Conference Paper

Model switching and other Unconventional Fun with Experts

Author(s):
de Rooij, Steven

Publication Date:
2012

Permanent Link:
https://doi.org/10.3929/ethz-a-007050435

Rights / License:
In Copyright - Non-Commercial Use Permitted
Model Switching and other Unconventional Fun with Experts

Steven de Rooij
Centrum Wiskunde & Informatica (CWI)
Science Park 123, Amsterdam, Netherlands
Email: s.de.rooij@cwi.nl

I. Universal Coding and Regret

Universal codes combine a number of codes into one that achieves code lengths close to those of the best of the bunch. We consider the strongest notion of universality, where we compete with the best code for any data sequence, not just in expectation. When this is the goal, a natural performance measure is regret: the difference between the universal code length and the length of the best original code. Given a fixed finite set of $K$ codes, a simple universal combination can be obtained with a two-part scheme, where the first $\log_2 K$ bits are used to identify the original codes to be used for the data, and the second part is the actual encoding of the data using that code. The regret of this code is bounded by the length of the first part of the code. (From here on, all logarithms are binary and we ignore the rounding of code lengths to integer numbers of bits.)

This idea can be taken a step further. Consider the case where a good code is available for any reasonably small segment of the input sequence, but either the data or the codes themselves are so changeable that none of the $K$ original codes perform very well on the entire data sequence. In such cases, a stronger regret guarantee is desirable. For simplicity, fix the number of source symbols at $n$. Then, starting from $K$ codes, first construct a new set of all “tracking” codes that split the input sequence into $m$ blocks, and use one of the $K$ original codes for each block. The goal is now to minimise the regret with respect to this extended set of codes. Counting how many such codes there are is a simple combinatorial exercise; we can subsequently use a two-part code as before to obtain a regret $R(x^n, m)$ satisfying

$$R(x^n, m) \leq \log\left(\frac{n - 1}{m - 1}\right) + m \log K.$$  

This solution is not quite satisfactory. First, it requires finding the best tracking code; while this can be done using dynamic programming, we will see that it is possible to obtain similar regret guarantees with much faster algorithms. Second, the number of segments $m$ is a parameter of the algorithm, but $m$ is typically unknown so we would like to have a regret guarantee that is valid for all $m$ simultaneously. Third, the code is not sequential, making it unusable in an online setting.

A number of very simple and elegant algorithms have been developed that avoid these issues. Some derive from the information theory literature [Wil07], [VW98], [SM99], but there are also important, and strongly related results in online learning that deserve more exposure [HW98], [Vov99], [BW02], [CBL06], [GLL08]. This text is based on our paper [KdR08], which outlines the common structure to these algorithms.

II. Expert Sequence Priors

Universal codes of the kind described above are most conveniently defined and analysed in the Bayesian framework. As a first step, we switch to probability theory and replace the length functions $L_1, \ldots, L_K$ of the $K$ original codes by probabilistic sources $P_1, \ldots, P_K$, such that $-\log P_k(x^n) = L_k(x^n)$ for all $k$ and all data sequences $x^n = x_1, \ldots, x_n \in \mathcal{X}^n$ (for simplicity assume the alphabet to be discrete). Such probabilistic sources can now be used in a sequential prediction setting; the accumulated logarithmic loss exactly corresponds to the code length:

$$\sum_{t=1}^{n} -\log P_k(x_t|x^{t-1}) = -\log P_k(x^n = x^n) = L(x^n).$$

In the online learning literature, the prediction strategies $P_k$ are called “experts” because, unless we have additional knowledge about the data generating process, which we do not wish to assume, we have to use their predictions.

We now extend the model class as follows. Let $k^n \in \{1, \ldots, K\}^n$ denote a sequence of expert identifiers. Define

$$P_{k^n}(x^n) = P_{k_1}(X_1) \cdot P_{k_2}(X_2|X_1) \cdot \ldots \cdot P_{k_n}(X_n|X_{n-1}).$$

We do not know which sequence of experts will yield the best predictions, but we can represent our uncertainty about the data generating process, which we do not wish to assume, we have to use their predictions. We can use this prior to make predictions by conditioning the Bayesian marginal distribution:

$$P(X_{n+1}|x^n) = \frac{P(x^{n+1})}{P(x^n)} = \frac{\sum_{k^{n+1}} P_{k^{n+1}}(x^{n+1}, X_{n+1})}{\sum_{k^n} P_{k^n}(x^n)}.$$  

It is now also very easy to obtain a reasonable regret bound, by dropping every term from the marginal likelihood except the one we are interested in. Choose any reference expert sequence $\tilde{k}^n$, and let $m$ be the number of blocks it has. Then

$$R(x^n, \tilde{k}^m) = -\log P(x^n) - (-\log P_{\tilde{k}^n})$$

$$= -\log \sum_{k^n} P_{k^n}(x^n)p(k^n) + \log P_{\tilde{k}^n}(x^n)$$

$$\leq -\log \left(P_{\tilde{k}^n}(x^n)p(\tilde{k}^m)\right) + \log P_{\tilde{k}^n}(x^n)$$

$$= -\log \pi(\tilde{k}^m).$$
A. Example

In general, the best sequence of experts may have a very low prior probability, so it will be hard to compete with. However, we can construct the prior such that expert sequences with relatively few blocks receive a high prior probability. For example, we may set

\[ \pi(k^n) = K^{-m} \alpha^{n-1} \left( \frac{1 - \alpha}{\alpha} \right)^{m-1}, \]

where \( m \) is the number of blocks in \( k^n \), and \( \alpha \in [0, 1] \) is a hyperparameter that can be interpreted as the “switching rate”; it is the probability of a new block starting at any given time. It may be verified that this prior sums to one.

An algorithm to efficiently compute the predictions of this distribution, called Fixed Share, was first published in COLT 1995; see [HW98] for the journal version. For Fixed Share, the regret bound (3) reduces to

\[ \mathcal{R}(x^n, k^n) \leq m \log K - n \log \alpha - (m - 1) \log \left( \frac{1 - \alpha}{\alpha} \right). \]

The bound is optimised by setting the hyperparameter to \( \alpha^* = (n - m + 1)/(n - 1) \), which yields

\[ \mathcal{R}(x^n, k^n) \leq m \log K + (n - 1)H(\alpha^*), \]

where \( H \) denotes the binary entropy function. Note that this bound is very close to (1); the slack is necessary because this new regret bound is valid for all \( m \), whereas (1) is tight for a single fixed \( m \).

B. Second Example: Hyperpriors

In case the expert sequence prior is burdened by an unknown parameter, we can “integrate it out” using a hyperprior. Let \( \pi^{(\alpha)} \) denote an expert sequence prior \( \pi \) with hyperparameter \( \alpha \), and let \( P^{(\alpha)} \) denote the tracking distribution with expert sequence prior \( \pi^{(\alpha)} \). Now impose a hyperprior \( \rho \) on \( \alpha \). We then obtain the following parameterless distribution:

\[ P(X^n) = \int P^{(\alpha)}(X^n) \rho(\alpha) \, d\alpha = \sum_k P_k(X^n) \int \pi^{(\alpha)}(k^n) \rho(\alpha) \, d\alpha. \]

Note that the final integral itself defines an expert sequence prior. We will call it \( \sigma \). Thus, to obtain a parameterless tracking distribution we do not have to look at the data; it suffices to mix the expert sequence prior using \( \rho \).

In case of the example of the previous section, with \( \pi^{(\alpha)} \) given as in (4) and hyperprior \( \rho(\alpha) = \text{Beta}(\frac{1}{2}, \frac{1}{2}) \), we find

\[ \sigma(k^n) = K^{-m} \frac{\Gamma(n - m + \frac{1}{2}) \Gamma(m + \frac{1}{2})}{\pi \Gamma(n + 1)}. \]

Around the time that Fixed Share was published, but completely independently, Paul Volf and Frans Willems developed an algorithm dubbed the “Switching Method”, that efficiently predicts according to this distribution [Wii07], [VW98]. It can be shown that its regret exceeds (5) by at most \( \frac{1}{2} \log n + 1 \) bits, a modest price to pay for a completely parameterless strategy.

In the next section we proceed to show how Fixed Share and the Switching Method can be implemented efficiently.

III. Computation using HMMs

In general, the predictions of an expert distribution based on an expert sequence prior (2) may require summing an exponential number of terms, but the complexity can be reduced dramatically if the expert sequence prior admits a simple sufficient statistic. This is the case for the two example priors of the previous section: in Fixed Share, the marginal prior distribution on the next expert \( \pi(K_{n+1} | k^n) \) can be expressed as a function of only the current expert \( k_n \). Namely,

\[ \pi(K_{n+1} = k | k^n) = \frac{\alpha}{K} + 1 \{ k = k_n \}(1 - \alpha). \]

Such expert distributions can be represented by a Hidden Markov Model (HMM), where the hidden state is such a sufficient statistic. The value of the hidden variable determines the distribution on experts; the experts’ predictions are weighed according to this distribution. It is convenient to represent the prior in the form of a state transition diagram for the hidden states. For the Fixed Share algorithm with four experts, the diagram looks as follows:

All straight transitions have probability \( 1 - \alpha \); they represent that no new block is started at that sample size. The transitions into the centre nodes have probability \( \alpha \); they represent a switch to a new block. The centre nodes are “silent states” where no prediction has to be produced; such states redistribute some of the probability mass over the experts. The arcs out of the silent states all have probability \( 1/K \).

Using the Forward algorithm with these state transitions, the distribution on the hidden state can be updated in \( O(K) \), so that the entire data set can be processed in \( O(n \cdot K) \).

The switching method is only slightly more complicated. In this case, the currently used expert is not sufficient to determined the probability of starting a new block: we also need to know how many blocks there were previously. The following state transition diagram visualises the state transition diagram required to calculate \( \sigma(K_{n+1} | k^n) \). For simplicity it is for two experts, and we also assume that the first expert always predicts first. Then the current expert is determined by the number of switches modulo 2.
Here the upward arcs represent the starting of a new block, and the straight arcs represent staying in the same block. For the $n$th node from the left, the $s$th row from the bottom, the probability of the upward arc is $(s + 1/2)/(n + 1)$ and the probability of the straight arc is one minus that; in other words the Krichevsky-Trofimov estimator is used to predict whether or not a switch occurs.

Note that the time complexity of the forward algorithm on all symbols has increased to $O(n^2 \cdot K)$.

IV. THERE IS MORE

The power of expert tracking algorithms is twofold. First, they are black box: any online prediction strategy can be used as an expert. As such they can be applied very generally. One example is in [vEGdR11], where the estimators associated with each a series of model classes are used as “experts”. The paper subsequently uses an algorithm similar to Fixed Share to improve the performance of Bayesian Model Averaging under many circumstances. Second, very attractive computational complexity can sometimes be achieved. For example, the Fixed Share algorithm runs exactly as fast as Bayesian Model Averaging itself.

Based on our paper [KdR08], we have seen two expert tracking algorithms as examples of how you can do more with a bunch of prediction strategies than what everyone always seems to be doing, which is to just take the Bayesian mixture. The preprint of the full paper discusses several other expert sequence priors for tracking, as well as more sophisticated methods to analyse the resulting performance.

Much of current online learning research is related to prediction with expert advice, but it may not be instantly recognisable as such, as over time the problem has been generalised substantially. Rather than that the learner uses a set of experts who make predictions in the form of probability distributions, is subsequently penalized with logarithmic loss, before finally observing the actual outcome, in the more general setting both experts and outcomes are abstracted away altogether: the learner chooses an action from some space of allowed actions, and receives an arbitrary loss. This more general setup allows the theory to encompass online linear optimisation and prediction problems with loss functions other than logarithmic loss. It also makes the problem harder, and perhaps less directly applicable to information theory. As a starting point, see [CBL06].

REFERENCES