Adaptive Online Learning

Dmitry Adamskiy

Computer Learning Research Centre and
Department of Computer Science,
Royal Holloway, University of London,
United Kingdom

2013

A dissertation submitted in fulfilment of the degree of Doctor of Philosophy.

Declaration

I declare that this dissertation was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text and that this work has not been submitted for any other degree of professional qualification except as specified.

Dmitry Adamskiy

Abstract

The research that constitutes this thesis was driven by the two related goals in mind. The first one was to develop new efficient online learning algorithms and to study their properties and theoretical guarantees. The second one was to study real-world data and find algorithms appropriate for the particular real-world problems. This thesis studies online prediction with few assumptions about the nature of the data. This is important for real-world applications of machine learning as complex assumptions about the data are rarely justified. We consider two frameworks: conformal prediction, which is based on the randomness assumption, and prediction with expert advice, where no assumptions about the data are made at all.

Conformal predictors are set predictors, that is a set of possible labels is issued by Learner at each trial. After the prediction is made the real label is revealed and Learner's prediction is evaluated. In case of classification the label space is finite so Learner makes an error if the true label is not in the set produced by Learner. Conformal prediction was originally developed for the supervised learning task and was proved to be valid in the sense of making errors with a prespecified probability. We will study possible ways of extending this approach to the semi-supervised case and build a valid algorithm for this task. Also, we will apply conformal prediction technique to the problem of diagnosing tuberculosis in cattle.

Whereas conformal prediction relies on just the randomness assumption, prediction with expert advice drops this one as well. One may wonder whether it is possible to make good predictions under these circumstances. However Learner is provided with predictions of a certain class of experts (or prediction strategies) and may base his prediction on them. The goal then is to perform not much worse than the best strategy in the class. This is achieved by carefully mixing (aggregating) predictions of the base experts. However, often the nature of data changes over time, such that there is a region where one expert is good, followed by a region where another is good and so on. This leads to the algorithms which we call adaptive: they take into account this structure of the data. We explore the possibilities offered by the framework of specialist experts to build adaptive algorithms. This line of thought allows us then to provide an intuitive explanation for the mysterious Mixing Past Posteriors algorithm and build a new algorithm with sharp bounds for Online Multitask Learning.

Acknowledgements

First I would like to thank my supervisors Alex Gammerman and Volodya Vovk for providing constant support during my research.

Next I would like to thank my coauthors. Thanks to Alexey Chernov, whose ideas sparkled my interest in virtual specialists. Thanks to Wouter M. Koolen, who got enthusiastic about virtual specialists and told me about Mixing Past Posteriors. I would like to also thank Wouter for numerous life-hacks, hobby side-projects and TEX tricks he taught me (as well as for keeping me fit during the bike rides he organized). Many thanks to Manfred Warmuth for the idea of applying virtual specialists to multitask learning. Thanks to Ilia Nouretdinov for the joint work on conformal prediction.

Special thanks to Yuri Kalnishkan for the great number of fruitful discussions during the coffee-breaks.

This work was supported by Veterinary Laboratories Agency of DEFRA grant and I would like to thank Nick Coldham and Andy Mitchell for the discussions and for providing Bovine TB data for the experiments.

I am grateful to all of the students and staff in the Computer Science department for the friendly environment they created. I am grateful to the department for the financial support which made it possible to present the results of our work at conferences.

The work is not possible without occasional breaks from it, so I would like to thank all members of SLOW(South London Orienteers) for the great orienteering weekends.

I would like to thank my housemates James Lewis, Will Horner, Sam Heron and Stella di Virgilio. And last but not the least, very special thanks to my wife Natasha for her patience during all these years!

Contents

1 Introduction **6**
1.1 Motivation 6
1.2 Main contributions 9
1.2.1 Conformal prediction 9
1.2.2 Prediction with expert advice 9
1.3 Publications 10
1.4 The structure of this thesis 10

2 Conformal prediction **11**
2.1 Background: conformal prediction for supervised learning . . . 12
2.1.1 Formal definitions 13
2.1.2 Forced prediction with confidence 18
2.1.3 Performance measures 19
2.1.4 Label-conditional conformal predictors 19
2.1.5 Non-conformity measures based on a loss function . . . 21
2.1.6 Example of nonconformity measure 21
2.2 Conformal prediction for semi-supervised learning 22
2.2.1 Semi-supervised learning problem 22
2.2.2 Learning on manifolds 22
2.2.3 Conformal predictor with external information 23
2.2.4 Geo-NN: example of NCM for semi-supervised learning 24
2.3 Applying conformal prediction to the Bovine TB diagnosis . . 27

3 Prediction with expert advice: specialists **32**
3.1 Background: prediction with expert advice and specialists . . 33
3.1.1 Mix loss and the Aggregating Algorithm 35
3.1.2 Specialist experts 37
3.1.3 Fixed Share and Mixing Past Posteriors 40
3.2 A closer look at Adaptive Regret 42
3.2.1 Specialist experts approach to adaptivity 43
3.2.2 Fixed Share recovered 44
3.2.3 Restarts approach to adaptivity 46
3.2.4 Fixed Share worst-case adaptive regret 48
3.2.5 Fixed Share is the optimal adaptive algorithm 52
3.3 Mixing Past Posteriors demystified 55
3.3.1 Construction 55
3.3.2 MPP recovered 56
3.3.3 A simple Markov chain circadian prior 58
3.4 Online Sparse Multitask Learning 61
3.4.1 Construction 62
3.4.2 Implementation 62
3.4.3 Multitask Learning experiment 65

A Adaptive regret for mix loss transfers to mixable losses **67**

B Worst-case adaptive regret data for Fixed Share **70**

Chapter 1

Introduction

1.1 Motivation

Machine learning is a broad field of Artificial Intelligence studying algorithms that learn from data. In recent years machine learning algorithms penetrated a great variety of fields, such as medical diagnosis, character recognition, email filtering, to name a few.

Normally, learning from data is understood in the sense that the performance of the algorithms in some task improves with experience. Naturally, this is a very broad definition and there is a great variety of methods falling within it. There are two broad learning frameworks, *batch* and *online*. In the former we get a training set of data right from the start. This thesis is mostly concerned with online learning. In the online learning setting the data comes sequentially rather than in the form of a training set and Learner has to make a prediction about the data on each trial or choose some action. Real-world examples include weather prediction, sequential investment, sequential coding etc.

We are interested in studying online learning algorithms that can provide strong theoretical guarantees while making very weak (if any) assumptions about the nature of the data. This is a natural approach when real-world

data is concerned as complex assumptions about it are rarely justified. In this thesis we consider two frameworks: *conformal prediction*, which is based on the randomness assumption, and *prediction with expert advice*, where no assumptions about the data are made at all.

In the conformal prediction scenario Learner's task is to predict the label of the data object at each trial. Conformal predictors are *set predictors*, that is the set of possible labels is issued at each trial. After the prediction is made the real label is revealed and Learner's prediction is evaluated. In case of *classification* the label space is finite so the natural measure of the quality of predictions is the number of errors. Here we say that Learner makes an error if the true label is not in the set produced by Learner. Conformal prediction was originally developed for the supervised learning task. However in the real world labeled data is sometimes expensive to obtain. This gave rise to semi-supervised learning where the algorithms try to use the easy-to-obtain unlabeled data to improve their performance. We studied a possible way of extending conformal prediction approach to the semi-supervised case.

Whereas conformal prediction relies on just the randomness assumption, prediction with expert advice makes no assumptions about the nature of the data at all. One may wonder whether it is possible to make good predictions under these circumstances. However Learner is provided with predictions of a certain class of experts (or prediction strategies) and may base his prediction on that. The goal then is to perform not much worse than the best strategy in the class. This is achieved by carefully mixing (aggregating) predictions of the experts.

In 1997 Freund et al. in [17] extended the known technique of aggregating experts predictions to the class of experts who can abstain from prediction. They called those experts specialists. This idea provides inspiration to construct virtual specialists that correspond to the real ones but can abstain from predictions.

This research started with the ambition to build algorithms with good

adaptive regret using the virtual specialists, that is, that compete well against the real experts on all the time intervals simultaneously. Surprisingly, we arrived at the well-known algorithm called Fixed Share, known, however, for the other type of bounds. This suggested that the class of algorithms covered by the idea of virtual specialists is rather wide. And indeed it provided a unifying framework for already known algorithms such as Fixed Share and Mixing Past Posteriors as well as a new algorithm for online multitask learning. We call these algorithms adaptive because they manage to adjust to the changes in the nature of data and exploit the structure in it. In the case of tracking algorithms (Fixed Share and Mixing past Posteriors), this structure means that there is a region where one expert is good, followed by a region where another is good and so on.

The research that constitutes this thesis was driven by two related goals in mind. The first one was to develop new efficient online learning algorithms and to study their properties and theoretical guarantees. We consider the algorithms in the frameworks of online prediction with confidence and prediction with expert advice. The second one was to study real-world data and find algorithms appropriate for particular real-world problems. Our focus was on the data provided by the VLA and was related to the problem of diagnosing tuberculosis in cattle. We applied conformal prediction to the VETNET database, which allowed us to reduce the number of false positives of the skin test while controlling for the number of mistakes. The other task was to predict the result of the skin test itself in an attempt to work towards the more targeted testing. We used this task as a testing ground for the new multitask learning algorithm.

1.2 Main contributions

1.2.1 Conformal prediction

We studied an extension of conformal prediction technique to the semi-supervised learning case and built a valid algorithm for this task. Our empirical study on the subset of USPS dataset showed an increase of confidence due to using the unlabeled examples.

We applied conformal prediction to the Bovine TB prediction task and found a way to increase the positive predictive rate of the skin test.

1.2.2 Prediction with expert advice

The main theoretical result of the thesis is the development of the specialist predictors framework. In 1997 Freund et al. [17] propose the extension of the well-known Aggregating Algorithm to the case where experts are allowed to abstain from prediction. They call such experts *specialists* and prove that the bounds guaranteed by the original algorithm transfer to this wider class of comparators.

This is further generalized in [15] where the crucial idea of assigning the loss of Learner to the abstaining experts is introduced and the mixable losses are considered.

But the real power and elegance of this method is revealed when it is applied to the virtual specialists created from the real ones. In this way well-known existing algorithms for tracking problems such as Fixed Share [26] and Mixing Past Posteriors [9] can be recovered. As a nice side-effect, new tight bounds for the adaptive regret of Fixed Share were obtained. Furthermore, applying the idea of virtual specialists to Online Multitask problems led to an algorithm achieving better guarantees than the current state of the art [1].

1.3 Publications

The results of Section 2.2 on semi-supervised conformal predictors were published as a book chapter [5].

The application of conformal prediction to the Bovine TB data (section 2.3) was presented at the AIAB workshop in 2011 and published in the proceedings [4].

The work on conformal prediction was done in collaboration with VLA.

Results on Adaptive Regret (section 3.2) were presented at ALT2012 and published in the proceedings [3].

Results on Mixing Past Posteriors and online multitask learning were presented at NIPS2012 and published in the proceedings [28].

1.4 The structure of this thesis

This introductory chapter presents the motivation behind this research and summarizes the main contributions. The rest of the thesis is organized as follows.

Chapter 2 is devoted to the theory of conformal prediction. We present the necessary background in Section 2.1. Section 2.2 is concerned with the extension of conformal prediction to the semi-supervised learning. Application to the Bovine TB diagnosis is the subject of Section 2.3.

Chapter 3 covers the main theoretical results of the thesis, that is the development of the specialist experts framework. It starts with the necessary preliminaries on prediction with expert advice and specialists in Section 3.1. The virtual specialists corresponding to the Fixed Share algorithm and the analysis of adaptive regret of Fixed Share is the subject of Section 3.2. More complex virtual specialists are constructed in Section 3.3 to recover the Mixing Past Posteriors algorithm. Section 3.4 presents the new algorithm for multitask learning based on "task subset specialists".

Chapter 2

Conformal prediction

This chapter is devoted to methods that allow Learner to express confidence in individual predictions. Suppose, for example, that the goal is to predict the labelling of the grey-scale images representing scanned digits (as in the classical USPS dataset). Traditional machine learning algorithms could be trained to output a hypothesis that will map any grey-scale image to its label. However, some images are easier to classify than others, so we would like to have an algorithm that will output a statement about how strong it believes in the prediction along with the prediction itself. Naturally, some idea of how certain the algorithm is about the particular example could be extracted from the traditional algorithms as well. Hyperplane-based predictors (e.g. Support Vector Machines) are "more confident" in their prediction if the example is further away from the hyperplane. The problem here is that we cannot convert this "confidence" to a property that we could rely upon. Informally speaking, we would like to have such a predictor that if it issues a prediction with, say, 95% confidence, we are sure that the probability of error is indeed at most 0.05. This sort of guarantee will be called *validity* (precise definition to follow) and this is what conformal predictors achieve automatically.

This approach was first applied to the underlying Support Vector Machine

predictor in a 1998 paper by Gammerman et al. [19]. It was then generalized by Saunders et al. [37] in 1999. Efficient implementation of conformal prediction for the regression case was studied in [34]. A comprehensive summary of the results in the field of prediction with confidence could be found in the book [45].

In this chapter we give an overview of the existing theory of conformal prediction in Section 2.1, present our extension of conformal prediction framework to the semi-supervised case in Section 2.2 and apply conformal prediction to the Bovine TB database in Section 2.3.

2.1 Background: conformal prediction for supervised learning

In this section we consider the online supervised learning. Here *online* means that the examples are presented one by one. *Supervised* means that that the examples consist of *objects* and *labels* and that on each trial Learner predicts a label after observing the object. Then after the prediction is made the true label is revealed.

Learning tasks vary and the examples could be of different nature. For instance, in the case of Optical Character Recognition, the objects could be grey-scale images and the labels are the symbols these images represent and that the algorithm tries to recognize. For the typical learning task of predicting house prices the objects are individual house attributes like the size, the number of bedrooms, etc. and the labels are prices. Depending on the type of labels, supervised learning tasks are typically divided into *classification* and *regression*. Classification deals with finite label sets whereas regression deals with continuous ones. In this chapter we are interested in classification.

What could be the goal of Learner? There is more than one answer to this. In practice the quality of prediction is often estimated using the held-out test set or the cross-validation techniques. In *Statistical Learning Theory*

(see [39]) the emphasis is on the Probably Approximately Correct (PAC) predictions. Informally, that means that we want to be able to guarantee under certain assumptions on the data source that we will be able to predict reasonably well most of the time.

Conformal prediction [45] is a way of making valid predictions with confidence which does not require any assumption other than the "randomness assumption": the examples are generated from the same probability distribution independently of each other.

The most remarkable feature of conformal predictors is their validity, that is making errors with a prespecified probability. Also, it is possible to estimate confidence in the prediction of the given individual example. Detailed explanation of the theory of conformal prediction can be found in [45]; here we present the bits required for the following sections.

2.1.1 Formal definitions

We consider the following protocol. Nature outputs a sequence of examples sequentially:

$$(x_1, y_2), (x_2, y_2), \ldots, (x_n, y_n), \ldots .$$

Examples consist of *objects* $x_i \in X$ and *labels* $y_i \in Y$ where X and Y are measurable spaces. Here X is called *object space* and Y is called *label space*. Sometimes we will use the notation $z_i = (x_i, y_i)$ for examples and will call $Z = X \times Y$ the *example space*.

The standard assumption which is often made in Machine Learning is the *randomness assumption*: the examples are drawn independently from the same probability distribution P on Z. Actually, conformal predictors require the slightly weaker *exchangeability assumption* and this difference will be important in the extension to the semi-supervised setting.

Definition 1. A distribution P on Z^∞ is *exchangeable* if for every positive integer n, every permutation π of $\{1,\dots,n\}$ and every measurable set $E \subseteq Z^n$,

$$\begin{aligned} &P\{(z_1,z_2,\dots) \in Z^\infty : (z_1,\dots,z_n) \in E\} \\ &= P\{(z_1,z_2,\dots) \in Z^\infty : (z_{\pi(1)},\dots,z_{\pi(n)}) \in E\} \end{aligned}$$

The goal of Learner at trial n is to predict the label y_n given all the past examples $z_1, z_2, \dots, z_{n-1}$ and the object x_n. First we define "simple predictors".

Definition 2. A *simple predictor* is a function of type

$$D : Z^* \times X \to Y$$

However, we want to be able to express the (un)certainty of our predictions by issuing a prediction set based on the significance level ϵ instead of a single element of Y.

Definition 3. A *confidence predictor* is a measurable function of type

$$\Gamma : Z^* \times X \times (0,1) \to 2^Y$$

such that the subsets shrink as ϵ increases:

$$\Gamma(z_1,\dots,z_{n-1},x_n,\epsilon_1) \subseteq \Gamma(z_1,\dots,z_{n-1},x_n,\epsilon_2)$$

whenever $\epsilon_1 \geq \epsilon_2$. In what follows we will position ϵ as a superscript instead of placing it with the other arguments. Measurable here means that for each n the set of sequences $\epsilon, x_1, y_1, \cdots, x_n, y_n$ satisfying

$$y_n \in \Gamma^\epsilon(x_1, y_1, \cdots, x_{n-1}, y_{n-1}, x_n)$$

is a measurable subset of $(0,1) \times (X \times Y)^n$. The complementary value $1 - \epsilon$ is called the *confidence level*.

Now we turn to the desired properties of the confidence predictors, that is, validity and efficiency. There are different notions of validity; here we shall restrict ourselves to *conservative validity*. The confidence predictor *makes an error* on trial n if the true label y_n is not in the prediction set. For a sequence of outcomes $\omega = (x_1, y_1, x_1, y_2, \dots)$ we define

$$\mathrm{err}_n^\epsilon(\Gamma, \omega) := \begin{cases} 1 & \text{if } y_n \notin \Gamma^\epsilon(x_1, y_1, \dots, x_{n-1}, y_{n-1}, x_n) \\ 0 & \text{otherwise,} \end{cases} \tag{2.1}$$

.

Definition 4. A confidence predictor is *conservatively valid* if for any exchangeable probability distribution P on Z^∞ there exists a probability space with two families

$$(\xi_n^{(\epsilon)} : \epsilon \in (0,1), n = 1, 2, \dots), \quad (\eta_n^{(\epsilon)} : \epsilon \in (0,1), n = 1, 2, \dots)$$

of $\{0,1\}$-valued random variables such that:

- for a fixed $\epsilon, \xi_n^{(\epsilon)}, n = 1, 2, \dots$ is a sequence of independent Bernoulli random variables with parameter ϵ;
- for all n and ϵ, $\eta_n^{(\epsilon)} \le \xi_n^{(\epsilon)}$;
- the joint distribution of $\mathrm{err}_n^\epsilon(\Gamma, w)$, with $w \sim P$, $\epsilon \in (0,1), n = 1, 2, \dots$, coincides with the joint distribution of $\eta_n^{(\epsilon)}, \epsilon \in (0,1), n = 1, 2, \dots$

The motivation for this definition is as follows. Learner would like to control the number of mistakes in the following sense: for the prespecified significance level ϵ errors are made independently with probability ϵ. This property is called *exact validity* and it could be shown that it is impossible to

achieve it without certain randomization ([45], Theorem 2.1). However in the definition above we allow the probabilities to be even less than ϵ, so that the sequence of errors is dominated in distribution by a sequence of independent Bernoulli random variables. This is achievable by conformal predictors.

Finite horizon. We will also be interested in the finite horizon version of validity. Suppose that we have a bag of N examples and the data is drawn from it one by one at random. Clearly, in this scenario the data is not i.i.d. anymore. However, the probability of each permutation of the examples is $\frac{1}{n!}$ so the distribution is exchangeable. It turns out that conformal predictors continue to be valid in this setting, where the definition of validity is adapted in the obvious way: P is now the uniform distribution on the permutations of N examples and instead of $n = 1, 2, \dots$ one should read $n = 1, 2, \dots, N$.

Now we are ready to define conformal predictors. They are built around the notion of nonconformity measure. Intuitively, we would like to predict those labels that make the example similar to the previous ones in a certain sense and a nonconformity measure is a function expressing how different a given example is from the bag of other examples.

However, formally it is a very simple object.

Definition 5. A *nonconformity measure* is a measurable function

$$A : Z^{(*)} \times Z \to \overline{\mathbb{R}}$$

where $Z^{(*)}$ is the set of all bags (multisets) of elements of Z.

Any nonconformity measure defines a conformal predictor in the following manner. The algorithm tries each possible label y of x_n. Then it calculates nonconformity scores for each example:

$$\alpha_j^y = A(\{z_1, \dots, z_{n-1}, (x_n, y)\}, z_j)$$

for $j = 1, \dots, n-1$ and $\alpha_n^y = A(\{z_1, \dots, z_{n-1}, (x_n, y)\}, (x_n, y))$. Here $\{z_1, \dots, z_n\}$

is a bag consisting of elements $z_1, \ldots, z_n$ some of which may be identical with each other.

The p-value of this label is then defined as

$$p(y) = \frac{\#\{j = 1, \ldots, n : \alpha_j^y \geq \alpha_n^y\}}{n}.$$

Conformal predictor outputs the prediction set consisting of those labels for which the p-value is greater than the prespecified significance level ϵ:

$$\Gamma_n^\epsilon = \{y : p(y) > \epsilon\}$$

The main result in the theory of conformal prediction is the following proposition (Proposition 2.3 in [45]):

Proposition 6. *All conformal predictors are conservatively valid.*

We will also need the finite horizon version of it.

Proposition 7. *Suppose that Learner is presented with the finite bag of examples. If at each trial the example is drawn (without replacement) from this bag, then any conformal predictor is valid in the sense described above.*

This last result follows from Theorem 8.2 in [45]. The version of it for the exchangeability assumption is actually an ingredient in proving Proposition 6 as shown in [44].

This means that no matter what nonconformity measure we choose the resulting conformal predictor will be valid. As an extreme example, consider the constant nonconformity measure. One can immediately verify that it result in a conformal predictor that will include all possible labels in the prediction set. This means that it makes no mistakes at all and thus is valid (but extremely inefficient). This shows that in order to obtain an efficient predictor one should carefully select a suitable nonconformity measure. Intuitively, it should measure strangeness so that when the new example is

complemented with the "wrong" label it becomes very strange w.r.t. the bag of old examples.

Algorithm 1 Conformal Predictor for classification

Input: data examples $(x_1, y_1), (x_2, y_2), \ldots, (x_{n-1}, y_{n-1}) \in X \times Y$
Input: a new object $x_n \in X$
Input: a non-conformity measure $A : Z^{(*)} \times Z \to \mathbb{R}$
Input: a significance level ϵ
$z_1 = (x_1, y_1), \ldots, z_{n-1} = (x_{n-1}, y_{n-1})$
for $y \in Y$ **do**
  $z_n = (x_n, y)$
  for j in $1, 2, \ldots, n$ **do**
    $\alpha_j = A(\lbag z_1, \ldots, z_{n-1}, z_n \rbag, z_j)$
  end for
  $p(y) = \frac{\#\{j=1,\ldots,n : \alpha_j \geq \alpha_n\}}{n}$
end for
Output: prediction set $\Gamma^\epsilon_n = \{y : p(y) > \epsilon\}$
Output: any forced prediction $\hat{y}_n \in \arg\max_y \{p(y)\}$
Output: confidence
$\mathrm{conf}(\hat{y}_n) = 1 - \max_{y \neq \hat{y}_n} \{p(y)\}$

2.1.2 Forced prediction with confidence

In the case of finite Y (classification problem), there is an alternative way to represent the output: single ("forced") prediction and some measure of confidence in this prediction. It does not require the significance level ϵ as input. The single prediction is selected by largest p-value. *Confidence* is defined as

$$\sup\{1 - \epsilon : |\Gamma^\epsilon| \leq 1\}.$$

Here we assume that there are no ties among the p-values. For example: $Y = \{1, 2, 3\}$ and $p(1) = 0.005, p(2) = 0.05, p(3) = 0.5$, then forced prediction is 3, confidence in it is $1 - 0.05 = 0.95$.

In terms of confidence, validity implies the following: with probability at

least $1 - \epsilon$ either the forced prediction is correct or confidence in it is not larger than $1 - \epsilon$.

2.1.3 Performance measures

Suppose that there are two nonconformity measures, so there are two versions of conformal prediction algorithm and thus both are valid. How to check their efficiency on a real data set?

If the same confidence level is selected for several methods then we can compare their efficiency by the *uncertainty rate*, percentage of uncertain prediction for a fixed confidence level. If the selected nonconformity measure is not adequate for the problem, the number of uncertain predictions will be too high. For example, if nonconformity measure is a constant, then all predictions will be uncertain.

In terms of forced prediction with confidence, a prediction is uncertain at level ϵ if its confidence is less than $1 - \epsilon$. So uncertainty rate is percentage of predictions with the individual confidence smaller than $1 - \epsilon$. To have an overall performance measure independent from ϵ, we can use *median (or mean) confidence*.

2.1.4 Label-conditional conformal predictors

Algorithm 1 is valid in the sense that under the randomness assumption it makes errors independently on each trial with probability at most ϵ. However, sometimes there are natural categories of the examples (such as men and women for the medical predictions) and we would like to have "category-wise" validity. For instance, suppose that the examples fall into "easy to predict" and "hard to predict" categories; then the overall validity will be reached by any conformal predictor, but the individual error rate for "hard to predict" objects could be worse.

In order to overcome this, Mondrian conformal predictor was presented

in [46]. Here we present a special case of it, label-conditional conformal predictor (see Algorithm 2). In our case the difference from the original conformal predictor is in the definition of p-values: now only the nonconformity scores of the examples with the same label are taken into account.

Algorithm 2 Mondrian conformal predictor for classification

Input: data examples $(x_1, y_1), (x_2, y_2), \ldots, (x_{n-1}, y_{n-1}) \in X \times Y$
Input: a new object $x_n \in X$
Input: a non-conformity measure $A : Z^{(*)} \times Z \to \mathbb{R}$
Input(optional): a significance level ϵ
$z_1 = (x_1, y_1), \ldots, z_{n-1} = (x_{n-1}, y_{n-1})$
for $y \in Y$ **do**
  $z_n = (x_n, y)$
  for j in $1, 2, \ldots, n$ **do**
    $\alpha_j = A(\lbag z_1, \ldots, z_{n-1}, z_n \rbag, z_j)$
  end for
  $p(y) = \frac{\#\{j=1,\ldots,n : y_j = y, \alpha_j \geq \alpha_n\}}{|y = y_j|}$
end for
Output(optional): prediction set $\Gamma^\epsilon_n = \{y : p(y) > \epsilon\}$
Output: forced prediction $\hat{y}_n = \arg\max_y \{p(y)\}$
Output: confidence
$\mathrm{conf}(\hat{y}_n) = 1 - \max_{y \neq \hat{y}_n} \{p(y)\}$

This modifications of conformal predictor is known to be *label-wise valid*, that is the conditional probability of making an error given the labels of the past and the current examples is at most the prespecified value.

We are going to be interested in one-sided label-wise validity in section 2.3.

2.1.5 Non-conformity measures based on a loss function

A universal way to define nonconformity measure is to use an underlying method of basic prediction

$$F : Z^{(*)} \times X \to Y$$

and a loss (discrepancy) function

$$L : Y \times Y \to \mathbb{R}$$

and then to use them directly:

Input: examples $z_1 = (x_1, y_1), \ldots, z_n = (x_n, y_n)$

Input: z_j (one of examples above)

$\alpha_j = L(y_j, F(x_1, y_1, \ldots, x_{j-1}, y_{j-1}, x_{j+1}, y_{j+1}, \ldots, x_n, y_n, x_j))$

Although this method is usually applied for regression (see e.g. [34, 32]) we will see that in its pure form it is not the best way for classification.

2.1.6 Example of nonconformity measure

Suppose that the there is a distance function on X. The following nonconformity measure can then be used for 1 nearest neighbour underlying method:

Input: bag of examples $\wr z_1, \ldots, z_n \int$

Input: z_j (one of objects above)

$A(\wr z_1, \ldots, z_n \int, z_j) = \frac{\min\{\text{dist}(x_j, x_k); k \neq j, y_k = y_j\}}{\min\{\text{dist}(x_j, x_k); y_k \neq y_j\}}$

We can see that the nonconformity score is high when the example is close to an object with a different label and far from any object with the same label.

2.2 Conformal prediction for semi-supervised learning

2.2.1 Semi-supervised learning problem

Now we turn to an extension of conformal predictors to the semi-supervised learning setting. Semi-supervised learning is a recently developed framework naturally arising in many practical tasks. Namely, for some tasks labelled examples for training could be quite expensive to obtain, whereas unlabelled ones are readily available in abundance. The standard examples of such problems are linguistic tasks where the objects are portions of texts. Unlabelled texts could be found on the Internet whereas labelling requires a skilled linguist. Therefore the question is: could we improve the quality of the predictor given the unlabelled data? Semi-supervised learning algorithms differ based on the assumptions they make and range from simple algorithms like self-learning [47] and co-training [8] to more complex manifold learning algorithms [7]. See [14] and [2] for the overview of the existing semi-supervised methods.

2.2.2 Learning on manifolds

If there is no specific relation between the conditional distribution of the label given the object and the marginal distribution of objects then the knowledge of unlabelled data is of no help. Thus some additional assumptions are usually made, and based on the nature of these assumptions different semi-supervised algorithms arise. In what follows we present an example of utilizing two of such assumptions: the manifold assumption and the cluster assumption.

Manifold assumption: The marginal distribution $P(x)$ is supported on a low-dimensional manifold and the conditional distribution $P(y|x)$ is smooth as a function of x on the manifold.

Cluster assumption: We believe that the data in the same cluster is more likely to have the same label.

Usually, the distribution of unlabelled examples (and thus the manifold) is unknown to Learner. Thus we have to model it from the (large) sample of unlabelled examples.

This is usually done by building the *neighbourhood graph*. The problem of dealing with both labelled and unlabelled data can be viewed as either labelling partially labelled data set or as labelling the held out test set. We shall follow the first setting as in [7]; thus the neighbourhood graph is built using both labelled and unlabelled points.

2.2.3 Conformal predictor with external information

Algorithm 3 Conformal Predictor for SSL

Input: data examples $(x_1, y_1), (x_2, y_2), \ldots, (x_{n-1}, y_{n-1}) \in X \times Y$
Input: a new object $x_n \in X$
Input: unlabelled objects $u_1, u_2, \ldots, u_s \in X$
Input: a nonconformity measure $A : (z_i, \lbag z_1, \ldots, z_n \rbag, U) \mapsto \alpha_i$ on pairs $z_i \in X \times Y$ dependent on external information $U = \lbag u_1, u_2, \ldots, u_s \rbag$.
Input(optional): a significance level ϵ
$z_1 = (x_1, y_1), \ldots, z_{n-1} = (x_{n-1}, y_{n-1})$
for $y \in Y$ **do**
  $z_n = (x_n, y)$
  for j in $1, 2, \ldots, n$ **do**
    $\alpha_j = A(z_j, \{z_1, \ldots, z_{n-1}, z_n\}, U)$
  end for
  $p(y) = \frac{\#\{j=1,\ldots,n : \alpha_j \geq \alpha_n\}}{n}$
end for
Output(optional): prediction set $\Gamma^\epsilon_n = \{y : p(y) > \epsilon\}$
Output: any forced prediction $\hat{y}_n \in \arg\max_y \{p(y)\}$
Output: individual confidence
$\mathrm{conf}(\hat{y}_n) = 1 - \max_{y \neq \hat{y}_n} p(y)$

General scheme is provided by algorithm 3. Its main difference from the

conformal predictor for supervised learning (algorithm 1) is that nonconformity measure depends on an external argument. In particular, it may be the set of unlabelled objects.

In the online setting it could be viewed as follows:

1. A bag of unlabelled examples is presented to Learner. Learner uses this bag to fix the specific nonconformity measure from the possible family of nonconformity measures.

2. Examples are randomly drawn from this bag without repetition and Learner tries to predict their label.

3. After the prediction is made, the label for each example is revealed.

This is in some way similar to the online learning on graphs setting [25] where the graph is known to Learner and its task is to predict sequentially the labels of vertices.

Proposition 8. *Algorithms 3 is conservatively valid.*

Proof. As all the permutations are of equal probability and the nonconformity measure and the role of unlabeled examples is restricted to the selection of a particular nonconformity measure it is possible to apply the finite horizon proposition stating conformal prediction validity (Proposition 7). □

Thus even if Learner's choice of additional assumption for the semi-supervised learning is not correct, the resulting predictor will still be valid.

2.2.4 Geo-NN: example of NCM for semi-supervised learning

One simple way of using the unlabelled data under the manifold (and cluster) assumptions is to adjust the nearest neighbours nonconformity measure.

Instead of using the metric in the ambient space we can use the "geodesic" distance approximated as a distance on the neighbourhood graph built using both labelled and unlabelled data. In the case where the manifold or cluster assumption is true it is indeed the distance we are interested in, because the conditional distribution of labels is smooth along the manifold and for a given point we want to find its nearest (labelled) neighbours on the manifold.

Algorithm 4 Geodesic nearest neighbours NCM

Input: data set $z_1 = (x_1, y_1), \ldots, z_n = (x_n, y_n)$
Input: z_j (one of objects above)
Input: Unlabelled data: $x_{n+1}, \ldots, x_{n+u}$
Fix the distance function $\mathrm{dist}_0(x_i, x_j)$ in the ambient object space.
Build the neighbourhood graph based on k-nn (or ϵ-balls) using all the examples available (labelled and unlabelled).
for $i, j = 1, \ldots, l + u$ **do**
  $A_{ij} = 1$ if either x_i is one of the k-nearest neighbours of x_j of vice versa.
  $A_{ij} = \infty$ otherwise.
end for
Apply Floyd (of Dijkstra) algorithm to calculate distances on the graph.
Define geodesic distance $\mathrm{dist}_1(x_i, x_j)$ to be the resulting distance function – length of the shortest path between x_i and x_j on the graph (here we make use of the "labelling partially labelled set" setting).
$d_s(j) = \min\{\mathrm{dist}_1(x_j, x_k) | k \neq j, y_k = y_j\}$
$d_o(j) = \min\{\mathrm{dist}_1(x_j, x_k) | y_k \neq y_j\}$
$\alpha_j = d_s(j)/d_o(j)$

Obviously the construction of neighbourhood graph in Algorithm 4 is done only once when the bag of unlabelled examples is presented to Learner. We use the unweighted graph here, however using the weighted graph is also possible. After this is done the distance function is fixed and we proceed with the original 1-NN conformal prediction. The predictor is valid but we may also hope for some improvement in efficiency when the number of labelled examples is not big.

Another possible way of constructing nonconformity measure for the semi-

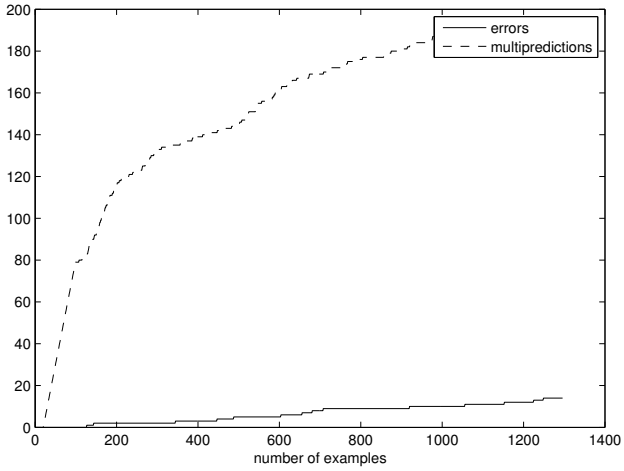

Figure 2.1: Validity of the algorithm 4 ("geodesic"-NN NCM)

supervised setting is to derive it from the semi-supervised basic predictor in the general way mentioned above.

Algorithm 5 SSL-based NCM

Input: data set $z_1 = (x_1, y_1), \ldots, z_n = (x_n, y_n)$
Input: unlabelled objects $u_1, u_2, \ldots, u_s \in X$
Input: z_j (one of labelled objects above)
Train a basic semi=supervised predictor D.
$\alpha_j = L(y_j, D(x_j, \wr z_1, \ldots, z_{j-1}, z_{j+1}, ..., z_n \int, u_1, u_2, \ldots, u_s))$

Figure 2.1 shows the number of errors and the number of uncertain predictions for the confidence level of 99% on a subset of the USPS dataset. We can see that the predictor is indeed valid (the solid line shows that the frequency of errors if around 1%). However, what is the performance benefit?

Figure 2.2 shows the result of applying the "geodesic-NN" conformal predictor and the 1NN-Threshold conformal predictor in the online mode to the subset of USPS dataset consisting of digits 4 and 9 (known to be rather hard to separate). This graph shows a single experiment run with $k = 5$. We see

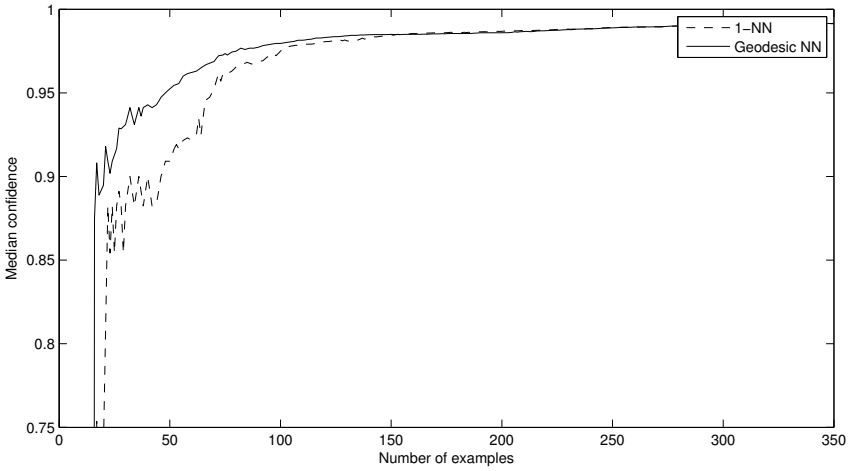

Figure 2.2: Median confidence of 1-nearest neighbour and "geodesic"-nearest neighbour conformal predictors

that the median confidence for the "geodesic-NN" is higher when the number of labelled examples is small and the performance levels off as the number of examples grows. This is what one would expect from semi-supervised setting.

2.3 Applying conformal prediction to the Bovine TB diagnosis

In this section we apply the nearest-neighbours conformal predictor to the VETNET database in order to increase the positive prediction rate of the existing Skin Test. Conformal prediction framework allows us to do so while controlling the risk of misclassifying true positives.

Bovine Tuberculosis (bTB) is an infectious disease of cattle caused by the bacterium Mycobacterium bovis (M.bovis). The disease is widespread in certain areas of the UK (particularly South West England) and of major economic importance, costing the UK Government millions of pounds each

year, since positive animals are slaughtered and compensation is paid to the cattle owners.

The main testing tool for diagnosing TB in cows is the Single Intradermal Cervical Tuberculin (SICCT) skin test. The procedure involves administering intradermally both Bovine and Avian Tuberculin PPDs and measuring the thickening of the skin. The difference of thickenings $(B_2 - B_1) - (A_2 - A_1)$ is the actual numeric result of the test, where A_1, B_1 are the skin thickness measurements before administering Avian(A) and Bovine(B) Tuberculinum and A_2 and B_2 are the measurements after injecting Tuberculinum. Avian Tuberculinum is used to exclude the non-specific reactions. The test is considered positive if the result is greater than a certain threshold (namely 4mm).

The issue of considerable interest to VLA was to reduce the number of false positives of the skin test. Each cow that is positive is slaughtered and the post-mortem analysis is conducted which could find the TB lesions. Furthermore, some of the cows are sent to the culture analysis. If either of those two tests is positive, the cow is considered a true reactor. If both come back negative, it is considered to be a mistake of the test. The goal is then to reduce these mistakes. We use the framework of conformal prediction to achieve this goal while simultaneously controlling the price we have to pay for it: the number of true reactors we are going to classify incorrectly.

The data on all the reactors is stored in the VETNET database. The data stored there includes (per reactor) the numeric test results A_1, A_2, B_1, B_2, and such features as age, herd identifier, date of the test, post-mortem and culture results (if any), type of herd, type of the test and some others.

As there are two tests that can confirm the diagnosis after the cow is slaughtered, the definition of true positives could vary and here we use logical OR as such (the cow is true positive if there are visible lesions or if the culture test was positive). The data in VETNET alone is not enough to judge about the efficiency of the skin test, as the post-mortem tests are not performed for the negative animals. However, it is believed that the positive prediction

Test time	CPHH (herd ID)	Lesions	Culture	Age	$A_2 - A_1$	$B_2 - B_1$
1188860400	35021001701	1	1	61	4	22
1218495600	37079003702	0	0	68	1	9

Table 2.1: Two entries from the VETNET database

rate could be improved by taking into account some other factor apart from just the binary result of the skin test.

We used the positively tested cows from VEBUS subset of the original VETNET database. It includes 12873 false positives and 18673 true positives. In what follows the words "true positives" and "false positives" will refer to the skin test results.

After the preliminary study, it was discovered that the most relevant attributes for classification are the numeric value of skin test result (that is, $(B_2 - B_1) - (A_2 - A_1)$), age and either the ID of the given test which is a herd identifier combined with a test date, or just the identifier of a herd (that may cover several tests performed at different time).

The extract from the VetNet database containing those features is shown in Table 2.3.

The first of these examples is a True Positive (Lesions or Culture test is positive) and the second is a False Positive (both Lesions and Culture tests are negative). The task is to distinguish between these two classes to decrease number of cows being slaughtered. This means that we wish to discover as many cows as possible to be False Positives. On the other hand, the number of True Positives misclassified as False Positives should be strictly limited. So unlike standard conformal prediction, the role of two classes is different.

Thus we present a one-sided version of conformal predictor. Each new example is assigned only one p-value, that corresponds to True Positive hypothesis. Then for a selected significance level ϵ we mark a cow as a False Positive if $p < \epsilon$. This allows us to set the level at which we tolerate the marking of true positive and we aim to mark as many false positives as possible.

The property of validity is interpreted in the following way: if a cow is True Positive, it is marked as a False Positive with probability at most ϵ. A trivial way to achieve this is to mark any cow with probability ϵ. So the result of conformal prediction can be considered as efficient only if the percentage of marked False Positives is essentially larger.

In our experiments we used the nonconformity measure presented in algorithm 6 based on k-Nearest-Neighbour algorithm.

Algorithm 6 kNN Nonconformity Measure for VetNet database

Input: a bag of data examples $z_1 = (x_1, y_1), z_2 = (x_2, y_2), \ldots, z_n = (x_n, y_n) \in X \times Y$
Input: an example $z_i = (x_i, y_i)$ from this bag;
Input: a distance function $d : X \times X \to \mathbb{R}^+$
$A(z_i, \lbag z_1, z_2, \ldots, z_n \rbag) = |\{j : x_j$ is amongst k nearest neighbours of x_i in $x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n$ according to the distance d, and $y_j \neq y_n\}|$

A possible version of efficient predictor can be done setting $k = 50$ and using the following distance, which assigns the highest importance to comparison of herd IDs, second priority is given to the numerical value $(B_2 - B_1) - (A_2 - A_1)$ of skin test, and age is used as an additional source of information"

$$\mathrm{dist}(x_1, x_2) = 100S(x_1, x_2) + T(x_1, x_2) + |\log_{10}(\mathrm{age}(x_1)) - \log_{10}(\mathrm{age}(x_2))|$$

where $S(x_1, x_2) = 0$ if x_1 and x_2 belong to the same herd and 1 otherwise, $T(x_1, x_2)$ is the difference between numerical values of test results on x_1 and x_2. Thus, first of all the animals within given test are considered as neighbours and then all the others. Experiments showed that this distance resulted in an efficient predictor though the validity property holds for other parameters as well. The results on a subset of VetNet database are summarized in the following table. The experiment was performed in an online mode with the data sorted by test date (as in real life).

Significance level	Marked reactors within FP	Marked reactors within TP
1%	1267/12873	138/18673
5%	4880/12873	919/18673
10%	7971/12873	1904/18673

The randomness assumption is clearly a simplification here, but as in some other applications of conformal predictors (see [33]) we can see that it is not essentially broken and we can see that the validity property holds: the fraction of marked reactors within true positives is indeed close to the significance level that was set. The efficiency could be judged by the number of marked reactors within false positives: at a cost of misclassifying 10% of true positives it is possible to identify almost two thirds of test mistakes.

Chapter 3

Prediction with expert advice: specialists

What if we have no clue about the nature of the data? Is it possible to make good predictions in an adversarial environment? These questions are the subject of prediction with expert advice theory where no assumptions at all are made about the data. Instead Learner has access to the pool of prediction strategies or experts and has somehow to base his actions on them. The goal then is to be not much worse than the best of those strategies under some measure of the prediction quality.

Prediction with expert advice started with works of Vovk [41] and Littlestone and Warmuth [30] although the idea of using exponentially weighted averages already appears (in the log-loss context) in the paper of De Santis, Markowski and Wegman [16]. The Aggregating Algorithm, which we use as a main tool in our analysis, was proposed in [41]. Prediction with expert advice has links to the theory of sequential coding [49] and Bayesian decision theory.

In 1997 Freund et al. proposed the modification of the prediction task by considering "experts that specialize" in [17]. This means that sometimes the prediction of a certain strategy (expert) is not available (the name comes

from the fact that human predictors could have their areas of expertise and refuse to issue predictions outside those). Then for each expert Learner wants some guarantee on those trials when this expert did issue predictions. It was shown that an effective algorithm for this setting is obtained by slight modification of the Aggregating Algorithm. The elegant explanation for this fact along with the generalization to the mixable loss functions appeared in [15]. It turns out that the specialist experts when they abstain from prediction could be viewed as the ones silently agreeing with Learner.

Other modifications of prediction task were also studied. Herbster and Warmuth in [26] introduced the notion of tracking regret and proposed Fixed Share, an algorithm competing well against the shifting sequences of experts. Bousquet and Warmuth in 2001 [9] gave the efficient MPP algorithm competing against *sparse* compound experts.

This chapter presents the main theoretical contribution of the thesis, the virtual specialist experts framework. We present the necessary background from the theory of prediction with expert advice and the Aggregating Algorithm (AA). We then present the specialist experts and show how the AA can be modified to accommodate them. Introducing virtual specialists allows us to provide a unifying interpretation to Fixed Share and Mixing Past Posteriors and build a new algorithm with sharp bounds for multitask online learning.[1]

3.1 Background: prediction with expert advice and specialists

We start with describing the setup of prediction with expert advice.

Nature generates outcomes step by step. At every trial Learner tries to predict the outcome. Then the actual outcome is revealed and the quality of

[1]The idea of applying virtual specialists in multitask setting was suggested by Manfred K. Warmuth and was implemented by Wouter Koolen and myself.

Protocol 1 Prediction with expert advice

for $t = 1, 2, \ldots$ **do**
  Experts issue predictions $\gamma_t^n \in \Gamma, n = 1, \ldots, N$
  Learner announces his prediction $\gamma_t \in \Gamma$
  Reality announces the outcome $\omega_t \in \Omega$
  Experts and Learner suffer losses $l_t^n = \lambda(\gamma_t^n, \omega_t)$ and $l_t = \lambda(\gamma_t, \omega_t)$
end for

Learner's prediction is measured by a loss function.

No assumptions are made about the nature of the data. Instead, at every trial Learner is presented with the predictions of a pool of experts and he may base his predictions on these. The goal of Learner in the classical setting is to guarantee small regret, that is, to suffer cumulative loss that is not much larger than that of the best (in hindsight) expert from the pool.

Formally, this setting is presented in Protocol 1. Here, Γ is the prediction space, Ω is the outcome space and $\lambda : \Gamma \times \Omega \to \mathbb{R}$ is a loss function, mapping predictions and outcomes to the losses. The triple $(\Omega, \Gamma, \lambda)$ is sometimes called a *game*.

We are interested in designing algorithms that provide good regret guarantees in the adversarial environment, that is, no matter what the moves of experts and Nature are the regret w.r.t. any expert is small.

Definition 9. The regret w.r.t to the ith expert over the first T trials is the difference between Learner's total loss on trials $1, \ldots, T$ and i-th expert total loss on those trials:

$$R_T^i = L_T - L_T^i,$$

where $L_T := \sum_{t=1}^T l_t$ is the cumulative loss of Learner over T trials and $L_T^i := \sum_{t=1}^T l_t^i$ is the cumulative loss of i-th expert.

The nature of our guarantees depends on the loss function. Consider, for example, a *simple game of prediction*. Here, the prediction and outcome

spaces are both $\{0,1\}$ and the loss function also takes values 0 or 1:

$$\lambda(\gamma,\omega) = \begin{cases} 0 \text{ if } \gamma = \omega \\ 1 \text{ if } \gamma \neq \omega \end{cases}$$

In this case any algorithm can be forced to suffer regret linear in time. To see this consider two experts, one predicting 0 all the time and the other predicting 1 all the time and Nature choosing on every trial the outcome not selected by Learner. Learner's loss on T trial will be T, but at least one of the experts will have his loss not more than $T/2$ making the regret greater than $T/2$. But for some important classes of games Learner can do much better.

3.1.1 Mix loss and the Aggregating Algorithm

In this thesis we will present our results for a specific protocol called *mix loss* prediction (see Protocol 2). It could be understood as follows, in the terms of sequential investment.

There are N types of assets. On each trial Learner has to split his capital between those (one of them may be cash). Nature then reveals the numbers $\ell_t^i \in [-\infty, +\infty]$ such that $e^{-\ell_i}$ is the multiplicative factor indicating the price change of the ith asset. Learner's capital is then multiplied by a factor of $e^{-\ell_t} = \sum_n u_t^n e^{-\ell_t^n}$. As Learner then has to reinvest his capital again, after T trials his capital is multiplied by e^{-L_t} where $L_t = \sum_{t=1,\dots,T} \ell_t$. Regret then has a meaning that $e^{-R_T^i}$ is the multiplicative factor between the Learner's capital after T trials and what it would have been had he invested all of it in the ith asset all the time.

We choose this fundamental setting because it is universal, in the sense that many other common settings reduce to it. For example probability forecasting and data compression are straightforward instances [13]. In addition, mix loss is the baseline for the wider class of *mixable loss functions*, which in-

Protocol 2 Mix loss prediction

for $t = 1, 2, \ldots$ **do**
  Learner announces probability vector $\vec{u}_t \in \triangle_N$
  Reality announces loss vector $\vec{\ell}_t \in [-\infty, \infty]^N$
  Learner suffers loss $\ell_t := -\ln \sum_n u_t^n e^{-\ell_t^n}$
end for

cludes e. g. square loss [43]. Classical regret bounds of Aggregating Algorithm transfer from mix loss to mixable losses almost by definition, and the same reasoning extends to adaptive regret bounds. In addition, mix loss results carry over in the usual modular ways(via Hoeffding and related bounds) to non-mixable games, which include the Hedge setting [18] and Online Convex Optimisation [48].

Now we can present our main tool, the Aggregating Algorithm (AA). The *Aggregating Algorithm* [42] is parametrised by a prior distribution $\vec{u}_1$ on $[N]$ (where $[N]$ denotes the set $\{1, \ldots, N\}$). At trial t the weights of the experts are set to

$$u_t^n := \frac{u_1^n e^{-\sum_{s<t} \ell_s^n}}{\sum_n u_1^n e^{-\sum_{s<t} \ell_s^n}} = \frac{u_1^n e^{-L_{t-1}^n}}{\sum_n u_1^n e^{-L_{t-1}^n}}, \tag{3.1a}$$

which we may also maintain incrementally using the update rule

$$u_{t+1}^n = \frac{u_t^n e^{-\ell_t^n}}{\sum_n u_t^n e^{-\ell_t^n}}. \tag{3.1b}$$

In the log loss setting the Aggregating Algorithm is also known as Bayes mixture. The weights u_t are the prior weights on trial t and the update of (3.1b) becomes the Bayes rule.

Lemma 10. *The losses of Learner satisfy for any T the following:*

$$e^{-L_T} = \sum_{n=1}^{N} u_1^n e^{-L_T^n}. \tag{3.2}$$

Proof.

$$\begin{aligned} e^{-L_T} &= \prod_{t=1\ldots T} e^{-\ell_t} = \prod_{t=1\ldots T} \frac{\sum_{n=1}^N u_1^n e^{-L_t^n}}{\sum_{n=1}^N u_1^n e^{-L_{t-1}^n}} \\ &= \sum_{n=1}^N u_1^n e^{-L_T^n} \end{aligned}$$

□

Theorem 11. *The Aggregating Algorithm in the mix loss case guarantees for every t and n*

$$L_t \le L_t^n - \ln u_1^n. \tag{3.3}$$

Proof. Leaving only one summand corresponding to the n-th expert in the right-hand side of (3.2) and taking $-\ln$ of both sides immediately yields the statement of the theorem. □

If there is no prior knowledge about experts the obvious choice for the prior is uniform, $u_1^n = 1/N$, leading to the regret bound

$$L_t \le L_t^n + \ln N.$$

3.1.2 Specialist experts

Now we turn to the other main building block we are going to use: the specialist experts framework.

It is reasonable to claim that sometimes the prediction of the expert could be not available. Human forecasters may be specialized, unreachable or too expensive, algorithms may run out of memory or simply take too long. Thus it seems natural to generalise the setting to abstaining experts.

This generalization of PEA setting to the specialist experts was first addressed in 1997 paper by Freund et al. [17]. They considered the log-loss case and proposed an algorithm (which they called SBayes) which guaranteed the regret of $\log N$ w.r.t. any specialist expert.

In what follows we call the specialist *asleep* on the trials when he abstains and *awake* on all the other trials.

Now the goal of Learner is slightly different. We want to guarantee that, for every expert, our loss on the trials where this expert was awake is not much bigger than his loss.

Mix loss for specialists. Mix loss setting is adapted to specialists as follows. On every trial Learner is presented with awake set of experts W_t, $|W_t| = k$ and has to play weights on them, $\vec{u}_t \in \triangle_k$. Then the losses of awake experts are announced by Reality and Learner suffers loss $\ell_t := -\ln \sum_{n \in W_t} u_t^n e^{-\ell_t^n}$. In terms of the investment interpretation this means that some assets are not available for trading at certain times and Learner has to split his capital between the others. The cumulative losses L_T^n and $L_T^{(n)}$ are the losses of n-th expert and Learner respectively summed over the trials when n-th expert was awake.

AA for specialists. Surprisingly, this generalization does not make Learner's task any harder. It is possible to slightly modify the existing algorithm (AA) to make it work with specialists while retaining the regret bound. The algorithm keeps the normalized weights is such a way that on each trial the weights of the sleeping specialists are unchanged and the weights of awake specialists are updated in the normal way and renormalized to the original weight of the awake set (see Algorithm 7 for the mix-loss version of it).

Theorem 12. *Learner has a strategy (Algorithm 7) that guarantees in the game of prediction with N specialist experts advice under mix-loss function*

Algorithm 7 AA for specialist experts (mix loss)

Start with weights $u_1^n, \sum_n u_1^n = 1$
for $t = 1, 2, \ldots$ **do**
  Read the awake set W_t.
  Predict with the weights $w_t^n = u_t^n / \sum_{n' \in A_t} u_t^{n'}$
  Update $u_{t+1}^n = u_t e^{l_t - l_t^n}$ for $n \in W_t$
end for

λ for all T and for all $n = 1, \ldots, N$, that

$$L_T^{(n)} \leq L_T^n + \ln N \tag{3.4}$$

Proof. From the definition of mix-loss we get that

$$e^{-\ell_t} = \sum_{k \in W_t} w_t^k e^{-\ell_t^k} = \sum_{k \in W_t} \frac{u_t^k e^{-\ell_t^k}}{\sum_{k \in W_t} u_t^k} \tag{3.5}$$

On the other hand, suppose that all the sleeping experts are instead awake and suffer the same loss as the right-hand side of (3.5). Let's see whether this changes the loss of Learner:

$$e^{-\ell_t} = \sum_{k \in W_t} u_t^k e^{-\ell_t^k} + (1 - \sum_{k \in W_t} u_t^k) \sum_{k \in W_t} \frac{u_t^k e^{-\ell_t^k}}{\sum_{k \in W_t} u_t^k} = \sum_{k \in W_t} \frac{u_t^k e^{-\ell_t^k}}{\sum_{k \in W_t} u_t^k}$$

The weight update also stays the same thus showing that the sleeping experts can be substituted with the awake ones suffering the same loss as Learner. Now the result of the theorem follows from Theorem 11. □

A slightly more elaborate version of this theorem could be found in [17].

Theorem 13 ([17, Theorem 1])**.** *Let p^n be any distribution on a set of specialists with awake sets $W_1, W_2, \ldots$ Then for any T, AA guarantees*

$$\sum_{n=1}^{N} p^n \left(L_T^{(n)} - L_T^n \right) \;\leq\; \triangle\big(p^n \big\| u_1^n\big),$$

where $\triangle\big(p^n\big\|u^n\big)$ denotes the relative entropy $\sum_{n=1}^{N} p^n \ln \frac{p^n}{u^n}$ between the distributions p^n and u^n.

This theorem is easily generalized to the mixable loss-functions, see [15].

3.1.3 Fixed Share and Mixing Past Posteriors

Typically the nature of the data is changing with time: in an initial segment one experts predicts well, followed by a second segment in which another expert has small loss and so forth. For this scenario the natural comparator class is the set of *partition experts* which divide the sequence of T outcomes into B segments and specify the expert that predicts in each segment. By running AA on all exponentially many partition experts comprising the comparator class, we can guarantee regret $\ln \binom{T-1}{B-1} + B \ln N$, which is optimal. The goal then is to find *efficient* algorithms with approximately the same guarantee as full AA. This task is called *tracking* the best expert. In this case this is achieved by the Fixed Share [26] predictor. It assigns a certain prior to all partition experts for which the exponentially many posterior weights collapse to N weights that can be maintained efficiently. Modifications of this algorithm achieve essentially the same bound for all T, B and N simultaneously [40, 29]. Now we present the update of this algorithm.

Fixed Share [26], in addition to a prior $\vec{u}_1$ requires a sequence of switching rates $\alpha_2, \alpha_3, \ldots$ Intuitively, α_t is the probability of a switch in the sequence of "best-at-the-step" experts *before* trial t. The weights are now updated as

$$u_{t+1}^n \;:=\; \frac{\alpha_{t+1}}{N-1} + \left(1 - \frac{N}{N-1}\alpha_{t+1}\right) \frac{u_t^n e^{-\ell_t^n}}{\sum_n u_t^n e^{-\ell_t^n}}\,. \tag{3.6}$$

(We see that the Aggregating Algorithm is the special case when all α_t are 0.) The tracking regret bound for Fixed Share with uniform prior $\vec{u}_1$ and constant $\alpha_t = \alpha$ switching rate states that for any reference sequence $j_1, \ldots, j_T$ of

experts with m blocks (and hence $m-1$ switches)

$$\sum_{t=1}^{T} \ell_t - \sum_{t=1}^{T} \ell_t^{j_t} \;\le\; \ln N + (m-1)\ln(N-1) - (m-1)\ln\alpha - (T-m)\ln(1-\alpha).$$

In an open problem Yoav Freund [9] asked whether there are algorithms that have small regret against *sparse* partition experts where the base experts allocated to the segments are from a small subset of N of the M experts. The Aggregating Algorithm when run on all such partition experts achieves regret $\ln\binom{M}{N} + \ln\binom{T-1}{B-1} + B\ln N$, but contrary to the non-sparse case, emulating this algorithm is NP-hard. However in a breakthrough paper, Bousquet and Warmuth in 2001 [9] gave the efficient MPP algorithm with only a slightly weaker regret bound. Like Fixed Share, MPP maintains M "posterior" weights, but it instead mixes in a bit of all past posteriors in each update. This causes weights of previously good experts to "glow" a little bit, even if they perform poorly locally. When the data later favours one of those good experts, its weight is pulled up quickly. However the term "posterior" is a misnomer because no Bayesian interpretation for this curious self-referential update was known. Understanding the MPP update is a very important problem because in many practical applications [22, 20][2] it significantly outperforms Fixed Share. Now we present the MPP update rule.

Mixing Past Posteriors [9] algorithm is parameterized by a so-called *mixing scheme*, which is a sequence $\gamma_1, \gamma_2, \ldots$ of distributions, each γ_t with support $\{0, \ldots, t-1\}$. MPP predicts with weights v_t^m defined recursively by

$$v_t^m := \sum_{s=0}^{t-1} \tilde{v}_{s+1}^m \, \gamma_t(s) \quad \text{where} \quad \tilde{v}_1^m := \frac{1}{N} \quad \text{and} \quad \tilde{v}_{t+1}^m := \frac{e^{-\ell_t^m} v_t^m}{\sum_n v_t^n e^{-\ell_t^n}}. \quad (3.7)$$

[2]The experiments reported in [22] are based on precursors of MPP. However MPP outperforms these algorithms in later experiments we have done on natural data for the same problem (not shown).

The auxiliary distribution $\tilde{v}_{t+1}(m)$ is formally the (incremental) posterior from prior $v_t(m)$. The predictive weights $v_t(m)$ are then the pre-specified γ_t mixture of all such past posteriors. Two special choices of γ_t are studied in the original paper by Bousqet and Warmuth: uniform past and decaying past. The former one yields better running time, while the latter results in a better bound. Our work eliminates this trade-off and gets the best of both worlds.

3.2 A closer look at Adaptive Regret

In this section we focus on the task of obtaining small *adaptive* regret, a notion first considered in [30] and later studied in [21]. The adaptive regret of an algorithm on a time interval $[t_1, t_2]$ is the loss of the algorithm there, minus the loss of the best expert for that interval:

$$R_{[t_1,t_2]} := L_{[t_1,t_2]} - \min_j L^j_{[t_1,t_2]}$$

The goal is now to ensure small regret on all intervals simultaneously. Note that adaptive regret was defined in [21]. There for a fixed time horizon T adaptive regret of an algorithm is defined as a maximum regret it achieves over any time interval within $[1, T]$. However, we need the fine-grained dependence on the endpoint times to be able to prove matching upper and lower bounds. It also appears in the context of online regression in [23] under the name of "local loss bound".

We started this work by studying the very natural way of obtaining adaptive algorithms using the specialist experts framework. However, the resulting algorithm turned out to be Fixed Share, a well-known *tracking* algorithm. Then we turned our attention to the alternative approach studied by Hazan et al. in [21]. However, the resulting algorithm (called Follow the Leading History) is also an instance of Fixed Share. So different attempts to create

an adaptive algorithm all resulted in the same old Fixed Share known for tracking but whose adaptive propertied were not studied before. Intrigued by this fact, we decided to study adaptive regret of Fixed Share thoroughly and found the exact expression for the worst-case regret. Furthermore, we were able to prove that *Fixed Share is the optimal adaptive algorithm*, in a sense that any other algorithm will be dominated by a certain instance of Fixed Share. The rest of this section is structured as follows. First, we describe our initial approach in section 3.2.1. Then we show how restarts approach from [21] is reduced to Fixed Share in section 3.2.3. Section 3.2.4 presents the worst-case regret of Fixed Share and the special cases for the specific choices of switching rate.

3.2.1 Specialist experts approach to adaptivity

Our way of getting an adaptive algorithm is the following. We create a pool of virtual specialist experts. For each real expert n and time t, we include a virtual specialist that sleeps for the first $t-1$ rounds and predicts as expert n from trial t onward. Then the classical regret w.r.t. this virtual expert on $[1,T]$ is the same as the adaptive regret w.r.t. the real expert n on $[t,T]$ because on the first $t-1$ trials the loss of the virtual expert equals Learner's loss. For fixed t_2, the uniform prior on wake-up time $t_1 \leq t_2$ and expert n would lead to adaptive regret $\ln(Nt_2)$. It turns out that the same holds even without knowledge of t_2.

At first glance, it is very inefficient to maintain weights of TN specialists with a growing T. However, we do not need to, since we may merge the weights of all awake specialists associated to the same real expert, resulting in Algorithm 8. To verify this, denote this merged (unnormalised) weight in trial t by v_t^n for each real expert n. The merged (unnormalised) weight v_{t+1}^n of this real expert n in the next trial $t+1$ consists of the prior weight of the newly awaken virtual specialist plus v_t^n, the sum of the old weights, each multiplied by the same factor $e^{\ell^t - \ell_t^n}$ (as they were all awake). Thus we can

update the sum directly, and this is reflected by our update rule.

Note that for simplicity, we have taken the prior on experts and wake-up times independent of the experts, i. e.

$$p^{(n,t)} \;=\; p(t)\,.$$

Here $p(t)$ plays the role of the distribution on the awake times. It is not normalized but for the predictions of algorithm on trials $[1,T]$ only the first T weights play role. Normalization happens at the prediction step on each trial. Thus if we multiply all the weights in the sequence $p^{(n,t)}$ by a constant the predictions remain unchanged.

Algorithm 8 Adaptive Aggregating Algorithm

Require: Prior nonnegative weights $p(t)$, $t = 1, 2, \ldots$

$v_1^n \coloneqq p(1), n = 1, \ldots, N$

for $t = 1, 2, \ldots$ **do**

Play weights $u_t^n \coloneqq \frac{v_t^n}{\sum_{k=1}^N v_t^k}$

Read the experts losses ℓ_t^n, $n = 1, \ldots, N$

Set $v_{t+1}^n \coloneqq p(t+1) + v_t^n \frac{e^{-\ell_t^n}}{\sum_{k=1}^N u_t^k e^{-\ell_t^k}}$, $n = 1, \ldots, N$

end for

3.2.2 Fixed Share recovered

Now we will see that Algorithm 8 turns out to be Fixed Share with variable switching rate. In the rest of this section we derive this. Let $P(t) = \sum_{s=1}^t p(s)$.

Fact 14. *The update step of Algorithm 8 preserves the following:*

$$\sum_n v_t^n = \sum_n \sum_{s \le t} p(s) = N P(t)\,.$$

Proof. This follows immediately from expanding the one-step update rule:

$$\begin{aligned}
\sum_n v_{t+1}^n &= \sum_n p(t+1) + \sum_n v_t^n \frac{e^{-\ell_t^n}}{\sum_k u_t^k e^{-\ell_t^k}} \\
&= \sum_n p(t+1) + \sum_n v_t^n \frac{e^{-\ell_t^n}}{\sum_k \frac{v_t^k}{\sum_j v_t^j} e^{-\ell_t^k}} \\
&= Np(t+1) + \sum_n v_t^n \overset{\text{Induction}}{=} NP(t+1)\,.
\end{aligned}$$

□

We now show that Algorithm 8 can be seen as Fixed Share (and vice versa).

Lemma 15. *Say that α_t is the probability of a Fixed Share switch before trial t, and $p(t)$ is the prior weight of specialist waking up in trial t in Algorithm 8. If $\alpha_t \in (0, \frac{N-1}{N}]$ for all t then the following conversion preserves behaviour:*

$$p(t) = \frac{\frac{N}{N-1}\alpha_t}{\prod_{s=2}^t (1 - \frac{N}{N-1}\alpha_s)}\,, \qquad \alpha_t = \frac{N-1}{N}\frac{p(t)}{\sum_{s=1}^t p(s)}\,,$$

where we use the convention that $\alpha_1 = \frac{N-1}{N}$.

Proof. Let us rewrite the update step of Algorithm 8 for the normalised weights.

$$\begin{aligned}
u_{t+1}^n &= \frac{v_{t+1}^n}{\sum_k v_{t+1}^k} = \frac{p(t+1)}{NP(t+1)} + \frac{1}{NP(t+1)} v_t^n \frac{e^{-\ell_t^n}}{\sum_k u_t^k e^{-\ell_t^k}} \\
&= \frac{p(t+1)}{NP(t+1)} + \frac{1}{NP(t+1)} NP(t) u_t^n \frac{e^{-\ell_t^n}}{\sum_k u_t^k e^{-\ell_t^k}} \\
&= \frac{\alpha_{t+1}}{N-1} + \frac{P(t+1) - p(t+1)}{P(t+1)} u_t^n \frac{e^{-\ell_t^n}}{\sum_k u_t^k e^{-\ell_t^k}} \\
&= \frac{\alpha_{t+1}}{N-1} + \left(1 - \frac{N}{N-1}\alpha_{t+1}\right) u_t^n \frac{e^{-\ell_t^n}}{\sum_k u_t^k e^{-\ell_t^k}}.
\end{aligned}$$

We see that the weight update is the update of the Fixed Share algorithm with variable switching rate α_t. □

Note about big α_t. For $\alpha_t \in (\frac{N-1}{N}, 1]$ the conversion is not possible. Later we will see that these algorithms are not effective in terms of adaptive regret and are dominated by Fixed Share with $\alpha_t = \frac{N-1}{N}$. The intuitive reason is as follows. $\alpha_t = \frac{N-1}{N}$ is equivalent to forgetting about the past at trial t (predicting with the uniform distribution no matter what the past losses were). Even bigger α_t means favouring the switch, that is putting less weights on experts that were good in the past. This is clearly an "anti-adaptive" behaviour. Sometimes (see for example, [27]) Fixed Share is presented using an alternative parametrization β_t allowing "self switching". Then $\alpha_t = \frac{N-1}{N}\beta_t$ and the conversion could be done for any $\beta_t \in [0, 1]$. However we will follow the original notation of [26]. In what follows we assume that $\alpha \in [0, \frac{N-1}{N}]$.

3.2.3 Restarts approach to adaptivity

A second intuitive method to obtain adaptive regret bounds, called Follow the Leading History (FLH), was introduced in [21]. One starts with a base algorithm that ensures low classical regret. FLH then obtains low adaptive regret by restarting a copy of this base algorithm each trial, and aggregating the predictions of these copies. To get low adaptive regret w.r.t. N experts, it is natural to take the AA as the base algorithm. We now show that FLH with this choice equals Fixed Share with switching rate $\alpha_t = \frac{N-1}{Nt}$.

For each n, s and $t \geq s$, let $p_t^{n|s}$ denote the weight allocated to expert n by the copy of the AA started at time s. By definition $p_s^{n|s} = 1/N$, and these weights evolve according to (3.1b). We denote by p_t^s the weight allocated by FLH in trial $t \geq s$ to the copy of AA started at time s. In [21], these weights

are defined as follows. Initially $p_1^1 = 1$ and subsequently

$$p_{t+1}^{t+1} \;=\; \frac{1}{t+1}, \qquad p_{t+1}^{s} \;=\; \left(1 - p_{t+1}^{t+1}\right) \frac{p_t^s e^{-\left(-\ln \sum_n p_t^{n|s} e^{-\ell_t^n}\right)}}{\sum_{r=1}^{t} p_t^r e^{-\left(-\ln \sum_n p_t^{n|r} e^{-\ell_t^n}\right)}}.$$

We now show that this construction is a reparametrisation of Fixed Share. In fact, this is true for any choice of the restart probabilities p_t^t.

Lemma 16. *For mix loss, FLH with AA as the base algorithm issues the same predictions as Fixed Share with learning rate* $\alpha_t = \frac{N-1}{N} p_t^t$.

Proof. We prove by induction on t that the FS and FLH weights coincide:

$$u_t^n \;=\; \sum_{s=1}^{t} p_t^{n|s} p_t^s.$$

The base case $t = 1$ is obvious. For the induction step we expand

$$\begin{aligned}
\sum_{s=1}^{t+1} p_{t+1}^{n|s} p_{t+1}^s \;&=\; \sum_{s=1}^{t} p_{t+1}^{n|s} p_{t+1}^s + p_{t+1}^{t+1}/N \\
&=\; \left(1 - p_{t+1}^{t+1}\right) \sum_{s=1}^{t} \left(\frac{p_t^{n|s} e^{-\ell_t^n}}{\sum_n p_t^{n|s} e^{-\ell_t^n}} \frac{p_t^s \left(\sum_n p_t^{n|s} e^{-\ell_t^n}\right)}{\sum_{r=1}^{t} p_t^r \left(\sum_n p_t^{n|r} e^{-\ell_t^n}\right)} \right) + \frac{1}{N} p_{t+1}^{t+1} \\
&=\; \left(1 - p_{t+1}^{t+1}\right) \frac{\sum_{s=1}^{t} p_t^s p_t^{n|s} e^{-\ell_t^n}}{\sum_{r=1}^{t} \sum_n p_t^r p_t^{n|r} e^{-\ell_t^n}} + \frac{1}{N} p_{t+1}^{t+1} \\
&\overset{\text{Induction}}{=}\; \left(1 - p_{t+1}^{t+1}\right) \frac{u_t^n e^{-\ell_t^n}}{\sum_n u_t^n e^{-\ell_t^n}} + \frac{1}{N} p_{t+1}^{t+1} \;=\; u_{t+1}^n,
\end{aligned}$$

and find the Fixed Share update equation (3.6) for switching rate $\alpha_t = \frac{N-1}{N} p_t^t$. □

3.2.4 Fixed Share worst-case adaptive regret

We have seen in the previous section that both intuitive approaches to obtain algorithms with low adaptive regret result in Fixed Share. We take this convergence to mean that Fixed Share is the most fundamental adaptive algorithm. The tracking regret for Fixed Share is already well-studied. In this section we thoroughly analyse the adaptive regret of Fixed Share. We obtain the worst-case adaptive regret for mix loss. This result implies the known tracking regret bounds.

We also show an information-theoretic lower bound for mix loss that must hold for any algorithm, and which is tight for Fixed Share. This proves that Fixed Share is a Pareto-optimal algorithm for the mix loss game, in the sense that no other algorithm can guarantee essentially better adaptive regret.

The exact worst-case adaptive regret for mix loss. In this section we first compute the exact worst-case adaptive regret of Fixed Share with arbitrary switching rate α_t. Then we obtain certain regret bounds of interest, including the tracking regret bound, for particular choices of α_t.

Theorem 17. *The worst-case adaptive regret of Fixed Share with N experts on interval $[t_1, t_2]$ equals*

$$-\ln\left(\frac{\alpha_{t_1}}{N-1}\prod_{t=t_1+1}^{t_2}(1-\alpha_t)\right). \tag{3.8}$$

Proof. The proof consists of two parts. First we claim that the worst-case data for the interval $[t_1, t_2]$ in the setting of Protocol 2 is rather simple: on the interval there is one *good* expert (all others get infinite losses) and on the single trial before the interval (if $t_1 > 1$) this expert suffers infinite loss while others do not. The proof of this can be found in Appendix B.

Now we will compute the regret on this data. The regret of Fixed Share on the interval $[t_1, t_2]$ is $-\ln$ of the product of the weights put on the good

expert (say, j) on this interval:

$$R^{\mathrm{FS}}_{[t_1,t_2]} = -\ln \prod_{t_1 \le t \le t_2} u_t^j \,.$$

It is straightforward to derive $u^j_{t_1}$ from (3.6):

$$u^j_{t_1} \;=\; \frac{\alpha_{t_1}}{N-1} \qquad \text{and} \qquad u^j_t \;=\; 1-\alpha_t \qquad \text{for } t \in [t_1+1, t_2]$$

from which the statement follows. □

Example 1: constant switching rate. This is the original Fixed Share [26].

Corollary 18. *Fixed Share with constant switching rate $\alpha_t = \alpha$ for $t > 1$ (recall that $\alpha_1 = \frac{N-1}{N}$) has worst-case adaptive regret equal to*

$$\begin{aligned} &\ln(N-1) - \ln\alpha - (t_2-t_1)\ln\bigl(1-\alpha\bigr) && \text{for } t_1 > 1, \text{ and} \\ &\ln N - (t_2-1)\ln\bigl(1-\alpha\bigr) && \text{for } t_1 = 1. \end{aligned}$$

A slightly weaker upper bound was obtained in [12]. The clear advantage of our analysis with equality is that we can obtain the standard Fixed Share tracking regret bound by summing the above adaptive regret bounds on individual intervals. Comparing with the best sequence of experts S on the interval $[1,T]$ with m blocks, we obtain the bound

$$L^{\mathrm{FS}}_{[1,T]} - L^S_{[1,T]} \;\le\; \ln N + (m-1)\ln(N-1) - (m-1)\ln\alpha - (T-m)\ln(1-\alpha)\,,$$

which is exactly the Fixed Share standard bound. So we see that the reason why Fixed Share can effectively compete with switching sequences is that it can, in fact, effectively compete with an expert on any interval, that is, has small adaptive regret.

Example 2: slowly decreasing switching rate. The idea of slowly decreasing the switching rate was considered in [38] in the context of source coding, and later analysed for expert switching in [29]; we saw in Section 3.2.3 that it also underlies Follow the Leading History of [21]. It results in tracking regret bounds that are almost as good as the bounds for constant α *with optimally tuned* α. These tracking bounds are again implied by the following corresponding adaptive regret bound.

Corollary 19. *Fixed Share with switching rate* $\alpha_t = 1/t$ *(except for* $\alpha_1 = \frac{N-1}{N}$*) has worst-case adaptive regret*

$$-\ln\left(\frac{1}{(N-1)t_1}\prod_{t=t_1+1}^{t_2}\frac{t-1}{t}\right) = \ln(N-1)+\ln t_2 \qquad \text{for } t_1 > 1, \text{ and} \tag{3.9a}$$

$$-\ln\left(\frac{1}{N}\prod_{t=2}^{t_2}\frac{t-1}{t}\right) = \ln N + \ln t_2 \qquad \text{for } t_1 = 1. \tag{3.9b}$$

Example 3: quickly decreasing switching rate. The bounds we have obtained so far depend on t_2 either linearly or logarithmically. To get bounds that depend on t_2 sub-logarithmically, or even not at all, one may instead decrease the switching rate faster than $1/t$, as analysed in [38, 27]. To obtain a controlled trade-off, we consider setting the switching rate to $\alpha_t = \frac{1}{t\ln t}$, except for $\alpha_1 = \frac{N-1}{N}$. This leads to adaptive regret at most

$$\begin{aligned}\ln(N-1)+\ln t_1+\ln\ln t_1 - \sum_{t=t_1+1}^{t_2}\ln\left(1-\frac{1}{t\ln t}\right)& \\ \leq \ln(N-1)+\ln t_1+\ln\ln t_2+1.28& \end{aligned} \tag{3.10a}$$

when $t_1 > 1$ and

$$\ln N - \sum_{t=2}^{t_2} \ln\left(1 - \frac{1}{t \ln t}\right) \;\leq\; \ln N + \ln\ln t_2 + 1.65 \tag{3.10b}$$

when $t_1 = 1$ (remember that $\ln\ln 1$ is understood to be 0). The constant 1.28 in (3.10a) is needed because t_1 and t_2 can take small values; e.g., if we only consider $t_1 \geq 10$, we can replace 1.28 by 0.05, and we can replace 1.28 by an arbitrarily small $\delta > 0$ if we only consider $t_1 \geq c$ for a sufficiently large c.

The dependence on t_2 in (3.10) is extremely mild. We can suppress it completely by increasing the dependence on t_1 just ever so slightly. If we set $\alpha_t = t^{-1-\epsilon}$, where $\epsilon > 0$, then the sum of the series $\sum_{t=1}^{\infty} \alpha_t$ is finite and the bound becomes

$$\ln(N-1) + (1+\epsilon)\ln t_1 + c_\epsilon \qquad \text{for } t_1 > 1, \text{ and} \tag{3.11a}$$

$$\ln N + c_\epsilon \qquad \text{for } t_1 = 1, \tag{3.11b}$$

where $c_\epsilon = -\sum_{t=2}^{\infty} \ln(1 - t^{-1-\epsilon})$. It is clear that the bound (3.11a) is far from optimal when t_1 is large: c_ϵ can be replaced by a quantity that tends to 0 as $O(t_1^{-\epsilon})$ as $t_1 \to \infty$. In particular, for $\epsilon = 1$ we have the bound

$$\ln N + 2\ln t_1 + \ln 2.$$

An interesting feature of this switching rate is that for the full interval $[t_1, t_2] = [1, T]$ the bound differs from the standard AA bound only by an additive term less than 1. In words, the overhead for small adaptive regret is negligible.

3.2.5 Fixed Share is the optimal adaptive algorithm

We started by considering several intuitive constructions for adaptive algorithms, and saw that they all result in Fixed Share. We then obtained the worst-case adaptive regret of Fixed Share. Intuitive as it may be, we have not answered the question whether Fixed Share is a *good* algorithm in the sense that its worst-case adaptive regret bounds are *small*. It is conceivable that there are smarter algorithms with better adaptive regret guarantees. See for example the palette of tracking algorithms by [29]. And even if no better algorithms exist, there may still be algorithms that exhibit different trade-offs, in the sense that their worst-case adaptive regret is incomparable to that of Fixed Share.

So in this section we start from the other end and derive lower bounds that hold for any algorithm. As expected, we conclude that the bounds of Fixed Share (with any switching rate sequence α_t) are Pareto optimal. But it came to us as a shocking surprise that actually *all other bounds are strictly dominated*. No matter how smart the algorithm; its worst-case adaptive regret will be dominated by that of an instance of Fixed Share.

To keep the notation in this section simple, we consider the number N of experts fixed. We call a mapping ϕ of intervals to losses a *candidate guarantee*. Such a candidate guarantee is *realisable* if there is an algorithm for mix loss prediction (Protocol 2) with adaptive regret at most ϕ. That is, we demand

$$R_{[t_1,t_2]} \;\leq\; \phi(t_1,t_2)$$

for all sequences of expert losses $\vec{\ell}_1, \vec{\ell}_2, \ldots$ in $[-\infty,\infty]^N$ and all choices of the interval $1 \leq t_1 \leq t_2$. We say that a realisable guarantee ϕ is *Pareto optimal* if there is no realisable ϕ that is strictly better, i.e. $\leq$ on all intervals and $<$ on at least one interval.

We are interested in Pareto-optimal guarantees. Every such guarantee is, by definition, the worst-case regret of an algorithm. Let us make that

precise:

Definition 20. We say that ϕ is the *worst-case adaptive regret* of a given algorithm if

$$\phi(t_1, t_2) \;=\; \sup_{\vec{\ell}_1, \vec{\ell}_2, \ldots} R_{[t_1,t_2]} \qquad \text{for all } 1 \le t_1 \le t_2.$$

Note that the worst-case regret may not be attained.

The main step in characterising the Pareto optimal guarantees is showing that worst-case adaptive regrets can not be too small.

Theorem 21. *Let ϕ be the worst-case adaptive regret of some algorithm. Then*

$$\phi(t,t) \;\ge\; \ln N \qquad \textit{for all } 1 \le t \textit{ and} \tag{3.12a}$$

$$\phi(t_1,t_2) \;\ge\; \phi(t_1,t_1) + \sum_{t=t_1+1}^{t_2} -\ln\bigl(1-(N-1)e^{-\phi(t,t)}\bigr) \qquad \textit{for all } 1 \le t_1 < t_2. \tag{3.12b}$$

Proof. Suppose (3.12a) fails for some t. Since at any point the N weights must sum to at most one, the smallest weight must be $\le 1/N$. Hence by hitting all others with infinite loss we can force loss at least $\ln N$. Now suppose (3.12a) holds throughout, and consider any interval $[t_1, t_2]$. Fix $\epsilon > 0$. Let $\vec{\ell}_1, \ldots, \vec{\ell}_{t_1}$ be data on which the worst-case regret exceeds $\phi(t_1,t_1) - \epsilon$. Let i^* be the (any) best expert in trial t_1. Now for all $t \in (t_1, t_2]$ choose $\ell_t^{i^*} = 0$ and $\ell_t^j = \infty$ for all $j \ne i^*$. In any trial t, the algorithm must allocate at least weight $e^{-\phi(t,t)}$ to each expert to guarantee ϕ on the singleton interval $\{t\}$. Since the weights sum to one, the weight allocated to expert i^* during each trial $t \in (t_1, t_2]$ can be at most $1-(N-1)e^{-\phi(t,t)}$. Hence the loss of the algorithm must be at least $-\ln\bigl(1-(N-1)e^{-\phi(t,t)}\bigr)$. Since the loss of

the best expert i^* on $(t_1, t_2]$ is zero, the regret is at least the right-hand side of (3.12b) minus ϵ. The worst-case regret $\phi(t_1, t_2)$ must be at least as large. Since this holds for all ϵ, we proved (3.12b). □

A realisable guarantee is witnessed by some algorithm, and is therefore dominated by the worst-case adaptive regret of that algorithm. We proved that this worst-case adaptive regret must satisfy (3.12). We now show that any guarantee satisfying (3.12) is realised by an instance of Fixed Share.

Theorem 22. *Let ϕ satisfy* (3.12)*. Then Fixed Share with switching rate sequence $\alpha_2, \alpha_3, \ldots$ where $\alpha_t = (N-1)e^{-\phi(t,t)}$ guarantees ϕ.*

Proof. From (3.12a) we know that $\alpha_t \leq (N-1)/N$, so the worst case regret of Fixed Share is given by Theorem 17. In particular (3.8) with our choice of α_t equals the right-hand side of (3.12b), and so FS guarantees ϕ. Note the fact that α_1 is always set to the specific value $(N-1)/N$ only works in our favour here. □

By combining the preceding two theorems, we obtain canonical representatives of the Pareto guarantees for adaptive regret.

Corollary 23. *Let ϕ be a candidate guarantee. The following are equivalent:*

- *ϕ is realisable*
- *$\exists \psi \leq \phi$ such that ψ satisfies* (3.12)
- *ϕ is dominated by the worst-case adaptive regret of a Fixed Share.*

And more importantly, the following are also equivalent:

- *ϕ is Pareto optimal*
- *ϕ satisfies* (3.12a)*, with equality for $t = 1$, and* (3.12b) *with equality throughout.*
- *ϕ is the worst-case adaptive regret of a Fixed Share.*

3.3 Mixing Past Posteriors demystified

Now we turn our attention towards more complex virtual specialists. Namely we provide a specialist experts explanation for Mixing Past Posteriors (MPP), an effective but somewhat mysterious algorithm by Bousquet and Warmuth [9]. In this section we craft the specialist experts and the prior on them to recover MPP. As Fixed Share is the special case of MPP (corresponding to a specific mixing scheme) it should come as no surprise that the construction here is more general then the Fixed Share one. Namely, now the experts are allowed to fall asleep after they wake up. Actually, we consider all possible wake-up sequences and create a virtual expert for every pair of such wake-up sequence and a real expert.

3.3.1 Construction

Each *partition specialist* (χ, m) is parameterized by an expert index m and a *circadian* (wake/sleep pattern) $\chi = (\chi_1, \chi_2, \ldots)$ with $\chi_t \in \{\mathsf{w}, \mathsf{s}\}$. We use infinite circadians in order to obtain algorithms that do not depend on a time horizon. The wake set W_t includes all partition specialists that are awake at time t, i.e. $W_t \coloneqq \{(\chi, m) \mid \chi_t = \mathsf{w}\}$. An awake specialist (χ, m) in W_t suffers the same loss as the base expert m, i.e. $l_t^{(\chi,m)} = l_t^m$ if $\chi_t = w$. The algorithm is now fully specified by choosing a prior on partition specialists. Here we enforce the independence $P(\chi, m) \coloneqq P(\chi)P(m)$ and define $P(m) \coloneqq 1/M$ uniform on the base experts. We now can apply Theorem 13 to bound the regret w.r.t. any partition expert with time horizon T by decomposing it into N partition specialists $(\chi^1_{\leq T}, \hat{m}^1), \ldots, (\chi^N_{\leq T}, \hat{m}^N)$ and choosing $U(\cdot) = 1/N$ uniform on these specialists:

$$\mathcal{R} \;\leq\; N \ln \frac{M}{N} + \sum_{n=1}^{N} - \ln P(\chi^n_{\leq T}). \tag{3.13}$$

The overhead of selecting N reference experts from the pool of size M closely approximates the information-theoretic ideal $N \ln \frac{M}{N} \approx \ln \binom{M}{N}$. This improves previous regret bounds [9, 1, 12] by an additive $N \ln N$. Next we consider two choices for $P(\chi)$: one for which we retrieve MPP, and a natural one which leads to efficient algorithms and sharper bounds.

3.3.2 MPP recovered

To see how one might be able to build a prior corresponding to the weighting schemes of Mixing Past Posteriors we expand the MPP weights 3.7:

$$v_t^n = v_0^n \sum_{m=1}^{t} \sum_{1 \le s_1 \le s_2 \le \cdots \le s_m = t} \gamma_{s_1}(0) \prod_{q=2}^{m} \gamma_{s_q}(s_{q-1}) \frac{e^{-\ell^n_{s_{q-1}}}}{\sum_n e^{-\ell^n_{s_{q-1}}} v^n_{s_{q-1}}} \tag{3.14}$$

where $s_0 = 0$.

For the specialist algorithm the weight corresponding to n-th real expert on trial t is obtained by summing the weights of the awake specialists corresponding to n-th expert normalized over the weight of the wake set.

$$u_t^n = \frac{1}{\sum_{W_t} u_t^{(\chi,n)}} \sum_{\chi_t = \mathsf{w}} u_0^{(\chi,n)} \prod_{s<t, \chi_s = \mathsf{w}} \frac{e^{-\ell^n_s}}{\sum_n e^{-\ell^n_s} u^n_s} \tag{3.15}$$

To match 3.15 and 3.14 we build a prior by only putting weights on the sequences with finite number of w-s in the following manner. Let m be the number of w-s in χ and $s_i, i = 1, \ldots, m$ index their positions in χ. Then,

$$u_0^{(\chi,n)} = \frac{1}{N} P(\chi) = \frac{1}{N} P_0(s_m) \prod_{q=1}^{m} \gamma_{s_q}(s_{q-1}) \tag{3.16}$$

where as before $s_0 = 0$ and $P_0(s_m)$ is any distribution on $\mathbb{N}$ with full support. Now we state several facts about this prior.

Fact 24. *$P(\chi)$ is the probability of a path χ in the following Markov chain. The chain has states $S_0, S_\infty, S_1, S_2, \ldots$. It starts at S_∞ and the transition probabilities are defined as follows.*

$$P(S_\infty, S_n) = P(n),$$
$$P(S_i, S_j) = \begin{cases} \gamma_i(j), \text{ if } i > j \\ 0, \text{otherwise.} \end{cases}$$

Fact 25. *$P(\chi_{<t}|\chi_t = \mathsf{w}) = \prod_{q=1}^{k} \gamma_{s_q}(s_{q-1})$, where $s_k = t$.*

Now we can prove by induction that the weights match. Expanding 3.15 with the constructed prior we get, using the Markovian property,

$$\begin{aligned} u_t^n &= \frac{u_0^n}{\sum_{W_t} u_t^{(\chi,n)}} \sum_{\chi_t=\mathsf{w}} P(\chi_t = \mathsf{w}) P(\chi_{<t}|\chi_t = \mathsf{w}) P(\chi_{>t}|\chi_t = \mathsf{w}) \prod_{s<t, \chi_s=\mathsf{w}} \frac{e^{-\ell_s^n}}{\sum_n e^{-\ell_s^n} u_s^n} \\ &= \frac{u_0^n}{\sum_{W_t} u_t^{(\chi,n)}} \sum_{\chi_t=\mathsf{w}} P(\chi_t = \mathsf{w}) \prod_{s_q<t, s.t. \chi_{s_q}=\mathsf{w}} \gamma_{s_q}(s_{q-1}) \prod_{s<t, \chi_s=\mathsf{w}} \frac{e^{-\ell_s^n}}{\sum_n e^{-\ell_s^n} u_s^n} \\ &= u_0^n \frac{P(\chi_t = \mathsf{w})}{\sum_{W_t} u_t^{(\chi,n)}} \sum_{m=1}^{t} \sum_{1 \le s_1 \le s_2 \le \cdots \le s_m = t} \gamma_{s_1}(0) \prod_{q=2}^{m} \gamma_{s_q}(s_{q-1}) \frac{e^{-\ell_{s_{q-1}}^n}}{\sum_n e^{-\ell_{s_{q-1}}^n} u_{s_{q-1}}^n} \end{aligned}$$

Now we apply the induction hypothesis and note that the weights coincide up to the normalization, which must, therefore, be 1.

This proves the following theorem.

Theorem 26. *For any mixing scheme $\gamma_1, \gamma_2, \ldots$ there is a prior (defined by 3.16) on the virtual specialist experts such that the prediction of MPP is equivalent to the prediction of The Aggregating Algorithm for specialists.*

3.3.3 A simple Markov chain circadian prior

In the previous section we recovered circadian priors corresponding to the MPP mixing schemes. Here we design priors afresh from first principles. Our goal is efficiency and good regret bounds. A simple and intuitive choice for prior $P(\chi)$ is a Markov chain on states $\{\mathsf{w}, \mathsf{s}\}$ with initial distribution $\theta(\cdot)$ and transition probabilities $\theta(\cdot|\mathsf{w})$ and $\theta(\cdot|\mathsf{s})$, that is

$$P(\chi_{\leq t}) \;\coloneqq\; \theta(\chi_1) \prod_{s=2}^{t} \theta(\chi_s|\chi_{s-1}). \tag{3.17}$$

By choosing low transition probabilities we obtain a prior that favors temporal locality in that it allocates high probability to circadians that are awake and asleep in contiguous segments. Thus if a good sparse partition expert exists for the data, our algorithm will pick up on this and predict well.

The resulting strategy (aggregating infinitely many specialists) can be executed efficiently.

Theorem 27. *The prediction of AA with Markov prior* (3.17) *is equal to the prediction* v_t^m *of Algorithm 9, which can be computed in* $\mathcal{O}(M)$ *time per outcome using* $\mathcal{O}(M)$ *space.*

Proof. We prove that the update of the algorithm reflects the fact that $v_t(b, m) = \sum_{\chi_t = b} u_t^{(\chi, m)}$ for each expert m and $b \in \{\mathsf{w}, \mathsf{s}\}$. The base case

$t = 1$ is automatic. For $t > 1$ step we expand

$$\begin{aligned} v_{t+1}(b,m) &= \sum_{\chi_{t+1}=b} u_{t+1}^{(\chi,m)} = \sum_{\chi_t=w,\chi_{t+1}=b} u_{t+1}^{(\chi,m)} + \sum_{\chi_t=s,\chi_{t+1}=b} u_{t+1}^{(\chi,m)} \\ &= \sum_{\chi_t=w,\chi_{t+1}=b} u_t^{(\chi,m)} e^{\ell_t - \ell_t^m} + \sum_{\chi_t=s,\chi_{t+1}=b} u_t^{(\chi,m)} \\ &= \sum_{\chi_t=w} u_t^{(\chi,m)} \theta(b|\mathsf{w}) e^{\ell_t - \ell_t^m} + \sum_{\chi_t=s} u_t^{(\chi,m)} \theta(b|\mathsf{s}) \\ &= v_t(\mathsf{w},m)\theta(b|\mathsf{w}) e^{\ell_t - \ell_t^m} + v_t(\mathsf{s},m)\theta(b|\mathsf{s}) \end{aligned}$$

□

Algorithm 9 AA with Markov circadian prior (3.17) (for Freund's problem)

Input: Distributions $\theta(\cdot)$, $\theta(\cdot|\mathsf{w})$ and $\theta(\cdot|\mathsf{s})$ on $\{\mathsf{w},\mathsf{s}\}$.
Initialize $v_1(b,m) \coloneqq \theta(b)/M$ for each expert m and $b \in \{\mathsf{w},\mathsf{s}\}$
for $t = 1, 2, \ldots$ **do**
  Predict with $v_t^m = \frac{v_t(\mathsf{w},m)}{\sum_{m'=1}^M v_t(\mathsf{w},m')}$
  Observe losses ℓ_t^n and suffer mix-loss $\ell_t = -\ln \sum_m v_t^m e^{-\ell_t^m}$.
  Update $v_{t+1}(b,m) \coloneqq \theta(b|\mathsf{w}) e^{l_t - l_t^m} v_t(\mathsf{w},m) + \theta(b|\mathsf{s}) v_t(\mathsf{s},m)$.
end for

The previous theorem establishes that we can predict fast. Next we show that we predict well.

Theorem 28. *Let $\hat{m}_1, \ldots, \hat{m}_T$ be an N-sparse assignment of M experts to T times with B segments. The regret of Algorithm 9 with tuning $\theta(\mathsf{w}) = 1/N$, $\theta(\mathsf{s}|\mathsf{w}) = \frac{B-1}{T-1}$ and $\theta(\mathsf{w}|\mathsf{s}) = \frac{B-1}{(N-1)(T-1)}$ is at most*

$$\begin{aligned} \mathcal{R} \;\le\; & N \ln \frac{M}{N} + N\,\mathcal{H}\left(\frac{1}{N}\right) + (T-1)\,\mathcal{H}\left(\frac{B-1}{T-1}\right) \\ & + (N-1)(T-1)\,\mathcal{H}\left(\frac{B-1}{(N-1)(T-1)}\right), \end{aligned}$$

where $\mathcal{H}(p) \coloneqq -p\ln(p) - (1-p)\ln(1-p)$ *is the binary entropy function.*

Proof. Without loss of generality assume $\hat{m}_t \in \{1, \dots, N\}$. For each expert n pick circadian $\chi^n_{\leq T}$ with $\chi^n_t = \mathsf{w}$ iff $\hat{m}_t = n$. Expanding the definition of the prior (3.17) we find

$$\prod_{n=1}^{N} P(\chi^n_{\leq T}) \;=\; \theta(\mathsf{w})\theta(\mathsf{s})^{N-1}\theta(\mathsf{s}|\mathsf{s})^{(N-1)(T-1)-(B-1)}\theta(\mathsf{w}|\mathsf{w})^{T-B}\theta(\mathsf{w}|\mathsf{s})^{B-1}\theta(\mathsf{s}|\mathsf{w})^{B-1},$$

which is in fact maximized by the proposed tuning. The theorem follows from (3.13). □

The information-theoretic ideal regret is $\ln\binom{M}{N} + \ln\binom{T-1}{B-1} + B\ln N$. Theorem 28 is very close to this except for a factor of 2 in front of the middle term; since $n\,\mathcal{H}(k/n) \leq k\ln(n/k) + k$ we have

$$\mathcal{R} \;\leq\; N\ln\frac{M}{N} + 2(B-1)\ln\frac{T-1}{B-1} + B\ln N + 2B.$$

The origin of this factor remained a mystery in [9], but becomes clear in our analysis: it is the *price of coordination* between the specialists that constitute the best partition expert. To see this, let us regard a circadian as a sequence of wake/sleep transition times. With this viewpoint (3.13) bounds the regret by summing the prior costs of all the reference wake/sleep transition times. This means that we incur overhead at each segment boundary of the comparator *twice*: once as the sleep time of the preceding expert, and once more as the wake time of the subsequent expert.

In practice the comparator parameters T, N and B are unknown. This can be addressed by standard orthogonal techniques. Of particular interest is the method inspired by [38, 29, 27] of changing the Markov transition probabilities as a function of time. It can be shown that by setting $\theta(\mathsf{w}) = 1/2$ and increasing $\theta(\mathsf{w}|\mathsf{w})$ and $\theta(\mathsf{s}|\mathsf{s})$ as $\exp(-\frac{1}{t\ln^2(t+1)})$ we keep the update time and space of the algorithm at $\mathcal{O}(M)$ and guarantee regret bounded for all T,

N and B as

$$\mathcal{R} \;\leq\; N \ln \frac{M}{N} + 2N + 2(B-1)\ln T + 4(B-1)\ln\ln(T+1).$$

At no computational overhead, this bound is remarkably close to the fully tuned bound of the theorem above, especially when the number of segments B is modest as a function of T.

3.4 Online Sparse Multitask Learning

Multitask learning [10] is an approach of learning several problems together, with the hope of exploiting similarities among them. We transition to an extension of the sequential prediction setup called online multitask learning [1, 35, 6, 31, 11, 36, 24]. The new ingredient in the online protocol is that before predicting in trial t we are given its *task number* $\kappa_t \in \{1, \ldots, K\}$. As before, we have access to M experts. If a single expert performs well on several tasks we want to figure this out quickly and exploit it. Simply ignoring the task number would not result in an adaptive algorithm. Applying a separate Bayesian predictor to each task independently would not result in any inter-task synergy. Nevertheless, it would guarantee regret at most $K \ln M$ overall. Now suppose each task is predicted well by some expert from a small subset of experts of size $N \ll M$. Running AA on all N-sparse allocations would achieve regret $\ln \binom{M}{N} + K \ln N$. However, emulating AA in this case is NP-hard [35]. The goal is to design efficient algorithms with approximately the same regret bound.

In [1] this multiclass problem is reduced to MPP, giving regret bound $N \ln \frac{M}{N} + B \ln N$. Here B is the number of same-task segments in the task sequence $\kappa_{\leq T}$. When all trials with the same task number are consecutive, i.e. $B = K$, then the desired bound is achieved. However the tasks may be interleaved, making the number of segments B much larger than K. We now eliminate the dependence on B, i.e. we solve a key open problem of [1].

We apply the method of specialists to multitask learning, and obtain regret bounds close to the information-theoretic ideal, which in particular do not depend on the task segment count B at all.

3.4.1 Construction

We create a *subset specialist* (S, m) for each basic expert index m and subset of tasks $S \subseteq \{1, \ldots, K\}$. At time t, specialists with the current task κ_t in their set S are awake, i.e. $W_t \coloneqq \{(S, m) \mid \kappa_t \in S\}$, and suffer the loss ℓ_t^m of expert m. We assign to subset specialist (S, m) prior probability $P(S, m) \coloneqq P(S)P(m)$ where $P(m) \coloneqq 1/M$ is uniform, and $P(S)$ includes each task independently with some fixed bias $\sigma(\mathsf{w})$

$$P(S) \;\coloneqq\; \sigma(\mathsf{w})^{|S|}\sigma(\mathsf{s})^{K-|S|}. \tag{3.18}$$

This construction has the property that the product of prior weights of two loners $(\{\kappa_1\}, \hat{m})$ and $(\{\kappa_2\}, \hat{m})$ is dramatically lower than the single pair specialist $(\{\kappa_1, \kappa_2\}, \hat{m})$, especially so when the number of experts M is large or when we consider larger task clusters. By strongly favoring it in the prior, any inter-task similarity present will be picked up fast.

3.4.2 Implementation

In this section we show that the resulting strategy involving $M2^K$ subset specialists can be implemented efficiently.

Theorem 29. *The predictions of AA with the set prior* (3.18) *equal the predictions* v_t^m *of Algorithm 10. They can be computed in* $\mathcal{O}(M)$ *time per outcome using* $\mathcal{O}(KM)$ *storage.*

Of particular interest is Algorithm 10's update rule for $f_{t+1}^{\kappa}(m)$. This would be a regular Bayesian posterior calculation if $v_t(m)$ were replaced by

$f_t^\kappa(m)$. In fact, $v_t(m)$ is the communication channel by which knowledge about the performance of expert m in other tasks is received.

Proof. The resource analysis follows from inspection, noting that the update is fast because only the weights $f_t^\kappa(m)$ associated to the current task κ are changed. We prove by induction on t that $v_t(m) = \sum_{S \ni \kappa_t} u_t^{(S,m)}$. In the base case $t = 1$ both equal $1/M$. For the induction step we expand $\sum_{S \ni \kappa_{t+1}} u_{t+1}^{(S,m)}$, which is by definition proportional to

$$\sum_{S \ni \kappa_{t+1}} \frac{1}{M} \sigma(\mathsf{w})^{|S|} \sigma(\mathsf{s})^{K-|S|} \left(\prod_{q \le t \,:\, \kappa_q \in S} e^{-\ell_q^m} \right) \left(\prod_{q \le t \,:\, \kappa_q \notin S} e^{-\ell_q} \right). \tag{3.19}$$

The product form of both set prior and likelihood allows us to factor this exponential sum of products into a product of binary sums. It follows from the induction hypothesis that

$$f_t^k(m) \;=\; \frac{\sigma(\mathsf{w})}{\sigma(\mathsf{s})} \prod_{q \le t \,:\, \kappa_q = k} e^{\ell_q - \ell_q^m}$$

Then we can divide (3.19) by $e^{-L_{t+1}} \sigma(\mathsf{s})^K$ and reorganize to

$$\frac{1}{M} f_t^{\kappa_{t+1}}(m) \prod_{k \neq \kappa_{t+1}} \left(f_t^k(m) + 1 \right) \;=\; \frac{1}{M} \frac{f_t^{\kappa_{t+1}}(m)}{f_t^{\kappa_{t+1}}(m) + 1} \prod_{k=1}^{K} \left(f_t^k(m) + 1 \right)$$

Since the algorithm maintains $\pi_t(m) = \prod_{k=1}^K (f_t^k(m) + 1)$ this is proportional to $v_{t+1}(m)$. □

The Bayesian strategy is hence emulated fast by Algorithm 10. We now show it predicts well.

Theorem 30. *Let $\hat{m}_1, \ldots, \hat{m}_K$ be an N-sparse allocation of M experts to K tasks. With tuned inclusion rate $\sigma(\mathsf{w}) = 1/N$, the regret of Bayes (Algo-*

Algorithm 10 Bayes with set prior (3.18) (for online multitask learning)

Input: Number of tasks $K \geq 2$, distribution $\sigma(\cdot)$ on $\{\mathsf{w}, \mathsf{s}\}$.
Initialize $f_1^k(m) := \frac{\sigma(\mathsf{w})}{\sigma(\mathsf{s})}$ for each task k and $\pi_1(m) := \prod_{k=1}^K (f_1^k(m) + 1)$.
for $t = 1, 2, \ldots$ **do**
  Observe task index $\kappa = \kappa_t$.
  Predict with $v_t(m) := \frac{f_t^\kappa(m)\,\pi_t(m)/(f_t^\kappa(m)+1)}{\sum_{i=1}^M f_t^\kappa(i)\,\pi_t(i)/(f_t^\kappa(i)+1)}$.
  Read losses ℓ_t^m of each expert m
  Suffer loss $\ell_t = -\ln \sum_m v_t^m e^{-\ell_m^t}$.
  Update $f_{t+1}^\kappa(m) := e^{\ell_t - \ell_t^m} f_t^\kappa(m)$ and keep $f_{t+1}^k(m) := f_t^k(m)$ for all $k \neq \kappa$.
  Update $\pi_{t+1}(m) := \frac{f_{t+1}^\kappa(m)+1}{f_t^\kappa(m)+1}\pi_t(m)$.
end for

rithm 10) is bounded by

$$\mathcal{R} \;\leq\; N \ln(M/N) + KN\,\mathcal{H}(1/N).$$

Proof. Without loss of generality assume that $\hat{m}_k \in \{1, \ldots, N\}$. Let $S_n := \{1 \leq k \leq K \mid \hat{m}_k = n\}$. The sets S_n for $n = 1, \ldots, N$ form a partition of the K tasks. By (3.18) $\prod_{n=1}^N P(S_n) = \sigma(\mathsf{w})^K \sigma(\mathsf{s})^{(N-1)K}$, which is maximized by the proposed tuning. The theorem now follows from (3.13). □

We achieve the desired goal since $KN\,\mathcal{H}(1/N) \approx K \ln N$. In practice N is of course unavailable for tuning, and we may tune $\sigma(\mathsf{w}) = 1/K$ pessimistically to get $K \ln K + N$ instead for all N simultaneously. Or alternatively, we may sacrifice some time efficiency to externally mix over all M possible values with decreasing prior, increasing the tuned regret by just $\ln N + \mathcal{O}(\ln \ln N)$. If in addition the number of tasks is unknown or unbounded, we may (as done in Section 3.3.3) decrease the membership rate $\sigma(\mathsf{w})$ with each new task encountered and guarantee regret $\mathcal{R} \leq N \ln(M/N) + K \ln K + 4N + 2K \ln \ln K$ where now K is the number of tasks actually received.

3.4.3 Multitask Learning experiment

To test our new Multitask Learning algorithm we used the TB data provided by the VLA. The task here is to try to predict the probability of the skin test being positive based on the per-animal attributes, such as age, sex and breed. We pose it as a multitask learning problem in the following way. The cows are partitioned into buckets based on triples {age group, sex, breed}, where we use three age groups of equal size. Each bucket is a task. At every trial Learner is given the task number and is predicting the probability of this animal being a reactor. Then the outcome of the test is revealed and Learner suffers log-loss.

The experts for this task are discretized Bernoulli experts so we would like to learn the optimal rate for a given task. It is known that running plain AA on Bernoulli experts with certain prior results in a well-known Krichevsky–Trofimov estimator.

Setup After the initial preprocessing (removing the entries with missing data and splitting cows into three age groups of the same size) 16572647 cows were split into the 450 buckets (3 age groups, 3 values for sex and 50 breeds). One of them was empty, so this resulted in 449 tasks.

The following algorithms were run.

- Basic Aggregating Algorithm (Krichevsky–Trofimov). This was learning the overall illness rate and does exploit the differences between tasks.
- Per-task Aggregating Algorithm (keeping separate weights on experts for each task)
- Aggregating Algorithm for Multitask Learning (Algorithm 10).

Also, the following experts were used for comparison:

AA	Best expert	Per-task AA	Multitask AA	Best per-task expert
8.7815e+05	8.7814e+05	8.4364e+05	8.4299e+05	8.4182e+05

Table 3.1: Multitask experiment: total losses

- Retrospectively best Bernoulli expert overall
- Retrospectively best per-task Bernoulli expert

The first one sets the goal for the basic AA, while the second one sets the goal for the per-task AA and the multitask one. We expect that by exploiting the similarities of some tasks Algorithm 10 will be better than the per-task AA.

Discussion Total losses after 16572647 trials are shown in Table 3.1.

First of all, we see that all the losses are not very far from each other. The reason here is that the level of TB varies between 0.05% and 1% for all the tasks. We see that AA gets very close to the best expert overall.

Per-task AA gets quite close to the Best per-task expert and the gap for Multitask algorithms to squeeze in is very small. However, Algorithm 10 manages to get even closer to the best expert, as expected.

Appendix A

Adaptive regret for mix loss transfers to mixable losses

The protocol of prediction with expert advice is given as Protocol 3. Predictions are made sequentially, their quality is measured by the loss function λ and Learner has access to a (finite) pool of N experts. Protocol 2 is the special case corresponding to the simplex prediction set $\Gamma = \triangle_N$, outcome space $\Omega = [-\infty, \infty]^N$ and mix loss. An important class of loss functions that allow for effective algorithms are mixable losses.

Protocol 3 Prediction with expert advice

$L_0 := 0$,
$L_0^n := 0$, $n = 1, \ldots, N$
for $t = 1, 2, \ldots$ **do**
  Expert n announces prediction $\gamma_t^n \in \Gamma$, $n = 1, \ldots, N$
  Learner announces prediction $\gamma_t \in \Gamma$
  Reality announces outcome $\omega_t \in \Omega$
  $L_t := L_{t-1} + \lambda(\gamma_t, \omega_t)$
  $L_t^n := L_{t-1}^n + \lambda(\gamma_t^n, \omega_t)$, $n = 1, \ldots, N$
end for

Definition 31. A loss function λ is called *η-mixable*, where $\eta > 0$, if for every K, every sequence of predictions $\gamma^1, \ldots, \gamma^K$ and every sequence of normalised

nonnegative weights $u^1, \ldots, u^K$ there exists a prediction $\gamma \in \Gamma$ such that for every outcome $\omega \in \Omega$

$$\lambda(\gamma, \omega) \;\leq\; -\frac{1}{\eta} \ln \left(\sum_{k=1}^{K} u^k e^{-\eta \lambda(\gamma^k, \omega)} \right). \tag{A.1}$$

A function Σ that maps every sequence of predictions $\gamma^1, \ldots, \gamma^K$ and every sequence of normalised nonnegative weights $u^1, \ldots, u^K$ of the same length to $\gamma \in \Gamma$ satisfying (A.1) is called an *η-perfect substitution function*.

The notion of mixability will not change if we set $K = 2$ in (A.1). Examples of mixable games can be found in [43]. Note that $1/\eta$-scaled mix loss is the baseline used in the definition of mixability: see (A.1). In this sense the mix loss is the hardest mixable loss. It is hence no surprise that adaptive regret bounds for mix loss immediately transfer to any mixable loss:

Fact 32. *Let X be a mix loss algorithm with w.c. adaptive regret $\phi(t_1, t_2, N)$. If λ is η-mixable then there is an algorithm Y with adaptive regret at most $\phi(t_1, t_2, N)/\eta$.*

Proof. Let Σ be an η-perfect substitution function for λ. We choose Y to be the algorithm that operates as follows. At each trial $t = 1, 2, \ldots$ it obtains prediction $\vec{u}_t$ from X, predicts with $\gamma_t = \Sigma(\vec{u}_t, \vec{\gamma}_t)$, and feeds into X losses $\ell_t^n = \eta \lambda(\gamma_t^n, \omega_t)$ (from the point of view of Y these are scaled losses rather than losses). Then for each interval $[t_1, t_2]$ and reference expert j we have

$$\begin{aligned} L^Y_{[t_1,t_2]} - L^j_{[t_1,t_2]} \;&=\; \sum_{t=t_1}^{t_2} \lambda(\gamma_t, \omega_t) - \sum_{t=t_1}^{t_2} \lambda(\gamma_t^j, \omega_t) \\ &\leq\; -\frac{1}{\eta} \sum_{t=t_1}^{t_2} \ln \sum_n u_t^n e^{-\eta \lambda(\gamma_t^n, \omega_t)} - \sum_{t=t_1}^{t_2} \lambda(\gamma_t^j, \omega_t) \\ = \; -\frac{1}{\eta} \sum_{t=t_1}^{t_2} \ln \sum_n u_t^n e^{-\ell_t^n} - \frac{1}{\eta} \sum_{t=t_1}^{t_2} \ell_t^j \;&=\; \frac{1}{\eta} \left(L^X_{[t_1,t_2]} - L^j_{[t_1,t_2]} \right) \;\leq\; \frac{1}{\eta} \phi(t_1, t_2, N), \end{aligned}$$

where the first inequality follows from the definition of an η-perfect substitution function and the last one from our assumption about X. $\square$

Fact 32 shows that all our performance guarantees for the mix loss protocol carry over to the protocol of prediction with expert advice with a mixable loss function.

Appendix B

Worst-case adaptive regret data for Fixed Share

In this subsection we prove that the worst-case data for Fixed Share has the following form. On the interval $[t_1, t_2]$ we are interested in all but one expert suffer infinite loss and on the step preceding t_1 (if $t_1 \neq 1$) this one expert suffers infinite loss himself. The construction is iterative and we start constructing the data from the end of the interval. Throughout this section we assume that $\alpha_t \in [0, \frac{N-1}{N}]$ for all t.

Lemma 33. *For any history prior to the step t_2 the adaptive regret $R^j_{[t_1,t_2]}$ w.r.t. expert j on the interval $[t_1, t_2]$ is maximised with $\ell^k_{t_2} = \infty$ for $k \neq j$.*

Proof. Let us differentiate the adaptive regret w.r.t. $\ell_{t_2}^k$:

$$\begin{aligned}\frac{\partial R_{[t_1,t_2]}^j}{\partial \ell_{t_2}^k} &= \frac{\partial}{\partial \ell_{t_2}^k}(\ell_{t_2} - \ell_{t_2}^j) \\ &= \frac{\partial}{\partial \ell_{t_2}^k}(\ell_{t_2}) - \mathbf{1}_{\{j=k\}} \\ &= -\frac{\partial}{\partial \ell_{t_2}^k} \ln \sum_i u_{t_2}^i e^{-\ell_{t_2}^i} - \mathbf{1}_{\{j=k\}} \\ &= \frac{u_{t_2}^k e^{-\ell_{t_2}^k}}{\sum u_{t_2}^i e^{-\ell_{t_2}^i}} - \mathbf{1}_{\{j=k\}}\end{aligned}$$

□

We can see that it is positive for all $k \neq j$ and becomes zero for $k = j$ when we plug in $\ell_{t_2}^k = \infty$ for those.

Lemma 34. *Fix an comparator expert j. Let $t \in [t_1, t_2]$. Suppose that the losses for steps $s = t+1, \ldots, t_2$ satisfy $\ell_s^k = \infty$ for $k \neq j$. Then the adaptive regret $R_{[t_1,t_2]}^j$ is maximised with $\ell_t^k = \infty$ for $k \neq j$.*

Proof. Let us start with showing that the if on the steps $t+1$ and $t+2$ the data is organised as we want to, that is j-th expert is good and all others suffer infinite loss, then Learner's loss on step $t+2$ is not dependent on what happens at time t and before. This follows immediately from (3.6), as

$$\ell_{t+2} = -\ln(1 - \alpha_{t+2}).$$

Now let us differentiate the adaptive regret $R_{[t_1,t_2]}^j$ w.r.t. ℓ_t^k assuming that the future losses are set up as we want. Let us show that the derivatives w.r.t. ℓ_t^k where $k \neq j$ are all positive. For those,

$$\frac{\partial R_{[t_1,t_2]}^j}{\partial \ell_t^k} = \frac{\partial \ell_t}{\partial \ell_t^k} + \frac{\partial \ell_{t+1}}{\partial \ell_t^k}$$

Expanding the second one gives (as before, $k \neq j$):

$$\begin{aligned}\frac{\partial \ell_{t+1}}{\partial \ell_t^k} &= \frac{\partial}{\partial \ell_t^k} - \ln\left(\frac{\alpha_{t+1}}{N-1} + (1 - \frac{N}{N-1}\alpha_{t+1})u_t^j e^{\ell_t - \ell_t^j}\right) \\ &= -\frac{(1 - \frac{N}{N-1}\alpha_{t+1})u_t^j e^{\ell_t - \ell_t^j} \frac{\partial}{\partial \ell_t^k}\ell_t}{\frac{\alpha_{t+1}}{N-1} + (1 - \frac{N}{N-1}\alpha_{t+1})u_t^j e^{\ell_t - \ell_t^j}}\end{aligned}$$

So we see that

$$\begin{aligned}\frac{\partial R^j_{[t_1,t_2]}}{\partial \ell_t^k} &= \frac{\partial \ell_t}{\partial \ell_t^k}\left(1 - \frac{(1 - \frac{N}{N-1}\alpha_{t+1})u_t^j e^{\ell_t - \ell_t^j}}{\frac{\alpha_{t+1}}{N-1} + (1 - \frac{N}{N-1}\alpha_{t+1})u_t^j e^{\ell_t - \ell_t^j}}\right) \\ &= \frac{\partial \ell_t}{\partial \ell_t^k}\left(\frac{\frac{\alpha_{t+1}}{N-1}}{\frac{\alpha_{t+1}}{N-1} + (1 - \frac{N}{N-1}\alpha_{t+1})u_t^j e^{\ell_t - \ell_t^j}}\right) > 0\end{aligned}$$

So our worst-case pattern of losses extends one trial backwards. □

Finally, we need to state the almost obvious fact that in order to maximise the adaptive regret we need to insert an infinite loss for the comparator expert right before the start of the interval, thus killing all the previous weight on him.

Lemma 35. *Fix a comparator expert j. Suppose that the losses for steps $s = t_1, \ldots, t_2$ satisfy $\ell_s^k = \infty$ for $k \neq j$. Then the adaptive regret $R^j_{[t_1,t_2]}$ is maximised with $\ell^j_{t-1} = \infty$.*

Proof. As before, the adaptive regret on steps starting from $t_1 + 1$ does not depend on $\ell^k_{t_1-1}$. So let us show that $\frac{\partial R^j_{[t_1,t_2]}}{\partial \ell^j_{t_1-1}} > 0$. We can reuse the proofs of previous lemmas for that:

$$\begin{aligned}\frac{\partial R^j_{[t_1,t_2]}}{\partial \ell^j_{t_1-1}} &= \frac{\partial \ell_{t_1}}{\partial \ell^j_{t_1-1}} \\ &= -\frac{(1 - \frac{N}{N-1}\alpha_{t_1})u^j_{t_1-1} e^{\ell_{t_1-1} - \ell^j_{t_1-1}}}{\frac{\alpha_{t_1}}{N-1} + (1 - \frac{N}{N-1}\alpha_{t_1})u^j_{t_1-1} e^{\ell_{t_1-1} - \ell^j_{t_1-1}}} \frac{\partial\left(\ell_{t_1-1} - \ell^j_{t_1-1}\right)}{\partial \ell^j_{t_1-1}} > 0,\end{aligned}$$

since $\frac{\partial\left(\ell_{t_1-1}-\ell^j_{t_1-1}\right)}{\partial\ell^j_{t_1-1}}$ is negative as follows from the proof of Lemma 33. $\square$

Bibliography

[1] Jacob Ducan Abernethy, Peter Bartlett, and Alexander Rakhlin. Multitask learning with expert advice. Technical report, University of California at Berkeley, January 2007.

[2] Steven Abney. *Semisupervised Learning for Computational Linguistics*. Chapman & Hall/CRC, 1st edition, 2007.

[3] Dmitry Adamskiy, Wouter M. Koolen, Alexey Chernov, and Vladimir Vovk. A closer look at adaptive regret. In *ALT*, pages 290–304, Berlin, Heidelberg, 2012. Springer.

[4] Dmitry Adamskiy, Ilia Nouretdinov, Andy Mitchell, Nick Coldham, and Alexander Gammerman. Applying conformal prediction to the bovine TB diagnosing. In *EANN/AIAI (2)*, pages 449–454, 2011.

[5] Mitya Adamskiy, Ilia Nouretdinov, and Alex Gammerman. *Conformal prediction in semi-supervised case*. Chapman and Hall, 2011.

[6] Alekh Agarwal, Alexander Rakhlin, and Peter Bartlett. Matrix regularization techniques for online multitask learning. Technical report, University of California Berkeley, October 2008. EECS-2008-138.

[7] Mikhail Belkin and Partha Niyogi. Semi-supervised learning on Riemannian manifolds. *Machine Learning*, 56:209–239, 2004.

[8] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *COLT*.

[9] Olivier Bousquet and Manfred K. Warmuth. Tracking a small set of experts by mixing past posteriors. *Journal of Machine Learning Research*, 3:363–396, 2002.

[10] Rich Caruana. Multitask learning. In *Machine Learning*, pages 41–75, 1997.

[11] Giovanni Cavallanti, Nicolò Cesa-Bianchi, and Claudio Gentile. Linear algorithms for online multitask classification. *J. Mach. Learn. Res.*, 11:2901–2934, December 2010.

[12] Nicolò Cesa-Bianchi, Pierre Gaillard, Gábor Lugosi, and Gilles Stoltz. A new look at shifting regret. *arXiv*, abs/1202.3323, 2012.

[13] Nicolò Cesa-Bianchi and Gábor Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

[14] O. Chapelle, B. Schölkopf, and A. Zien, editors. *Semi-Supervised Learning*. MIT Press, Cambridge, MA, 2006.

[15] Alexey Chernov and Vladimir Vovk. Prediction with expert evaluators' advice. In *ALT*, pages 8–22, 2009.

[16] Alfredo DeSantis, George Markowsky, and Mark N. Wegman. Learning probabilistic prediction functions. In *COLT*, pages 312–328, San Francisco, CA, USA, 1988. Morgan Kaufmann Publishers Inc.

[17] Y. Freund, R. E. Schapire, Y. Singer, and M. K. Warmuth. Using and combining predictors that specialize. In *Proc. 29th Annual ACM Symposium on Theory of Computing*, pages 334–343. ACM, 1997.

[18] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55:119–139, 1997.

[19] Alexander Gammerman, Vladimir Vovk, and Vladimir Vapnik. Learning by transduction. In *In Uncertainty in Artificial Intelligence*, pages 148–155. Morgan Kaufmann, 1998.

[20] Robert B. Gramacy, Manfred K. Warmuth, Scott A. Brandt, and Ismail Ari. Adaptive caching by refetching. In Suzanna Becker, Sebastian Thrun, and Klaus Obermayer, editors, *NIPS*, pages 1465–1472. MIT Press, 2002.

[21] Elad Hazan and C. Seshadhri. Efficient learning algorithms for changing environments. In *ICML*, page 50, 2009.

[22] David P. Helmbold, Darrell D. E. Long, Tracey L. Sconyers, and Bruce Sherrod. Adaptive disk spin-down for mobile computers. *ACM/Baltzer Mobile Networks and Applications (MONET)*, pages 285–297, 2000.

[23] Mark Herbster. Learning additive models online with fast evaluating kernels. In *COLT*, pages 444–460, London, UK, UK, 2001. Springer-Verlag.

[24] Mark Herbster, Stephen Pasteris, and Fabio Vitale. Online sum-product computation over trees. In *NIPS*, pages 2879–2887, 2012.

[25] Mark Herbster, Massimiliano Pontil, and Lisa Wainer. Online learning over graphs. In *ICML*, pages 305–312, 2005.

[26] Mark Herbster and Manfred K. Warmuth. Tracking the best expert. *Machine Learning*, 32:151–178, 1998.

[27] Wouter M. Koolen. *Combining Strategies Efficiently: High-quality Decisions from Conflicting Advice*. PhD thesis, Institute of Logic, Language and Computation (ILLC), University of Amsterdam, January 2011.

[28] Wouter M. Koolen, Dmitry Adamskiy, and Manfred K. Warmuth. Putting Bayes to sleep. In *NIPS*, pages 135–143, 2012.

[29] Wouter M. Koolen and Steven de Rooij. Combining expert advice efficiently. In *COLT*, pages 275–286, 2008.

[30] Nick Littlestone and Manfred K. Warmuth. The Weighted Majority Algorithm. *Inf. Comput.*, 108(2):212–261, 1994.

[31] Gábor Lugosi, Omiros Papaspiliopoulos, and Gilles Stoltz. Online multi-task learning with hard constraints. In *COLT*, 2009.

[32] Thomas Melluish, Craig Saunders, Ilia Nouretdinov, and Volodya Vovk. Comparing the Bayes and typicalness frameworks. In *ECML*, pages 360–371, 2001.

[33] Ilia Nouretdinov, Brian Burford, and Alexander Gammerman. Application of inductive confidence machine to ICMLA competition data. In *ICMLA*, pages 435–438, 2009.

[34] Ilia Nouretdinov, Thomas Melluish, and Volodya Vovk. Ridge regression confidence machine. In *ICML*, pages 385–392, 2001.

[35] Alexander Rakhlin, Jacob Abernethy, and Peter L. Bartlett. Online discovery of similarity mappings. In *Proceedings of the 24th international conference on Machine learning*, ICML '07, pages 767–774, New York, NY, USA, 2007. ACM.

[36] Avishek Saha, Piyush Rai, Hal Daumé III, and Suresh Venkatasubramanian. Online learning of multiple tasks and their relationships. In *AISTATS*, Ft. Lauderdale, Florida, 2011.

[37] Craig Saunders, Alexander Gammerman, and Volodya Vovk. Transduction with confidence and credibility. In *IJCAI*, pages 722–726, 1999.

[38] Gil I. Shamir and Neri Merhav. Low complexity sequential lossless coding for piecewise stationary memoryless sources. *IEEE Trans. Info. Theory*, 45:1498–1519, 1999.

[39] Vladimir Vapnik. *Statistical learning theory*. Wiley, New York, 1998.

[40] Paul A.J. Volf and Frans M.J. Willems. Switching between two universal source coding algorithms. In *Proceedings of the Data Compression Conference, Snowbird, Utah*, pages 491–500, 1998.

[41] Vladimir Vovk. Aggregating strategies. In *COLT*, pages 371–383. Morgan Kaufmann, 1990.

[42] Vladimir Vovk. A game of prediction with expert advice. *Journal of Computer and System Sciences*, 56:153–173, 1998.

[43] Vladimir Vovk. Competitive on-line statistics. *International Statistical Review*, 69:213–248, 2001.

[44] Vladimir Vovk. On-line confidence machines are well-calibrated. In *FOCS*, pages 187–196, 2002.

[45] Vladimir Vovk, Alexander Gammerman, and Glenn Shafer. *Algorithmic Learning in Random World*. Springer, 2005.

[46] Vladimir Vovk, David Lindsay, Ilia Nouretdinov, and Alex Gammerman. Mondrian confidence machine. Workingpaper, 2003. On-line Compression Modelling project, http://vovk.net/cp, Working Paper 4.

[47] David Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, ACL '95, pages 189–196, Stroudsburg, PA, USA, 1995. Association for Computational Linguistics.

[48] Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *ICML*, pages 928–936, 2003.

[49] Jacob Ziv. Coding theorems for individual sequences. *IEEE Transactions on Information Theory*, 24(4):405–412, 1978.