allows to compute the transductive bound. In the multi-class case, the bound can be evaluated only by approximating the posterior probabilities. Once we estimate the posterior probability, $K_{i,j}$ and the transductive conditional Gibbs risk are also directly approximated.

2.3.2 Transductive confusion matrix and transductive error rate

In this section, based on Theorem 2.3.2, we derive bounds for two other error measures: the *error rate* and the *confusion matrix* (Morvant et al., 2012). We define the transductive error rate and the transductive *joint* error rate of the Bayes classifier

B_Q over the unlabeled set $\mathrm{X}_{\mathcal{U}}$ given a vector $\boldsymbol{\theta} = (\theta_c)_{c=1}^K \in [0,1]^K$, as:

$$R_{\mathcal{U}}(B_Q) := \frac{1}{u} \sum_{\mathbf{x} \in \mathrm{X}_{\mathcal{U}}} \sum_{\substack{c \in \{1,\dots,K\} \\ c \neq B_Q(\mathbf{x})}} P(Y = c | \mathbf{X} = \mathbf{x}),$$

$$R_{\mathcal{U}^\wedge\boldsymbol{\theta}}(B_Q) := \frac{1}{u} \sum_{\mathbf{x} \in \mathrm{X}_{\mathcal{U}}} \sum_{\substack{c \in \{1,\dots,K\} \\ c \neq B_Q(\mathbf{x})}} P(Y = c | X = \mathbf{x}) \mathbb{I}(v_Q(\mathbf{x}, B_Q(\mathbf{x})) \geq \theta_{B_Q(\mathbf{x})}). \quad (2.11)$$

Then, we define the *transductive joint Bayes confusion matrix* for $\boldsymbol{\theta} \in [0,1]^K$, and $(i,j) \in \{1,\dots,K\}^2$, as follows:

$$\left[\mathbf{C}_h^{\mathcal{U}^\wedge\boldsymbol{\theta}}\right]_{i,j} = \begin{cases} 0 & i = j, \\ R_{\mathcal{U}^\wedge\boldsymbol{\theta}}(h,i,j) & i \neq j. \end{cases}$$

The following proposition links the error rate with the joint confusion matrix:

Proposition 2.3.3. *Let B_Q be the majority vote classifier. Given a vector $\boldsymbol{\theta} \in [0,1]^K$, for $\mathbf{p} := \{u_i/u\}_{i=1}^K$, where $u_i = \sum_{\mathbf{x} \in \mathrm{X}_{\mathcal{U}}} P(Y = i | X = \mathbf{x})$, we have:*

$$R_{\mathcal{U}^\wedge\boldsymbol{\theta}}(B_Q) = \left\| \left(\mathbf{C}_{B_Q}^{\mathcal{U}^\wedge\boldsymbol{\theta}} \right)^\intercal \mathbf{p} \right\|_1. \quad (2.12)$$

Proof. To prove Eq. (2.12), combine the definition of transductive joint Bayes conditional risk given in Eq. (2.3) and Eq. (2.11) as follows:

$$\begin{aligned} R_{\mathcal{U}^\wedge\boldsymbol{\theta}}(B_Q) &= \frac{1}{u} \sum_{i=1}^K \sum_{\substack{j=1 \\ j \neq i}}^K \sum_{\mathbf{x} \in \mathrm{X}_{\mathcal{U}}} P(Y = i | X = \mathbf{x}) \mathbb{I}(B_Q(\mathbf{x}) = j) \mathbb{I}(v_Q(\mathbf{x}, j) \geq \theta_j) \\ &= \sum_{i=1}^K \frac{u_i}{u} \sum_{\substack{j=1 \\ j \neq i}}^K R_{\mathcal{U}^\wedge\boldsymbol{\theta}}(B_Q, i, j) = \left\| \left(\mathbf{C}_{B_Q}^{\mathcal{U}^\wedge\subseteq} \right)^\intercal \mathbf{p} \right\|_1. \end{aligned}$$

□

From Theorem 2.3.2, we derive corresponding transductive bounds for the confusion matrix norm and the error rate of the Bayes classifier. To simplify notations, we introduce a matrix $\mathbf{U}_{\boldsymbol{\theta}}$ of size $K \times K$ with zeros on the main diagonal and the following (i,j)-entries, $i \neq j$:

$$[\mathbf{U}_{\boldsymbol{\theta}}]_{i,j} := \inf_{\gamma \in [\theta_j, 1]} \left\{ I_{i,j}^{(\leq,<)}(\theta_j, \gamma) + \frac{1}{\gamma} \left| (K_{i,j} - M_{i,j}^<(\gamma) + M_{i,j}^<(\theta_j)) \right|_+ \right\},$$

which corresponds to the transductive bound proposed in Theorem 2.3.2.

Corollary 2.3.4. *For all $\boldsymbol{\theta} \in [0,1]^K$, we have:*

$$\|\mathbf{C}_{B_Q}^{\mathcal{U}\wedge\boldsymbol{\theta}}\| \leq \|\mathbf{U}_{\boldsymbol{\theta}}\|. \tag{2.13}$$

Moreover, we have the following bound:

$$R_{\mathcal{U}\wedge\boldsymbol{\theta}}(B_Q) \leq \left\|\mathbf{U}_{\boldsymbol{\theta}}^{\intercal}\,\mathbf{p}\right\|_1. \tag{2.14}$$

where $\|.\|$ is the spectral norm, $\mathbf{p} = \{u_i/u\}_{i=1}^K$ with $u_i = \sum_{\mathbf{x}\in\mathrm{X}_{\mathcal{U}}} P(Y=i|X=\mathbf{x})$.

Proof. The confusion matrix $\mathbf{C}_{B_Q}^{\mathcal{U}\wedge\boldsymbol{\theta}}$ is always non-negative, and from Theorem 2.3.2, each of its entries is smaller than the corresponding entry of $\mathbf{U}_{\boldsymbol{\theta}}$. Hence, from the property of spectral norm for two positive matrices $\mathbf{A}$ and $\mathbf{B}$:

$$\mathbf{0}_{K,K} \preceq \mathbf{A} \preceq \mathbf{B} \Rightarrow \|\mathbf{A}\| \leq \|\mathbf{B}\|,$$

where $\mathbf{A} \preceq \mathbf{B}$ denotes that each element of $\mathbf{A}$ is smaller than the corresponding element of $\mathbf{B}$, we deduce Eq. (2.13).

With the same computations, we observe the following inequality:

$$\left(\mathbf{C}_{B_Q}^{\mathcal{U}\wedge\boldsymbol{\theta}}\right)^{\intercal}\,\mathbf{p} \leq \mathbf{U}_{\boldsymbol{\theta}}^{\intercal}\,\mathbf{p}.$$

Elements of the left vector are non-negative. Hence the inequality holds for the ℓ_1-norm, and taking into account Proposition 2.3.3 we infer:

$$R_{\mathcal{U}\wedge\boldsymbol{\theta}}(B_Q) = \left\|\left(\mathbf{C}_{B_Q}^{\mathcal{U}\wedge\boldsymbol{\theta}}\right)^{\intercal}\,\mathbf{p}\right\|_1 \leq \left\|\mathbf{U}_{\boldsymbol{\theta}}^{\intercal}\,\mathbf{p}\right\|_1.$$

□

Note that the transductive bound of the Bayes error rate is obtained from Eq. (2.14) by taking $\boldsymbol{\theta}$ as the zero vector $\mathbf{0}_K$:

$$R_{\mathcal{U}}(B_Q) \leq \left\|\mathbf{U}_{\mathbf{0}_K}^{\intercal}\,\mathbf{p}\right\|_1. \tag{TB}$$

2.3.3 Tightness Guarantees

In this section, we assume that the Bayes classifier makes most of its error on unlabeled examples with a low prediction vote, i.e., class votes can be considered as indicators of confidence. In the following proposition, we show that the bound becomes tight under certain conditions. We remind that $\Gamma_j = \{\gamma_j^{(t)}\}$ is the set of unique votes for the unlabeled examples to the class j, and $b_{i,j}^{(t)}$ corresponds to the Bayes conditional risk on the examples with the vote $\gamma_j^{(t)}$ (see Lemma 2.3.1 for more details).

Proposition 2.3.5. *Let* $\Gamma_j^\tau := \{\gamma_j^{(t)} \in \Gamma_j | b_{i,j}^{(t)} > \tau\}$, *where* $\tau \in [0,1]$ *is a given threshold. If there exists a lower bound* $C \in [0,1]$ *such that for all* $\gamma \in \Gamma_j^\tau$*:*

$$\sum_{\mathbf{x}\in X_{\mathcal{U}}} P(Y=i|X=\mathbf{x})\mathbb{I}(B_Q(\mathbf{x})=j)\mathbb{I}(v_Q(\mathbf{x},j)<\gamma) \geq C \sum_{\mathbf{x}\in X_{\mathcal{U}}} P(Y=i|X=\mathbf{x})\mathbb{I}(v_Q(\mathbf{x},j)<\gamma), \quad (2.15)$$

then, the following inequality holds:

$$[\mathbf{U}_{\mathbf{0}_K}]_{i,j} - R_{\mathcal{U}}(B_Q,i,j) \leq \frac{1-C}{C} R_{\mathcal{U}}(B_Q,i,j) + r_{i,j}\left(\frac{1}{\gamma_j^*}-1\right),$$

where

- $\gamma_j^* := \sup\{\gamma_j^{(t)} \in \Gamma_j^\tau\}$ *is the highest vote which satisfies* $b_{i,j}^{(t)} > \tau$*, and*
- $r_{i,j} := \sum_{\mathbf{x}\in X_{\mathcal{U}}} P(Y=i|X=\mathbf{x}) v_Q(\mathbf{x},j)\mathbb{I}(B_Q(\mathbf{x})=j)\mathbb{I}(v_Q(\mathbf{x},j)>\gamma_j^*)/u_i$ *corresponds to the average of j-votes in the class i that greater than* γ_j^* *and on which the Bayes classifier makes the conditional mistake.*

Proof. First, it can be proved that for all $\mathbf{x} \in X_{\mathcal{U}}$, for all $(i,j) \in \{1,\ldots,K\}^2$, the following inequality holds:

$$R_{\mathcal{U}}(B_Q,i,j) \geq \frac{1}{u_i}\sum_{\mathbf{x}\in X_{\mathcal{U}}} P(Y=i|X=\mathbf{x})\mathbb{I}(B_Q(\mathbf{x})=j)\mathbb{I}(v_Q(\mathbf{x},j)<\gamma^*) + \frac{1}{\gamma^*}\left\lfloor \lfloor K_{i,j} - M_{i,j}^{<}(\gamma^*)\rfloor_+ - r_{i,j}\right\rfloor_+ + r_{i,j}, \quad (2.16)$$

where $\gamma^* := \sup\{\gamma \in \Gamma_j | \sum_{\mathbf{x}\in X_{\mathcal{U}}} P(Y=i|X=\mathbf{x})\mathbb{I}(B_Q(\mathbf{x})=j)\mathbb{I}(v_Q(\mathbf{x},j)=\gamma)/u_i > \tau\}$. We prove this result in Lemma 2.6.2 in Section 2.6. We remind that $I_{i,j}^{(\leq,<)}(0,\gamma^*) = \frac{1}{u_i}\sum_{\mathbf{x}\in X_{\mathcal{U}}} P(Y=i|X=\mathbf{x})\mathbb{I}(v_Q(\mathbf{x},j)<\gamma^*)$. Then, combining Eq. (2.16) and Eq. (2.15) we deduce the following:

$$R_{\mathcal{U}}(B_Q,i,j) \geq C\, I_{i,j}^{(\leq,<)}(0,\gamma^*) + \frac{1}{\gamma^*}\left\lfloor \lfloor K_{i,j} - M_{i,j}^{<}(\gamma^*)\rfloor_+ - r_{i,j}\right\rfloor_+ + r_{i,j}. \quad (2.17)$$

By definition of $\mathbf{U}_{\mathbf{0}_K}$ we have, for all $(i,j) \in \{1,\ldots,K\}^2$,

$$[\mathbf{U}_{\mathbf{0}_K}]_{i,j} \leq I_{i,j}^{(\leq,<)}(0,\gamma^*) + \frac{1}{\gamma^*}\left\lfloor K_{i,j} - M_{i,j}^{<}(\gamma^*)\right\rfloor_+. \quad (2.18)$$

Subtracting Eq. (2.17) from Eq. (2.18) we obtain:

$$[\mathbf{U}_{\mathbf{0}_K}]_{i,j} - R_{\mathcal{U}}(B_Q, i, j) \leq (1 - C) I_{i,j}^{(\leq,<)}(0, \gamma^*) + \frac{1}{\gamma^*} \left(\left\lfloor K_{i,j} - M_{i,j}^{<}(\gamma^*) \right\rfloor_+ - \left\lfloor \lfloor K_{i,j} - M_{i,j}^{<}(\gamma^*) \rfloor_+ - r_{i,j} \right\rfloor_+ \right) - r_{i,j}.$$

We can notice that for all $a, b \in \mathbb{R}^+ : \; b - \lfloor b - a \rfloor_+ \leq a$. Then, we have:

$$[\mathbf{U}_{\mathbf{0}_K}]_{i,j} - R_{\mathcal{U}}(B_Q, i, j) \leq (1 - C) I_{i,j}^{(\leq,<)}(0, \gamma^*) + r_{i,j} \left(\frac{1}{\gamma^*} - 1 \right). \tag{2.19}$$

Also, from Eq. (2.17) one can derive:

$$\begin{aligned} I_{i,j}^{(\leq,<)}(0, \gamma^*) &\leq \frac{1}{C} \left(R_{\mathcal{U}}(B_Q, i, j) - \frac{1}{\gamma^*} \left\lfloor \lfloor K_{i,j} - M_{i,j}^{<}(\gamma^*) \rfloor_+ - r_{i,j} \right\rfloor_+ - r_{i,j} \right) \\ &\leq \frac{R_{\mathcal{U}}(B_Q, i, j)}{C}. \end{aligned} \tag{2.20}$$

Taking into account Eq. (2.19) and Eq. (2.20), we infer:

$$[\mathbf{U}_{\mathbf{0}_K}]_{i,j} - R_{\mathcal{U}}(B_Q, i, j) \leq \frac{1 - C}{C} R_{\mathcal{U}}(B_Q, i, j) + r_{i,j} \left(\frac{1}{\gamma^*} - 1 \right).$$

□

This proposition states that if Eq. (2.15) holds, the difference between the transductive Bayes conditional risk and its upper bound does not exceed an expression that depends on a constant C and a threshold τ. When the majority vote classifier makes most of its mistake for the class j on observations with a low value of $v_Q(\mathbf{x}, j)$, with a reasonable choice of τ, $r_{i,j}$ and γ_j^* are decreasing. This also implies that Eq. (2.15) accepts a high value C (close to 1) and the bound will be tighter. The closer our framework to the deterministic one, the closer $r_{i,j}$ will be to 0 (in the deterministic case, τ can be set to 0, so $r_{i,j}$ will be 0), so the bound becomes tight. Although our bound is tight only under the condition of making mistakes on low prediction votes, the assumption is reasonable from the theoretical point of view, since if for some observation the Bayes classifier gives a relatively high vote to the class j, we expect that the observation is most probably from this class and not from the class i. From the practical point of view, this assumption requires the learning model to be well calibrated (Gebel, 2009).

2.4 Probabilistic C-Bound with Imperfect Labels

The transductive bound (TB) can give us insights for self-learning, since the transductive error can be evaluated on different values of the threshold $\boldsymbol{\theta}$. However, the

pseudo-labels obtained by self-learning may be still erroneous, and we do not know how to evaluate the classification error in this noisy case. In addition, (TB) can be regarded as a first-order bound, since it is linearly dependent on the classifier' votes, so it does not take into account the correlation between hypotheses. In this section, we overcome these two issues by deriving a new probabilistic C-bound on the generalization error in the presence of imperfect labels.

2.4.1 Ordinary C-Bound

Lacasse et al. (2007) proposed to upper bound the Bayes error by taking into account the mean and the variance of the prediction margin, which, we recall Eq. (2.2), is defined as $v_Q(\mathbf{x}, y) - \max_{c \in \mathcal{Y} \setminus \{y\}} v_Q(\mathbf{x}, c)$. A similar result was obtained in a different context by Breiman (2001). Laviolette et al. (2017) extended this bound to the multi-class case.

We remind that the generalization error is defined in the probabilistic setting as follows:

$$R(B_Q) := \mathbb{E}_{\mathbf{X}} \sum_{c \in \mathcal{Y} \setminus \{B_Q(\mathbf{x})\}} P(Y = c | \mathbf{X} = \mathbf{x}) = \mathbb{E}_{\mathbf{X}}[1 - P(Y = B_Q(\mathbf{x}) | \mathbf{X} = \mathbf{x})].$$

Below, we show how the C-bound of Laviolette et al. (2017) is derived in the probabilistic setting. The probabilistic formulation gives us insights that the bound can be applied not only in the classical supervised setting, but also in the case when every training example is supplied with probabilistic (soft) label which may arise in medical or crowdsourcing applications (Peng et al., 2014).

Theorem 2.4.1. *Let M be a random variable such that $[M|\mathbf{X} = \mathbf{x}]$ is a discrete random variable that is equal to the margin $M_Q(\mathbf{x}, c)$ with probability $P(Y = c|\mathbf{X} = \mathbf{x})$, $c = \{1, \dots, K\}$. Let μ_1^M and μ_2^M be the first and the second statistical moments of the random variable M, respectively. Then, for all choice of Q on a hypothesis space $\mathcal{H}$, and for all distributions $P(\mathbf{X})$ over $\mathcal{X}$ and $P(Y|\mathbf{X})$ over $\mathcal{Y}$, such that $\mu_1^M > 0$, we have:*

$$R(B_Q) \leq 1 - \frac{(\mu_1^M)^2}{\mu_2^M}. \tag{CB}$$

Proof. At first, we show that $R(B_Q) = P(M \leq 0)$. For a fixed $\mathbf{x}$, one can notice that

$$P(M \leq 0 | \mathbf{X} = \mathbf{x}) = \sum_{c=1}^{K} P(Y = c | \mathbf{X} = \mathbf{x}) \mathbb{I}(M_Q(\mathbf{x}, c) \leq 0) = \sum_{c \in \mathcal{Y} \setminus \{B_Q(\mathbf{x})\}} P(Y = c | \mathbf{X} = \mathbf{x}).$$

Applying the total probability law, we obtain:

$$P(M \leq 0) = \int_{\mathcal{X}} P(M \leq 0, \mathbf{X} = \mathbf{x}) \mathrm{d}\mathbf{x} = \mathbb{E}_{\mathbf{X}} P(M \leq 0 | \mathbf{X} = \mathbf{x}) = R(B_Q). \tag{2.21}$$

By applying the Cantelli-Chebyshev inequality (Lemma 2.6.3 in Appendix 2.6), we deduce:

$$P(M \leq 0) \leq \frac{\mu_2^M - (\mu_1^M)^2}{\mu_2^M - (\mu_1^M)^2 + (\mu_1^M)^2} = 1 - \frac{(\mu_1^M)^2}{\mu_2^M}. \tag{2.22}$$

Combining Eq. (2.21) and Eq. (2.22) gives the bound. □

The main advantage of the C-bound is the involvement of the second margin moment, which can be related to correlations between hypotheses' errors (Lacasse et al., 2007; Masegosa et al., 2020).

2.4.2 Mislabeling Error Model

When training data contains an unknown portion of imperfect labels, a common approach is to explicitly model mislabeling errors (Chittineni, 1980). The approach has been considered in both supervised (Natarajan et al., 2013; Scott, 2015; Xia et al., 2019) and semi-supervised settings (Amini and Gallinari, 2003). In most cases, a focus is on the estimation of the mislabeling errors to train a supervised classifier, and theoretical studies are limited by the binary case (Natarajan et al., 2013; Scott, 2015). We follow Chittineni (1980) and consider a general multi-class framework to evaluate the classification error.

We consider an imperfect output $\hat{Y}$, which has a different distribution from the true output Y. The label imperfection is summarized through the *mislabeling matrix* $\mathbf{P} = (p_{j,c})_{1 \leq j,c \leq K}$, defined by:

$$P(\hat{Y} = j | Y = c) := p_{j,c} \quad \forall (j,c) \in \{1, \ldots, K\}^2, \tag{2.23}$$

where $\sum_{j=1}^{K} p_{j,c} = 1$. Additionally, we assume that $\hat{Y}$ does not influence the true class distribution: $P(\mathbf{X}|Y, \hat{Y}) = P(\mathbf{X}, Y)$. This implies that

$$P(\hat{Y} = j | \mathbf{X} = \mathbf{x}) = \sum_{c=1}^{K} p_{j,c} P(Y = c | \mathbf{X} = \mathbf{x}). \tag{2.24}$$

This class-related model is a common approach to deal with the label imperfection (Amini and Gallinari, 2003; Natarajan et al., 2013; Scott, 2015).

In this section, we derive a bound that connects the error of the true and the imperfect label in misclassifying a particular example $\mathbf{x} \in \mathcal{X}$. For this, we use the following notations:

$$r(h, \mathbf{x}) = \sum_{\substack{c \in \mathcal{Y} \\ c \neq h(\mathbf{x})}} P(Y = c | \mathbf{X} = \mathbf{x}), \qquad \hat{r}(h, \mathbf{x}) = \sum_{\substack{c \in \mathcal{Y} \\ c \neq h(\mathbf{x})}} P(\hat{Y} = c | \mathbf{X} = \mathbf{x}).$$

First, we remind the result obtained by Chittineni (1980) for the oracle classifier.

Theorem 2.4.2 (Chittineni (1980, p. 284))**.** *Assume the mislabeling matrix* $\mathbf{P}$ *is given. Then, for all* $\mathbf{x} \in \mathcal{X}$*, we have*

$$r(O, \mathbf{x}) \leq 1 - \frac{1 - \hat{r}(O, \mathbf{x})}{\beta}, \tag{2.25}$$

where $\beta = \max_{j \in \{1,\dots,K\}} \left(\sum_{c=1}^{K} p_{j,c} \right)$.

Proof. Remind that for all $\mathbf{x} \in \mathcal{X}$, $O(\mathbf{x}) = \max_{c \in \mathcal{Y}} P(Y = c | \mathbf{X} = \mathbf{x})$. Then, we have

$$\begin{aligned} \hat{r}(O, \mathbf{x}) &= 1 - \max_{j \in \{1,\dots,K\}} \sum_{c=1}^{K} p_{j,c} P(Y = c | \mathbf{X} = \mathbf{x}) \geq 1 - O(\mathbf{x}) \max_{j \in \{1,\dots,K\}} \left(\sum_{c=1}^{K} p_{j,c} \right) \\ &= 1 - O(\mathbf{x})\beta = (1 - \beta) + \beta r(O, \mathbf{x}). \end{aligned}$$

The rest of the proof is straightforward. □

As one can see, the term $\beta \geq 1$ ($\beta = 1$ means no mislabeling) penalizes the bound in the case of mislabeling. However, β is constant with respect to $\mathbf{x}$, so the penalization of the error $\hat{r}(O, \mathbf{x})$ does not depend on the predicted label, which is limitation, since some classes can be well predicted, so they should be less penalized. Another limitation is that the bound holds only for the oracle classifier. Thus, we derive another bound that holds for an arbitrary classifier and is given by the following theorem.

Theorem 2.4.3. *Assume the mislabeling matrix* $\mathbf{P}$ *is given. Then, for any classifier* h*, for all* $\mathbf{x} \in \mathcal{X}$*, for all* $\lambda \geq 0$ *such that* $p_{j,j} > p_{j,c} - \lambda,\ \forall j, c \in \{1, \dots, K\}^2$*, we have*

$$r(h, \mathbf{x}) \leq \frac{\hat{r}(h, \mathbf{x})}{\lambda + \delta(\mathbf{x})} - \frac{1 - \lambda - \alpha(\mathbf{x})}{\lambda + \delta(\mathbf{x})}, \tag{2.26}$$

with

- $\delta(\mathbf{x}) := p_{h(\mathbf{x}),h(\mathbf{x})} - \max_{j \in \mathcal{Y} \setminus \{h(\mathbf{x})\}} p_{h(\mathbf{x}),j}$,
- $\alpha(\mathbf{x}) := p_{h(\mathbf{x}),h(\mathbf{x})}$.

Proof. First, from the definition of $\hat{r}(\mathbf{x})$ and applying Eq. (2.24) we obtain that

$$\begin{aligned} \hat{r}(h, \mathbf{x}) &= 1 - P(\hat{Y} = h(\mathbf{x}) | \mathbf{X} = \mathbf{x}) = 1 - \sum_{c=1}^{K} p_{h(\mathbf{x}),j} P(Y = c | \mathbf{X} = \mathbf{x}) \\ &= 1 - p_{h(\mathbf{x}),h(\mathbf{x})} P(Y = h(\mathbf{x}) | \mathbf{X} = \mathbf{x}) - \sum_{\substack{c=1 \\ c \neq h(\mathbf{x})}}^{K} p_{h(\mathbf{x}),j} P(Y = c | \mathbf{X} = \mathbf{x}) \end{aligned}$$

One can notice that

$$\sum_{\substack{c=1 \\ c\neq h(\mathbf{x})}}^{K} p_{h(\mathbf{x}),c} P(Y=c|\mathbf{X}=\mathbf{x}) \leq \max_{c\in\mathcal{Y}\setminus\{h(\mathbf{x})\}} p_{h(\mathbf{x}),c} \sum_{\substack{c=1 \\ c\neq h(\mathbf{x})}}^{K} P(Y=c|\mathbf{X}=\mathbf{x})$$
$$= \max_{c\in\mathcal{Y}\setminus\{h(\mathbf{x})\}} p_{h(\mathbf{x}),c}(1-P(Y=h(\mathbf{x})|\mathbf{X}=\mathbf{x})).$$

Finally, we infer the following inequality:

$$\begin{aligned}\hat{r}(h,\mathbf{x}) &\geq (p_{h(\mathbf{x}),h(\mathbf{x})} - \max_{c\in\mathcal{Y}\setminus\{h(\mathbf{x})\}} p_{h(\mathbf{x}),c})(1-P(Y=h(\mathbf{x})|\mathbf{X}=\mathbf{x})) + 1 - p_{h(\mathbf{x}),h(\mathbf{x})} \\ &= \delta(\mathbf{x}) r(h,\mathbf{x}) + 1 - \alpha(\mathbf{x}) \\ &\geq (\delta(\mathbf{x})+\lambda) r(h,\mathbf{x}) + 1 - \lambda - \alpha(\mathbf{x}).\end{aligned}$$

Taking into account the assumption that $p_{j,j} > p_{j,c} - \lambda,\ \forall i,j \in \{1,\ldots,K\}^2$, we deduce that $\lambda + \delta(\mathbf{x}) > 0$, which concludes the proof. □

This theorem gives us insights on how the true error rate can be bounded given the error rate of the imperfect label and the mislabeling matrix. With the quantities $\delta(\mathbf{x})$ and $\alpha(\mathbf{x})$, we perform a correction of $\hat{r}(\mathbf{x})$. Note that when there is no mislabeling, the left and right sides of Eq. (2.26) are equal with $\lambda = 0$, since $\alpha(\mathbf{x}) = 1$ and $\delta(\mathbf{x}) = 1$ in this case.

Note that this theorem holds also for a more general case when correction probabilities depend on the example $\mathbf{x}$. In this case, all probabilities $p_{i,j}$ are replaced by $p^{\mathbf{x}}_{i,j} := P(\hat{Y}=i|Y=j,\mathbf{X}=\mathbf{x})$. Since it is harder to estimate $p^{\mathbf{x}}_{i,j}$ compared to $p_{i,j}$, we stick to consider the class-related model described by Eq. (2.24).

In the theorem, the mislabeling matrix is assumed given, while in practice it has to be estimated. Since the number of matrix entries grows quadratically with the increase of K, a direct estimation of the true posterior probabilities from Eq. (2.24) may be more affected by the estimation error than the bound itself as the latter needs to know only $2K$ entries. We give more details about estimation of the mislabeling matrix in Section 2.5.

The theorem assumes that the bound holds for any λ such that $p_{j,j} > p_{j,c} - \lambda$ is satisfied. One can notice that with $\lambda = 0$ we assume that $p_{j,j} > p_{j,c}$, i.e., the diagonal entries of the mislabeling matrix are the largest elements in their corresponding columns, which means that the imperfect label is reasonably correlated with the true label. In practice, this condition may not hold, so we can pass the assumption by increasing λ. An interesting fact is that we can make the bound tighter by properly choosing λ. For example, when $\delta(\mathbf{x})$ is close to 0, λ can help to avoid an arbitrarily large bound. We illustrate the impact of λ in Section 2.4.5.

2.4.3 C-Bounds with Imperfect Labels

Based on Theorem 2.4.3, we bound the generalization error $R(B_Q)$, which is the expectation of $r(\mathbf{X})$. By taking expectation in Eq. (2.26), we obtain that

$$R(B_Q) = \mathbb{E}_{\mathbf{X}} r(B_Q, \mathbf{x}) \leq \mathbb{E}_{\mathbf{X}} \frac{\hat{r}(B_Q, \mathbf{x})}{\delta(\mathbf{x}) + \lambda} - \mathbb{E}_{\mathbf{x}} \frac{1 - \lambda - \alpha(\mathbf{x})}{\delta(\mathbf{x}) + \lambda}. \tag{2.27}$$

One can see that for every $\mathbf{x}$, $\hat{r}(B_Q, \mathbf{x})$ is multiplied by a positive weight $1/\delta(\mathbf{X}) > 0$, so the first term of the right-hand side is a weighted generalization error of the imperfect label. To cope with this, we derive a weighted C-bound by proposing the next theorem.

Theorem 2.4.4. *Let $\hat{M}$ be a random variable such that $[\hat{M}|\mathbf{X} = \mathbf{x}]$ is a discrete random variable that is equal to the margin $\hat{M}_Q(\mathbf{x}, i)$ with probability $P(\hat{Y} = i|\mathbf{X} = \mathbf{x})$, $i = \{1, \ldots, K\}$. Assume that the mislabeling matrix $\mathbf{P}$ is given. Then, for all choice of Q on a hypothesis space $\mathcal{H}$, and for all distributions $P(\mathbf{X})$ over $\mathcal{X}$ and $P(Y|\mathbf{X})$ over $\mathcal{Y}$, for all $\lambda \geq 0$ such that $p_{j,j} > p_{j,c} - \lambda$, $\forall j, c \in \{1, \ldots, K\}^2$ we have:*

$$R(B_Q) \leq \psi_{\mathbf{P}} - \frac{\left(\mu_1^{\hat{M},\mathbf{P}}\right)^2}{\mu_2^{\hat{M},\mathbf{P}}}, \tag{CBIL}$$

if $\mu_1^{\hat{M}_{\mathbf{P}}} > 0$, where

- $\psi_{\mathbf{P}} := \mathbb{E}_{\mathbf{X}} \frac{\alpha(\mathbf{X}) + \lambda}{\delta(\mathbf{X}) + \lambda}$ *with δ and α defined as in Theorem 2.4.3,*

- $\mu_1^{\hat{M},\mathbf{P}} := \int_{\mathcal{X} \times \mathbb{R}} \frac{m}{\delta(\mathbf{x}) + \lambda} P(\hat{M} = m, \mathbf{X} = \mathbf{x}) \mathrm{d}\mathbf{x}\mathrm{d}m$ *is the weighted 1st margin moment,*

- $\mu_2^{\hat{M},\mathbf{P}} := \int_{\mathcal{X} \times \mathbb{R}} \frac{m^2}{\delta(\mathbf{x}) + \lambda} P(\hat{M} = m, \mathbf{X} = \mathbf{x}) \mathrm{d}\mathbf{x}\mathrm{d}m$ *is the weighted 2nd margin moment.*

Proof. At first, let us introduce a normalization factor $\omega_{\mathbf{P}}$ defined as follows:

$$\omega_{\mathbf{P}} := \mathbb{E}_{\mathbf{X}} \frac{1}{\delta(\mathbf{X}) + \lambda} = \int_{\mathcal{X} \times \mathbb{R}} \frac{P(\hat{M} = m, \mathbf{X} = \mathbf{x})}{\delta(\mathbf{x}) + \lambda} \mathrm{d}\mathbf{x}\mathrm{d}m.$$

Remind that $\hat{r}(B_Q, \mathbf{x}) = P(\hat{M} \leq 0 | \mathbf{X} = \mathbf{x})$. Then, we can write:

$$\begin{aligned}
\mathbb{E}_{\mathbf{X}} \frac{\hat{r}(B_Q, \mathbf{x})}{\delta(\mathbf{x}) + \lambda} &= \int_{\mathcal{X}} \frac{1}{\delta(\mathbf{x}) + \lambda} P(\hat{M} \leq 0 | \mathbf{X} = \mathbf{x}) P(\mathbf{X} = \mathbf{x}) \mathrm{d}\mathbf{x} \\
&= \int_{-\infty}^{0} \int_{\mathcal{X}} \frac{P(\hat{M} = m, \mathbf{X} = \mathbf{x})}{\delta(\mathbf{x}) + \lambda} \mathrm{d}\mathbf{x} \mathrm{d}m \\
&= \omega_{\mathbf{P}} \int_{-\infty}^{0} \frac{\int_{\mathcal{X}} P(\hat{M} = m, \mathbf{X} = \mathbf{x}) / (\delta(\mathbf{x}) + \lambda) \mathrm{d}\mathbf{x}}{\int_{\mathcal{X} \times \mathbb{R}} P(\hat{M} = m, \mathbf{X} = \mathbf{x}) / (\delta(\mathbf{x}) + \lambda) \mathrm{d}\mathbf{x} \mathrm{d}m} \mathrm{d}m \qquad (2.28) \\
&= \omega_{\mathbf{P}} P(\hat{M}_\omega < 0), \qquad (2.29)
\end{aligned}$$

where Eq. (2.29) is given by a random variable $\hat{M}_\omega$ coming from the density f_ω defined as the expression inside the integral in Eq. (2.28).

We further notice that the weighted margin moments can be represented as:

$$\begin{aligned}
\mu_1^{\hat{M},\mathbf{P}} &= \int_{\mathcal{X} \times \mathbb{R}} \frac{m}{\delta(\mathbf{x}) + \lambda} P(\hat{M} = m, \mathbf{X} = \mathbf{x}) \mathrm{d}\mathbf{x} \mathrm{d}m = \omega_{\mathbf{P}} \mu_1^{\hat{M}_\omega}, \\
\mu_2^{\hat{M},\mathbf{P}} &= \int_{\mathcal{X} \times \mathbb{R}} \frac{m^2}{\delta(\mathbf{x}) + \lambda} P(\hat{M} = m, \mathbf{X} = \mathbf{x}) \mathrm{d}\mathbf{x} \mathrm{d}m = \omega_{\mathbf{P}} \mu_2^{\hat{M}_\omega}.
\end{aligned}$$

From this, we also obtain that $var(M_\omega) = \left(\mu_2^{\hat{M},\mathbf{P}} / \omega_{\mathbf{P}}\right) - \left(\mu_1^{\hat{M},\mathbf{P}} / \omega_{\mathbf{P}}\right)^2$. Then, using the Cantelli-Chebyshev inequality (Lemma 2.6.3) with $a = \mu_1^{\hat{M}_f} = \mu_1^{\hat{M},\mathbf{P}} / \omega_{\mathbf{P}}$ we deduce the following inequality:

$$P(\hat{M}_\omega < 0) \leq \frac{\left(\mu_2^{\hat{M},\mathbf{P}} / \omega_{\mathbf{P}}\right) - \left(\mu_1^{\hat{M},\mathbf{P}} / \omega_{\mathbf{P}}\right)^2}{\left(\mu_2^{\hat{M},\mathbf{P}} / \omega_{\mathbf{P}}\right) - \left(\mu_1^{\hat{M},\mathbf{P}} / \omega_{\mathbf{P}}\right)^2 + \left(\mu_1^{\hat{M},\mathbf{P}} / \omega_{\mathbf{P}}\right)^2} = 1 - \frac{\left(\mu_1^{\hat{M},\mathbf{P}}\right)^2}{\omega_{\mathbf{P}} \mu_2^{\hat{M},\mathbf{P}}}.$$

Combining this inequality with Eq. (2.27) we infer (CBIL):

$$\begin{aligned}
R(B_Q) &\leq \mathbb{E}_{\mathbf{X}} \frac{\hat{r}(B_Q, \mathbf{x})}{\delta(\mathbf{x}) + \lambda} - \mathbb{E}_{\mathbf{X}} \frac{1 - \lambda - \alpha(\mathbf{x})}{\delta(\mathbf{x}) + \lambda} = \omega_{\mathbf{P}} P(\hat{M}_\omega < 0) - \omega_{\mathbf{P}} + \psi_{\mathbf{P}} \\
&\leq \psi_{\mathbf{P}} - \frac{\left(\mu_1^{\hat{M},\mathbf{P}}\right)^2}{\mu_2^{\hat{M},\mathbf{P}}}.
\end{aligned}$$

□

Given data with imperfect labels, the direct evaluation of the generalization error rate may be biased, leading to an overly optimistic evaluation. Using the mislabeling matrix $\mathbf{P}$ we derive a more conservative C-bound, where the error of $\mathbf{x}$ is penalized by the factor $1/\delta(\mathbf{x}) + \lambda$, where $\delta(\mathbf{x})$ depends on the predicted class. If there is no mislabeling and $\lambda = 0$, then $\psi_{\mathbf{P}} = 1$, $\mu_1^{\hat{M},\mathbf{P}}$ and $\mu_2^{\hat{M},\mathbf{P}}$ are equivalent to

$\mu_1^{\hat{M}}$ and $\mu_2^{\hat{M}}$, so we obtain the regular C-bound (CB).

In particular, this general result can be used to evaluate the error rate in the semi-supervised setting when mislabeling arises from pseudo-labeling of unlabeled examples via self-learning. Comparing with the transductive bound (TB) obtained as a corollary of Theorem 2.3.2, (CBIL) directly upper bounds the error rate, so it will be tighter in most of cases. Particularly, it can be noticed that the value of (TB) is growing with the increase of the number of classes. Note that there exists other attempts to evaluate the C-bound in the semi-supervised setting. In the binary case, the second margin moment is expressed via the disagreement of hypotheses (see Theorem 1.1.2), so it can be estimated using available unlabeled data. However, this holds for the binary case only.

In this theorem, we have combined the mislabeling bound (2.26) with the supervised multi-class C-bound (Laviolette et al., 2017), however, another possibility could be to combine with the bound based on the second-order Markov's inequality (Masegosa et al., 2020). As pointed out by Masegosa et al. (2020), the latter can be regarded as a relaxation of the C-bound, but it is easier to estimate from data in practice. As the tightest bound does not always imply the lowest error, the use of C-bound in model selection tasks may be more advantageous as it involves both the individual strength of hypotheses and correlation between their errors, while the bound of Masegosa et al. (2020) is based on the error correlation only.

2.4.4 PAC-Bayesian Theorem for C-Bound Estimation

When the margin mean, the margin variance and the mislabeling matrix are empirically estimated from data, evaluation of (CBIL) may be optimistically biased. In this section, we analyze the behavior of the empirical estimate by deriving a PAC-Bayesian bound (McAllester, 1999, 2003), which additionally penalizes the estimate by the sample size and the Kullback-Leibler divergence between the posterior Q and the prior Q_0 over the hypothesis space $\mathcal{H}$. This bound is given by the following theorem, the proof of which is a combination of Propositions 2.6.7, 2.6.9 and 2.6.11 that are deferred to Section 2.6.

Theorem 2.4.5. *Under the notations of Theorem 2.4.4, for any set of classifiers $\mathcal{H}$, for any prior distribution Q_0 on $\mathcal{H}$ and any $\epsilon \in (0,1]$, with a probability at least $1-\epsilon$ over the choice of the sample of size $n = l + u$, for every posterior distribution Q over $\mathcal{H}$, if $\mu_1^{\hat{M}} > 0$ and $\tilde{\delta}(\mathbf{x}) > 0$, we have:*

$$R(B_Q) \leq \tilde{\psi} - \frac{\tilde{\mu}_1^2}{\tilde{\mu}_2}, \tag{2.30}$$

where

$$\tilde{\mu}_1 = \frac{1}{u} \sum_{\mathbf{x} \in X_{\mathcal{U}}} (\tilde{\delta}(\mathbf{x}) + \lambda)^{-1} \sum_{c=1}^{K} M_Q(\mathbf{x}, c) P(Y = c | \mathbf{X} = \mathbf{x}) - B_1 \Omega(Q_0, Q, u, \epsilon)$$

$$\tilde{\mu}_2 = \frac{1}{u} \sum_{\mathbf{x} \in X_{\mathcal{U}}} (\tilde{\delta}(\mathbf{x}) + \lambda)^{-1} \sum_{c=1}^{K} (M_Q(\mathbf{x}, c))^2 P(Y = c | \mathbf{X} = \mathbf{x}) + B_2 \Omega(Q_0, Q, u, \epsilon)$$

$$\Omega(Q_0, Q, u, \epsilon) := \sqrt{\frac{2}{u} \left[KL(Q \parallel Q_0) + \ln \frac{2\sqrt{u}}{\epsilon / \rho} \right]}$$

$$\tilde{\psi} = \frac{1}{u} \sum_{i=1}^{u} \frac{\tilde{\alpha}(\mathbf{x}_i) + \lambda}{\tilde{\delta}(\mathbf{x}_i) + \lambda} + B_3 \sqrt{\frac{2}{u} \ln \frac{2\sqrt{u}}{\epsilon / \rho}}$$

$$\tilde{\delta}(\mathbf{x}) = \hat{\delta}(\mathbf{x}) - \sqrt{\frac{1}{2 l_{c_\mathbf{x}}} \ln \frac{2\sqrt{l_{c_\mathbf{x}}}}{\epsilon / \rho}} - \sqrt{\frac{1}{2 l_{j_\mathbf{x}}} \ln \frac{2\sqrt{l_{j_\mathbf{x}}}}{\epsilon / \rho}}, \text{ with } c_\mathbf{x} := B_Q(\mathbf{x}), j_\mathbf{x} := \underset{j \in \mathcal{Y} \setminus \{c_\mathbf{x}\}}{\operatorname{argmin}} l_j,$$

$$\tilde{\alpha}(\mathbf{x}) = \hat{\alpha}(\mathbf{x}) + \sqrt{\frac{1}{2 l_{c_\mathbf{x}}} \ln \frac{2\sqrt{l_{c_\mathbf{x}}}}{\epsilon / \rho}},$$

and where $\hat{\delta}(\mathbf{x})$ and $\hat{\alpha}(\mathbf{x})$ are empirical estimates respectively of $\delta(\mathbf{x})$ and $\alpha(\mathbf{x})$ based on the available labeled set, $KL(Q \parallel Q_0)$ is the Kullback-Leibler divergence between Q and Q_0, $l_j = \sum_{i=1}^{l} \mathbb{I}(y_j = j)/l$ is the proportion of the labeled training examples from the true class j, and $\rho := 2K + 3$ comes from applying a union bound .

As one can see, with the growth of u, the penalization becomes less severe, so $\tilde{\mu}_1$ and $\tilde{\mu}_2$ are close to $\mu_1^{\hat{M}}$ and $\mu_2^{\hat{M}}$. Similarly, $\tilde{\delta}(\mathbf{x})$ and $\tilde{\alpha}(\mathbf{x})$ are closer to $\hat{\delta}(\mathbf{x})$ and $\hat{\alpha}(\mathbf{x})$ with the increase of the number examples used to estimate the mislabeling matrix, which we take l for the sake of simplicity. Note that, in contrast to the supervised case (Laviolette et al., 2017, Theorem 3), B_1 and B_2 can have a drastic influence on the bound's value, when $\tilde{\delta}(\mathbf{x})$ is close to 0, which motivates to properly choose λ.

The obtained bound may be used to estimate the Bayes error from data, with the pseudo-labeled unlabeled examples serving as a hold-out set for estimating the margin moments, and the labeled examples serving as a hold-out set for estimating the mislabeling matrix. In the case of the random forest, the latter can be performed in the out-of-bag fashion as in (Thiemann et al., 2017; Lorenzen et al., 2019). However, the bound does not appear tighter in practice compared to the supervised case (Laviolette et al., 2017) due to the additional penalization on estimation of the mislabeling matrix. Making this bound tighter could be a good direction for future work. Nevertheless, when the focus is set on model selection, a common choice is to simply use an empirical estimate of the C-bound as an optimization criterion (Bauvin et al., 2020).

2.4.5 Empirical Illustration of (CBIL)

In this section, we illustrate the value of (CBIL) evaluated on the unlabeled examples pseudo-labeled by a self-learning algorithm. We study how the bound's value is penalized by the mislabeling model, so we empirically compare it with the oracle C-bound (CB) evaluated as if the labels for the considered unlabeled data would be known.

Data set	# of label. ex., $\lvert Z_{\mathcal{L}}\rvert$	# of unlab. ex., $\lvert X_{\mathcal{U}}\rvert$	Dimension, d	# of classes, K
`Isolet`	389	7408	617	26
`HAR`	102	10197	561	6
`Fashion`	175	69825	784	10

Table 2.1: Characteristics of data sets used in our experiments.

To do so, we compute the value of the two bounds varying the number of examples used for evaluation with respect to the prediction confidence: the pseudo-labeled examples are sorted by the value of the prediction vote in the descending order, and we keep only the first ρ% of the examples for $\rho \in \{20, 40, 60, 80, 100\}$.

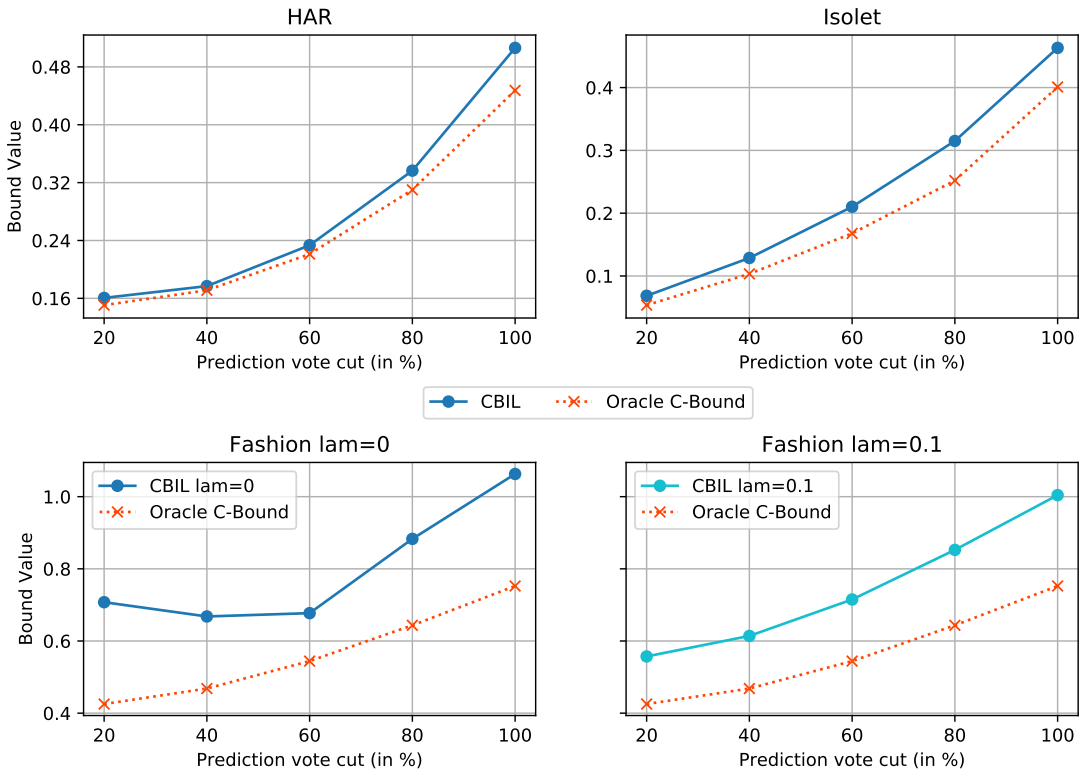


Figure 2-1: (CBIL) ($\lambda = 0$) and Oracle C-Bound when varying the number of pseudo-labels on 3 data sets. We keep the most confident one (with respect to prediction vote) from 20% to 100%. For `Fashion`, we illustrate also the results for $\lambda = 0.1$.

As ρ grows, we expect mislabeling to occur more often, so the (CBIL) is more penalized. In (CBIL), we use the true value of the mislabeling matrix (i.e., evaluated using the labels of unlabeled data) for clear illustration of the C-bound's penalization. In Section 2.5, we discuss the possible estimations of the mislabeling matrix.

The experiment is performed on 3 data sets `HAR`, `Isolet` and `Fashion` described in Table 2.1. The experimental results are illustrated in Figure 2-1, where λ is taken as 0 on all data sets, and additionally computing bound with $\lambda = 0.1$ for `Fashion`. As expected, the classifier makes mistakes mostly on low class votes, so the error increases when ρ grows. On `Isolet` and `HAR` one can see that (CBIL) is close to the oracle C-bound for small ρ, since most of pseudo-labels are true. When more noisy pseudo-labels are included, the difference between the two values becomes more evident, leading (CBIL) to be more pessimistic. This is probably connected with the choice of the mislabeling error model (2.23) that is class-related and not instance-related. Although we lose some flexibility, the class-related mislabeling matrix would be easier to estimate in practice. Finally, for `Fashion`, the penalization is more severe. It appears that with $\rho \in \{20\%, 40\%\}$ $\delta(\mathbf{x})$ can be close to 0, so the bound value grows and also leads to poor correlation with the true error. It is clearly seen that with $\lambda = 0.1$ the curve's shape becomes much more similar to the oracle C-bound. Eventually, a good correlation with the true error is crucial for practical applications, e.g., if (CBIL) is used as some optimization or selection criterion.

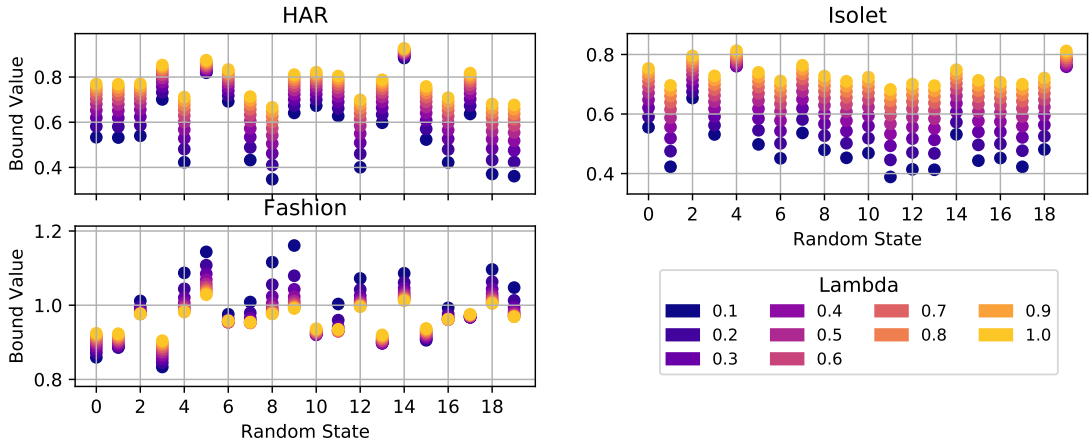


Figure 2-2: The value of (CBIL) with different λ over 20 different labeled/unlabeled splits of `HAR`, `Isolet` and `Fashion`.

In order to show more clearly the impact of λ on the bound's value, we also performed another experiment, where we evaluate the bound with $\lambda \in [0.1, 0.2, \ldots, 1]$. In Figure 2-2, we display the results of 20 random labeled/unlabeled splits for `HAR`, `Isolet` and `Fashion`. One can observe that when the bound is not penalized much (i.e., $\delta(\mathbf{x})$ is far from 0), then the increase of λ makes the bound looser, so $\lambda = 0.1$ is the tightest choice. Exactly the opposite situation is observed when $\delta(\mathbf{x})$ is small (most of trials for `Fashion`): higher values of λ diminish the influence of hyperbolic weights $1/\delta(\mathbf{x})$, so $\lambda = 1$ leads to the tightest bound.

2.5 Conclusion and Perspectives

In this chapter, we proposed a new probabilistic framework for the multi-class semi-supervised learning. At first, we derived a bound for the transductive conditional risk of the majority vote classifier. This probabilistic bound is based on the distribution of the class vote over unlabeled examples for a predicted class. We deduced corresponding bounds on the confusion matrix norm and the error rate as a corollary and determined a condition when the bounds are tight. The proposed bounds allows us to evaluate errors on the unlabeled examples, for which the majority vote outputs a class c with a vote above a threshold θ_c. This suggests that the bound can be evaluated at different thresholds in order to choose one that can be used for self-learning. In Chapter 3, we will show how this can be done.

Then, we proposed a mislabeling error model for analysis of the majority vote classifier on an imperfectly labeled data. We established the connection between the true and the imperfect label. Based on this result, we extended the C-bound to imperfect labels and derived a PAC-Bayesian Theorem for controlling the sample effect. The proposed bound allowed us to evaluate the performance of a learning model after pseudo-labeling the unlabeled data. We illustrated the influence of the mislabeling error model on the bound's value on several real data sets.

However, (CBIL) requires in practice estimation of the mislabeling matrix, which is a complex problem. While it is possible to estimate it using the labeled set, we have experimentally found it unsatisfying as the mislabeling matrix becomes biased by the labeled examples. This problem, nevertheless, is an active field of study (Natarajan et al., 2013). For example, in the semi-supervised setting, Krithara et al. (2008) learn the mislabeling matrix together with the classifier parameters through the classifier likelihood maximization for document classification. In supervised setting, a common approach is to detect anchor points whose labels are surely true (Scott, 2015). A potential idea would be to transfer this idea to the semi-supervised case in order to detect the anchor points in the unlabeled set and use them together with the labeled set for correct estimation of the noise in pseudo-labels; this may require additional assumptions such as the existence of clusters (Rigollet, 2007; Maximov et al., 2018) or manifold structure (Belkin and Niyogi, 2004).

We also point out possible applications of (CBIL). At first, in Chapter 4, we demonstrate that (CBIL) could be promisingly used as a selection criterion that guides a feature selection algorithm to choose an optimal feature subset based on the labeled and the pseudo-labeled sets. Next, (CBIL) can be used as a criterion to learn the posterior Q in the semi-supervised setting. This issue is actively studied in the supervised context, e.g., Roy et al. (2016); Bauvin et al. (2020) have developed the boosting-based C-bound optimization algorithms.
It should be noticed that for these two applications, the main objective is to rank models, so the best model has the minimal error on the unlabeled set. Hence, the bound analysis goes beyond the classical question of tightness: the tightest bound

does not always imply the minimal error, and we aim for seeking a bound that is well-correlated with the true error (as we saw in Figure 2-1).

2.6 Appendix

2.6.1 Mathematical Tools for Section 2.3

Tools for Theorem 3.2

Proof of Lemma 2.3.1. First, we obtain Eq. (2.4):

$$\begin{aligned}
R_{\mathcal{U}}(G_Q, i, j) &= \frac{1}{u_i}\mathbb{E}_{h\sim Q}\sum_{\mathbf{x}\in X_{\mathcal{U}}} P(Y=i|X=\mathbf{x})\mathbb{I}(h(\mathbf{x})=j) \\
&= \frac{1}{u_i}\sum_{\mathbf{x}\in X_{\mathcal{U}}} P(Y=i|X=\mathbf{x})v_Q(\mathbf{x},j) \\
&\geq \frac{1}{u_i}\sum_{\mathbf{x}\in X_{\mathcal{U}}} P(Y=i|X=\mathbf{x})v_Q(\mathbf{x},j)\mathbb{I}(B_Q(\mathbf{x})=j) \\
&= \frac{1}{u_i}\sum_{t=1}^{N_j}\sum_{\mathbf{x}\in X_{\mathcal{U}}} \left(P(Y=i|X=\mathbf{x})\mathbb{I}(B_Q(\mathbf{x})=j)\mathbb{I}(v_Q(\mathbf{x},j)=\gamma_j^{(t)})\right)\gamma_j^{(t)} \\
&= \sum_{t=1}^{N_j} b_{i,j}^{(t)}\gamma_j^{(t)}.
\end{aligned}$$

Then, we deduce Eq. (2.5):

$$\begin{aligned}
R_{\mathcal{U}\wedge\boldsymbol{\theta}}(B_Q, i, j) &= \frac{1}{u_i}\sum_{\mathbf{x}\in X_{\mathcal{U}}} P(Y=i|X=\mathbf{x})\mathbb{I}(B_Q(\mathbf{x})=j)\mathbb{I}(v_Q(\mathbf{x},j)\geq\theta_j) \\
&= \frac{1}{u_i}\sum_{t=1}^{N_j}\sum_{\mathbf{x}\in X_{\mathcal{U}}} P(Y=i|X=\mathbf{x})\mathbb{I}(B_Q(\mathbf{x})=j)\mathbb{I}(v_Q(\mathbf{x},j)=\gamma_j^{(t)})\mathbb{I}(\gamma_j^{(t)}\geq\theta_j) \\
&= \frac{1}{u_i}\sum_{t=k_j+1}^{N_j}\sum_{\mathbf{x}\in X_{\mathcal{U}}} P(Y=i|X=\mathbf{x})\mathbb{I}(B_Q(\mathbf{x})=j)\mathbb{I}(v_Q(\mathbf{x},j)=\gamma_j^{(t)}) \\
&= \sum_{t=k_j+1}^{N_j} b_{i,j}^{(t)}.
\end{aligned}$$

□

Lemma 2.6.1 (Lemma 4 in Amini et al. (2008)). *Let* $(g_i)_{i\in\{1,\dots,N\}}$ *be such that* $0 < g_1 < \cdots < g_N \leq 1$. *Consider also* $p_i \geq 0$ *for each* $i \in \{1,\dots,N\}$, $B \geq 0$, $k \in \{1,\dots,N\}$. *Then, the optimal solution of the linear program:*

$$\begin{cases} \max_{\mathbf{q}:=(q_1,\dots,q_N)} F(\mathbf{q}) := \max_{q_1,\dots,q_N} \sum_{i=k+1}^{N} q_i \\ 0 \leq q_i \leq p_i \quad \forall i \in \{1,\dots,N\} \\ \sum_{i=1}^{N} q_i g_i \leq B \end{cases}$$

will be $\mathbf{q}^*$ *defined as, for all* $i \in \{1,\dots,N\}$, $q_i^* = \min\left(p_i, \left\lfloor \frac{B-\sum_{j<i} q_j^* g_j}{g_i} \right\rfloor_+\right) \mathbb{I}(i > k)$; *where, the sign* $\lfloor\cdot\rfloor_+$ *denotes the positive part of a number,* $\lfloor x\rfloor_+ = x \cdot \mathbb{I}(x > 0)$.

Proof. It can be seen that the first k target variables should be zero for the optimal solution. Indeed, they do not influence explicitly the target function F. However, terms $g_i q_i$ for $i \in \{1,\dots,k\}$ are positive, so their increase leads to smaller values of q_i for $i \in \{k+1,\dots,N\}$, which in their turn decrease the value of F. Because of this, we look for a solution in a space $\mathcal{O} = \{0\}^k \times \prod_{i=k+1}^{N}[0, p_i]$. We aim to show that there is a unique optimal solution $\mathbf{q}^*$ in $\mathcal{O}$.

Existence. It is known that the linear program under consideration is a convex, feasible and bounded task. Hence, there is a feasible optimal solution $\mathbf{q}^{opt} \in \prod_{i=1}^{N}[0, p_i]$. Then, we define $\mathbf{q}^{opt,\mathcal{O}} \in \mathcal{O}$:

$$\begin{cases} q_i^{opt,\mathcal{O}} = q_i^{opt} & \text{if } i > k \\ q_i^{opt,\mathcal{O}} = 0 & \text{otherwise.} \end{cases}$$

It can be seen that this solution is feasible: $F(\mathbf{q}^{opt,\mathcal{O}}) = F(\mathbf{q}^{opt})$. Then, there exists an optimal solution in $\mathcal{O}$. Further, the optimal solution is again designated as $\mathbf{q}^*$.

Unique representation. We would like to find a representation of $\mathbf{q}^*$ that is, in fact, unique. Before doing it, one can notice that for $\mathbf{q}^*$ the following equation is necessarily true:

$$\sum_{i=1}^{N} q_i^* g_i = B.$$

Indeed, as g_i are fixed, $\mathbf{q}^*$ would not be optimal otherwise, and there would exist $\tilde{\mathbf{q}}$ such that $\sum_{i=1}^{N} \tilde{q}_i g_i > \sum_{i=1}^{N} q_i^* g_i$, which implies $F(\tilde{\mathbf{q}}) > F(\mathbf{q}^*)$.

Let's consider the lexicographic order $\succeq$:

$$\begin{aligned} &\forall(\mathbf{q}, \mathbf{q}') \in \mathbb{R}^N \times \mathbb{R}^N, \mathbf{q} \succeq \mathbf{q}' \\ &\qquad \Leftrightarrow \left\{\mathcal{I}(\mathbf{q}', \mathbf{q}) = \emptyset\right\} \vee \left\{\mathcal{I}(\mathbf{q}', \mathbf{q}) \neq \emptyset \wedge \min\left(\mathcal{I}(\mathbf{q}, \mathbf{q}')\right) < \min\left(\mathcal{I}(\mathbf{q}', \mathbf{q})\right)\right\}, \end{aligned}$$

where $\mathcal{I}(\mathbf{q}', \mathbf{q}) = \{i | q_i' > q_i\}$.

We aim for showing that the optimal solution is the greatest feasible solution in $\mathcal{O}$ for $\succeq$. Let $\mathcal{M}$ be the set$\{i > k | q_i^* < p_i\}$. Then, there are two cases:

- $M = \emptyset$. It means that for all $i > k$, $q_i^* = p_i$ and $\mathbf{q}^*$ is then the maximal element for $\succeq$ in $\mathcal{O}$.
- $M \neq \emptyset$. Let's consider $K = \min\{i > k | q_i^* < p_i\}$, $M = \mathcal{I}(\mathbf{q}, \mathbf{q}^*)$. By contradiction, suppose $\mathbf{q}^*$ is not the greatest feasible solution for $\succeq$ and there is $\mathbf{q} \in \mathbb{R}^N$ such that $\mathbf{q} \succ \mathbf{q}^*$.
  1. $M \leq k$. Then, $q_M > q_M^* = 0$. It implies that $\mathbf{q} \notin \mathcal{O}$.
  2. $k < M < K$. Then, $q_M > q_M^* = p_M$. The same, $\mathbf{q} \notin \mathcal{O}$.
  3. $M \geq K$. Then, $F(\mathbf{q}) > F(\mathbf{q}^*)$. But then $\sum_{i=1}^N q_i g_i > \sum_{i=1}^N q_i^* g_i = B$.

Hence, we conclude that if the solution is optimal then it is necessarily the greatest feasible solution for $\succeq$. Let's prove that if a solution is not the greatest feasible one then it can not be optimal. With this statement, uniqueness would be proven.

Consider $\mathbf{q} \in \mathcal{O}$ such that $\mathbf{q}^* \succ \mathbf{q}$.

- $\mathcal{I}(\mathbf{q}, \mathbf{q}^*) = \emptyset$. Then, $F(\mathbf{q}^*) > F(\mathbf{q})$ and $\mathbf{q}$ is not optimal.
- $\mathcal{I}(\mathbf{q}, \mathbf{q}^*) \neq \emptyset$. Let $K = \min(\mathcal{I}(\mathbf{q}^*, \mathbf{q}))$ and $M = \min(\mathcal{I}(\mathbf{q}, \mathbf{q}^*))$. Then, $q_M > q_M^* \geq 0$ and $K < M$. Denote $\lambda = \min\left(q_M, \frac{g_M}{g_K}(p_K - q_K)\right)$ and define $\mathbf{q}'$ by:

$$q_i' = q_i,\ i \notin \{K, M\}, \quad q_K' = q_K + \frac{g_M}{g_K}\lambda \quad q_M' = q_M - \lambda$$

It can be observed that $\mathbf{q}'$ satisfies the box constraints. Moreover, $F(\mathbf{q}') = F(\mathbf{q}) + \lambda(g_M/g_K - 1) > F(\mathbf{q})$ since $g_K < g_M$ and $\lambda > 0$. Thus, $\mathbf{q}$ is not optimal. Summing up, it is proven that there is the only optimal solution in $\mathcal{O}$ and it is the greatest feasible one for $\succeq$.

Then, let's obtain an explicit representation of this solution. As it is the greatest one in lexicographical order, we assign q_i for $i > k$ to maximal feasible values, which are p_i. It continues until the moment when $\sum_{j=1}^i q_i g_i$ is close to B. Denote by I the index such that $\sum_{i=1}^{I-1} p_i g_i \leq B$, but $\sum_{i=1}^{I} p_i g_i \geq B$.

- $\sum_{i=1}^{I-1} p_i g_i = B$. Then, $q_i = 0$ for $i \geq I$. It can be also written in the following way:

$$q_i = \left\lfloor \frac{B - \sum_{j<i} q_j g_j}{g_i} \right\rfloor_+, \qquad i \geq I$$

.

- $\sum_{i=1}^{I-1} p_i g_i < B$. Then, q_I is equal to residual:

$$q_I = \frac{B - \sum_{j<I} q_j g_j}{g_I} = \left\lfloor \frac{B - \sum_{j<I} q_j g_j}{g_I} \right\rfloor_+.$$

For the other q_i, $i > I$ we assign to 0.

□

Tools for Proposition 2.3.5

Lemma 2.6.2. *For all* $\mathbf{x} \in \mathrm{X}_{\mathcal{U}}$*, for all* $(i,j) \in \{1,\dots,K\}^2$*, the following inequality holds:*

$$R_{\mathcal{U}}(B_Q,i,j) \geq \frac{1}{u_i}\sum_{\mathbf{x}\in\mathrm{X}_{\mathcal{U}}} P(Y=i|X=\mathbf{x})\mathbb{I}(B_Q(\mathbf{x})=j)\mathbb{I}(v_Q(\mathbf{x},j)<\gamma^*) + \frac{1}{\gamma^*}\Big\lfloor \lfloor K_{i,j} - M^{<}_{i,j}(\gamma^*)\rfloor_+ - r_{i,j}\Big\rfloor_+ + r_{i,j}, \quad (2.31)$$

where $\gamma^* := \sup\{\gamma \in \Gamma_j \,|\, \sum_{\mathbf{x}\in\mathrm{X}_{\mathcal{U}}} P(Y=i|X=\mathbf{x})\mathbb{I}(B_Q(\mathbf{x})=j)\mathbb{I}(v_Q(\mathbf{x},j)=\gamma)/u_i > \tau\}$.

Proof. Denote $\gamma^* = \gamma_j^{(p)}$. According to Lemma 2.3.1, $K_{i,j} = \sum_{n=1}^{N_j} b_{i,j}^{(n)}\gamma_j^{(n)}$, where $b_{i,j}^{(n)} := \frac{1}{u_i}\sum_{\mathbf{x}\in\mathrm{X}_{\mathcal{U}}} P(Y=i|X=\mathbf{x})\mathbb{I}(B_Q(\mathbf{x})=j)\mathbb{I}(v_Q(\mathbf{x},j)=\gamma_j^{(n)})$. We can express $b_{i,j}^{(p)}$ in the following way:

$$b_{i,j}^{(p)} = \frac{K_{i,j} - \sum_{n=1}^{p-1} b_{i,j}^{(n)}\gamma_j^{(n)} - \sum_{n=p+1}^{N_j} b_{i,j}^{(n)}\gamma_j^{(n)}}{\gamma_j^{(p)}} = \frac{K_{i,j} - \sum_{n=1}^{p-1} b_{i,j}^{(n)}\gamma_j^{(n)} - r_{i,j}}{\gamma_j^{(p)}}.$$

Remind $B_{i,j}^{(n)} = \frac{1}{u_i}\sum_{\mathbf{x}\in\mathrm{X}_{\mathcal{U}}} P(Y=i|X=\mathbf{x})\mathbb{I}(v_Q(\mathbf{x},j)=\gamma_j^{(n)})$. From this we derive the following:

$$-\sum_{n=1}^{p-1} b_{i,j}^{(n)}\gamma_j^{(n)} \geq -\sum_{n=1}^{p-1} B_{i,j}^{(n)}\gamma_j^{(n)} = -M^{<}_{i,j}(\gamma_j^{(p)}) = -M^{<}_{i,j}(\gamma^*).$$

Taking into account this as well as $b_{i,j}^{(p)} \geq 0$, we deduce a lower bound for $b_{i,j}^{(p)}$:

$$b_{i,j}^{(p)} \geq \frac{1}{\gamma^*}\lfloor K_{i,j} - M^{<}_{i,j}(\gamma^*) - r_{i,j}\rfloor_+ = \frac{1}{\gamma^*}\Big\lfloor \lfloor K_{i,j} - M^{<}_{i,j}(\gamma^*)\rfloor_+ - r_{i,j}\Big\rfloor_+. \quad (2.32)$$

Then, one can notice that:

$$\begin{aligned} R_{\mathcal{U}}(B_Q,i,j) &= \sum_{n=1}^{N_j} b_{i,j}^{(n)} = \sum_{n=1}^{p-1} b_{i,j}^{(n)} + b_{i,j}^{(p)} + \sum_{n=p+1}^{N_j} b_{i,j}^{(n)} \\ &\geq \sum_{n=1}^{p-1} b_{i,j}^{(n)} + b_{i,j}^{(p)} + \sum_{n=p+1}^{N_j} b_{i,j}^{(n)}\gamma_j^{(n)} \\ &= \frac{1}{u_i}\sum_{\mathbf{x}\in\mathrm{X}_{\mathcal{U}}} P(Y=i|X=\mathbf{x})\mathbb{I}(B_Q(\mathbf{x})=j)\mathbb{I}(v_Q(\mathbf{x},j)<\gamma^*) + b_{i,j}^{(p)} + r_{i,j} \end{aligned}$$

By applying Eq. (2.32) to the last inequality we deduce Eq. (2.31). □

2.6.2 Mathematical Tools for Section 2.4

Tools for Theorem 2.4.1

Lemma 2.6.3 (Cantelli-Chebyshev inequality)**.** *[Ex 2.3 in Boucheron et al. (2013)] Let Z be a random variable with the mean μ and the variance σ^2. Then, for every $a > 0$, we have:*

$$P(Z \leq \mu - a) \leq \frac{\sigma^2}{\sigma^2 + a^2}.$$

Bounds for the Mislabeling Matrix' Entries

We remind that the imperfection is summarized through the mislabeling matrix $\mathbf{P} = (p_{i,j})_{1 \leq i,j \leq K}$ with

$$p_{i,j} := P(\hat{Y} = i | Y = j) \quad \text{for all } (i,j) \in \{1,\dots,K\}^2$$

such that $\sum_{i=1}^{K} p_{i,j} = 1$. Also, recall that $\delta(\mathbf{x}) := p_{B_Q(\mathbf{x}),B_Q(\mathbf{x})} - \max_{j \in \mathcal{Y} \setminus \{B_Q(\mathbf{x})\}} p_{B_Q(\mathbf{x}),j}$ and $\alpha(\mathbf{x}) = p_{B_Q(\mathbf{x}),B_Q(\mathbf{x})}$.

Proposition 2.6.4. *Let $\mathbf{P}$ be the mislabeling matrix, and assume that $p_{i,i} > p_{i,j}$, $\forall i,j \in \{1,\dots,K\}^2$. For any $\epsilon \in (0,1]$, with probability $1-\epsilon$ over the choice of the l sample, for all $(i,j) \in \{1,\dots,K\}^2$, for all $\mathbf{x} \in \mathcal{X}$,*

$$\hat{p}_{j,c} - r(l_c) \leq p_{j,c} \leq \hat{p}_{j,c} + r(l_c), \tag{2.33}$$

$$\alpha(\mathbf{x}) \leq \hat{\alpha}(\mathbf{x}) + r(l_{c_\mathbf{x}}), \tag{2.34}$$

$$\frac{1}{\delta(\mathbf{x})} \leq \frac{1}{\hat{\delta}(\mathbf{x}) - r(l_{c_\mathbf{x}}) - r(l_{j_\mathbf{x}})}, \quad \text{if } \hat{\delta}(\mathbf{x}) \geq r(l_{c_\mathbf{x}}) + r(l_{j_\mathbf{x}}), \tag{2.35}$$

where

- $r(l_k) = \sqrt{\frac{1}{2l_k} \ln \frac{2\sqrt{l_k}}{\epsilon}}$,
- $l_k = \sum_{i=1}^{l} \mathbb{I}(y_i = k)/l$ *is the proportion of the labeled training examples from the true class k,*
- $c_\mathbf{x} := B_Q(\mathbf{x})$, $j_\mathbf{x} := \operatorname{argmin}_{j \in \mathcal{Y} \setminus \{c_\mathbf{x})\}} l_j$,
- $\hat{p}_{j,c}$, $\hat{\alpha}(\mathbf{x})$ *and* $\hat{\delta}(\mathbf{x})$ *are empirical estimates respectively of* $p_{j,c}$, $\alpha(\mathbf{x})$ *and* $\delta(\mathbf{x})$ *based on the available l sample.*

Proof. Let S_j denote the subset of the available examples for which the true class is j. Consider the non-negative random variable $\exp\left\{2l_j(\hat{p}_{i,j} - p_{i,j})^2\right\}$. From the

Markov inequality we obtain that the following holds with probability at least $1-\epsilon$ over $S_j \sim P(\mathbf{X}|Y=j)^{l_j}$:

$$\exp\left\{2l_j(\hat{p}_{i,j}-p_{i,j})^2\right\} \leq \frac{1}{\delta}\mathbb{E}_{S_j}\exp\left\{2l_j(\hat{p}_{i,j}-p_{i,j})^2\right\}. \tag{2.36}$$

By successively applying Lemma 2.6.5 and Lemma 2.6.6, we deduce that

$$\mathbb{E}_{S_j}\exp\left\{2l_j(\hat{p}_{i,j}-p_{i,j})^2\right\} \leq \mathbb{E}_{S_j}\exp\left\{l_j\cdot kl(\hat{p}_{i,j}||p_{i,j})\right\} \leq 2\sqrt{l_j}. \tag{2.37}$$

Combining Eq. (2.36) and Eq. (2.37), we infer $2l_j(\hat{p}_{i,j}-p_{i,j})^2 \leq \ln\left(2\sqrt{l_j}/\delta\right)$. Eq. (2.33) is directly obtained from the last inequality, and hence, we derive also Eq. (2.34). To prove Eq. (2.35), let us define

$$k_{\mathbf{x}} := \underset{k\in\mathcal{Y}\setminus\{B_Q(\mathbf{x})\}}{\operatorname{argmax}} p_{c_{\mathbf{x}},k}, \qquad \hat{k}_{\mathbf{x}} := \underset{k\in\mathcal{Y}\setminus\{B_Q(\mathbf{x})\}}{\operatorname{argmax}} \hat{p}_{c_{\mathbf{x}},k}.$$

Then, we write:

$$\begin{aligned}\frac{1}{\delta(\mathbf{x})} &= \frac{1}{p_{c_{\mathbf{x}},c_{\mathbf{x}}}-p_{c_{\mathbf{x}},k_{\mathbf{x}}}} \leq \frac{1}{p_{c_{\mathbf{x}},c_{\mathbf{x}}}-p_{c_{\mathbf{x}},k_{\mathbf{x}}}-r(l_{c_{\mathbf{x}}})-r(l_{k_{\mathbf{x}}})}\\ &\leq \frac{1}{p_{c_{\mathbf{x}},c_{\mathbf{x}}}-p_{c_{\mathbf{x}},\hat{k}_{\mathbf{x}}}-r(l_{c_{\mathbf{x}}})-r(l_{j_{\mathbf{x}}})} = \frac{1}{\hat{\delta}(\mathbf{x})-r(l_{c_{\mathbf{x}}})-r(l_{j_{\mathbf{x}}})}.\end{aligned}$$

These transitions hold only when the denominator is positive, which is ensured if $\hat{\delta}(\mathbf{x}) \geq r(l_{c_{\mathbf{x}}})+r(l_{j_{\mathbf{x}}})$. □

Lemma 2.6.5 (Pinsker's Inequality for Bernoulli random variables, Theorem 4.19 in Boucheron et al. (2013))**.** *For all $p_1, p_2 \in [0,1]^2$,*

$$\begin{aligned}&2(p_2-p_1)^2 \leq kl(p_2||p_1)\\ &kl(p_2||p_1) := p_2\ln\frac{p_2}{p_1}+(1-p_2)\ln\frac{1-p_2}{1-p_1} = KL(P_2 \parallel P_1),\end{aligned}$$

where P_2 and P_1 are Bernoulli distributions with parameters p_2 and p_1 respectively.

Lemma 2.6.6 (Theorem 1 in Maurer (2004) and Lemma 19 in Germain et al. (2015))**.** *Let $\mathbf{X} = (X_1,\ldots,X_n)$ be a random vector, whose components X_i are i.i.d. with values $\in [0,1]$ and expectation μ. Let $\mathbf{X}' = (X_1',\ldots,X_n')$ denotes a random vector, where each X_i' is the unique Bernoulli random variable of the corresponding X_i: $P(X_i'=1) = \mathbb{E}X_i' = \mathbb{E}X_i = \mu$, $\forall i \in \{1,\ldots,n\}$. Then,*

$$\mathbb{E}\left[e^{nKL(\bar{X} \parallel \mu)}\right] \leq \mathbb{E}\left[e^{nKL(\bar{X}' \parallel \mu)}\right] \leq 2\sqrt{n},$$

where $\bar{X} = \frac{1}{n}\sum_{i=1}^n X_i$ and $\bar{X}' = \frac{1}{n}\sum_{i=1}^n X_i'$.

Lower Bound on the First Margin Moment

Proposition 2.6.7. *Let $\hat{M}$ be a random variable such that $[\hat{M}|\mathbf{X} = \mathbf{x}]$ is a discrete random variable that is equal to the margin $M_Q(\mathbf{x}, j)$ with probability $P(\hat{Y} = j|\mathbf{X} = \mathbf{x})$, $j = \{1, \ldots, K\}$. Let $\mu_1^{\hat{M}}$ be defined as in Theorem 2.4.4. Given the conditions of Proposition 2.6.4, for any set of classifiers $\mathcal{H}$, for any prior distribution Q_0 on $\mathcal{H}$ and any $\epsilon \in (0, 1]$, with a probability at least $1 - \epsilon$ over the choice of the n sample, for every posterior distribution Q over $\mathcal{H}$*

$$\mu_1^{\hat{M},\mathbf{P}} \geq \bar{\mu}_1^S - B_1 \sqrt{\frac{2}{n}\left[KL(Q \parallel Q_0) + \ln \frac{2\sqrt{n}}{\delta}\right]},$$

where

- $\bar{\mu}_1^S = \frac{1}{n}\sum_{i=1}^n (1/\tilde{\delta}(\mathbf{x}))\sum_{c=1}^K M_Q(\mathbf{x}, c)P(Y = c|\mathbf{X} = \mathbf{x})$ *is the empirical weighted margin mean based on the available n-sample S,*
- $\tilde{\delta}(\mathbf{x}) := \hat{\delta}(\mathbf{x}) - r(l_{c_\mathbf{x}}) - r(l_{j_\mathbf{x}})$,
- $B_1 := \max_{x \in \mathcal{X}} |(1/\tilde{\delta}(\mathbf{x}))\sum_{c=1}^K M_Q(\mathbf{x}, c)P(Y = c|\mathbf{X} = \mathbf{x})|$,
- *KL denotes the Kullback–Leibler divergence.*

Proof. Further, we denote the available sample with imperfect labels by S. Let $\mu_1^{\hat{M},\mathbf{P},h}$ and $\bar{\mu}_1^{S,h}$ be the random variables such that $\mu_1^{\hat{M},\mathbf{P}} = \mathbb{E}_{h\sim Q}\mu_1^{\hat{M},\mathbf{P},h}$ and $\bar{\mu}_1^S = \mathbb{E}_{h\sim Q}\bar{\mu}_1^{S,h}$.

We apply the Markov inequality to $\mathbb{E}_{h\sim Q_0} \exp\left\{\frac{n}{2B_1^2}(\bar{\mu}_1^{S,h} - \mu_1^{\hat{M},\mathbf{P},h})^2\right\}$, which is a non-negative random variable, and obtain that with probability at least $1 - \epsilon$ over $S \sim P(\mathbf{X}, \hat{Y})^n$:

$$\mathbb{E}_{h\sim Q_0} \exp\left\{\frac{n(\bar{\mu}_1^{S,h} - \mu_1^{\hat{M},\mathbf{P},h})^2}{2B_1^2}\right\} \leq \frac{1}{\epsilon}\mathbb{E}_S\mathbb{E}_{h\sim Q_0} \exp\left\{\frac{n(\bar{\mu}_1^{S,h} - \mu_1^{\hat{M},\mathbf{P},h})^2}{2B_1^2}\right\}. \quad (2.38)$$

Since the prior distribution Q_0 over $\mathcal{H}$ is independent on S, we can swap $\mathbb{E}_S$ and $\mathbb{E}_{h\sim Q_0}$. One can notice that

$$\frac{1}{2B_1^2}(\bar{\mu}_1^{S,h} - \mu_1^{\hat{M},\mathbf{P},h})^2 = 2\left[\frac{1}{2}(1 - \frac{\bar{\mu}_1^{S,h}}{B_1}) - \frac{1}{2}(1 - \frac{\mu_1^{\hat{M},\mathbf{P},h}}{B_1})\right]^2,$$

which is the squared of the difference of two random variables that are both between 0 and 1. Then, we successively apply Lemma 2.6.5 and Lemma 2.6.6 deriving

that:

$$\mathbb{E}_{h\sim Q_0}\mathbb{E}_S \exp\left\{2n\left[\frac{1}{2}\left(1-\frac{\bar{\mu}_1^{S,h}}{B_1}\right)-\frac{1}{2}\left(1-\frac{\mu_1^{\hat{M},\mathbf{P},h}}{B_1}\right)\right]^2\right\}$$
$$\leq \mathbb{E}_{h\sim Q_0}\mathbb{E}_S \exp\left\{n\cdot kl\left(\frac{1}{2}(1-\frac{\bar{\mu}_1^{S,h}}{B_1})\middle\|\frac{1}{2}(1-\frac{\mu_1^{\hat{M},\mathbf{P},h}}{B_1})\right)\right\} \leq \mathbb{E}_{h\sim Q_0} 2\sqrt{n} = 2\sqrt{n}.$$

We apply this result for Eq. (2.38), and by taking the natural logarithm from the both sides we obtain that:

$$\ln\left(\mathbb{E}_{h\sim Q_0}\exp\left\{\frac{n}{2B_1^2}(\bar{\mu}_1^{S,h}-\mu_1^{\hat{M},\mathbf{P},h})^2\right\}\right) \leq \ln\left(\frac{2\sqrt{n}}{\epsilon}\right). \tag{2.39}$$

Using the change of measure (Lemma 2.6.8) and the Jensen's inequalities, we derive that:

$$\ln\left(\mathbb{E}_{h\sim Q_0}\exp\left\{\frac{n}{2B_1^2}(\bar{\mu}_1^{S,h}-\mu_1^{\hat{M},\mathbf{P},h})^2\right\}\right) \geq \mathbb{E}_{h\sim Q}\frac{n}{2B_1^2}(\bar{\mu}_1^{S,h}-\mu_1^{\hat{M},\mathbf{P},h})^2 - KL(Q\parallel Q_0)$$
$$\geq \frac{n}{2B_1^2}\left(\bar{\mu}_1^{S}-\mu_1^{\hat{M},\mathbf{P}}\right)^2 - KL(Q\parallel Q_0).$$

Combining with Eq. (2.39), we derive:

$$\frac{n}{2B_1^2}(\bar{\mu}_1^{S}-\mu_1^{\hat{M},\mathbf{P}})^2 \leq \ln\left(\frac{2\sqrt{n}}{\epsilon}\right) + KL(Q\parallel Q_0). \tag{2.40}$$

The final inequality is directly inferred from Eq. (2.40).

Lemma 2.6.8 (Change of Measure Inequality Donsker and Varadhan (1975))**.** *For any measurable function ϕ defined on the hypothesis space $\mathcal{H}$ and all distributions Q_0, Q on $\mathcal{H}$, the following inequality holds:*

$$\mathbb{E}_{h\sim Q}\phi(h) \leq KL(Q\parallel Q_0) + \ln \mathbb{E}_{h\sim Q_0} e^{\phi(h)}.$$

□

Other Required Bounds

Proposition 2.6.9. *Let $\hat{M}$ be a random variable such that $[\hat{M}|\mathbf{X}=\mathbf{x}]$ is a discrete random variable that is equal to the margin $M_Q(\mathbf{x},j)$ with probability $P(\hat{Y}=j|\mathbf{X}=\mathbf{x})$, $j=\{1,\ldots,K\}$. Let $\mu_2^{\hat{M},\mathbf{P}}$ be defined as in Theorem 2.4.4. Given the conditions of Proposition 2.6.4, for any set of classifiers $\mathcal{H}$, for any prior distribution Q_0 on $\mathcal{H}$ and any $\epsilon\in(0,1]$, with a probability at least $1-\epsilon$ over the choice of the n sample, for*

every posterior distribution Q over $\mathcal{H}$

$$\mu_2^{\hat{M},\mathbf{P}} \leq \bar{\mu}_2^S + B_2\sqrt{\frac{2}{n}\left[2KL(Q \parallel Q_0) + \ln\frac{2\sqrt{n}}{\epsilon}\right]},$$

where

- $\bar{\mu}_2^S = \frac{1}{n}(1/\tilde{\delta}(\mathbf{x}))\sum_{i=1}^{n}\sum_{c=1}^{K}(M_Q(\mathbf{x}_i,c))^2 P(Y=c|\mathbf{X}=\mathbf{x}_i)$ *is the weighted 2nd margin moment estimated based on the available n-sample S,*
- $\tilde{\delta}(\mathbf{x}) := \hat{\delta}(\mathbf{x}) - r(l_{c_\mathbf{x}}) - r(l_{j_\mathbf{x}})$,
- $B_2 := \max_{x\in\mathcal{X}} |(1/\tilde{\delta}(\mathbf{x}))\sum_{c=1}^{K}(M_Q(\mathbf{x},c))^2 P(Y=c|\mathbf{X}=\mathbf{x})|$,
- *KL denotes the Kullback–Leibler divergence.*

Proof. The proof is similar to the one given for Proposition 2.6.7, but relies on the extension of the change of measure inequality (Lemma 2.6.10).

Lemma 2.6.10 (Change of Measure Inequality for Pairs of Voters (Lemma 1 in Laviolette et al. (2017)))**.** *For any set of voters $\mathcal{H}$, for any distributions Q_0, Q on $\mathcal{H}$, and for any measurable function $\phi : \mathcal{H}\times\mathcal{H} \to \mathbb{R}$, the following inequality holds:*

$$\mathbb{E}_{(h,h')\sim Q^2}\phi(h,h') \leq 2KL(Q \parallel Q_0) + \ln \mathbb{E}_{(h,h')\sim Q_0^2} e^{\phi(h,h')}.$$

□

Proposition 2.6.11. *Given the conditions of Proposition 2.6.4, for any $\epsilon \in (0,1]$, with a probability at least $1-\epsilon$ over the choice of the n sample,*

$$\psi_\mathbf{P} \leq \frac{1}{n}\sum_{i=1}^{n}\frac{\hat{\alpha}(\mathbf{x}_i) + r(l_{c_\mathbf{x}})}{\hat{\delta}(\mathbf{x}_i) - r(l_{c_\mathbf{x}}) - r(l_{j_\mathbf{x}})} + B_3\sqrt{\frac{2}{n}\ln\frac{2\sqrt{n}}{\epsilon}},$$

where $B_3 := \max_{x\in\mathcal{X}}[\hat{\alpha}(\mathbf{x}_i) + r(l_{c_\mathbf{x}})]/[\hat{\delta}(\mathbf{x}_i) - r(l_{c_\mathbf{x}}) - r(l_{j_\mathbf{x}})]$.

Proof. First, we take into consideration the result of Proposition 2.6.4 and deduce that $\psi_\mathbf{P} \leq \mathbb{E}_\mathbf{X}[(\hat{\alpha}(\mathbf{x}_i) + r(l_{c_\mathbf{x}}))/(\hat{\delta}(\mathbf{x}_i) - r(l_{c_\mathbf{x}}) - r(l_{j_\mathbf{x}}))]$. The rest of proof is similar to those are given for Proposition 2.6.4 and for Proposition 2.6.7. □

Chapter 3

Self-learning and Application

In semi-supervised learning, self-learning is usually performed with the confidence threshold that is fixed at a certain value or in such a way that it controls the number of pseudo-labeled examples at each iteration. However, these parameters can not be tuned in the case of real data, so it is not clear what value the threshold should be set to. In Section 3.1 of this chapter, we propose a strategy to effectively find this threshold at every iteration as a trade-off between the number of pseudo-labeled examples and the bounded transductive error evaluated on them. We present empirical evidence showing that the proposed self-learning policy is effective compared to several state-of-the-art approaches.

Then, in Section 3.2, we apply a self-learning approach for a biological application related to classification of DNA sequences where some classes are naturally absent in the labeled data. We propose an extension of the self-learning algorithm to deal with this problem. A new approach can refrain from predicting any observed class label and find clusters in unlabeled data that are successively included in the labeled training set as new classes. We empirically validate the approach via an ablative study and show that we are able to detect regions of potentially new classes.

3.1 Multi-class Self-learning Algorithm

3.1.1 The Proposed Approach

In this section, we propose to apply the transductive bound on the error rate derived in Section 2.3 for self-learning (Tür et al., 2005). We remind that the self-learning algorithm starts from a supervised base classifier initially trained on available labeled examples. Then, it iteratively assigns pseudo-labels at each iteration to those unlabeled examples that have a confidence score above a certain threshold. The pseudo-labeled examples are then included in the training set, and the classifier is retrained. The process is repeated until no examples for pseudo-labeling are left.

The central question of applying the self-learning algorithm in practice is how to choose the threshold. Intuitively, the threshold can manually be set to a very high

value, since only examples with a very high degree of confidence will be pseudo-labeled in this case. However, the confidence measure is biased by the small labeled set, so every iteration of the self-learning may still induce an error and shift the boundary in the wrong direction. In addition, the fact that a large number of iterations makes the algorithm computationally expensive drives us to choose the threshold carefully.

To overcome this problem, we extend the strategy proposed by Amini et al. (2008) to the multi-class setting. We consider the majority vote as the base classifier and the prediction vote as an indicator of confidence. Given a threshold vector $\boldsymbol{\theta}$, we introduce the *conditional Bayes error rate* $R_{\mathcal{U}|\boldsymbol{\theta}}(B_Q)$, defined in the following way:

$$R_{\mathcal{U}|\boldsymbol{\theta}}(B_Q) := \frac{R_{\mathcal{U}\wedge\boldsymbol{\theta}}(B_Q)}{\pi(v_Q(\mathbf{x},k) \geq \theta_k)}, \tag{3.1}$$

where $\pi(v_Q(\mathbf{x},k) \geq \theta_k) := \sum_{\mathbf{x}\in X_{\mathcal{U}}} \mathbb{1}_{v_Q(\mathbf{x},k)\geq\theta_k}/u$ and $k := B_Q(\mathbf{x})$. The numerator reflects the proportion of mistakes on the unlabeled set when the threshold is equal to $\boldsymbol{\theta}$, whereas the denominator computes the proportion of unlabeled observations with the vote larger than the threshold for the predicted class. Thus, we propose to find the threshold that yields the minimal value of $R_{\mathcal{U}|\boldsymbol{\theta}}(B_Q)$, making a trade-off between the error we induce by pseudo-labeling and the number of pseudo-labeled examples. In Algorithm 1 we summarize our algorithm, which is further denoted by `MSLA`[1].

To evaluate the transductive error, we bound the numerator of Eq. (3.1) using Corollary 2.3.4. However, the bound can practically be computed only with assumptions, since the posterior probabilities $P(Y = c|X = \mathbf{x})$ for unlabeled examples are not known. In this work, we approximate the posterior $P(Y = c|X = \mathbf{x})$ by $v_Q(\mathbf{x},c)$ of the base classifier trained on labeled examples only (the initial step of `MSLA`). Although this approximation is optimistic, by formulating the bound as probabilistic we keep some chances for other classes so the error of the supervised classifier can be smoothed. However, it must be borne in mind that the hypothesis space should be diverse enough so that the entropy of $(v_Q(\mathbf{x},c))_{c=1}^K$ would not be always zero, and the errors are made mostly on low prediction votes. In our experiments, as the base classifier we use the random forest (Breiman, 2001) that aggregates predictions from trees learned on different bootstrap samples. In Section 3.1.5, we validate the proposed approximation by empirically comparing it with the case when the posterior probabilities are set to $1/K$, i.e., when we treat all classes as equally probable.

Similarly to the work of Amini et al. (2008), in practice, to find an optimal $\boldsymbol{\theta}^*$ we perform a grid search over the hypercube $(0,1]^K$. The same algorithm is used for computing the optimal γ^* that provides the value of an upper bound for the conditional risk (see Theorem 2.3.2). In contrast to the binary self-learning, the

[1]The code source of the algorithm can be found at `https://github.com/vfeofanov/trans-bounds-maj-vote`.

Algorithm 1 Multi-class self-learning algorithm (MSLA)

Input:
Labeled observations $Z_{\mathcal{L}}$
Unlabeled observations $X_{\mathcal{U}}$
Initialization:
A set of pseudo-labeled instances, $Z_{\mathcal{P}} \leftarrow \emptyset$
A classifier B_Q trained on $Z_{\mathcal{L}}$
repeat

1. Compute the threshold $\boldsymbol{\theta}^*$ that minimizes the conditional Bayes error rate:

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta} \in (0,1]^K}{\operatorname{argmin}} R_{\mathcal{U}|\boldsymbol{\theta}}(B_Q). \qquad (\star)$$

2. $S_{\text{high}} \leftarrow \{(\mathbf{x}, \hat{y}) | \mathbf{x} \in X_{\mathcal{U}}; [v_Q(\mathbf{x}, \hat{y}) \geq \theta^*_{\hat{y}}] \wedge [\hat{y} = \operatorname{argmax}_{c \in \mathcal{Y}} v_Q(\mathbf{x}, c)]\}$
3. $Z_{\mathcal{P}} \leftarrow Z_{\mathcal{P}} \cup S_{\text{high}}$, $X_{\mathcal{U}} \leftarrow X_{\mathcal{U}} \setminus S_{\text{high}}$
4. Re-train B_Q on $Z_{\mathcal{L}} \cup Z_{\mathcal{P}}$ with the following instance weights s_i:

$$s_i = \begin{cases} (l + |Z_{\mathcal{P}}|)/l & \text{for } i \in \{1, \ldots, l\}, \\ (l + |Z_{\mathcal{P}}|)/|Z_{\mathcal{P}}| & \text{for } i \in \{l+1, \ldots, l + |Z_{\mathcal{P}}|.\} \end{cases}$$

until $X_{\mathcal{U}}$ or S_{high} are $\emptyset$
Output: The final classifier B_Q

direct grid search in the multi-class setting costs $O\left(R^K\right)$, where R is the sampling rate of the grid. As

$$\begin{aligned} R_{\mathcal{U}|\boldsymbol{\theta}}(B_Q) &= \sum_{j=1}^{K} \frac{R^{(j)}_{\mathcal{U}\wedge\boldsymbol{\theta}}(B_Q)}{\sum_{c=1}^{K} \frac{1}{u} \sum_{\mathbf{x} \in X_{\mathcal{U}}} \mathbb{1}_{v_Q(\mathbf{x},c) \geq \theta_c} \mathbb{1}_{B_Q(\mathbf{x})=c}} \leq \sum_{j=1}^{K} \frac{R^{(j)}_{\mathcal{U}\wedge\boldsymbol{\theta}}(B_Q)}{\frac{1}{u} \sum_{\mathbf{x} \in X_{\mathcal{U}}} \mathbb{1}_{v_Q(\mathbf{x},j) \geq \theta_j} \mathbb{1}_{B_Q(\mathbf{x})=j}} \\ &\leq \sum_{j=1}^{K} \frac{R^{(j)}_{\mathcal{U}\wedge\boldsymbol{\theta}}(B_Q)}{\pi\{(v_Q(\mathbf{x}, j) \geq \theta_j) \wedge (B_Q(\mathbf{x}) = j)\}}, \end{aligned} \qquad (*)$$

where $R^{(j)}_{\mathcal{U}\wedge\boldsymbol{\theta}}(B_Q) = \sum_{i=1}^{K} u_i R_{\mathcal{U}\wedge\boldsymbol{\theta}}(B_Q, i, j)/u$, the sum might be minimized term by term, tuning independently each component of $\boldsymbol{\theta}$. This replaces the K-dimensional minimization task by K tasks of 1-dimensional minimization.

3.1.2 Experimental Setup

In this section, we present our experimental setup for validation of the proposed approach. All experiments were performed on a cluster with an `Intel(R) Xeon(R) CPU E5-2640 v3` at `2.60GHz`, `32` cores, `256GB` of RAM, the `Debian 4.9.110-3 x86_-64` OS.

Data set	# of labeled examples, l	# of unlabeled examples, u	Dimension, d	# of classes, K
Vowel	99	891	10	11
Protein	129	951	77	8
DNA	31	3155	180	3
PageBlocks	1094	4379	10	5
Isolet	389	7408	617	26
HAR	102	10197	561	6
Pendigits	109	10883	16	10
Letter	400	19600	16	26
Fashion	175	69825	784	10
MNIST	175	69825	784	10
SensIT	49	98479	100	3

Table 3.1: Characteristics of data sets used in our experiments ordered by the size of the training set ($n = l + u$).

Experiments are conducted on publicly available data sets (Dua and Graff, 2017; Chang and Lin, 2011; Xiao et al., 2017). Since we are interested in the practical use of our approach in the semi-supervised context, we would like to see if it has good performance when $l \ll u$. Therefore, we do not use the train/test splits that are proposed by data sources. Instead, we propose our own splits that makes a situation closer to the semi-supervised context. Each experiment is conducted 20 times, by randomly splitting an original data set on a labeled and an unlabeled parts keeping fixed their respective size at each iteration. The reported performance results are averaged over the 20 trials. We evaluate the performance as the accuracy score over the unlabeled training set (ACC-U).

In all our experiments, we consider the Random Forest algorithm (Breiman, 2001) (denoted by RF) with 200 trees and the maximal depth of trees as the majority vote classifier with the uniform posterior distribution. We use the implementation of Pedregosa et al. (2011) and keep the default values for the other parameters. For an observation $\mathbf{x}$, we evaluate the vector of class votes $\{v(\mathbf{x}, i)\}_{i=1}^{K}$ by averaging over the trees the vote given to each class by the tree. A tree computes a class vote as the fraction of training examples in a leaf belonging to a class.

Experiments are conducted on 11 real data sets. The associated applications are image classification with the Fashion data set, the Pendigits and the MNIST databases of handwritten digits; a signal processing application with the SensIT database for vehicle type classification and the human activity recognition HAR data set; speech recognition using the Vowel, the Isolet and the Letter data sets; document recognition using the Page Blocks database; and finally applications to bioinformatics with the Protein and DNA data sets. The main characteristics of these data sets are summarized in Table 3.1.

The proposed MSLA that automatically finds the threshold by minimizing the conditional Bayes error rate, is compared with the following baselines:

- a fully supervised RF trained using only labeled examples. The approach is obtained at the initialization step of MSLA and once learned it is directly applied to predict the class labels of the whole unlabeled set;

- the scikit-learn's implementation (Pedregosa et al., 2011) of the graph based, label spreading algorithm (Zhou et al., 2004) denoted by LS;

- a transductive support vector machine (Joachims, 1999) using the Quasi-Newton scheme proposed by Gieseke et al. (2014) [2] and denoted further as QN-S3VM), which we extend to the multi-class case by the one-versus-all rule;

- a semi-supervised extension of the linear discriminant analysis (Semi-LDA) that is based on the contrastive pessimistic likelihood estimation proposed by Loog (2015);

- a semi-supervised extension of the random forest (DAS-RF) proposed by Leistner et al. (2009) where the classifier is repeatedly re-trained on the labeled and all the unlabeled examples with pseudo-labels optimized via deterministic annealing;

- the multi-class extension of the classical self-learning approach (denoted by FSLA) described by Tür et al. (2005) with a fixed prediction vote threshold;

- a self-learning approach (denoted by CSLA) where the threshold is defined via curriculum learning by taking it as the $(1 - t \cdot \Delta)$-th percentile of the prediction vote distribution at the step $t = 1, 2, \ldots$ (Cascante-Bonilla et al., 2020).

As the size of the labeled training examples $|Z_{\mathcal{L}}|$ is small, the hyperparameter tuning can not be performed properly. At the same time, the performance of baselines may be sensitive to some of their hyperparameters. For this reason, we compute LS, QN-S3VM, Semi-LDA, DAS-RF on a grid of parameters' values, and then choose a hyperparameter for which the performance is the best in average on 20 trials. We tune the RBF kernel parameter $\sigma \in \{10, 1.5, 0.5, 10^{-1}, 10^{-2}, 10^{-3}\}$ for LS, the regularization parameters $(\lambda, \lambda') \in \{10^{-1}, 10^{-2}, 10^{-3}\}^2$ for QN-S3VM, the learning rate $\alpha \in \{10^{-4}, 10^{-3}, 10^{-2}\}$ for Semi-LDA, the initial temperature $T_0 \in \{10^{-3}, 5 \cdot 10^{-3}, 10^{-2}\}$ for DAS-RF. Other hyperparameters for these algorithms are left to their default values. Particularly, in DAS-RF the strength parameter and the number of iterations are respectively set to 0.1 and 10.

While the aforementioned parameters are rather data-dependent, the choice of θ for FSLA and Δ for CSLA depend more on how the prediction vote given by the base classifier is distributed. After manually testing different values, we have found that $\text{FSLA}_{\theta=0.7}$ and $\text{CSLA}_{\Delta} = 1/3$ are good choices for the random forest. For FSLA, we terminate the learning procedure as soon as the algorithm makes 10 iterations,

[2] The source code for the binary QN-S3VM is available at http://www.fabiangieseke.de/index.php/code/qns3vm.

Data set	RF	LS	QN-S3VM	Semi-LDA	DAS-RF	FSLA $_{\theta=0.7}$	CSLA $_{\Delta=1/3}$	MSLA
Vowel	.586 ± .028	**.602** ± .026	.208$^{\downarrow}$ ± .029	.432$^{\downarrow}$ ± .029	.587 ± .028	.531$^{\downarrow}$ ± .034	.576$^{\downarrow}$ ± .031	.586 ± .026
Protein	.764$^{\downarrow}$ ± .032	.825 ± .028	.72$^{\downarrow}$ ± .034	**.842** ± .029	.768$^{\downarrow}$ ± .036	.687$^{\downarrow}$ ± .036	.771$^{\downarrow}$ ± .035	.781$^{\downarrow}$ ± .034
DNA	.693$^{\downarrow}$ ± .074	.584$^{\downarrow}$ ± .038	**.815** ± .025	.573$^{\downarrow}$ ± .037	.693$^{\downarrow}$ ± .083	.521$^{\downarrow}$ ± .095	.671$^{\downarrow}$ ± .112	.702$^{\downarrow}$ ± .082
PageBlocks	.965 ± .003	.905$^{\downarrow}$ ± .004	.931$^{\downarrow}$ ± .003	.935$^{\downarrow}$ ± .009	.965 ± .003	.964 ± .004	.965 ± .003	**.966** ± .002
Isolet	.854$^{\downarrow}$ ± .016	.727$^{\downarrow}$ ± .01	.652$^{\downarrow}$ ± .016	.787$^{\downarrow}$ ± .019	.859$^{\downarrow}$ ± .018	.7$^{\downarrow}$ ± .04	.843$^{\downarrow}$ ± .021	**.875** ± .014
HAR	.851 ± .024	.215$^{\downarrow}$ ± .05	.78$^{\downarrow}$ ± .02	.743$^{\downarrow}$ ± .043	.852 ± .024	.81$^{\downarrow}$ ± .041	.841 ± .029	**.854** ± .026
Pendigits	.863$^{\downarrow}$ ± .022	**.916** ± .013	.675$^{\downarrow}$ ± .022	.824$^{\downarrow}$ ± .012	.872$^{\downarrow}$ ± .023	.839$^{\downarrow}$ ± .036	.871$^{\downarrow}$ ± .029	.884$^{\downarrow}$ ± .022
Letter	.711 ± .011	.664$^{\downarrow}$ ± .01	.064$^{\downarrow}$ ± .013	.589$^{\downarrow}$ ± .016	.718 ± .012	.651$^{\downarrow}$ ± .015	**.72** ± .013	.717 ± .013
Fashion	.718 ± .022	NA	NA	.537$^{\downarrow}$ ± .027	.722 ± .023	.64$^{\downarrow}$ ± .04	.713 ± .026	**.723** ± .023
MNIST	.798$^{\downarrow}$ ± .015	NA	NA	.423$^{\downarrow}$ ± .029	.822$^{\downarrow}$ ± .017	.705$^{\downarrow}$ ± .055	.829$^{\downarrow}$ ± .02	**.857** ± .013
SensIT	**.723** ± .022	NA	NA	.647$^{\downarrow}$ ± .042	**.723** ± .022	.692$^{\downarrow}$ ± .023	.713 ± .024	.722 ± .021

Table 3.2: Classification performance on different data sets described in Table 3.1. The performance is computed using the accuracy score on the unlabeled training examples (ACC-U). The sign $^{\downarrow}$ shows if the performance is statistically worse than the best result on the level 0.01 of significance. NA indicates the case when the time limit was exceeded.

which reduces the computation time and may also improve the performance, since, in this case, the algorithm is less affected by noise. Cascante-Bonilla et al. (2020) used for CSLA a slightly other architecture for self-learning, where the set of selected pseudo-labeled examples included just for one iteration (like if in Algorithm 1 Step 3 would be replaced by $Z_{\mathcal{P}} \leftarrow S$). In our context, we have found that the performance of CSLA is identical for both two architectures.

3.1.3 Experimental Results

In our setup, a time deadline is set: we stop computation for an algorithm if one trial takes more than 4 hours. Table 3.2 summarizes results obtained by RF, LP, QN-S3VM, Semi-LDA, DAS-RF, FSLA, CSLA and MSLA. We used bold face to indicate the highest performance rates and the symbol $\downarrow$ indicates that the performance is significantly worse than the best result, according to Mann-Whitney U test (Mann and Whitney, 1947) used at the p-value threshold of 0.01.

From these results it comes out that

- in 5 of 11 cases, the MSLA performs better than its opponents. On data sets Isolet and MNIST it significantly outperforms all the others, and it significantly outperforms the baseline RF on Isolet, Pendigits and MNIST (6% improvement);

- the LS and the QN-S3VM did not pass the scale over larger data sets (Fashion, MNIST and SensIT), while the MSLA did not exceeded 2 minutes per trial on these data sets (see Table 3.3);

- the performance of LS and Semi-LDA performance varies greatly on different data sets, which may be caused by the topology of data. In contrast, MSLA has more stable results over all data sets as it is based on the predictive score, and the RF is used as the base classifier;

- since the QN-S3VM is a binary classifier by nature, its one-versus-all extension is not robust with respect to the number of classes. This can be observed on Vowel, Isolet and Letter, where the number of classes is high;

- from our observation, both LS and QN-S3VM are highly sensitive to the choice of the hyperparameters. However, it is not very clear whether these hyperparameters can be properly tuned given a insufficient number of labeled examples. The same concern is applied to all the other semi-supervised baselines, while MSLA does not require any particular tuning since it finds automatically the threshold $\boldsymbol{\theta}$;

- while the approach proposed by Loog (2015) always guarantees an improvement of the likelihood compared to the supervised case, we have observed that the classification accuracy is not always improved for Semi-LDA and may even degrade over the supervised linear discriminant analysis;

- compared to the fully supervised approach, RF, the use of pseudo-labeled unlabeled training data (in DAS-RF, FSLA, CSLA or MSLA) may generally give no benefit or even degrade performance in some cases (Vowel, PageBlocks, SensIT). This may be due to the fact that the learning hypotheses are not met regarding the data sets where this effect is observed;

- although for DAS-RF the performance is usually not degraded when T_0 is properly chosen, it has rather little improvement compared to RF. The performance of FSLA degrades most of the time, while degradation for CSLA is observed on 6 data sets. The latter suggests that the choice of the threshold for pseudo-labeling is crucial and challenging in the multi-class framework. Using the proposed criterion based on Eq. (3.1), we can find the threshold efficiently;

- from the results it can be seen that self-learning is also sensitive to the choice of the initial classifier. On some data sets, the number of labeled examples might be too small leading to a bad initialization of the first classifier trained over the labeled set. This implies that the initial votes are biased, so even with a well picked threshold we do not expect a great increase in performance (see Appendix 3.1.5 for more details).

We also analyze the behavior of the various algorithms for growing the initial number of labeled data in the training set. Figure 3-1 illustrates this by showing the accuracy on a subsample of 3500 observations from MNIST of RF, QN-S3VM, FSLA$_{\boldsymbol{\theta}=0.7}$ and MSLA with respect to the percentage of the labeled training examples. In this

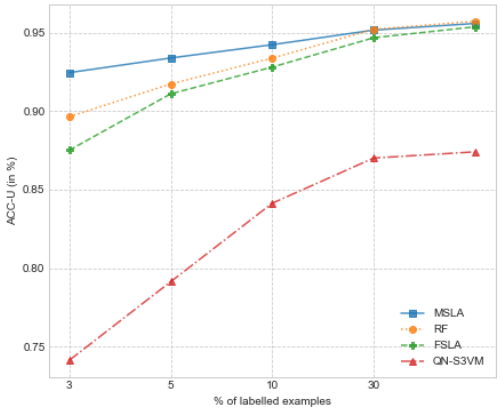


Figure 3-1: Classification accuracy with respect to the proportion of unlabeled examples for the MNIST data set (a subsample of 3500 examples). On the graph, dots represent the average performance on the unlabeled examples over 20 random splits. For simplicity of illustration, the other considered algorithms are not displayed.

graph, the performance of LS is not depicted, since it is significantly lower compared to the other methods under consideration. As expected, all performance curves increase monotonically with respect to the additional labeled data. When there are sufficient labeled training examples, MSLA, FSLA and RF actually converge to the same accuracy performance, suggesting that the labeled data carries out sufficient information and no additional information could be extracted from unlabeled examples.

3.1.4 Complexity Analysis

Further, we present a comparison of the learning algorithms under consideration by analyzing their complexity. The time complexity of the random forest RF is $O(Td\tilde{l}\log^2\tilde{l})$ (Louppe, 2014), where T is the number of decision trees in the forest and $\tilde{l} \approx 0.632 \cdot l$ is the number of training examples used for each tree. Since RF is employed in DAS-RF and self-learning, the time complexity of DAS-RF, FSLA and CSLA is $O(CTd\tilde{n}\log^2\tilde{n})$, where C is the number of times RF has been learned, $\tilde{n} \approx 0.632 \cdot n$, $n := l + u$[3]. In our experimental setup, $C = 11$ for FSLA and DAS-RF, and $C = 1/\Delta + 1 = 4$ for CSLA.

The time required for finding the optimal threshold at every iteration of the MSLA is $O(K^2R^2n)$, where R is the sampling rate of the grid. From this we deduce that the complexity of MSLA is $O(C\max(Tdn\log^2 n, K^2R^2n))$. As n grows, the complexity is

[3]For sake of simplicity, we write u, but the total number of pseudo-labeled examples can be generally smaller. However, it is not the case for MSLA and CSLA.

written as $O(dn\log^2 n)$, since C, T, R are constant. This indicates a good scalability of all considered pseudo-labeling methods for large-scale data as they also have a memory consumption proportional to nd, so the computation can be performed on a regular PC even for the large-scale applications.

In the label spreading algorithm, an iterative procedure is performed, where at every step the affinity matrix is computed. Hence, the time complexity of the LS is $O(Mn^2d)$, where M is the maximal number of iterations. From our observation, the convergence of LS is highly influenced by the value of σ and the data topology. The time complexity of the QN-S3VM is $O(n^2d)$ (Gieseke et al., 2014). Both algorithms suffer from high run-time for large-scale applications. Since LS and QN-S3VM evaluate respectively the affinity matrix and the kernel matrix of size n by n, these algorithms have also large space complexity proportional to n^2. From our observation, for the large-scale data (Fashion, MNIST, SensIT) the maximal resident set size[4] of LS and QN-S3VM may reach up to 200GB of RAM, which is practically infeasible with lack of resources.

Finally, the time complexity of Semi-LDA is $O(M\max(nd^2, d^3))$, where M is the maximal number of iterations and $O(\max(nd^2, d^3))$ is the complexity of the linear discriminant analysis assuming $n > d$ (Cai et al., 2008), and the space complexity is $O(nd)$. The approach pass the scale well with respect to the sample size, but may significantly slow down in the case of very large dimension.

Data set	RF	LS	QN-S3VM	Semi-LDA	DAS-RF	FSLA$_{\theta=0.7}$	CSLA$_{\Delta=1/3}$	MSLA
Vowel	1 s	6 s	2 s	3 s	7 s	11 s	2 s	5 s
Protein	1 s	22 s	4 m	5 s	6 s	10 s	2 s	4 s
DNA	1 s	1 m	26 s	1 s	9 s	7 s	3 s	4 s
PageBlocks	1 s	2 m	2 m	14 s	9 s	12 s	3 s	6 s
Isolet	1 s	1 m	1 h	10 s	38 s	16 s	5 s	28 s
HAR	1 s	18 m	32 m	3 s	42 s	23 s	6 s	13 s
Pendigits	1 s	30 m	10 m	37 s	13 s	13 s	3 s	14 s
Letter	1 s	3 h	40 m	1 m	20 s	16 s	5 s	1 m
Fashion	1 s	>4 h	>4 h	1 m	2 m	1 m	29 s	1 m
MNIST	1 s	>4 h	>4 h	1 m	2 m	1 m	29 s	1 m
SensIT	1 s	>4 h	>4 h	2 m	3 m	2 m	30 s	1 m

Table 3.3: The average run-time of the learning algorithms under consideration on the data sets described in Table 3.1. *s* stands for seconds, *m* for minutes and *h* for hours.

To empirically verify the complexity analysis, we present the run-time of all the algorithms empirically compared in Section 3.1.3. The results are depicted in

[4]Maximal resident set size (maxRSS) is the peak portion of memory that was occupied in RAM during the run.

Table3.3. In general, the obtained run-time is coherent with the complexity analysis. LS and QN-S3VM have a very large run-time when they converge slowly, and they are generally slower than the other algorithms. Semi-LDA is fast on the considered data sets, though it may slow down on data of large dimension not considered in our experimental setup.

It can be seen that DAS-RF is slower than the self-learning algorithms, which is due to the fact that the classifier is trained on all labeled and unlabeled examples at each iteration. CSLA is the fastest approach since it re-trains the base classifier only 3 times compared to 10 times for FSLA. From our observation, MSLA needs usually around 3-5 iterations to pseudo-label the whole unlabeled set, but it takes more time than CSLA, since it searches at each iteration the threshold by minimizing the conditional Bayes error. We have implemented the search in a single core, but it can be potentially parallelized. Nevertheless, the MSLA still runs fast.

3.1.5 Approximation of Posterior Probabilities for Self-learning

In this section, we analyze the behavior of MSLA depending on how the transductive bound given by Eq. ($TB_{i,j}$) is evaluated. Since the posterior probabilities for unlabeled data are unknown, we have proposed to estimate them as the votes of the base supervised classifier learned using the labeled data only (Sup. Estimation), which has been used to run MSLA in Section 3.1.3. We compare this approach with a strategy that considers the (worst) case of uniform posterior probabilities (Unif. Estimation), i.e., $P(Y=i|\mathbf{X}=\mathbf{x}) = 1/K,\ \forall \mathbf{x} \in \mathrm{X}_{\mathcal{U}},\ \forall i \in \{1,\ldots,K\}$. Finally, we provide the performance of MSLA when the labels of unlabeled data are given, which means that the transductive bound is truly estimated (Oracle).

Data set	MSLA		
	Unif. Estimation	Sup. Estimation	Oracle
Vowel	.586 ± .029	.586 ± .026	.599 ± .028
Protein	.773 ± .034	.781 ± .034	.805 ± .036
DNA	.697 ± .079	.702 ± .082	.721 ± .09
Page Blocks	.965 ± .002	.966 ± .002	.966 ± .002
Isolet	.869 ± .015	.875 ± .014	.885 ± .012
HAR	.852 ± .025	.854 ± .026	.856 ± .022
Pendigits	.873 ± .024	.884 ± .022	.892 ± .016
Letter	.716 ± .013	.717 ± .013	.723 ± .012
Fashion	.722 ± .022	.723 ± .023	.728 ± .024
MNIST	$.834^{\downarrow}$ ± .016	.857 ± .013	.87 ± .012
SensIT	.722 ± .021	.722 ± .021	.722 ± .021

Table 3.4: The performance comparison of MSLA depending on how the posterior probabilities are estimated in the evaluation of the transductive bound ($TB_{i,j}$).

Table 3.4 illustrates the performance results. As we can see, the supervised approximation generally outperforms the uniform one (significantly on MNIST). This might be explained by the fact that the supervised votes may give some additional information on the most probable labels for each example. In addition, we have observed that on the last iterations the votes of MSLA tend to be biased, so such posteriors can play a role of regularization. The performance results of the oracle show that better estimation of the posteriors can give an improvement, though not significantly on most of data sets. Note that the performance of the oracle is not perfect, because the true labels are used only for the bound estimation, and the votes are used for pseudo-labeling.

3.1.6 Discussion on Confirmation Bias

We have experimentally showed that the self-learning algorithm can be a very powerful semi-supervised algorithm, if the threshold on the prediction vote is properly chosen. However, the use of self-learning in practice has some limitations. For example, a post-analysis of a self-learning model is perplexing due to the so-called *confirmation bias*: at every iteration, the self-learning includes into the training set unlabeled examples with highly confident predictions, which arise from classifier's overconfidence to its initial decisions that could be erroneous. This implies that the hypotheses will have small disagreement on the unlabeled set after pseudo-labeling, so the votes are no more adequate for measuring prediction confidence.

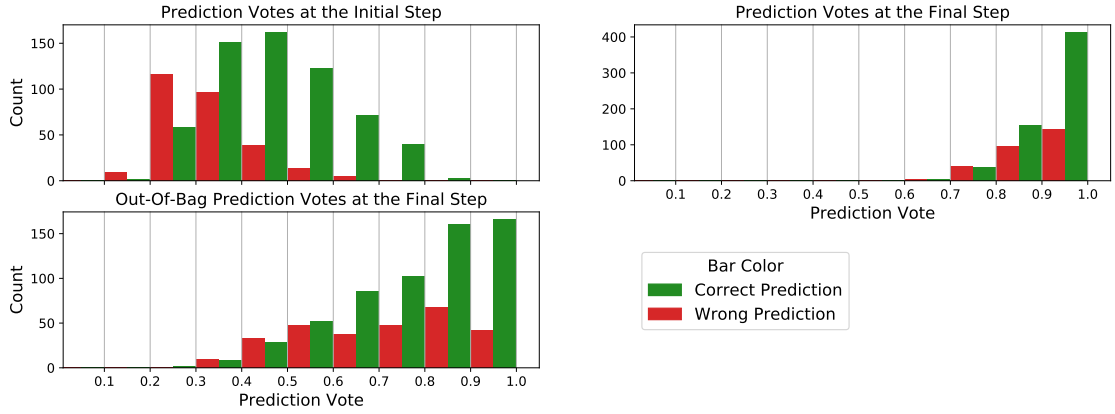


Figure 3-2: Illustration of the confirmation bias on Vowel data set. We present the prediction vote's distribution on the unlabeled set at the initial and the final steps of MSLA. In addition, we show for the final step how the out-of-bag prediction votes are distributed. Every histogram's column is divided by two in order to show how many examples were predicted correctly and incorrectly.

In order to illustrate this phenomena, in Figure 3-2 we plotted the distribution of the prediction vote on all the unlabeled examples of Vowel at the initial step (RF trained on the labeled examples only) and at the final step (all the unlabeled

examples are included with pseudo-labels in the training set). At the initial step, we mistake roughly on low prediction votes, while the final classifier gives mostly high prediction vote both for correctly and wrongly predicted examples. One can argue that the bias in the latter case could be caused by inclusion of the unlabeled examples in the training set. Then, it could be a better idea to evaluate votes in the out-of-bag fashion (Eq. (1.6)), i.e., to consider only those trees that did not contain a training example in their corresponding bootstrap samples. For instance, we will show in Chapter 4 that the out-of-bag error evaluated on the pseudo-labeled examples can be a good selection criterion. However, from the bottom left plot of Figure 3-2, where the distribution of the out-of-bag prediction votes is illustrated, one can see that the conclusion is the same: although the out-of-bag votes are more diverse, evaluation of the prediction confidence does not seem reliable.

3.2 Self-learning for a Biological Application

3.2.1 Introduction

In recent years, a great attention in microbiology is focused on the clustered regularly interspaced short palindromic repeats (CRISPR) that represent specific DNA sequences found in the genomes of bacteria and archaea. Serving as a memory bank of previous infections, a CRISPR array together with CRISPR-associated (Cas) proteins forms a CRISPR-Cas system that plays a role in adaptive immunity against phages and other exogenous mobile genetic elements (Hille et al., 2018). The CRISPR-Cas system is an active subject of study, and it has found a breakthrough application in the development of a new generation of genome editing systems (Pickar-Oliver and Gersbach, 2019).

Due to the fast evolution of the CRISPR–Cas loci, the number of different found CRISPR-Cas systems is growing every year, which challenges to build a robust classification scheme. For example, the most recent classification includes 6 types and 33 subtypes, compared with 5 types and 16 subtypes in 2015 (Makarova et al., 2020). In parallel, there is a growing interest in machine learning tools that are capable of automatically detecting the CRISPR-Cas system types and subtypes. For example, Russel et al. (2020) use a gradient boosting tree approach and aim to predict a CRISPR-Cas subtype based on extracted tetramers of an input CRISPR array. However, this model has two significant limitations. First, they follow the classical supervised setting, so their approach will poorly generalize on new data, where new subtypes may appear. Secondly, all their training sequences are supplied with true labels, which are possible to acquire only if a full genome is available. This is not always the case, so some training sequences rest naturally unlabeled.

In this section, we propose a novel view on this problem by considering the open-world classification (Bendale and Boult, 2015) framework, which represents a semi-supervised problem, where it is additionally assumed that unlabeled examples

contain classes unseen in the labeled data. By studying the CRISPR-Cas subtype classification in this context, our goal is three-fold: i) we want to automatically classify subtypes observed in the training set, ii) discover new subtypes by providing regions of grouped examples with low prediction confidence for their successive analysis by experts, iii) refrain from predicting any label for non-grouped examples with low prediction confidence.

In order to deal with this problem, we assume that unlabeled examples from new classes lie in homogeneous clusters. We propose to use the semi-supervised decision tree of Kim (2016) and the one-versus-all rule to construct a random forest classifier that is able to give low confidence to unlabeled examples from isolated regions. Based on this classifier and a density-based clustering algorithm, we propose an extension of self-learning that is able to pseudo-label not only unlabeled examples with high prediction confidence, but also clustered unlabeled examples with low confidence treating them as new classes. To the best of our knowledge, the use of decision trees and self-learning in the open-world context is a new result. We validate our approach on the real data set of CRISPR arrays extracted from genomic sequences by a CRISPRCasFinder software (Couvin et al., 2018).

The rest of this section is organized as follows. Section 3.2.2 formally defines our problem and the learning objective. In Section 3.2.3, we introduce the proposed open-world self-learning algorithm. In Section 3.2.4, we describe the data set and present the results of the performed experiments.

3.2.2 Framework

In this section, we define a semi-supervised problem in a non-classical manner. As before, we consider an input space $\mathcal{X} \subset \mathbb{R}^d$, but define an output space as union of two disjoint sets $\mathcal{Y} = \mathcal{Y}_{\mathcal{O}} \cup \mathcal{Y}_{\emptyset}$, where $\mathcal{Y}_{\mathcal{O}} = \{1, \ldots, K\},\ K \geq 2$, is the set of observed classes, whereas $\mathcal{Y}_{\emptyset} = \{K+1, \ldots, K+C\},\ C \geq 1$, is the set of unseen classes. Then, we assume an available set of labeled training examples from the observed classes $Z_{\mathcal{L}} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{l} \in (\mathcal{X} \times \mathcal{Y}_{\mathcal{O}})^l$ and an available set of unlabeled training examples $X_{\mathcal{U}} = \{\mathbf{x}_i\}_{i=l+1}^{l+u} \in \mathcal{X}^u$ for which it is assumed known that some of these examples belong to unseen classes, i.e., the true (not available) class label $y \in \mathcal{Y} = \mathcal{Y}_{\mathcal{O}} \cup \mathcal{Y}_{\emptyset}$ for every $\mathbf{x} \in X_{\mathcal{U}}$[5].

Generally speaking, the absence of some classes in the labeled set may arise in different situations. In the case of i.i.d. data generation, this may happen if the labeled set is scarce, and the problem is highly imbalanced, hence some classes may have little chances to appear. Another situation is when data labeling is performed according to a some criterion (Cortes et al., 2008), i.e., the i.i.d. data assumption is violated. In this chapter, we consider the problem from an algorithmic point of view, so we do not assume explicitly the reason of the absence of some classes.

However, it is clear that classical learning objectives would not be appropriate

[5]For sake of simplicity, we consider the deterministic setting in this section.

in this framework. Thus, we propose to slightly modify the goal of learning. First, we assume that unlabeled data contain some homogeneous regions (clusters) that are prone to error. These regions may or may not contain new classes, but the main restriction is that every cluster shares the same label. Then we formulate the goal as to seek for a classifier h based on $Z_{\mathcal{L}} \cup X_{\mathcal{U}}$ that is able to i) predict correctly unlabeled examples of high confidence from the observed classes, ii) cluster unlabeled examples of low confidence into homogeneous regions, iii) leave completely unlabeled those examples that are neither classified nor clustered, minimizing their number. We use the following notations: for $\mathbf{x} \in X_{\mathcal{U}}$, $h(\mathbf{x}) \in \mathcal{Y}_{\mathcal{O}}$ if h predicts one of the observed classes, $h(\mathbf{x}) \in \mathcal{C}$ if h assigns $\mathbf{x}$ to one of the found clusters, where $\mathcal{C} = \{K+1, \ldots\}$ are cluster indices, $h(\mathbf{x}) = -1$, if h abstain from predicting any label.

In order to fulfill criteria i), ii) and iii), we define the following evaluation measures. Let $S_{\mathcal{U}\to\mathcal{O}} := \{\mathbf{x} \in X_{\mathcal{U}} : h(\mathbf{x}) \in \mathcal{Y}_{\mathcal{O}}\}$ be the set of unlabeled examples that are assigned by h to one of the observed classes. Then, to satisfy i), we aim for maximizing the balanced accuracy score (BACC) computed on the unlabeled examples from $S_{\mathcal{U}\to\mathcal{O}}$:

$$\texttt{BACC}_{\mathcal{U}\to\mathcal{O}} := \frac{1}{K}\sum_{k=1}^{K}\frac{\sum_{\mathbf{x}\in S_{\mathcal{U}\to\mathcal{O}}} \mathbb{I}(h(\mathbf{x})=k)\mathbb{I}(y(\mathbf{x})=k)}{\sum_{\mathbf{x}\in S_{\mathcal{U}\to\mathcal{O}}} \mathbb{I}(y(\mathbf{x})=k)}, \tag{3.2}$$

where $y(\mathbf{x})$ denotes the true label of the unlabeled example $\mathbf{x}$. For the task ii), we expect that in each cluster $c \in \mathcal{C}$ their unlabeled examples share the same class label. Let $S_{\mathcal{U}\to c} = \{\mathbf{x} \in X_{\mathcal{U}} : h(\mathbf{x}) = c\}$ denotes the set of unlabeled examples that belong to the cluster c. Then, we define the label agreement inside c as the proportion of the most represented class label inside this cluster. Thus, we aim for maximizing the average label agreement (ALA) over the whole set of clusters $\mathcal{C}$ defined as follows:

$$\texttt{ALA}_{\mathcal{U}\to\mathcal{C}} := \frac{1}{|\mathcal{C}|}\sum_{c\in\mathcal{C}}\frac{1}{|S_{\mathcal{U}\to c}|}\max_{k\in\mathcal{Y}_{\mathcal{O}}\cup\mathcal{Y}_{\varnothing}}\sum_{\mathbf{x}\in S_{\mathcal{U}\to c}}\mathbb{I}(y(\mathbf{x})=k). \tag{3.3}$$

$\texttt{ALA}_{\mathcal{U}\to\mathcal{C}} = 1$ means that in every cluster there are unlabeled examples only of one true label, which implies that the cluster is perfect. This measure can also be interpreted as follows: if we take one example from each cluster, ask to label each example by a human expert (assuming it possible) and propagate the label of each example to its cluster members, then $\texttt{ALA}_{\mathcal{U}\to\mathcal{C}}$ corresponds to the balanced accuracy in expectation over all examples. Finally, to fulfill iii), we prefer a classifier with a minimal abstention.

3.2.3 Open-World Self-learning Algorithm

In this section, we propose a modification of the self-learning algorithm with the random forest as the base classifier to satisfy the framework's requirements. We

improve uncertainty estimation of the random forest by considering the low density separation split criterion of Kim (2016) and using the one-versus-all rule. The unlabeled examples of high confidence are pseudo-labeled by one of the observed class labels and included in the training set, while the examples of low confidence are sent to a clustering algorithm. For each found cluster, all their examples are pseudo-labeled with a newly created class label and are included in the training set. We use a simplified density-based approach for clustering. We call the whole algorithm the open self-learning algorithm (OSLA), describe it Algorithm 2, and give the implementation details below.

Uncertainty Estimation

As we discussed in Section 1.2.1, in the case of a scarce labeled set, decision trees may poorly estimate the confidence in regions, where there are few or no training examples. This may lead to a paradox situation when a large prediction confidence is assigned for some misclassified examples. To overcome this, we use the approach proposed by Kim (2016) and described in Section 1.2.1, which consists in using a semi-supervised split criterion. At every node, we search a feature f and its split s such that it decreases the impurity of the labeled examples ($\mathcal{S}^{\text{sup}}$, Eq. (1.4)) and splits the labeled and unlabeled data in a place of low density ($\mathcal{S}^{\text{unsup}}$, Eq. (1.14)), which results in maximization of the following criterion: $\mathcal{S}^{\text{ssl}}(f,s) := (1-\lambda)\mathcal{S}^{\text{sup}}(f,s) + \lambda\mathcal{S}^{\text{unsup}}(f,s)$, where $\lambda \in [0,1]$ is a balance parameter. Further, we refer this approach as the semi-supervised decision tree, while a random forest of such trees is called the semi-supervised random forest (SSRF). By default, we use further the semi-supervised random forest until the opposite is said. We implement this approach slightly different from Kim (2016). To split a node based on $\mathcal{S}^{\text{ssl}}(f,s)$, we need to have at least two labeled examples. In our implementation, when the node contains unlabeled examples only, we use replace the criterion by $\mathcal{S}^{\text{unsup}}(f,s)$. Then, for purely unlabeled leafs, the tree outputs uniform posterior probabilities. Our motivation stems from the desire to find regions isolated from the labeled data and give to such unlabeled examples a low prediction confidence. To avoid excessive fragmentation, we have fixed the maximum tree depth to 12.

We additionally employ to our framework an option to reject predicting any class. For this, instead of using a single K-class random forest classifier B_{RF}, we employ a one-vs-all classifier $B^{\text{ova}}_{\text{RF}}$ that aggregates predictions of K binary classifiers $B^{(1)}_{\text{RF}}, \ldots, B^{(K)}_{\text{RF}}$, where $B^{(k)}_{\text{RF}}(\mathbf{x})$ outputs 1 when the class k is predicted for $\mathbf{x}$, and -1 otherwise. When for all k, $B^{(k)}_{\text{RF}}(\mathbf{x})$ is -1, then label prediction for $\mathbf{x}$ is rejected:

$$B^{\text{ova}}_{\text{RF}} = \begin{cases} k & \text{if } B^{(k)}_{\text{RF}}(\mathbf{x}) = 1, \\ -1 & \text{if } \forall k \in \mathcal{Y}_{\mathcal{O}}, B^{(k)}_{\text{RF}}(\mathbf{x}) = -1. \end{cases}$$

There is (rather small) probability that the one-vs-all classifier attributes to more

than one class, which can be solved by randomly predicting one of these classes. In order to define the prediction confidence of the label predicted by $B_{\text{RF}}^{\text{ova}}$ for an example $\mathbf{x}$, we use the class vote $v_{\text{RF}}^{(k)}(\mathbf{x},1)$ of the classifier $B_{\text{RF}}^{(k)}$, where $k = B_{\text{RF}}^{\text{ova}}(\mathbf{x})$. To simplify notations, we use $\nu_{\text{RF}}^{\text{ova}}(\mathbf{x}) := v_{\text{RF}}^{(k)}(\mathbf{x},1),\ k = B_{\text{RF}}^{\text{ova}}(\mathbf{x})$.

Consequently, we can determine a set of predictions in which we are confident (S_{high}) by comparing $\nu_{\text{RF}}^{\text{ova}}(\mathbf{x})$ with a high value threshold θ (set to 0.8). Similarly, we can define a set of examples (S_{low}) that are suspect to come from other classes. For this, we check if for all $k \in \{1,\ldots,K\}$, $v_{\text{RF}}^{(k)}(\mathbf{x},1)$ is lower than a small value threshold τ. When $\tau = 0.5$, we reject only those examples for which all $B_{\text{RF}}^{(k)}$ predict - 1, but τ can be set to a higher value. We have fixed the parameter to $\tau = 0.55$.

Algorithm 2 Open-world Self-learning Algorithm (OSLA)

Input:
Labeled observations $\text{Z}_{\mathcal{L}}$ and unlabeled observations $\text{X}_{\mathcal{U}}$,
Thresholds θ and τ s.t. $0 < \tau < \theta < 1$ and the maximum of iterations in self-learning $\bar{n}_{\text{iter}}$,
Number of nearest neighbors n_{nbr} and minimum cluster size $\underline{n}_{cls}$,
A one-vs-all random forest $B_{\text{RF}}^{\text{ova}}$ and a density-based clustering algorithm $\mathcal{Q}$.
Initialization: A set of pseudo-labeled examples, $\text{Z}_{\mathcal{P}} \leftarrow \emptyset$
Train $B_{\text{RF}}^{\text{ova}}$ on $\text{Z}_{\mathcal{L}}$ and $\text{X}_{\mathcal{U}}$.
repeat

1. Among the unlabeled examples $\mathbf{x} \in \text{X}_{\mathcal{U}}$ find those that have high and low prediction confidence:

$$\text{S}_{\text{high}} \leftarrow \{\mathbf{x} \in \text{X}_{\mathcal{U}} : \nu_{\text{RF}}^{\text{ova}}(\mathbf{x}) \geq \theta\}, \qquad \text{S}_{\text{low}} \leftarrow \{\mathbf{x} \in \text{X}_{\mathcal{U}} : \wedge_k (v_{\text{RF}}^{(k)}(\mathbf{x},1) \leq \tau)\}.$$

2. Pseudo-label the examples from S_{high} and move them into the training set:

$$\text{Z}_{\mathcal{P}} \leftarrow \text{Z}_{\mathcal{P}} \cup \left(\text{S}_{\text{high}}, \{B_{\text{RF}}^{\text{ova}}(\mathbf{x})\}_{\mathbf{x}\in\text{S}_{\text{high}}}\right), \quad \text{X}_{\mathcal{U}} \leftarrow \text{X}_{\mathcal{U}} \setminus \text{S}_{\text{high}}.$$

3. Find a set of $\mathcal{C}_t$ clusters inside S_{low}:

$$\{\text{S}_{\mathcal{U}\rightarrow c}\}_{c=1}^{\mathcal{C}_t} \leftarrow \mathcal{Q}(\text{S}_{\text{low}}, n_{\text{nbr}}, \underline{n}_{cls}).$$

4. Pseudo-label all examples from the cluster $\mathbf{x} \in \text{S}_{\mathcal{U}\rightarrow c},\ c \in \mathcal{C}_t$ and move them into the training set:

$$\text{Z}_{\mathcal{P}} \leftarrow \text{Z}_{\mathcal{P}} \cup \left(\text{S}_{\mathcal{U}\rightarrow c}, \{c\}_{\mathbf{x}\in S_{\mathcal{U}\rightarrow c}}\right), \quad \text{X}_{\mathcal{U}} \leftarrow \text{X}_{\mathcal{U}} \setminus S_{\mathcal{U}\rightarrow c}.$$

5. Re-train $B_{\text{RF}}^{\text{ova}}$ on $\text{Z}_{\mathcal{L}} \cup \text{Z}_{\mathcal{P}}$ and $\text{X}_{\mathcal{U}}$.
until $\text{X}_{\mathcal{U}}$ or S_{susp} is $\emptyset$, or the number of iterations $= \bar{n}_{\text{iter}}$
Output: $\text{Z}_{\mathcal{L}} \cup \text{Z}_{\mathcal{P}}$, $(\text{X}_{\mathcal{U}}, \{-1\}_{\mathbf{x}\in\text{X}_{\mathcal{U}}})$.

Finding New Clusters

Then, we search homogeneous structures among examples from S_{low} in order to find unseen regions or classes. For this, we perform a clustering of these examples by an algorithm $\mathcal{Q}$ that is described further. We construct a graph such that vertices are examples $\mathbf{x} \in S_{low}$ and two examples are connected by an edge if one of them belongs to a set of n_{nbr}-nearest neighbors of another. We have considered 5-nearest neighbors. Then, we construct clusters by assigning any two examples to one cluster c if there is a path between them. Note that this algorithm reminds the DBSCAN algorithm (Ester et al., 1996) with an unbounded radius. However, the difference is that we search the nearest neighbors relatively to the whole set of examples $X_{\mathcal{L}} \cup X_{\mathcal{U}}$. Also, we additionally check the size of found clusters and consider those that have size more than $\underline{n}_{cls}$ (fixed to 4).

Self-learning

To sum up, the proposed self-learning is organized as follows. At each iteration, we pseudo-label the confident examples $\mathbf{x} \in S_{high}$ by $B^{ova}_{RF}(\mathbf{x})$ and include them into the training set. Then, using the clustering algorithm $\mathcal{Q}$, we find clusters in S_{low}. For each cluster, we create a unique class label c and use it to pseudo-label all the examples from this cluster. While the inclusion of the confident examples may improve the prediction of the observed classes, by pseudo-labeling the clustered examples, the algorithm treats the clusters as new classes so that they can be enlarged in subsequent iterations. We re-train the classifier B^{ova}_{RF} on the training set augmented by the pseudo-labeled examples. The described steps are repeated until the maximum number of iterations $\bar{n}_{iter}$ is reached or there is nothing to pseudo-label.

3.2.4 CRISPR-Cas Subtype Prediction

Data Set

The data set covers 21 subtypes out of 6 types and contains 45216 sequences extracted by the CRISPRCasFinder (Couvin et al., 2018). In our experiments, we focus on well represented types I and II and subset to those sequences that are CRISPR of the large evidence level according to the software and are associated with only one Cas cluster. We describe a sequence by means of tetramers and RNA-fold descriptors. As the sequence orientation is unknown, we extract canonical tetramers by summing counts of the direct sequence and its reverse complement and keeping the mers that are lexicographically smaller than their reverse complements. Using the ViennaRNA package (Hofacker, 2003), we derive the RNA-fold descriptors at two degrees: 37° and 75°. For each sequence and its reverse complement, we compute the RNA-fold energy at 37° keeping the sequence with the lowest energy, so the RNA-fold descriptive variables at 37° and 75° are evaluated only for the chosen sequence. In total, we have 136 canonical tetramers and $7\times$ 2 RNA-fold descriptors.

Experimental Setup

To validate the proposed approach for the open-world classification, we perform two CRISPR-Cas subtype prediction experiments: in the first case, a training set composed of the subtypes of Type I, whereas the subtypes of Type II are considered for the second experiment. To emulate a situation where unlabeled data is more diverse than labeled one and may contain unseen classes, we split the data set by labeled and unlabeled observations in proportion 10% : 90% and hide the true labels for the latter ones. In addition, some classes are placed entirely in the unlabeled set. More specifically, for Type I we take the set of observed classes as $\mathcal{Y}_{\mathcal{O}} = \{\text{IB}, \text{IC}, \text{IE}, \text{IF}\}$ and the set of new classes as $\mathcal{Y}_{\not\mathcal{O}} = \{\text{IA}, \text{ID}\}$. For Type II, the set of observed classes is $\mathcal{Y}_{\mathcal{O}} = \{\text{IIA}, \text{IIC}\}$, and the set of unseen classes is $\mathcal{Y}_{\not\mathcal{O}} = \{\text{IIB}\}$. The full information about sample sizes can be seen in Table 3.5a for Type I and in Table 3.5b for Type II. We perform randomly 10 labeled/unlabeled splits preserving the proportion and report the performance results averaged over these splits.

Subtype	IA	IB	IC	ID	IE	IF
Labeled	0	33	78	0	482	97
Unlabeled	15	298	703	27	4338	874

(a) CRISPR-Cas Type I

Subtype	IIA	IIB	IIC
Labeled	49	0	62
Unlabeled	440	25	559

(b) CRISPR-Cas Type II

Table 3.5: The distribution of subtypes of CRISPR-Cas Type I and Type II across labeled and unlabeled sets used in experiments.

Learning Model and Baselines

To validate the proposed approach, further referred as `SSRF-OSLA`, we perform an ablative study by empirically comparing with the following baselines:

1. instead of considering the semi-supervised one-versus-all random forest as the base classifier in `OSLA`, use the supervised one-versus-all random forest, i.e., take $\mathcal{S}^{\text{sup}}$ as the split criterion. We refer this approach as `RF-OSLA`;

2. compare with the same algorithm but without iteratively re-training the classifier. In other words, we threshold the examples using θ and τ, assign pseudo-labels and include in the training set, and then terminate the algorithm. This corresponds to a single execution of steps 1-4 of Algorithm 2. We denote this approach by `SSRF-1`.

Experimental Results for Type I

At first, we display in Table 3.6 the distribution of all algorithms' output. One can see that `SSRF-OSLA` assigned in average 84.8% of unlabeled examples to one of the ob-

served classes, 13% classified to one of new clusters and 2.2% left unlabeled. Comparing with SSRF-1, we can observe that at the initial step of self-learning we leave unlabeled 10.6% of examples, which is reduced to 2.2% by iteratively re-training the classifier. The use of the supervised base classifier appears to produce a more confident model in its decisions: RF-OSLA predicts the observed classes more often, which is reasonable, since the supervised classifier does not take into account the unlabeled data. We will see further that this actually deteriorates the performance.

		Method		
		RF-OSLA	SSRF-1	SSRF-OSLA
Predictions (in %)	Observed Class	91.9 ± 0.4	84.8 ± 0.8	84.8 ± 0.8
	New In-Cluster	6.3 ± 0.2	4.6 ± 0.5	13 ± 0.8
	New No-Cluster	1.8 ± 0.2	10.6 ± 0.8	2.2 ± 0.2

Table 3.6: Proportions of unlabeled examples categorized to one of the observed classes, to one of new clusters, or to no cluster (label -1) for the Type I.

Now, we look at performance of the algorithms with respect to the balanced accuracy $\texttt{BACC}_{\mathcal{U}\to\mathcal{O}}$ and the average label agreement $\texttt{ALA}_{\mathcal{U}\to\mathcal{C}}$ defined by Eq. (3.2) and Eq. (3.3) respectively. The performance results are shown in Table 3.7. One can see that when the supervised random forest is used inside OSLA, the performance results are worse in average than for the semi-supervised random forest, which is probably connected with wrong confidence estimation for some unlabeled examples. SSRF-1 and SSRF-OSLA predict almost perfectly the observed classes. It appears that the iterative re-training of the base classifier helps to expand the clusters including more similar examples, so SSRF-OSLA outperforms SSRF-1 in terms of $\texttt{ALA}_{\mathcal{U}\to\mathcal{C}}$.

Method	RF-OSLA	SSRF-1	SSRF-OSLA
$\texttt{BACC}_{\mathcal{U}\to\mathcal{O}}$	$.979 \pm .06$	$\mathbf{.999} \pm .0$	$\mathbf{.999} \pm .0$
$\texttt{ALA}_{\mathcal{U}\to\mathcal{C}}$	$.915 \pm .02$	$.931 \pm .033$	$\mathbf{.962} \pm .011$

Table 3.7: The performance results for the Type I experiment averaged over 10 random labeled / unlabeled splits with respect to the accuracy score on examples assigned to the observed classes ($\texttt{BACC}_{\mathcal{U}\to\mathcal{O}}$) and the average label agreement among the predicted clusters ($\texttt{ALA}_{\mathcal{U}\to\mathcal{C}}$). The bold face indicated the best in average performance.

We have additionally checked the algorithm's behavior particularly on the examples from IA and ID, which are the classes unseen in the training set. The results are promising: none of these observations were classified to one of the observed classes. Figure 3-3 and Figure 3-4 depict to which cluster every example was assigned with respect to the other examples from the same class. The results are averaged over

10 labeled/unlabeled splits. One can see that 13 of 15 examples from IA are always placed in one cluster, whereas 2 examples (No. 2 and 13) are systematically placed into another cluster. We have manually checked and it appears that these two examples are indeed quite different from the other examples from IA with respect to RNAfold parameters. In the case of the subtype ID, all the examples tend to be placed in one cluster with some occasional exceptions.

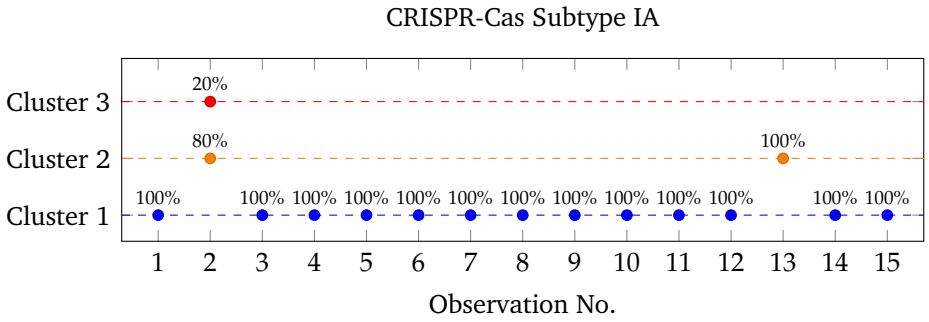


Figure 3-3: Distribution of the unlabeled observations from the subtype `IA` across different clusters. Cluster 1, 2, 3 denote clusters with members from `IA`, and the clusters are sorted by the number of members from `IA` in the decreasing order. The graph displays to which clusters every example belongs to in average over 10 labeled/unlabeled splits. A perfect situation is when all the examples of `IA` belong to Cluster 1.

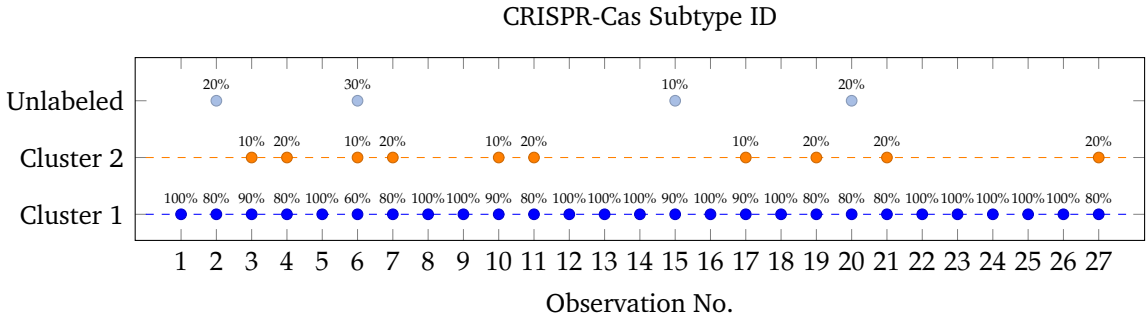


Figure 3-4: Distribution of the unlabeled observations from the subtype `ID` across different clusters. Cluster 1, 2, 3 denote clusters with members from `ID`, and the clusters are sorted by the number of members from `ID` in the decreasing order. The graph displays to which clusters every example belongs to in average over 10 labeled/unlabeled splits. The grey color means that the example was left unlabeled. A perfect situation is when all the examples of `ID` belong to Cluster 1.

Experimental Results for Type II

Now, we discuss the experimental results performed for subtypes of Type II (Table 3.5b). Table 3.8 illustrates the distribution of all algorithms' output. Compared to

the Type I experiment, this time our model hesitates more. SSRF-OSLA predicts one of the observed class labels just for 52% of the unlabeled examples, while we know that approximately 98% of the unlabeled examples are from the observed classes. This is probably due to the fixed θ, so its automatic selection can be a good direction for future work (see Section 3.3). Comparing our approach with with SSRF-1 and RF-OSLA, we make the same conclusions that SSRF-1 leaves too many examples unlabeled, while RF-OSLA tends to be more confident in their predictions, which, again, can be considered rather as a disadvantage in terms of the performance results that are displayed in Table 3.9.

		Method		
		RF-OSLA	SSRF-1	SSRF-OSLA
Predictions (in %)	Observed Class	69.5 ± 5	52 ± 3.7	52 ± 3.7
	New In-Cluster	25.9 ± 4.5	8.7 ± 4.4	43.4 ± 4.4
	New No-Cluster	4.6 ± 1.2	39.3 ± 3.4	4.6 ± 0.9

Table 3.8: Proportions of unlabeled examples categorized to one of the observed classes, to one of new clusters, or to no cluster (label -1) for the Type II.

From the performance results, we again see that SSRF-1 and SSRF-OSLA outperforms RF-OSLA in terms of the balanced accuracy. RF-OSLA pseudo-label more unlabeled examples as the observed classes, which comes with the cost of inducing more errors. This particularly indicates that learning an open-world classifier is about making a trade-off between uncertainty and accuracy. In our experimental setup, it appears that SSRF-OSLA satisfies this trade-off the most, being the best in terms of both $\texttt{BACC}_{\mathcal{U}\to\mathcal{O}}$ and $\texttt{ALA}_{\mathcal{U}\to\mathcal{C}}$.

Method	RF-OSLA	SSRF-1	SSRF-OSLA
$\texttt{BACC}_{\mathcal{U}\to\mathcal{O}}$	$.995 \pm .005$	$\mathbf{.998} \pm .001$	$\mathbf{.998} \pm .001$
$\texttt{ALA}_{\mathcal{U}\to\mathcal{C}}$	$.886 \pm .031$	$.913 \pm .06$	$\mathbf{.925} \pm .03$

Table 3.9: The performance results for the Type II experiment averaged over 10 random labeled / unlabeled splits with respect to $\texttt{BACC}_{\mathcal{U}\to\mathcal{O}}$ and $\texttt{ALA}_{\mathcal{U}\to\mathcal{C}}$. The bold face indicated the best in average performance.

Figure 3-5 demonstrates the prediction results particularly for the examples from the subtype IIB. Same as for IA and ID, none of the examples were misclassfied to one of the observed classes. The examples are primarily placed into two clusters (Cluster 1 and Cluster 2), while all the examples were a part of one cluster on 3 of 10 labeled/unlabeled splits. We found that all observations, attributed in 100% of cases to Cluster 1, actually belong to the same species (legionella pneumophila), while the rest are belong to other species. This particularly means that observations from the same class can naturally split into multiple clusters.

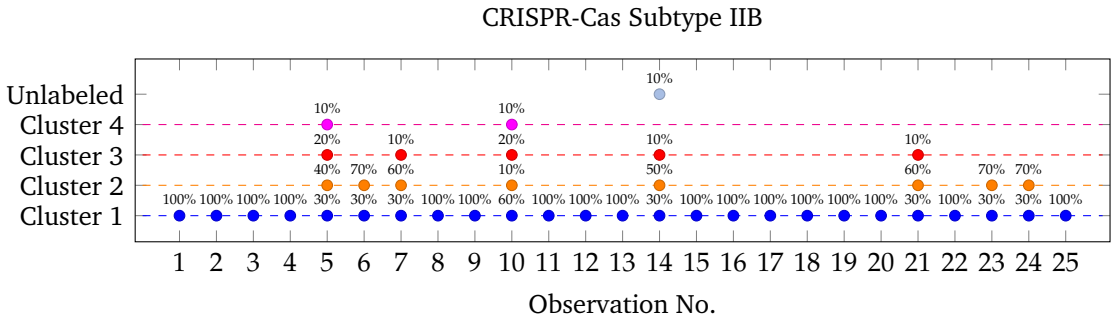


Figure 3-5: Distribution of the unlabeled observations from the subtype `IIB` across different clusters. Cluster 1, 2, 3 denote clusters with members from `IIB`, and the clusters are sorted by the number of members from `IIB` in the decreasing order. The graph displays to which clusters every example belongs to in average over 10 labeled/unlabeled splits. The grey color means that the example was left unlabeled. A perfect situation is when all the examples of `IIB` belong to Cluster 1.

3.3 Conclusion and Perspectives

In this chapter, we proposed a multi-class self-learning algorithm where the threshold for pseudo-labeling unlabeled data is automatically found from minimization of the transductive bound on the majority vote error rate. From the numerical results, it came out that the self-learning algorithm is sensitive to the supervised performance of the base classifier, but it pass well the scale and noticeably outperform other self-learning policies.

The proposed self-learning policy was experimentally validated when it was coupled with the random forest, but combining it with deep learning methods would also be interesting to test. This, however, is not straightforward. It is well known that the modern neural networks are not well calibrated, and examples are often misclassified with a high prediction vote (Guo et al., 2017). This is a significant limitation in our case, since we make an assumption that the classifier makes its mistakes on examples with low prediction votes, which is used for the bound's approximation. Possible solutions include the use of neural network ensembles or temperature scaling. Another direction of future work is to propose a way to deal with the confirmation bias (Section 3.1.6). Zou et al. (2019) studied the impact of different regularizers for solving this problem in the deep learning context. It is an open question how this can be implemented in the case of decision trees.

The second contribution of this chapter is a new self-learning approach that is able to discover potentially new classes in unlabeled data. Assuming that unseen classes lie in homogeneous clusters, we employed in self-learning a density-based clustering approach that finds clusters among examples with low prediction confidence and pseudo-label them as new classes. We showed that the estimation of prediction confidence is improved by considering the semi-supervised one-versus-all random forest as the base classifier.

The approach was empirically studied for the CRISPR-Cas subtype classification task, and the experimental results demonstrate the prospects of using unlabeled data for better generalization. Indeed, in this area, one can observe a steadily growing interest in machine learning tools, which have been applied for automatically identifying CRISPR patterns (Alkhnbashi et al., 2021; Mitrofanov et al., 2021) and predicting the CRISPR-Cas system types and subtypes (Russel et al., 2020). However, all these approaches consider the classical supervised setting. Moreover, Russel et al. (2020) focused only on well represented subtypes removing those with few labeled examples. This implies that their analysis is generally limited by the extracted data set and can not be further generalized. Thus, the open-world semi-supervised classification framework unfolds new perspectives to develop systems that utilize available unlabeled data and detect new regions that have to be analyzed by experts, since new subtypes may appear there.

As a future work, we would like to conduct our experiments on the complete database. In particular, we have restricted to consider observations with only one Cas cluster in their genomes. However, it is known that some subtypes like ID are known to work with other subtypes, so they rarely appear alone in a genome. This explains why we observed few examples from ID in our training data. From the algorithmic point of view, it would be great to test the algorithm on other data sets and compare with existing approaches (Miller and Browning, 2003; Nie et al., 2010; Bendale and Boult, 2015). We have considered a rather simple clustering algorithm, but more sophisticated ones can be tried as well. Finally, another direction would be to reformulate the problem and introduce the cost of refraining from predicting one of the observed classes, which aims to penalize models that are too uncertain in their predictions. Therefore, we can extend to this case the approach proposed in Section 3.1 to automatically find the two confidence thresholds τ and θ that have been fixed in our experiments.

Chapter 4

Semi-supervised Feature Selection

In this chapter, we consider learning on labeled and unlabeled examples for feature selection. We propose a feature selection approach based on a new modification of the genetic algorithm that creates and evaluates candidate feature subsets through an evolutionary process, taking into account feature weights and eliminating irrelevant features. To increase data variety, unlabeled observations are pseudo-labeled by the multi-class self-learning algorithm (described in Chapter 3) and are employed in the feature selection process. Empirical results on different data sets show the efficiency of our framework compared to several state-of-the-art semi-supervised feature selection approaches.

4.1 Introduction

We focus on semi-supervised classification problems where observations are described by a large number of characteristics. In this case, the original set of features may contain *irrelevant* or *redundant* characteristics to the output, which with the lack of labeled information leads to inefficient learning models. In practice, the removal of such features has been shown to provide important keys for the interpretability of results and yield better prediction performance (Guyon and Elisseeff, 2003; Chandrashekar and Sahin, 2014).

Feature selection techniques have been widely developed and, depending on the availability of class labels, can be supervised, unsupervised or semi-supervised. Being agnostic to the target variable, unsupervised approaches generally ignore the discriminative power of features, so their use may lead to poor performance. In contrast, supervised feature selection algorithms benefit from abundant labeled examples, so they effectively select the subset of relevant characteristics. When there is no access to a large number of observations, performance of supervised approaches degrades, so selection of relevant features becomes an intricate issue. In semi-supervised learning (Sheikhpour et al., 2017), in addition to the scarce labeled set, a large collection of unlabeled data is assumed available, so the aim is

to perform feature selection by exploiting both available labeled and unlabeled examples in order to provide a solution that preserves important structures of data, reducing significantly the original dimension and leading to high performance in accuracy.

As searching of the optimal feature subset by exhaustive search would be computationally infeasible, many classical methods are based on sequential search algorithms, such as forward or backward selection. However, such methods are also computationally heavy for large-scale applications. Heuristic search algorithms, like the genetic algorithm (Goldberg and Holland, 1988), significantly reduce the computational time (Siedlecki and Sklansky, 1993; Xue et al., 2015). The genetic algorithm is an evolutionary optimization algorithm inspired by the natural selection process, where a *fitness function* is optimized by evolving iteratively a *population* of *candidates* (possible feature subsets). Starting from a randomly drawn population, at every *generation* a new population is produced by preserving *parents* from the last generation and creating *offspring* from the parents using operation of *crossover* and *mutation* (Figure 4-1). After a predefined number of generations the algorithm is stopped, and a candidate with the best fitness score in the last population is returned. The approach can be very effective when the number of features is very large. However, it has two main drawbacks: it may have a high variability on large-dimensional data sets; the solution that the algorithm outputs is generally not as sparse as it could be, because any information like feature importance is ignoring during the crossover.

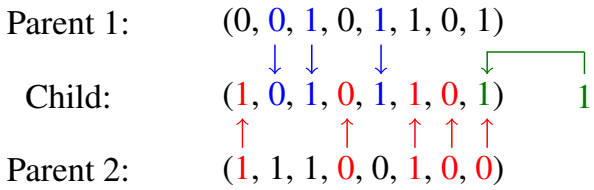


Figure 4-1: Illustration of how a new child is generated from two parents. The crossover procedure (red and blue colors) is followed by mutation (green color).

In this chapter, we propose a new semi-supervised feature selection method, denoted by `MSLA-FSGA`, that first pseudo-labels unlabeled data by the multi-class self-learning algorithm, which increases the variety of training data minimizing the number of label errors. This data is used to perform a feature subset search by a new modification of the genetic algorithm, called *Feature Selection Genetic Algorithm* (`FSGA`) that reduces the variance in selection and ensures a high level of sparsity. This is achieved by considering feature weights during the optimization phase and iteratively removing features found to be irrelevant. We guide `FSGA` by the out-of-bag score of the Random Forest classifier (Breiman, 2001), and we empirically show that the proposed approach is fast, accurate on benchmarks of large dimension, and it outperforms the supervised baseline. In addition, we show the prospects of using the C-bound with imperfect labels (CBIL) as a selection criterion.

The remainder of this chapter is organized as follows: Section 4.2 describes related work. Section 4.3 introduces the semi-supervised wrapper we propose, which is based on a modification of the genetic algorithm. Section 4.4 presents the experiments conducted on several data sets. Lastly, Section 4.5 concludes the chapter and discusses the future work.

4.2 Related Work

Feature selection methods can be classified into three main families. *Filter methods* score features following a criterion and perform selection before the construction of a learning model (Yang et al., 2010; Zhao et al., 2008). *Embedded techniques* perform model-based feature selection in order to infer the importance of features during the training process (Chen et al., 2017). Finally, *wrapper approaches* use a learner to effectively find a subset of features that are discriminatively powerful together (Kohavi and John, 1997; Ren et al., 2008). In the specific case of semi-supervised learning, some works have been initiated recently according to these three directions.

Most of the semi-supervised feature selection algorithms are extensions of popular supervised or unsupervised filters. The Semi-Fisher Score (`SFS`, Yang et al. (2010)) extends the supervised Fisher score by embedding the graph Laplacian computed on labeled and unlabeled data. The Semi-Supervised Laplacian Score (`SSLS`, Zhao et al. (2008)) is a graph-based approach that uses unlabeled examples to identify which features are able to preserve the local structure of the data, and labeled examples to maximize distance between observations from different classes. The main disadvantage of the filter approaches is that feature importance is evaluated individually, so there is a risk to discard features that are strong only in combination with others (Guyon and Elisseeff, 2003).

The Rescaled Linear Square Regression (`RLSR`, Chen et al. (2017)) is an embedded method that performs the least square regression for feature selection. It scales the regression coefficients with a set of scale factors and ranks the features using the projection matrix. As an embedded method, feature selection takes part of the training process, but, compared to wrappers, it is inflexible to the objective. In supervised feature selection, a popular embedded approach is the recurrent feature elimination (`RFE`, Guyon et al. (2002)) that recursively re-learns a learning algorithm removing a portion of features with the lowest feature weights at each iteration. Although this approach effectively removes irrelevant features, it may also remove some informative variables that are weak individually (Darst et al., 2018). In Section 4.4, we illustrate the empirical performance of an extension to semi-supervised learning.

Ren et al. (2008) proposed a semi-supervised wrapper algorithm (`CoT-FSS`), which incorporates unlabeled data to the training set by means of co-training, and find the best feature subset using forward sequential search. The approach performs

co-training inside the wrapper, which makes it computationally expensive. In Han et al. (2011), it was shown that the complexity may be reduced by pseudo-labeling the unlabeled examples just once and then performing the wrapper feature selection on the augmented the data set. As the structure of wrapper methods is more flexible, the choice of the criterion is not necessarily limited to the accuracy score, and other learning-based metrics can be used (Song et al., 2007). This is particularly attractive for semi-supervised learning where the criterion may be evaluated using both labeled and unlabeled data.

4.3 New Semi-supervised Wrapper: `MSLA-FSGA`

We consider the multi-class classification framework with the output space $\mathcal{Y} = \{1, \ldots, K\}$, $K \geq 2$, and the input space $\mathcal{X} \subset \mathbb{R}^d$, where d is the total number of features. We suppose given a set of labeled examples $Z_\mathcal{L} = \{\mathbf{x}_i, y_i\}_{i=1}^{l} \in (\mathcal{X} \times \mathcal{Y})^l$ and a set of unlabeled examples $X_\mathcal{U} = \{\mathbf{x}_i\}_{i=l+1}^{l+u} \in \mathcal{X}^u$. Given a level of sparsity $d' \ll d$, the goal is to find a feature subset $S^* \subseteq \{1, \ldots, d\}$, $|S^*| = d'$ based on $Z_\mathcal{L} \cup X_\mathcal{U}$ that leads to the highest classification performance among all possible feature subsets of size d'.

Below, we present our semi-supervised framework for wrapper feature selection using both labeled and unlabeled data. The approach consists of two phases: first, we increase variety of the training data by pseudo-labeling the unlabeled examples using the multi-class self-learning algorithm described in Section 3.1. On this augmented data set, we perform the feature selection in a wrapper fashion by a proposed genetic algorithm named Feature Selection Genetic Algorithm, described in Section 4.3.1, that uses a random forest of decision trees as a learning algorithm, which is described in Section 1.1.2.

One of the main advantages of a decision tree is that it infers feature weights during its learning process. A tree h_t outputs a weight ω_j^t for a feature j by computing the total impurity decrease the feature yields. Then, the random forest `RF` outputs feature weights by averaging them over the trees: for a feature j, $w_j = \frac{1}{T}\sum_{t=1}^{T} \omega_j^t$. As a criterion to measure the feature subset strength, we use the out-of-bag error of the random forest, which is defined for a training set $Z_\mathcal{L}$ and feature subset S as:

$$F_\mathcal{L}(S) := \frac{1}{l}\sum_{i=1}^{l} \mathbb{I}(y_i \neq \operatorname*{argmax}_{c \in \mathcal{Y}} v_\mathcal{L}(\mathbf{x}_i^{[S]}, c)), \tag{4.1}$$

where $\mathbf{x}_i^{[S]}$ denotes the projection of $\mathbf{x}_i$ on the set of features S, and the out-of-bag vote is evaluated as the proportion of trees that did not contain the example $\mathbf{x}$ in

their respective bootstrap sample:

$$v^{\mathcal{L}}_{OOB}(\mathbf{x}, c) = \frac{1}{|\{t : \mathbf{x} \notin \mathrm{B}_t\}|} \sum_{t:\ \mathbf{x} \notin \mathrm{B}_t} \mathbb{I}(h_t(\mathbf{x}) = c).$$

It is known that the out-of-bag error is an unbiased estimator of the generalization error (Breiman, 2001) being robust to perturbations in output (Dietterich, 2000).

4.3.1 FSGA: Feature Selection Genetic Algorithm

After obtaining an augmented data set via MSLA, we perform a heuristic search using a genetic algorithm (Goldberg and Holland, 1988). The classical genetic algorithm, CGA, ignores during the crossover any information like feature importance, since a child inherits features from its parents at random. Moreover, the larger is the number of features, the larger is the search space, so the algorithm becomes highly variable which affects the performance (Xue et al., 2015).

In this connection, we propose a new genetic algorithm for feature selection that tackles these two problems: 1) the algorithm takes into account the importance of features during the generation of a new population by using a weighted crossover, 2) it iteratively removes variables that are found to be irrelevant, which accelerates the convergence and reduces the search space. The main steps of this algorithm are summarized in Fig. 4-2 and are described as follows.

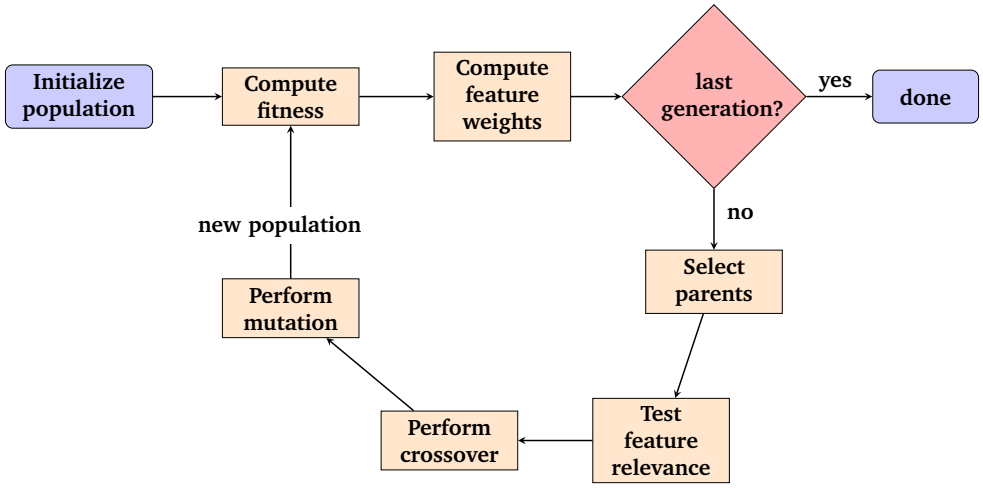


Figure 4-2: Feature Selection Genetic Algorithm (FSGA) in a nutshell.

Initialization: the population $\mathscr{P}_0$ is initialized by randomly generating feature subsets of a fixed length d'. Each candidate $S \in \mathcal{P}_0$ corresponds to a feature subset.

Evaluation of Fitness and Feature Weights: for the generation $g \geq 0$ and for each candidate $S \in \mathscr{P}_g$, a random forest model is learned on the labeled and the pseudo-labeled examples. Feature weights $\{w^S_j\}_{j \in S}$ are derived from the Random Forest classifier restricted to the feature subset S. To evaluate the strength of the

subset S, the out-of-bag score OOB is considered as a fitness score. It evaluates the generalization error without subsampling (compared to the cross-validation for example), reducing the computational cost. The introduced fitness criterion is computed on labeled and pseudo-labeled data with projection on a feature subset S as follows:

$$F_{\mathcal{L}\cup\mathcal{P}}(S) := \frac{1}{l+u_p}\sum_{i=1}^{l+u_p}\mathbb{I}(\hat{y}_i \neq \operatorname*{argmax}_{c\in\mathcal{Y}} v_{OOB}^{\mathcal{L}\cup\mathcal{P}}(\mathbf{x}_i^{[S]}, c)), \tag{4.2}$$

where $u_p \leq u$ corresponds to the number of examples that have been pseudo-labeled, $\hat{y}_i = y_i$ for $i = \{1,\ldots,l\}$, $\hat{y}_i$ corresponds to the pseudo-labels for $i = \{l+1,\ldots,l+u_p\}$.

Parent Selection: among the population $\mathscr{P}_g$, p candidates with the best fitness scores are selected, used for the next population $\mathscr{P}_{g+1}$ and for producing a new offspring. There exists various policies in the literature of genetic algorithms to select parents (Goldberg and Deb, 1991) such as tournament selection or proportionate reproduction. However, we have experimentally observed no benefit from such additional randomness, so we stick to the simplest policy of choosing candidates with the best fitness score (which corresponds to the tournament of maximal size).

Relevance Test: a test is performed to eliminate irrelevant to response variables. We embed an approach of Tuv et al. (2009) to compare variables with their copies using randomly permuted values. At first, we consider the features that appear at least in one of the candidates: $S_g = \{j \in \{1,\ldots,d\} : \exists S \in \mathscr{P}_g \text{ s.t. } j \in S\}$. We compute their average weights as:

$$\bar{w}_j = \frac{\sum_{S\,:[S\in\mathscr{P}_g]\wedge[j\in S]} w_j^S}{\sum_{S\in\mathscr{P}_g}\mathbb{I}(j\in S)}, \quad j \in S_g.$$

We define the set S_{out} of suspicious features S_{out} as a percentage[1] of features that have the smallest average weight. The principle of the test is to learn a classifier on a new data set, composed by: the features detected by the best parent, suspicious irrelevant features and their randomly permuted copies. We construct R times the data set with permuted copies and learn the classifier. For each $r \in \{1,\ldots,R\}$, we look at the feature weights of the noisy counterparts and evaluate a high percentile from their distribution, denoted by τ_r, which serves as a threshold to distinguish informative variables from noisy ones. Thus, for each feature $j \in S_{\text{out}}$ we have a sample of R feature weights retrieved from the R classifiers. This sample is compared with the sample of thresholds $(\tau_r)_{r=1}^R$ using the one-sided Mann-Whitney U test with a suitably small p-value. The hypothesis rejection for a feature j implies that its feature weight is statistically close to feature weights of the noisy counterparts, so this feature is called irrelevant, removed and will not be further considered

[1] In our experiments, we fix it to 30%.

by the algorithm. Since the size of selected parents may be less than the initial d' after this test, we perform backfilling by randomly including features into the subset to reach the size d'. We set the weights of these features to be 10^{-10} so they have little chance to participate in the next crossover.
The test requires to set such parameters as the percentile and the p-value. The higher percentile is taken, the higher thresholds $(\tau_r)_{r=1}^{R}$ are set, while high p-values suggest a more drastic assessment of irrelevance. We have noticed that the relevance test becomes more qualitative with the increase of classes[2] as in this case the difference between informative and irrelevant variables become more evident, so the test is not very sensitive to the choice of the percentile and the p-value, and the number of experiments R can be even reduced. Originally, this test has been proposed for supervised feature selection, and it was performed just once on the whole feature set (Tuv et al., 2009). In semi-supervised learning, the number of labeled examples is often much smaller than the number of features, so the features weights may be biased leading to not correct relevance estimation. Although incorporation of the pseudo-labeled unlabeled examples may help to reduce the bias, we have observed that using the relevance test iteratively at each generation improves the performance results (see more details in Section 4.4.4).

Crossover and Mutation: A new child is generated by mating two parents. In contrast to `CGA`, we inherit variables according to their weights: for each parent, its features are sorted by their weights in the decreasing order. The crossover point that characterizes the proportion of features inherited from the first parent is taken randomly, and we fill the child by its sorted features until we reach the quota. The rest of the features are taken from the second parent, ensuring no repetitions. This type of crossover suggests to increase exploitation of informative features with large feature weights.
To increase the diversity of candidates and prevent "deadlocks", mutation is used: for each child, a random number of features (not greater than a parameter $\text{mut}_{\max}$) from the subset S are replaced by the same number of features out of S. Since the proposed weighted crossover operator is highly exploitative, an aggressive mutation (i.e. large values of $\text{mut}_{\max}$) is a reasonable choice to balance exploration.

Those steps are repeated to generate new populations for several generations, and a candidate with the best fitness in the final population is output[3].

4.3.2 Time Complexity

In this section, we discuss the time complexity of our method and compare it with the other semi-supervised feature selection approaches discussed in Section 4.2. The conclusion of this section is empirically illustrated in Section 4.4.3.

Selecting $d' = \sqrt{d}$ variables, for each candidate feature subset, `FSGA` evalu-

[2]Assuming we are not dealing with a high class-imbalance.
[3]The code for our approach is available at `https://github.com/vfeofanov/TSLA-FSGA`.

ates the fitness based on the random forest classifier that has the average time complexity $O(\sqrt{d}T\tilde{n}\log^2\tilde{n})$, where $\tilde{n} \approx 0.632 \cdot (l + u_p)$ corresponds to the number of labeled and pseudo-labeled examples that are used in a bootstrap sample, T is the number of trees (Louppe, 2014). Then, the complexity of FSGA is $O(\sqrt{d}TN_g(N_c + 2R)\tilde{n}\log^2\tilde{n})$, where N_g is the number of generations, N_c is the population size, R is the number of experiments in the relevance test. In our experimental setup, we have set fixed T, N_g, N_c, R, so the complexity can be written as $O(\sqrt{d}\tilde{n}\log^2\tilde{n})$, which indicates a good scalability of the algorithm. Note also that the trees of RF as well as the learning models for fitness evaluation are naturally parallelized, which can significantly speed up the algorithm.

Compared to FSGA, the wrapper algorithms like Ren et al. (2008); Han et al. (2011) are time consuming for high-dimensional data, because they are based on sequential feature subset searching, which yields a complexity cubic in the dimension and quasilinear in the sample size [4]. The complexity of semi-supervised filter approaches like SFS (Yang et al., 2010) and SSLS (Zhao et al., 2008) are linear with respect to the dimension, but quadratic with respect to the sample size because of the Laplacian matrix's evaluation. In a large-scale setting, the complexity of RLSR (Chen et al., 2017) is high, being cubic in the dimension (or linear in the dimension and quadratic in the sample size when the sample size is large).

4.4 Experimental Results

The benefit of our approach is illustrated on a simulated data set as well as 10 publicly available data sets.

The synthetic data is generated based on the algorithm[5] that was used to create the Madelon data set (Guyon, 2003). The size of training labeled and training unlabeled sets are set respectively to 100 and 900. We fixed the number of classes to 3; the number of features to 20 wherein 8 features (No. 1-8) are informative, 6 redundant features (No. 9-14) are exact copies of the first informative variable, and 6 features (No. 15-20) are irrelevant to the target. We have observed that the first informative variable is individually strong, whereas the second one is weak, so the redundant features may be more attractive for selection algorithms rather than the second variable.

Benchmark data sets are coming from Chang and Lin (2011); Guyon (2003); Xiao et al. (2017); LeCun et al. (1998); Li et al. (2018); Dua and Graff (2017). Their characteristics are summarized in Table 4.1. The associated applications are image recognition, with Fashion, MNIST, Coil20 and Gisette databases; text classification data sets PCMAC, Relathe, Basehock; bioinformatics with Protein data set; feature selection with Madelon; and speech recognition with Isolet (we use

[4] In the case of using a decision-tree based classifier inside the wrapper.

[5] We use the implementation of Pedregosa et al. (2011).

Isolet-5). MNIST and Fashion data sets have been restricted to a subset of 10000 observations. To imitate the semi-supervised setting, we do not use the train / test splits that are proposed by data sources, but we use our own splits such that $l \ll u$.

Data set	# of lab. examples, l	# of unlab. examples, u	Dimension, d	# of classes, K
Protein	108	972	77	8
Madelon	260	2340	500	2
Isolet	156	1404	617	26
Fashion	100	9900	784	10
MNIST	100	9900	784	10
Coil20	144	1296	1024	20
PCMAC	195	1748	3289	2
Relathe	143	1284	4322	2
Basehock	200	1793	4862	2
Gisette	70	6930	5000	2

Table 4.1: Characteristics of data sets used in our experiments and ordered by dimension d.

We use the scikit-learn implementation of the random forest with 100 trees of maximal depth (Pedregosa et al., 2011). The latter is used as the majority vote classifier for MSLA.

To perform comparison with state-of-the-art approaches, for all feature selection algorithms, we first find a feature subset using a feature selection method, where the number of selected features d' is fixed to $\lfloor \sqrt{d} \rfloor$. Then, we train MSLA on the selected features and compute its performance, the classification accuracy on the unlabeled set (ACC-U).

For all experimental results, we perform 20 random labeled / unlabeled splits of the initial collection and report the average classification accuracy over the 20 trials on the unlabeled training set. We set a time limit to 1 hour per split and terminate an algorithm if the limit is exceeded. These cases are indicated as NA. All experiments were performed on a cluster with an Intel(R) Xeon(R) CPU E5-2640 v3 at 2.60GHz, 32 cores, 256GB of RAM, the Debian 4.9.110-3 x86_64 OS.

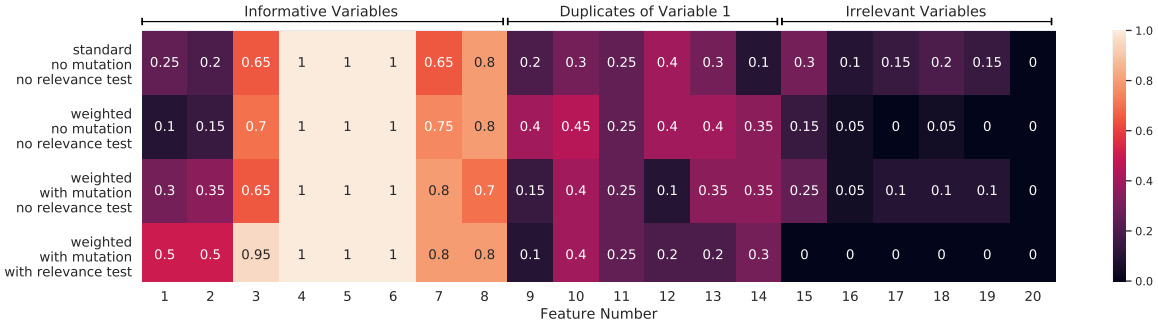
The experiments are organized as follows. First, we validate the Feature Selection Genetic Algorithm (FSGA) through an ablation study showing benefit of each step of the algorithm for finding an optimal feature subset. Then, we study how the choice of a learning model and the use of pseudo-labels have an impact on the selection criterion's strength. Finally, a comparison of the full method MSLA-FSGA with state-of-the-art methods is performed.

4.4.1 Validation of the Feature Selection Genetic Algorithm

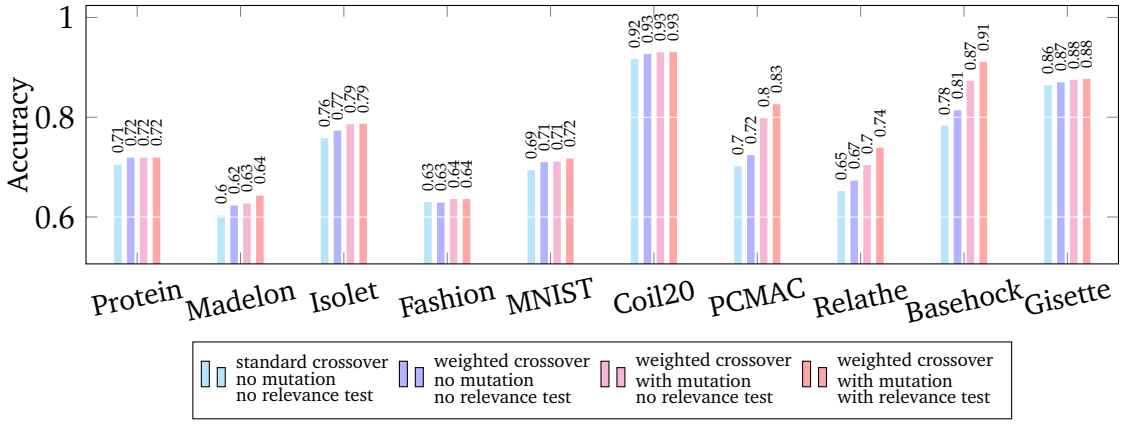
In this section, we want to demonstrate the benefit of the proposed FSGA by showing that the weighted crossover and the relevance test provides clear improvement

over the classical genetic algorithm (CGA). All versions of the genetic algorithm considered in this section use MSLA for pseudo-labeling and the pseudo-supervised out-of-bag score given in Eq. (4.2) as the fitness criterion.

In the implementation of the genetic algorithms, the number of generations is fixed to 25, the population size to 40 and the number of parents to 8. The maximum number of mutations is set to $\lfloor \sqrt{d}/2 \rfloor$. In the relevance test, we consider the 30% worst variables as suspicious, learn 10 classifiers with randomly permuted values, use the 95-th percentile to find the threshold, and set the p-value for the hypothesis test to 0.05. To highlight the benefit of the weighted crossover, we first run the genetic algorithm without mutation. Then, we refine the algorithm by successively adding mutation and relevance test, which finally leads to FSGA.



(a) Results on the synthetic data set. The features are sorted in the following order: 8 informative features, 6 redundant features, 6 irrelevant ones. On the graph, each cell represents the number of times when a feature was chosen by a feature selection method divided by the number of experiments (20).



(b) Comparison on the benchmark data sets described in Table 4.1. The accuracy on the unlabeled set (ACC-U) of a classifier trained on the final feature subset is illustrated.

Figure 4-3: Comparison of 4 different versions of the genetic algorithm: the standard crossover comparing with the weighted crossover, adding to the latter successively mutation operator and then the relevance test. Note that the proposed approach, FSGA, corresponds to the last version.

First, the algorithms are compared on the synthetic data set, and the results are illustrated in Fig. 4-3a. By looking at the irrelevant variables (15-20), we can observe that the weighted crossover is less prone to select these variables compared to the standard one. At the same time, when the weighted crossover is used alone, the subset search has little exploration concentrating on the individually strong features, as illustrated on variables 2 (weak feature) or 9 to 14 (strong features). This is solved by activating the mutation step, which helps to generate more diverse subsets. However, with mutation the output variance increases, so the irrelevant variables are again become selected more often. Then, this variance is reduced by activating the relevance test, which removes from consideration features found to be irrelevant to the target. Thus, we also reduce the search space, so the selection quality is generally improved (variables 2, 3 and 8 are selected more often, whereas variables 9-14 less often).

Then, the algorithms are compared on the benchmark data sets, and the experimental results are depicted in Fig. 4-3b. For most of data sets, more refinements yield better performance, which particularly leads to a high difference between the first and the last columns on `PCMAC`, `Relathe` and `Basehock` with an improvement of around 10%. According to the Mann and Whitney U test on level 0.01, the weighted crossover significantly outperforms the standard one on `Madelon`, `MNIST` and `Basehock` data sets. Adding the mutation step leads to a significant improvement on `PCMAC` and Basehock. Finally, the relevance test is a useful procedure to reduce the search space, and a significant improvement is observed on 3 data sets (`PCMAC, Relathe, Basehock`).

4.4.2 Improvement from Pseudo-labeling Unlabeled Data

In this section, we set the search scheme to `FSGA` and investigate how the choice of the learning algorithm and the fitness criterion impact the feature selection quality and what contribution unlabeled data can make.

At first, we evaluate two supervised baselines: when a single decision tree is used as a learning algorithm with the 5-fold cross-validation score as the fitness criterion (`Sup-Tree`); and when the random forest is learned with the out-of-bag score as the criterion (`Sup-RF`). In both cases, only the labeled examples are used for learning and fitness evaluation, and by comparing these two criteria we study whether the choice of a more sophisticated learning approach improves the results.

Then, we analyze the utility of unlabeled data for feature selection by considering three semi-supervised approaches. At the beginning, we use a self-learning algorithm to pseudo-label the unlabeled examples. Then, the labeled and the pseudo-labeled examples are used for training the genetic algorithm, where the random forest is set as a learning algorithm. To study how the quality of pseudo-labels influence the performance of the feature subset search, we compare two self-learning policies: 1) when at each step randomly picked 10% of the unlabeled examples

are added with their corresponding predictions (RSLA), 2) when we add at each step unlabeled examples with prediction vote higher than a threshold empirically learned from minimization of the transductive bound (MSLA). Note that the former policy is similar to a mechanism used for co-training in the wrapper approach of Ren et al. (2008). In the both cases, the strength of a feature subset S is evaluated by the pseudo-supervised out-of-bag score $F_{\mathcal{L}\cup\hat{\mathcal{L}}}(S)$ defined in (4.2). Finally, to see the impact of pseudo-labels on the fitness criterion alone, we introduce a third approach, where the pseudo-labels are acquired by MSLA and used inside the genetic algorithm, but the fitness score is evaluated on validation sets consisting only of the purely labeled examples. In other words, for each subset S, the learning algorithm is trained on both the labeled and the pseudo-labeled examples, but the out-of-bag score is computed on the labeled examples, i.e. we use $F_{\mathcal{L}}$ as the fitness criterion.

Data set	Sup-Tree	Sup-RF	RSLA: $F_{\mathcal{L}\cup\hat{\mathcal{L}}}$		MSLA: $F_{\mathcal{L}\cup\hat{\mathcal{L}}}$		MSLA: $F_{\mathcal{L}}$
	ACC-U	ACC-U	ACC-U	%N	ACC-U	%N	ACC-U
Protein	.707 ± .043	.719 ± .037	**.725** ± .045	25.5%	.719 ± .04	21.4%	.702 ± .04
Madelon	.606$^{\downarrow}$ ± .04	.617$^{\downarrow}$ ± .023	.632 ± .02	39.6%	.643 ± .031	42%	**.651** ± .034
Isolet	.744$^{\downarrow}$ ± .026	.751$^{\downarrow}$ ± .027	.77 ± .025	18.3%	**.787** ± .02	14.1%	.766$^{\downarrow}$ ± .022
Fashion	.604$^{\downarrow}$ ± .027	.627 ± .021	.631 ± .021	32.5%	**.636** ± .023	30.6%	.631 ± .014
MNIST	.639$^{\downarrow}$ ± .029	.676$^{\downarrow}$ ± .024	.714 ± .02	19.5%	**.717** ± .021	17.4%	.694$^{\downarrow}$ ± .021
Coil20	.903$^{\downarrow}$ ± .022	.922 ± .018	.922 ± .017	6.8%	**.931** ± .016	5.7%	.917 ± .022
PCMAC	.783$^{\downarrow}$ ± .022	.822 ± .014	.824 ± .021	17.4%	**.826** ± .019	15.9%	**.826** ± .018
Relathe	.684$^{\downarrow}$ ± .033	.735 ± .024	.708$^{\downarrow}$ ± .042	28.6%	**.739** ± .027	23.1%	.738 ± .028
Basehock	.84$^{\downarrow}$ ± .032	.902$^{\downarrow}$ ± .008	.9$^{\downarrow}$ ± .009	8.5%	**.911** ± .01	7.8%	.902 ± .018
Gisette	.849$^{\downarrow}$ ± .024	**.88** ± .01	**.88** ± .012	11.7%	.877 ± .01	12%	.874 ± .015

Table 4.2: The classification performance (accuracy) of different approaches to evaluate the fitness score in FSGA: two supervised baselines Sup-Tree and Sup-RF, three semi-supervised approaches, where MSLA and RSLA denote which self-learning policy is used for pseudo-labeling, while $F_{\mathcal{L}\cup\hat{\mathcal{L}}}$ and $F_{\mathcal{L}}$ denote which fitness criterion is taken. In addition, a % of wrong pseudo-labeled unlabeled examples (%N) is provided for the approaches that use $F_{\mathcal{L}\cup\hat{\mathcal{L}}}$ as the criterion. $^{\downarrow}$ indicates statistically significantly worse performance than the best result (shown in bold), according to the Mann-Whitney U test ($p < 0.01$).

The performance results are summarized in Table 4.2. At first, we can see that the random forest provides always more qualitative selection compared to the single tree (higher accuracy and smaller variance). Hence, the use of ensemble methods improves the selection quality, which would be connected with their robustness to overfitting. Then, we can observe that the selection becomes more qualitative when the pseudo-labeled examples are used in the algorithm, so all the three semi-supervised approaches generally outperform its supervised baseline Sup-RF. MSLA with $F_{\mathcal{L}\cup\hat{\mathcal{L}}}$ as the criterion benefits the most from unlabeled data, and it significantly

outperforms Sup-RF on 4 data sets, RSLA on Relathe and Basehock, MSLA with $F_{\mathcal{L}}$ on Isolet and MNIST.

By comparing RSLA and MSLA, we can see that more careful pseudo-labeling based on the transductive guarantees leads to the highest performance, and the larger portions of noisy pseudo-labels generally lead to worse results. However, both for RSLA and MSLA, the performance is not degraded with respect to the baseline Sup-RF on most of data sets, which validates our choice of the out-of-bag score inside the criterion. Thus, we conclude that the selection becomes more qualitative when unlabeled data are explored with the self-learning.

When we compare the two criteria based on MSLA, $F_{\mathcal{L}\cup\hat{\mathcal{L}}}$ and $F_{\mathcal{L}}$, we infer that the use of pseudo-labeled data for evaluation of feature strength is actually helpful. This may be connected with the fact that the few labeled examples bias the fitness score, and trusting pseudo-labels would give more benefit than harm. This is coherent with a general observation that the traditional supervised model selection based on validation is not effective in the semi-supervised setting (Madani et al., 2005). Note that MSLA with $F_{\mathcal{L}}$ nevertheless outperforms the baseline Sup-RF, since the pseudo-labels are still used to compute the feature weights.

Using (CBIL) as a Fitness Criterion

In Section 2.4, we proposed the C-bound with imperfect labels, which can be a convenient model selection criterion, since its minimization implies simultaneously maximization of the margin mean and minimization of the margin variance. In addition, (CBIL) penalizes the margin of every example depending on the predicted class and the corresponding mislabeling probabilities. Hence, (CBIL) can be used as an alternative fitness criterion, which is aware of possible label errors. To illustrate the prospects of using (CBIL), we employ it to MSLA-FSGA as a fitness criterion and compare with the pseudo-supervised out-of-bag score ($F_{\mathcal{L}\cup\hat{\mathcal{L}}}$). For (CBIL), we compute the true mislabeling matrix, i.e., it is evaluated as if the labels for the pseudo-labeled examples would be known.

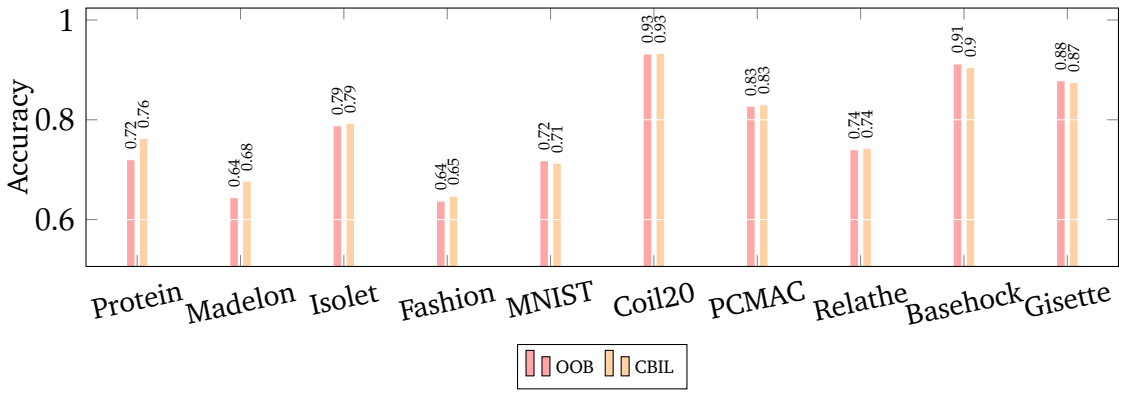


Figure 4-4: Comparison between the out-of-bag score and (CBIL) used as selection criteria on the benchmark data sets.

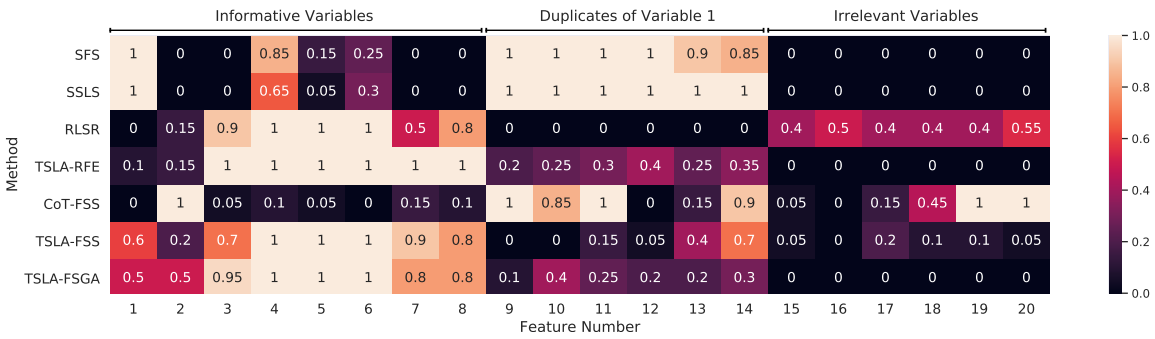
Figure 4-4 illustrates the experimental results. As it can be seen, on Protein and Madelon (CBIL) significantly outperforms the out-of-bag score (around 4% in both cases), which indicates that estimation of the class-conditional mislabeling matrix is a promising research direction. On the other data sets, (CBIL) is either insignificantly better or insignificantly worse, which means that the assumption of instance-independent mislabeling does not hold on some data sets.

4.4.3 Comparison with the State-of-the-Art

Finally, we validate the proposed approach referred as MSLA-FSGA, by comparing its performance with the state-of-the-art. It is compared with RLSR, SFS, SSLS and CoT-FSS (with decision tree as the learning algorithm inside the wrapper, the co-training for pseudo-labeling and the 5-fold cross-validation score as the selection criterion), all introduced in Section 4.2. In order to additionally validate FSGA, we also compare with two more selection algorithms, for which the unlabeled examples are pseudo-labeled before the feature selection step using MSLA for a fair comparison. Then, the first approach (denoted by MSLA-FSS), similarly to Han et al. (2011), performs the forward sequential search by minimizing the pseudo-supervised out-of-bag score (Eq. (4.2)), whereas the second approach (denoted by MSLA-RFE) applies the recursive feature elimination based on the random forest.

Due to the lack of labeled training examples, the hyperparameters of all methods are set to their default values. Namely, for RLSR, the regularization parameter γ is set to 0.1; for SSLS and SFS, the number of nearest neighbors is set to 20, and the bandwidth for constructing the graph Laplacian is determined using the median distance heuristic (Schölkopf, 1997). For MSLA-FSS, at each step we add 10% best features into the model, and for MSLA-RFE, at each step we remove 10% worst ones.

First, we compare the considered approaches on the synthetic data set. In Fig. 4-5a the feature selection results averaged over 20 trials are reported. One can observe that the filter approaches, SFS and SSLS, perfectly detect and discard the irrelevant features. However, since the importance of features is evaluated independently, only individually strong informative variables are selected. Thus, some informative features are rarely or never selected (2-3, 5-8), and redundant features (9-14), which bring no new information, are preferred. In contrast, RLSR is able to find redundancies but does not succeed to eliminate the irrelevant variables (15-20). From the results, CoT-FSS has the most difficulties to select relevant variables without a clear selection pattern. All MSLA-RFE, MSLA-FSS and MSLA-FSGA perform quite well. FSS is less effective in eliminating the irrelevant variables than RFE and FSGA that discard them as perfect as the filter methods. Although RFE is slightly better in selecting variables 7 and 8, FSGA outperforms it in selecting variable 2. As it was mentioned before, variable 2 is an individually weak variable, so RFE often discards it preferring to keep one of the duplicates of variable 1. In turn, FSGA evaluates features *jointly* focusing explicitly on how the selected features are combined.



(a) Results on the synthetic data set. The features are sorted in the following order: 8 informative features, 6 redundant features, 6 irrelevant ones. On the graph, each cell represents the number of times when a feature was chosen by a feature selection method divided by the number of experiments (20).

(b) Performance results on benchmark data sets. The classification accuracy is computed on the unlabeled set. $^{\downarrow}$ indicates statistically significantly worse performance than the best result (shown in bold), according to the Mann-Whitney U test ($p < 0.01$).

Data set	Filters		Embedded		Wrappers		
	SFS	SSLS	RLSR	MSLA-RFE	CoT-FSS	MSLA-FSS	MSLA-FSGA
Protein	.676$^{\downarrow}$ ± .035	.634$^{\downarrow}$ ± .037	.695 ± .034	**.719** ± .039	.583$^{\downarrow}$ ± .09	**.719** ± .033	**.719** ± .04
Madelon	.589$^{\downarrow}$ ± .035	.526$^{\downarrow}$ ± .038	.516$^{\downarrow}$ ± .015	**.658** ± .027	.514$^{\downarrow}$ ± .032	.649 ± .032	.643 ± .031
Isolet	.56$^{\downarrow}$ ± .037	.549$^{\downarrow}$ ± .03	.638$^{\downarrow}$ ± .05	.755$^{\downarrow}$ ± .03	.438$^{\downarrow}$ ± .058	.656$^{\downarrow}$ ± .053	**.787** ± .02
Fashion	.348$^{\downarrow}$ ± .03	.35$^{\downarrow}$ ± .033	.419$^{\downarrow}$ ± .048	.615$^{\downarrow}$ ± .025	.462$^{\downarrow}$ ± .043	.444$^{\downarrow}$ ± .041	**.636** ± .023
MNIST	.112$^{\downarrow}$ ± .0	.176$^{\downarrow}$ ± .029	.129$^{\downarrow}$ ± .013	.671$^{\downarrow}$ ± .029	.394$^{\downarrow}$ ± .068	.514$^{\downarrow}$ ± .027	**.717** ± .021
Coil20	.748$^{\downarrow}$ ± .046	.743$^{\downarrow}$ ± .045	.858$^{\downarrow}$ ± .029	.902$^{\downarrow}$ ± .02	.817$^{\downarrow}$ ± .038	.817$^{\downarrow}$ ± .025	**.931** ± .016
PCMAC	.613$^{\downarrow}$ ± .056	.545$^{\downarrow}$ ± .027	.773$^{\downarrow}$ ± .033	**.832** ± .019	NA	.813 ± .042	.826 ± .019
Relathe	.613$^{\downarrow}$ ± .025	.584$^{\downarrow}$ ± .017	.719 ± .034	**.746** ± .028	NA	.694 ± .032	.739 ± .027
Basehock	.733$^{\downarrow}$ ± .051	.582$^{\downarrow}$ ± .081	.875$^{\downarrow}$ ± .02	**.913** ± .008	NA	NA	.911 ± .01
Gisette	.851$^{\downarrow}$ ± .019	.521$^{\downarrow}$ ± .01	.51$^{\downarrow}$ ± .01	.871 ± .017	NA	NA	**.877** ± .01

Figure 4-5: Comparison of the state-of-the-art methods with our method to select relevant features in semi-supervised learning.

Figure 4-5b summarizes the performance results on the 10 benchmark data sets. On 4 data sets, Isolet, Fashion, MNIST and Coil20, our approach significantly outperforms all the other methods, while it is never significantly worse in cases when MSLA-FSGA is not the best. Compared to our approach, performances of other wrapper-based methods (CoT-FSS and MSLA-FSS) are significantly worst in most of the cases, which indicates the superiority of a genetic algorithm over a sequential search as the search scheme. We can also see that the performance of RLSR fluctuates from one data set to another. This could be connected with its sensitivity to the value of its regularization parameter γ, which is difficult to tune with few labeled examples. The filter methods, SFS and SSLS, are significantly worst in all situations. One can conclude that the filters are more suitable as a pre-processing step rather

Data set	SFS	SSLS	RLSR	MSLA-RFE	CoT-FSS	MSLA-FSS	MSLA-FSGA
Protein	1 s	1 s	10 s	9 s	22 s	26 s	2 m
Madelon	15 s	14 s	1 m	9 s	5 m	4 m	4 m
Isolet	5 s	1 s	2 m	13 s	9 m	6 m	4 m
Fashion	4 m	4 m	3 m	24 s	16 m	13 m	7 m
MNIST	5 m	4 m	3 m	24 s	18 m	14 m	7 m
Coil20	8 s	1 s	3 m	13 s	16 m	7 m	3 m
PCMAC	45 s	31 s	23 m	13 s	>1 h	26 m	3 m
Relathe	29 s	22 s	38 m	13 s	>1 h	29 m	4 m
Basehock	1 m	51 s	44 m	16 s	>1 h	>1 h	4 m
Gisette	17 m	17 m	21 m	40 s	>1 h	>1 h	6 m

Table 4.3: The average run-time of the feature selection algorithms under consideration on the benchmark data sets. *s* stands for seconds, *m* for minutes and *h* for hours.

than as a complete feature selection process.

Finally, the recurrent feature elimination, MSLA-RFE, has the best performance on the data sets with many irrelevant features (Madelon, PCMAC, Relathe, Basehock). However, when there are plenty of different informative features (Isolet, Fashion, MNIST, Coil20), RFE tends to underselect some weak but important for classification variables, so it becomes significantly worse than MSLA-FSGA. In addition, all these data sets are multi-class, which may arise additional difficulties for the feature selection task. In contrast, MSLA-FSGA successfully outputs feature subsets that are sparse and highly discriminative at the same time. Also, with the help of the relevance test, our approach detects and explicitly eliminates irrelevant variables, so the algorithm performs very well on Madelon, PCMAC, Relathe and Basehock as well.

Run-time

We present in Table 4.3 the run-time of all the algorithms, to illustrate theoretical complexities introduced in Section 4.3.2. A particular attention should be taken on MNIST, Fashion, Gisette with respect to the large sample size as well as on Relathe, Basehock, Gisette with respect to the large dimension. Although the genetic algorithm FSGA is slower than RFE, it still passes the scale well both with respect to the sample size and the dimension. Being very fast on small data sets, the filter methods SFS and SSLS significantly slow down with the increase of sample size. In turn, when the dimension is large, RLSR becomes expensive too. On large data sets, CoT-FSS and MSLA-FSS are computationally infeasible. In general, the results clearly illustrate the complexity discussion standing in Section 4.2.

4.4.4 Additional Ablative Study

In Section 4.3.1, we proposed to use the relevance test at every generation by testing just a portion of features, whereas originally the test was proposed to be performed once on the whole feature set (Tuv et al., 2009). To validate our choice, we compare two versions of the genetic algorithm: 1) at first, the relevance test is performed on the whole feature set, features found to be irrelevant are removed, and then FSGA is run without the relevance test step; 2) relevance test is used at every generation of FSGA as described in Section 4.3.1.

The performance results on the benchmarks are illustrated in Figure 4-6. As one can see, when the relevance test is iteratively used, the classification accuracy is noticeably higher on most of data sets, particularly on those with a high number of irrelevant features (Madelon, PCMAC, Relathe, Basehock). This suggests that the model learned on the whole feature set suffers from the curse of dimensionality, so some irrelevant variables are not detected based on derived feature weights.

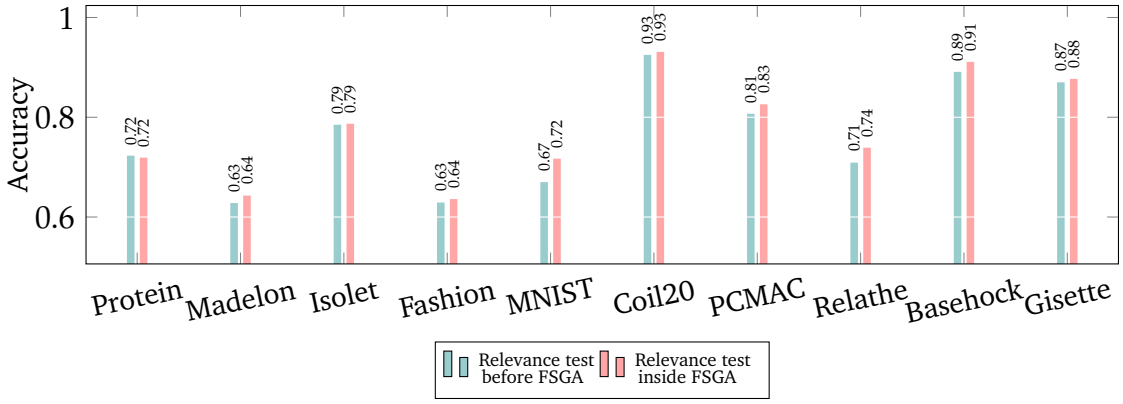


Figure 4-6: Comparison of 2 different versions of FSGA on the benchmark data sets: the relevance test performed just once before the genetic algorithm compared with performing the relevance test at every generation as it is described in Section 4.3.1. The accuracy on the unlabeled set (ACC-U) of a classifier trained on the final feature subset is illustrated.

In addition, we show that the relevance test also significantly improves the standard crossover, which is illustrated in Figure 4-7. Nevertheless, the proposed weighted crossover generally outperforms the standard one, which additionally validates our contribution.

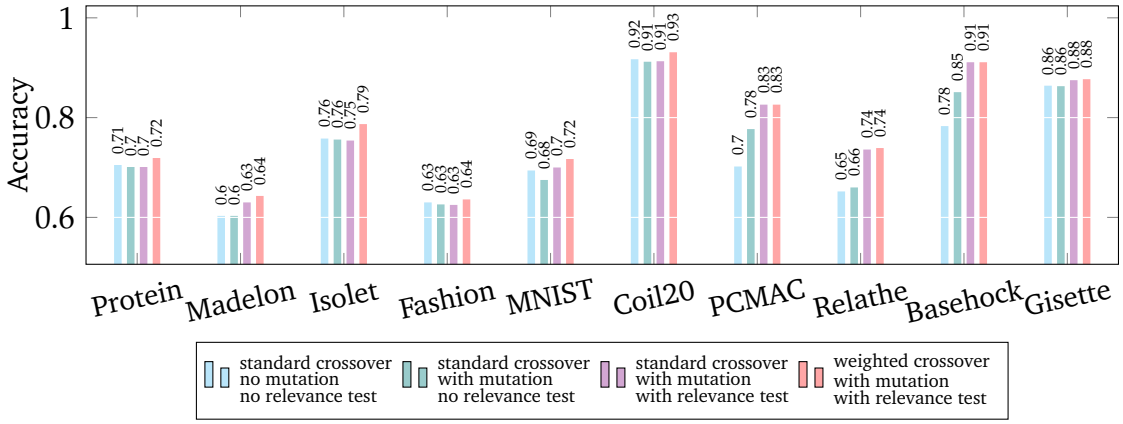


Figure 4-7: Comparison of 4 different versions of the genetic algorithm on the benchmark data sets: the standard crossover, to which the mutation operator and the relevance test are successively added, and the FSGA. Note that the second column corresponds to the classical genetic algorithm (CGA). The accuracy on the unlabeled set (ACC-U) of a classifier trained on the final feature subset is illustrated.

4.5 Conclusion and Perspectives

In this chapter, we proposed a new framework for semi-supervised wrapper feature selection. To increase the diversity of labeled data, unlabeled examples are pseudo-labeled using a self-learning algorithm. To produce a sparse solution, we proposed a modification of the genetic algorithm by taking into account feature weights during its evolutionary process and eliminating variables irrelevant to the target. The proposed model has been empirically validated through an ablative study and a comparison with several feature selection approaches.

As a future work, it would be interesting to detect automatically the level of sparsity, set to $\lfloor \sqrt{d} \rfloor$ in our paper. In this case, simple criteria like the out-of-bag score trivially lead to the situation when a large number of selected features is chosen. One solution would be to add the regularization term to impose the sparsity level, for example, as proposed by Frohlich et al. (2003); Da Silva et al. (2011).

Another direction would be to improve evaluation of the feature strength by developing a fitness criterion aware of possible errors in pseudo-labels. In Section 4.4.2, we showed that consideration of a mislabeling error model can be a good approach. Alternatively, this can be achieved by introducing an additional assumption to determine data regions of low confidence, where the pseudo-labels are prone to error.

Conclusion and Perspectives

This dissertation has studied the problem of semi-supervised classification, a situation where a large amount of data is naturally unlabeled, while labeled data are scarce. Our theoretical studies have focused on majority vote classifiers, while algorithmic designs have been proposed based on the self-learning paradigm. We have also made algorithmic contributions to semi-supervised feature selection and have conducted applied research on the CRISPR-Cas subtype prediction with the proposed new subtype discovery tool.

Summary of Contributions

In Chapter 2, we derived guarantees on the error of the majority vote classifier in the multi-class case. First, we proposed a transductive bound for the majority vote classifier by taking into account the class vote distribution and considering the class confusion matrix as an error measure. We proved that the bound is tight when the majority vote classifier makes its errors primarily on examples with a low prediction vote. Then, we derived a generalization bound for the majority vote classifier in the presence of imperfect labels. We explicitly modeled possible mislabeling and established the link between the true label and the imperfect label in misclassifying a particular example. Based on this, we extended the supervised C-bound to the case of imperfect labels and analyzed its estimation from data via the PAC-Bayesian theorem. The two proposed bounds give us insights for theoretical analysis of the error of the self-learning algorithm.

In Chapter 3, we proposed a new multi-class self-learning algorithm, where the confidence threshold is automatically found based on minimization of the transductive bound proposed in Chapter 2. The experimental results demonstrated the prospects of the proposed approach compared with other self-learning policies and semi-supervised approaches. Then, we applied self-learning to the CRISPR-Cas classification task, where the particular challenge is that new classes may be present in unlabeled data. Assuming that unseen classes are located in homogeneous regions, we employed a density-based clustering approach that finds clusters among examples with low prediction confidence in order to pseudo-label them as new classes. We empirically showed on a data set of CRISPR-Cas subtypes that we are able to detect subtypes that were completely unlabeled during the training phase.

In Chapter 4, we studied feature selection in the case of partially-labeled data. We proposed a new modification of the genetic algorithm that takes into account feature weights and eliminates irrelevant features during its evolutionary process. We applied the multi-class self-learning algorithm proposed in Chapter 3 in order to increase diversity of data used for the strength evaluation of candidate feature subsets. We performed an extensive ablative study and showed that the effectiveness of our approach compared to several semi-supervised feature selection approaches.

Future Work

The transductive bound proposed in Section 2.3 is based on how the hypotheses predictions agree on each unlabeled example, but it does not take into account how the hypotheses predictions correlate with each other on average over all the unlabeled examples. Incorporation of information about correlations could allow us to use this bound not only for selecting the threshold, but also for the posterior optimization over the hypothesis space. Another way for optimizing the posterior in the semi-supervised setting is to minimize the proposed C-bound with imperfect labels. This, however, requires correct estimation of the mislabeling matrix in the semi-supervised case, which can be another direction for future work. As we pointed out in Section 2.5, this can be achieved by introducing additional assumptions on a link between the labeled and unlabeled data to identify regions of unlabeled data prone to mislabeling. As estimation of mislabeling for each example would be a very challenging task, considering the class-related mislabeling model appears more reasonable. In Section 4.4.2, we showed that incorporating class-related mislabeling errors into the selection criterion can be a perspective approach to improve the feature selection quality.

In Section 3.1.6, we noted that the confirmation bias is a significant limitation, which naturally underlies the self-learning algorithm. Nevertheless, we concluded that a reliable estimation of the C-bound with imperfect labels (CBIL) can be used to evaluate the error of the self-learning algorithm, which can be important for model selection tasks. Another consequence of the confirmation bias is that the final model is poorly calibrated, i.e., it may output high prediction confidence for misclassified examples. Although some methods have already been proposed to improve the confidence assessment (Zou et al., 2019), to the best of our knowledge, there is no comprehensive review and analysis of this problem in the general case.

Another area of future work could be the continuation of work on the problem of open-world semi-supervised classification, where some classes may be completely unlabeled. Our motivation stems from the fact that this problem is inherent in biological applications like the one presented in Section 3.2, while there is little research in machine learning related to the open-world framework (Miller and Browning, 2003; Nie et al., 2010; Bendale and Boult, 2015).

Finally, as it was pointed out in Section 4.5, a great direction for future work

is to extend the proposed feature selection genetic algorithm to the case when the number of selected features is determined automatically, since for many practical tasks it is not clear what level of sparsity should be set. For classical wrappers approaches, this problem is difficult as the search space grows exponentially with the dimension. Nevertheless, this is made possible by the iterative elimination of irrelevant features proposed in Section 4.3, which allows us to significantly reduce the search space.

List of Figures

1-1 Illustration of three assumptions typically made in semi-supervised learning 16
1-2 Self-Learning Algorithm SLA. 17

2-1 (CBIL) and Oracle C-Bound when varying the number of pseudo-labels. 42
2-2 The value of (CBIL) with different λ. 43

3-1 Classification accuracy of MSLA with respect to the proportion of unlabeled examples. 62
3-2 Illustration of the confirmation bias on Vowel data set. 65
3-3 Distribution of the unlabeled observations from the subtype IA across different clusters. 74
3-4 Distribution of the unlabeled observations from the subtype ID across different clusters. 74
3-5 Distribution of the unlabeled observations from the subtype IIB across different clusters. 76

4-1 Illustration of how a new child is generated from two parents. 80
4-2 Feature Selection Genetic Algorithm (FSGA) in a nutshell. 83
4-3 Comparison of 4 different versions of the genetic algorithm. 88
4-4 Comparison between the out-of-bag score and (CBIL) used as selection criteria 91
4-5 Comparison of the state-of-the-art methods with our method to select relevant features in semi-supervised learning. 93
4-6 Comparison of 2 different versions of FSGA with the relevance test. . 95
4-7 Comparison of FSGA with an improved CGA. 96

List of Tables

2.1 Characteristics of data sets used in our experiments to illustrate (CBIL). 42

3.1 Characteristics of data sets used in our experiments to validate MSLA. 58
3.2 Classification performance of MSLA on different data sets. 60
3.3 The average run-time of MSLA. 63
3.4 The performance of MSLA depending on how the posterior probabilities are estimated. 64
3.5 The distribution of subtypes of CRISPR-Cas Type I and Type II across labeled and unlabeled sets used in experiments. 72
3.6 Proportions of unlabeled examples categorized to one of the observed classes, to one of new clusters, to no cluster for the Type I. . . 73
3.7 The performance results for the Type I experiment. 73
3.8 Proportions of unlabeled examples categorized to one of the observed classes, to one of new clusters, to no cluster for the Type II. . . 75
3.9 The performance results for the Type II experiment. 75

4.1 Characteristics of data sets used for feature selection experiments. . . 87
4.2 The classification performance of different approaches to evaluate the fitness score in FSGA. 90
4.3 The average run-time of the feature selection algorithms under consideration. 94

Bibliography

Alkhnbashi, O. S., Mitrofanov, A., Bonidia, R., Raden, M., Tran, V. D., Eggenhofer, F., Shah, S. A., Öztürk, E., Padilha, V. A., Sanches, D. S., et al. (2021). Crispr-loci: comprehensive and accurate annotation of crispr–cas systems. *Nucleic Acids Research*.

Amini, M., Laviolette, F., and Usunier, N. (2008). A transductive bound for the voted classifier with an application to semi-supervised learning. In *Advances in Neural Information Processing Systems*, pages 65–72.

Amini, M. and Usunier, N. (2015). *Learning with Partially Labeled and Interdependent Data*. Springer.

Amini, M.-R. and Gallinari, P. (2003). Semi-supervised learning with explicit misclassification modeling. In Gottlob, G. and Walsh, T., editors, *IJCAI-03, Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, Acapulco, Mexico, August 9-15, 2003*, pages 555–560. Morgan Kaufmann.

Bauvin, B., Capponi, C., Roy, J.-F., and Laviolette, F. (2020). Fast greedy c-bound minimization with guarantees. *Machine Learning*, 109(9):1945–1986.

Bégin, L., Germain, P., Laviolette, F., and Roy, J.-F. (2014). PAC-Bayesian Theory for Transductive Learning. In Kaski, S. and Corander, J., editors, *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics*, volume 33 of *Proceedings of Machine Learning Research*, pages 105–113, Reykjavik, Iceland. PMLR.

Belkin, M. and Niyogi, P. (2004). Semi-supervised learning on riemannian manifolds. *Machine Learning*, 56(1-3):209–239.

Bendale, A. and Boult, T. (2015). Towards open world recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1893–1902.

Bishop, C. M. (2006). Pattern recognition. *Machine learning*, 128(9).

Blum, A. and Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory (COLT)*, pages 92–100.

Boucheron, S., Lugosi, G., and Massart, P. (2013). *Concentration Inequalities - A Nonasymptotic Theory of Independence*. Oxford University Press.

Breiman, L. (1996). Bagging predictors. *Machine learning*, 24(2):123–140.

Breiman, L. (2001). Random forests. *Machine Learning*, 45(1):5–32.

Breiman, L., Friedman, J., Stone, C. J., and Olshen, R. A. (1984). *Classification and regression trees*. CRC press.

Brown, G. and Wyatt, J. L. (2003). The use of the ambiguity decomposition in neural network ensemble learning methods. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 67–74.

Cai, D., He, X., and Han, J. (2008). Training linear discriminant analysis in linear time. In *2008 IEEE 24th International Conference on Data Engineering*, pages 209–217. IEEE.

Cascante-Bonilla, P., Tan, F., Qi, Y., and Ordonez, V. (2020). Curriculum labeling: Revisiting pseudo-labeling for semi-supervised learning.

Catoni, O. (2007). Pac-bayesian supervised classification: the thermodynamics of statistical learning. *arXiv preprint arXiv:0712.0248*.

Chandrashekar, G. and Sahin, F. (2014). A survey on feature selection methods. *Computers & Electrical Engineering*, 40(1):16–28.

Chang, C.-C. and Lin, C.-J. (2011). LIBSVM: A Library for Support Vector Machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27:1–27:27.

Chapelle, O., Schölkopf, B., and Zien, A. (2010). *Semi-Supervised Learning*. The MIT Press, 1st edition.

Chapelle, O. and Zien, A. (2005). Semi-supervised classification by low density separation. In *AISTATS*, volume 2005, pages 57–64. Citeseer.

Chen, X., Yuan, G., Nie, F., and Huang, J. Z. (2017). Semi-supervised feature selection via rescaled linear regression. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, volume 2017, pages 1525–1531.

Chittineni, C. (1980). Learning with imperfectly labeled patterns. *Pattern Recognition*, 12(5):281–291.

Cortes, C., Mohri, M., Riley, M., and Rostamizadeh, A. (2008). Sample selection bias correction theory. In *International conference on algorithmic learning theory*, pages 38–53. Springer.

Couvin, D., Bernheim, A., Toffano-Nioche, C., Touchon, M., Michalik, J., Néron, B., Rocha, E. P., Vergnaud, G., Gautheret, D., and Pourcel, C. (2018). Crisprcasfinder, an update of crisrfinder, includes a portable version, enhanced performance and integrates search for cas proteins. *Nucleic acids research*, 46(W1):W246–W251.

Cozman, F. G., Cohen, I., and Cirelo, M. (2002). Unlabeled data can degrade classification performance of generative classifiers. In *Flairs conference*, pages 327–331.

Da Silva, S. F., Ribeiro, M. X., Neto, J. d. E. B., Traina-Jr, C., and Traina, A. J. (2011). Improving the ranking quality of medical image retrieval using a genetic feature selection method. *Decision support systems*, 51(4):810–820.

d'Alché-Buc, F., Grandvalet, Y., and Ambroise, C. (2001). Semi-supervised marginboost. *Advances in Neural Information Processing Systems*, 14:553–560.

Darst, B. F., Malecki, K. C., and Engelman, C. D. (2018). Using recursive feature elimination in random forest to account for correlated variables in high dimensional data. *BMC genetics*, 19(1):1–6.

Derbeko, P., El-Yaniv, R., and Meir, R. (2004). Explicit learning curves for transduction and application to clustering and compression algorithms. *Journal of Artificial Intelligence Research*, 22(1):117–142.

Dietterich, T. G. (2000). An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine learning*, 40(2):139–157.

Donsker, M. D. and Varadhan, S. R. S. (1975). Asymptotic evaluation of certain markov process expectations for large time, i. *Communications on Pure and Applied Mathematics*, 28(1):1–47.

Dua, D. and Graff, C. (2017). UCI machine learning repository.

Efron, B. (1992). Bootstrap methods: another look at the jackknife. In *Breakthroughs in statistics*, pages 569–593. Springer.

El-Yaniv, R. and Pechyony, D. (2006). Stable transductive learning. In *International Conference on Computational Learning Theory*, pages 35–49. Springer.

El-Yaniv, R. and Pechyony, D. (2009). Transductive rademacher complexity and its applications. *Journal of Artificial Intelligence Research*, 35:193–234.

Ester, M., Kriegel, H.-P., Sander, J., Xu, X., et al. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231.

Feofanov, V., Devijver, E., and Amini, M.-R. (2019). Transductive bounds for the multi-class majority vote classifier. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):3566–3573.

Feofanov, V., Devijver, E., and Amini, M.-R. (2021a). Multi-class probabilistic bounds for self-learning. *Journal submission (under review), arXiv preprint arXiv:2109.14422*.

Feofanov, V., Devijver, E., and Amini, M.-R. (2021b). Wrapper feature selection with partially labeled data. *Journal submission (under review), arXiv preprint arXiv:1911.04841*.

Feofanov, V., Gallopin, M., Devijver, E., Amini, M.-R., Charbit, P.-A., and Pourcel, C. (2021c). Crispr subtype prediction using open-world semi-supervised learning (in preparation for publication).

Freund, Y. and Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139.

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232.

Frohlich, H., Chapelle, O., and Scholkopf, B. (2003). Feature selection for support vector machines by means of genetic algorithm. In *Proceedings. 15th IEEE International Conference on Tools with Artificial Intelligence*, pages 142–148. IEEE.

Gebel, M. (2009). *Multivariate calibration of classifier scores into the probability space*. PhD thesis, The University of Dortmund.

Germain, P., Lacasse, A., Laviolette, F., March, M., and Roy, J.-F. (2015). Risk bounds for the majority vote: From a pac-bayesian analysis to a learning algorithm. *Journal of Machine Learning Research*, 16(26):787–860.

Germain, P., Lacasse, A., Laviolette, F., and Marchand, M. (2009). Pac-bayesian learning of linear classifiers. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 353–360.

Gieseke, F., Airola, A., Pahikkala, T., and Kramer, O. (2014). Fast and simple gradient-based optimization for semi-supervised support vector machines. *Neurocomputing*, 123:23–32.

Goldberg, D. E. and Deb, K. (1991). A comparative analysis of selection schemes used in genetic algorithms. In *Foundations of genetic algorithms*, volume 1, pages 69–93. Elsevier.

Goldberg, D. E. and Holland, J. H. (1988). Genetic algorithms and machine learning. *Machine learning*, 3(2):95–99.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. (2017). On calibration of modern neural networks. In *International Conference on Machine Learning*, pages 1321–1330. PMLR.

Guyon, I. (2003). Design of experiments of the nips 2003 variable selection benchmark. In *NIPS 2003 workshop on feature extraction and feature selection*.

Guyon, I. and Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of machine learning research*, 3(Mar):1157–1182.

Guyon, I., Weston, J., Barnhill, S., and Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. *Machine learning*, 46(1):389–422.

Han, Y., Park, K., and Lee, Y.-K. (2011). Confident wrapper-type semi-supervised feature selection using an ensemble classifier. In *2011 2nd International Conference on Artificial Intelligence, Management Science and Electronic Commerce (AIMSEC)*, pages 4581–4586. IEEE.

Hansen, L. K. and Salamon, P. (1990). Neural network ensembles. *IEEE transactions on pattern analysis and machine intelligence*, 12(10):993–1001.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media.

Hille, F., Richter, H., Wong, S. P., Bratovič, M., Ressel, S., and Charpentier, E. (2018). The biology of crispr-cas: backward and forward. *Cell*, 172(6):1239–1259.

Hofacker, I. L. (2003). Vienna rna secondary structure server. *Nucleic acids research*, 31(13):3429–3431.

Joachims, T. (1999). Transductive inference for text classification using support vector machines. In *Proceedings of the Sixteenth International Conference on Machine Learning*, ICML '99, pages 200–209, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Kim, K. (2016). A hybrid classification algorithm by subspace partitioning through semi-supervised decision tree. *Pattern Recognition*, 60:157–163.

Kohavi, R. and John, G. H. (1997). Wrappers for feature subset selection. *Artificial intelligence*, 97(1-2):273–324.

Kohavi, R., Wolpert, D. H., et al. (1996). Bias plus variance decomposition for zero-one loss functions. In *ICML*, volume 96, pages 275–83.

Krithara, A., Amini, M., Renders, J., and Goutte, C. (2008). Semi-supervised document classification with a mislabeling error model. In *Advances in Information Retrieval , 30th European Conference on IR Research, ECIR 2008, Glasgow, UK, March 30-April 3, 2008. Proceedings*, pages 370–381.

Krogh, A. and Vedelsby, J. (1995). Neural network ensembles, cross validation, and active learning. In *Advances in neural information processing systems*, pages 231–238.

Kuncheva, L. I. (2014). *Combining pattern classifiers: methods and algorithms*. John Wiley & Sons.

Lacasse, A., Laviolette, F., Marchand, M., Germain, P., and Usunier, N. (2007). Pac-bayes bounds for the risk of the majority vote and the variance of the gibbs classifier. In *Advances in Neural information processing systems*, pages 769–776.

Langford, J. (2005). Tutorial on practical prediction theory for classification. *J. Mach. Learn. Res.*, 6:273–306.

Langford, J. and Shawe-Taylor, J. (2002). PAC-Bayes & margins. In *Proceedings of the 15th International Conference on Neural Information Processing Systems*, NIPS'02, pages 439–446, Cambridge, MA, USA. MIT Press.

Laviolette, F., Morvant, E., Ralaivola, L., and Roy, J.-F. (2017). Risk upper bounds for general ensemble methods with an application to multiclass classification. *Neurocomputing*, 219:15–25.

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.

Leistner, C., Saffari, A., Santner, J., and Bischof, H. (2009). Semi-supervised random forests. In *2009 IEEE 12th international conference on computer vision*, pages 506–513. IEEE.

Letarte, G., Germain, P., Guedj, B., and Laviolette, F. (2019). Dichotomize and generalize: Pac-bayesian binary activated deep neural networks. In *Advances in Neural Information Processing Systems*, pages 6872–6882.

Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. P., Tang, J., and Liu, H. (2018). Feature selection: A data perspective. *ACM Computing Surveys (CSUR)*, 50(6):94.

Liu, Y. and Yao, X. (1999). Ensemble learning via negative correlation. *Neural networks*, 12(10):1399–1404.

Loog, M. (2015). Contrastive pessimistic likelihood estimation for semi-supervised classification. *IEEE transactions on pattern analysis and machine intelligence*, 38(3):462–475.

Lorenzen, S. S., Igel, C., and Seldin, Y. (2019). On pac-bayesian bounds for random forests. *Machine Learning*, 108(8-9):1503–1522.

Louppe, G. (2014). Understanding random forests: From theory to practice.

Madani, O., Pennock, D. M., and Flake, G. W. (2005). Co-validation: Using model disagreement on unlabeled data to validate classification algorithms. In *Advances in neural information processing systems*, pages 873–880.

Makarova, K. S., Wolf, Y. I., Iranzo, J., Shmakov, S. A., Alkhnbashi, O. S., Brouns, S. J., Charpentier, E., Cheng, D., Haft, D. H., Horvath, P., et al. (2020). Evolutionary classification of crispr–cas systems: a burst of class 2 and derived variants. *Nature Reviews Microbiology*, 18(2):67–83.

Mann, H. B. and Whitney, D. R. (1947). On a test of whether one of two random variables is stochastically larger than the other. *The Annals of Mathematical Statistics*, 18(1):50–60.

Masegosa, A., Lorenzen, S., Igel, C., and Seldin, Y. (2020). Second order pac-bayesian bounds for the weighted majority vote. *Advances in Neural Information Processing Systems*, 33.

Mason, L., Baxter, J., Bartlett, P., and Frean, M. (1999). Boosting algorithms as gradient descent. In *Proceedings of the 12th International Conference on Neural Information Processing Systems*, pages 512–518.

Maurer, A. (2004). A note on the pac bayesian theorem.

Maximov, Y., Amini, M.-R., and Harchaoui, Z. (2018). Rademacher complexity bounds for a penalized multi-class semi-supervised algorithm. *Journal of Artificial Intelligence Research*, 61(1):761–786.

McAllester, D. (2003). Simplified PAC-Bayesian margin bounds. In Schölkopf, B. and Warmuth, M. K., editors, *Learning Theory and Kernel Machines*, pages 203–215, Berlin, Heidelberg. Springer Berlin Heidelberg.

McAllester, D. A. (1999). Some pac-bayesian theorems. *Machine Learning*, 37(3):355–363.

Miller, D. J. and Browning, J. (2003). A mixture model and em-based algorithm for class discovery, robust classification, and outlier rejection in mixed labeled/unlabeled data sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(11):1468–1483.

Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill, Inc., New York, NY, USA, 1 edition.

Mitrofanov, A., Alkhnbashi, O. S., Shmakov, S. A., Makarova, K. S., Koonin, E. V., and Backofen, R. (2021). Crispridentify: identification of crispr arrays using machine learning approach. *Nucleic acids research*, 49(4):e20–e20.

Mohri, M., Rostamizadeh, A., and Talwalkar, A. (2018). *Foundations of machine learning*. MIT press.

Morvant, E., Koço, S., and Ralaivola, L. (2012). Pac-bayesian generalization bound on confusion matrix for multi-class classification. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*.

Natarajan, N., Dhillon, I. S., Ravikumar, P. K., and Tewari, A. (2013). Learning with noisy labels. In *Advances in Neural Information Processing Systems*, pages 1196–1204.

Nie, F., Xiang, S., Liu, Y., and Zhang, C. (2010). A general graph-based semi-supervised learning with novel class discovery. *Neural Computing and Applications*, 19(4):549–555.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Peng, P., Wong, R. C.-W., and Yu, P. S. (2014). Learning on probabilistic labels. In *Proceedings of the 2014 SIAM International Conference on Data Mining*, pages 307–315. SIAM.

Pickar-Oliver, A. and Gersbach, C. A. (2019). The next generation of crispr–cas technologies and applications. *Nature reviews Molecular cell biology*, 20(8):490–507.

Quinlan, J. R. (1986). Induction of decision trees. *Machine learning*, 1(1):81–106.

Ren, J., Qiu, Z., Fan, W., Cheng, H., and Yu, P. S. (2008). Forward semi-supervised feature selection. In Washio, T., Suzuki, E., Ting, K. M., and Inokuchi, A., editors, *Advances in Knowledge Discovery and Data Mining*, pages 970–976, Berlin, Heidelberg. Springer Berlin Heidelberg.

Rigollet, P. (2007). Generalization error bounds in semi-supervised classification under the cluster assumption. *Journal of Machine Learning Research*, 8(Jul):1369–1392.

Roy, J.-F., Marchand, M., and Laviolette, F. (2016). A column generation bound minimization approach with pac-bayesian generalization guarantees. In *Artificial Intelligence and Statistics*, pages 1241–1249.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning internal representations by error propagation. In Rumelhart, D. E. and Mcclelland, J. L., editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations*, pages 318–362. MIT Press, Cambridge, MA.

Russel, J., Pinilla-Redondo, R., Mayo-Muñoz, D., Shah, S. A., and Sørensen, S. J. (2020). CRISPRCasTyper: An automated tool for the identification, annotation and classification of CRISPR-Cas loci. *bioRxiv*, page 2020.05.15.097824.

Schölkopf, B. (1997). *Support vector learning*. PhD thesis, Oldenbourg München, Germany.

Scott, C. (2015). A rate of convergence for mixture proportion estimation, with application to learning from noisy labels. In *Artificial Intelligence and Statistics*, pages 838–846.

Sheikhpour, R., Sarram, M. A., Gharaghani, S., and Chahooki, M. A. Z. (2017). A survey on semi-supervised feature selection methods. *Pattern Recognition*, 64(C):141–158.

Siedlecki, W. and Sklansky, J. (1993). A note on genetic algorithms for large-scale feature selection. In *Handbook of pattern recognition and computer vision*, pages 88–107. World Scientific.

Song, L., Smola, A., Gretton, A., Borgwardt, K. M., and Bedo, J. (2007). Supervised feature selection via dependence estimation. In *Proceedings of the 24th international conference on Machine learning*, pages 823–830.

Stone, M. (1974). Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society: Series B (Methodological)*, 36(2):111–133.

Thiemann, N., Igel, C., Wintenberger, O., and Seldin, Y. (2017). A strongly quasi-convex pac-bayesian bound. In *International Conference on Algorithmic Learning Theory*, pages 466–492. PMLR.

Tropp, J. A. (2012). User-Friendly Tail Bounds for Sums of Random Matrices. *Foundations of Computational Mathematics*, 12(4):389–434.

Tumer, K. and Ghosh, J. (1996). Analysis of decision boundaries in linearly combined neural classifiers. *Pattern Recognition*, 29(2):341–348.

Tür, G., Hakkani-Tür, D. Z., and Schapire, R. E. (2005). Combining active and semi-supervised learning for spoken language understanding. *Speech Communication*, 45:171–186.

Tuv, E., Borisov, A., Runger, G., and Torkkola, K. (2009). Feature selection with ensembles, artificial variables, and redundancy elimination. *Journal of Machine Learning Research*, 10:1341–1366.

Vapnik, V. (1982). *Estimation of Dependences Based on Empirical Data: Springer Series in Statistics (Springer Series in Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Vapnik, V. N. (1998). *Statistical Learning Theory*. Wiley-Interscience.

Viallard, P., Germain, P., Habrard, A., and Morvant, E. (2021). Self-bounding majority vote learning algorithms by the direct minimization of a tight pac-bayesian c-bound. *arXiv preprint arXiv:2104.13626*.

Xia, X., Liu, T., Wang, N., Han, B., Gong, C., Niu, G., and Sugiyama, M. (2019). Are anchor points really indispensable in label-noise learning? In *Advances in Neural Information Processing Systems*, pages 6838–6849.

Xiao, H., Rasul, K., and Vollgraf, R. (2017). Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms.

Xue, B., Zhang, M., Browne, W. N., and Yao, X. (2015). A survey on evolutionary computation approaches to feature selection. *IEEE Transactions on Evolutionary Computation*, 20(4):606–626.

Yang, M., Chen, Y.-J., and Ji, G.-L. (2010). Semi_fisher score: A semi-supervised method for feature selection. In *2010 International Conference on Machine Learning and Cybernetics*, volume 1, pages 527–532. IEEE.

Zhao, J., Lu, K., and He, X. (2008). Locality sensitive semi-supervised feature selection. *Neurocomputing*, 71(10-12):1842–1849.

Zhou, D., Bousquet, O., Lal, T. N., Weston, J., and Schölkopf, B. (2004). Learning with local and global consistency. In *Advances in Neural Information Processing Systems*, pages 321–328.

Zhu, X., Ghahramani, Z., and Lafferty, J. D. (2003). Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 912–919.

Zou, Y., Yu, Z., Liu, X., Kumar, B., and Wang, J. (2019). Confidence regularized self-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5982–5991.