Minimum Description Length Model Selection

de Rooij, S.

Publication date
2008

Chapter 4

Interlude: Prediction with Expert Advice

We cannot predict exactly how complicated processes such as the weather, the stock market, social interactions and so on, will develop into the future. Nevertheless, people do make weather forecasts and buy shares all the time. Such predictions can be based on formal models, or on human expertise or intuition. An investment company may even want to choose between portfolios on the basis of a combination of these kinds of predictors. In such scenarios, predictors typically cannot be considered "true". Thus, we may well end up in a position where we have a whole collection of prediction strategies, or *experts*, each of whom has *some* insight into *some* aspects of the process of interest. We address the question how a given set of experts can be combined into a single predictive strategy that is as good as, or if possible even better than, the best individual expert.

The setup is as follows. Let Ξ be a finite set of experts. Each expert $\xi \in \Xi$ issues a distribution $P_\xi(\boldsymbol{x}_{n+1}|x^n)$ on the next outcome $\boldsymbol{x}_{n+1}$ given the previous observations $x^n := x_1, \ldots, x_n$. Here, each outcome x_i is an element of some countable space $\mathcal{X}$, and random variables are written in bold face. The probability that an expert assigns to a sequence of outcomes is given by the chain rule: $P_\xi(x^n) = P_\xi(x_1) \cdot P_\xi(x_2|x_1) \cdot \ldots \cdot P_\xi(x_n|x^{n-1})$.

A standard Bayesian approach to combine the expert predictions is to define a prior w on the experts Ξ which induces a joint distribution with mass function $P(x^n, \xi) = w(\xi)P_\xi(x^n)$. Inference is then based on this joint distribution. We can compute, for example: (a) the *marginal probability* of the data $P(x^n) = \sum_{\xi\in\Xi} w(\xi)P_\xi(x^n)$, (b) the *predictive distribution* on the next outcome $P(\boldsymbol{x}_{n+1}|x^n) = P(x^n, \boldsymbol{x}_{n+1})/P(x^n)$, which defines a prediction strategy that combines those of the individual experts, or (c) the *posterior distribution* on the experts $P(\boldsymbol{\xi}|x^n) = P_{\boldsymbol{\xi}}(x^n)w(\boldsymbol{\xi})/P(x^n)$, which tells us how the experts' predictions should be weighted. This simple probabilistic approach has the advantage that it is computationally easy: predicting n outcomes using $|\Xi|$ experts requires only $O(n \cdot |\Xi|)$ time. Additionally, this Bayesian strategy guarantees that the overall

probability of the data is only a factor $w(\hat{\xi})$ smaller than the probability of the data according to the best available expert $\hat{\xi}$. On the flip side, with this strategy we never do any *better* than $\hat{\xi}$ either: we have $P_{\hat{\xi}}(x^n) \geq P(x^n) \geq P_{\hat{\xi}}(x^n)w(\hat{\xi})$, which means that potentially valuable insights from the other experts are not used to our advantage!

More sophisticated combinations of prediction strategies can be found in the literature under various headings, including (Bayesian) statistics, source coding and universal prediction. In the latter the experts' predictions are not necessarily probabilistic, and scored using an arbitrary loss function. Here we consider only logarithmic loss, although our results can undoubtedly be generalised to the framework described in, e.g. [95].

The three main contributions of this paper are the following. First, we introduce prior distributions on *sequences* of experts, which allows unified description of many existing models. Second, we show how HMMs can be used as an intuitive graphical language to describe such priors and obtain computationally efficient prediction strategies. Third, we use this new approach to describe and analyse several important existing models, as well as one recent and one completely new model for expert tracking.

Overview

In Section 4.1 we develop a new, more general framework for combining expert predictions, where we consider the possibility that the *optimal* weights used to mix the expert predictions may *vary over time*, i.e. as the sample size increases. We stick to Bayesian methodology, but we define the prior distribution as a probability measure on *sequences of experts* rather than on experts. The prior probability of a sequence $\xi_1, \xi_2, \ldots$ is the probability that we rely on expert ξ_1's prediction of the first outcome and expert ξ_2's prediction of the second outcome, etc. This allows for the expression of more sophisticated models for the combination of expert predictions. For example, the nature of the data generating process may evolve over time; consequently different experts may be better during different periods of time. It is also possible that not the data generating process, but the experts themselves change as more and more outcomes are being observed: they may learn from past mistakes, possibly at different rates, or they may have occasional bad days, etc. In both situations we may hope to benefit from more sophisticated modelling.

Of course, not all models for combining expert predictions are computationally feasible. Section 4.2 describes a methodology for the specification of models that allow efficient evaluation. We achieve this by using hidden Markov models (HMMs) on two levels. On the first level, we use an HMM as a formal specification of a distribution on sequences of *experts* as defined in Section 4.1. We introduce a graphical language to conveniently represent its structure. These graphs help to understand and compare existing models and to design new ones.

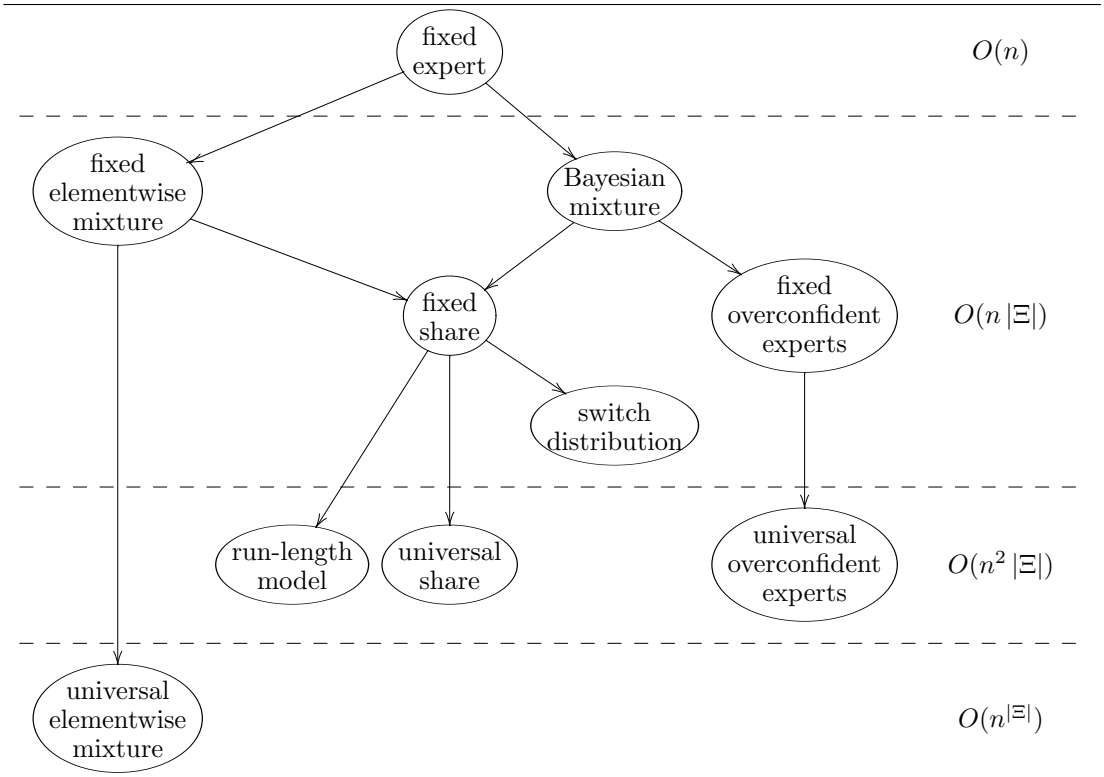
We then modify this first HMM to construct a second HMM that specifies the distribution on sequences of *outcomes*. Subsequently, we can use the standard dynamic programming algorithms for HMMs (forward, backward and Viterbi) on both levels to efficiently calculate most relevant quantities, most importantly the marginal probability of the observed outcomes $P(x^n)$ and posterior weights on the next expert given the previous observations $P(\boldsymbol{\xi_{n+1}}|x^n)$.

It turns out that many existing models for prediction with expert advice can be specified as HMMs. We provide an overview in Section 4.3 by giving the graphical representations of the HMMs corresponding to the following models. First, universal elementwise mixtures (sometimes called mixture models) that learn the optimal mixture parameter from data. Second, Herbster and Warmuth's fixed share algorithm for tracking the best expert [45, 46]. Third, universal share, which was introduced by Volf and Willems as the "switching method" [94] and later independently proposed by Bousquet [15]. Here the goal is to learn the optimal fixed-share parameter from data. The last considered model safeguards against overconfident experts, a case first considered by Vovk in [95]. We render each model as a prior on sequences of experts by giving its HMM. The size of the HMM immediately determines the required running time for the forward algorithm. The generalisation relationships between these models as well as their running times are displayed in Figure 4.1. In each case this running time coincides with that of the best known algorithm. We also give a loss bound for each model, relating the loss of the model to the loss of the best competitor among a set of alternatives in the worst case. Such loss bounds can help select between different models for specific prediction tasks.

Besides the models found in the literature, Figure 4.1 also includes two new generalisations of fixed share: the switch distribution and the run-length model. These models are the subject of Section 4.4. In Chapter 5 the switch distribution is used to improve Bayes/Minimum Description Length prediction to achieve the optimal rate of convergence in nonparametric settings. Here we give the concrete HMM that allows for its linear time computation, and we prove that it matches its parametric definition. The run-length model is based on a distribution on the number of successive outcomes that are typically well-predicted by the same expert. Run-length codes are typically applied directly to the data, but in our novel application they define the prior on expert sequences instead. Again, we provide the graphical representation of their defining HMMs as well as loss bounds.

Then in Section 4.5 we discuss a number of extensions of the above approach, such as approximation methods to speed up calculations for large HMMs.

Figure 4.1 Expert sequence priors: generalisation relationships and run time



4.1 Expert Sequence Priors

In this section we explain how expert tracking can be described in probability theory using expert sequence priors (ES-priors). These ES-priors are distributions on the space of infinite sequences of experts that are used to express regularities in the development of the relative quality of the experts' predictions. As illustrations we render Bayesian mixtures and elementwise mixtures as ES-priors. In the next section we show how ES-priors can be implemented efficiently by hidden Markov models.

Notation For $n \in \mathbb{N}$, we abbreviate $\{1, 2, \ldots, n\}$ by $[n]$, with the understanding that $[0] = \emptyset$. We also define $[\infty] = \mathbb{Z}^+$. For any natural number n, we let the variable q^n range over the n-fold Cartesian product Q^n, and we write $q^n = \langle q_1, \ldots, q_n \rangle$. We also let q^∞ range over Q^∞ — the set of infinite sequences over Q — and write $q^\infty = \langle q_1, \ldots \rangle$. We read the statement $q^\lambda \in Q^{\leq \infty}$ to first bind $\lambda \leq \infty$ and subsequently $q^\lambda \in Q^\lambda$. If q^λ is a sequence, and $\kappa \leq \lambda$, then we denote by q^κ the prefix of q^λ of length κ.

Forecasting Systems Let $\mathcal{X}$ be a countable outcome space. We use the notation $\mathcal{X}^*$ for the set of all finite sequences over $\mathcal{X}$ and let $\triangle(\mathcal{X})$ denote the set of

all probability mass functions on $\mathcal{X}$. A *(prequential) $\mathcal{X}$-forecasting system* (PFS) is a function $P : \mathcal{X}^* \to \triangle(\mathcal{X})$ that maps sequences of previous observations to a predictive distribution on the next outcome. Prequential forecasting systems were introduced by Dawid in [27].

Distributions We also require probability measures on spaces of infinite sequences. In such a space, a basic event is the set of all continuations of a given prefix. We identify such events with their prefix. Thus a distribution on $\mathcal{X}^\infty$ is defined by a function $P : \mathcal{X}^* \to [0, 1]$ that satisfies $P(\epsilon) = 1$, where ϵ is the empty sequence, and for all $n \geq 0$, all $x^n \in \mathcal{X}^n$ we have $\sum_{x \in \mathcal{X}} P(x_1, \ldots, x_n, x) = P(x^n)$. We identify P with the distribution it defines. We write $P(x^n|x^m)$ for $P(x^n)/P(x^m)$ if $0 \leq m \leq n$.

Note that forecasting systems continue to make predictions even after they have assigned probability 0 to a previous outcome, while distributions' predictions become undefined. Nonetheless we use the same notation: we write $P(x_{n+1}|x^n)$ for the probability that a forecasting system P assigns to the $n + 1$st outcome given the first n outcomes, as if P were a distribution.

ES-Priors The slogan of this chapter is, *we do not understand the data.* Instead of modelling the data, we work with experts. We assume that there is a fixed set of experts Ξ, and that each expert $\xi \in \Xi$ predicts using a forecasting system P_ξ.

We are interested in switching between different forecasting systems at different sample sizes. For a sequence of experts with prefix ξ^n, the combined forecast, where expert ξ_i predicts the ith outcome, is denoted

$$P_{\xi^n}(x^n) := \prod_{i=1}^{n} P_{\xi_i}(x_i|x^{i-1}).$$

Adopting Bayesian methodology, we impose a prior π on infinite sequences of experts; this prior is called an *expert sequence prior* (ES-prior). Inference is then based on the distribution on the joint space $(\mathcal{X} \times \Xi)^\infty$, called the *ES-joint*, which is defined as follows:

$$P\Big(\langle \xi_1, x_1 \rangle, \ldots, \langle \xi_n, x_n \rangle\Big) := \pi(\xi^n) P_{\xi^n}(x^n). \tag{4.1}$$

We adopt shorthand notation for events: when we write $P(S)$, where S is a subsequence of ξ^n and/or of x^n, this means the probability under P of the set of sequences of pairs which match S exactly. For example, the marginal probability of a sequence of outcomes is:

$$P(x^n) = \sum_{\xi^n \in \Xi^n} P(\xi^n, x^n) = \sum_{\xi^n} P\Big(\langle \xi_1, x_1 \rangle, \ldots, \langle \xi_n, x_n \rangle \Big). \tag{4.2}$$

Compare this to the usual Bayesian statistics, where a model $\{P_\theta \mid \theta \in \Theta\}$ is also endowed with a prior distribution w on Θ. Then, after observing outcomes x^n, inference is based on the posterior $P(\boldsymbol{\theta}|x^n)$ on the parameter, which is never actually observed. Our approach is exactly the same, but we always consider $\Theta = \Xi^\infty$. Thus as usual our predictions are based on the posterior $P(\boldsymbol{\xi}^\infty|x^n)$. However, since the predictive distribution of $\boldsymbol{x}_{n+1}$ only depends on ξ_{n+1} (and x^n) we always marginalise as follows:

$$P(\xi_{n+1}|x^n) = \frac{P(\xi_{n+1}, x^n)}{P(x^n)} = \frac{\sum_{\xi^n} P(\xi^n, x^n) \cdot \pi(\xi_{n+1}|\xi^n)}{\sum_{\xi^n} P(\xi^n, x^n)}. \tag{4.3}$$

At each moment in time we predict the data using the posterior, which is a mixture over our experts' predictions. Ideally, the ES-prior π should be chosen such that the posterior coincides with the optimal mixture weights of the experts at each sample size. The traditional interpretation of our ES-prior as a representation of belief about an unknown "true" expert sequence is tenuous, as normally the experts do not generate the data, they only predict it. Moreover, by mixing over different expert sequences, it is often possible to predict significantly better than by using any single sequence of experts, a feature that is crucial to the performance of many of the models that will be described below and in Section 4.3. In the remainder of this chapter we motivate ES-priors by giving performance guarantees in the form of bounds on running time and loss.

4.1.1 Examples

We now show how two ubiquitous models can be rendered as ES-priors.

Example 9 (Bayesian Mixtures)**.** Let Ξ be a set of experts, and let P_ξ be a PFS for each $\xi \in \Xi$. Suppose that we do not know which expert will make the best predictions. Following the usual Bayesian methodology, we combine their predictions by conceiving a prior w on Ξ, which (depending on the adhered philosophy) may or may not be interpreted as an expression of one's beliefs in this respect. Then the standard Bayesian mixture P_{bayes} is given by

$$P_{\text{bayes}}(x^n) = \sum_{\xi \in \Xi} P_\xi(x^n) w(\xi), \quad \text{where} \quad P_\xi(x^n) = \prod_{i=1}^{n} P_\xi(x_i|x^i). \tag{4.4}$$

The Bayesian mixture is not an ES-joint, but it can easily be transformed into one by using the ES-prior that assigns probability $w(\xi)$ to the identically-ξ sequence for each $\xi \in \Xi$:

$$\pi_{\text{bayes}}(\xi^n) = \begin{cases} w(k) & \text{if } \xi_i = k \text{ for all } i = 1, \ldots, n, \\ 0 & \text{o.w.} \end{cases}$$

We will use the adjective "Bayesian" generously throughout this paper, but when we write *the standard Bayesian ES-prior* this always refers to π_{bayes}. $\Diamond$

Example 10 (Elementwise Mixtures)**.** The *elementwise mixture*[1] is formed from some mixture weights $\alpha \in \triangle(\Xi)$ by

$$P_{\mathrm{mix},\alpha}(x^n) := \prod_{i=1}^{n} P_\alpha(x_i|x^{i-1}), \quad \text{where} \quad P_\alpha(x_i|x^{i-1}) = \sum_{\xi\in\Xi} P_\xi(x_i|x^{i-1})\alpha(\xi).$$

In the preceding definition, it may seem that elementwise mixtures do not fit in the framework of ES-priors. But we can rewrite this definition in the required form as follows:

$$\begin{aligned} P_{\mathrm{mix},\alpha}(x^n) &= \prod_{i=1}^{n}\sum_{\xi\in\Xi} P_\xi(x_i|x^{i-1})\alpha(\xi) = \sum_{\xi^n\in\Xi^n}\prod_{i=1}^{n} P_{\xi_i}(x_i|x^{i-1})\alpha(\xi_i) \\ &= \sum_{\xi^n} P_{\xi^n}(x^n)\pi_{\mathrm{mix},\alpha}(\xi^n), \end{aligned} \tag{4.5a}$$

which is the ES-joint based on the prior

$$\pi_{\mathrm{mix},\alpha}(\xi^n) := \prod_{i=1}^{n}\alpha(\xi_i). \tag{4.5b}$$

Thus, the ES-prior for elementwise mixtures is just the multinomial distribution with mixture weights α. ◊

We mentioned above that ES-priors cannot be interpreted as expressions of belief about individual expert sequences; this is a prime example where the ES-prior is crafted such that its posterior $\pi_{\mathrm{mix},\alpha}(\xi_{n+1}|\xi^n)$ exactly coincides with the desired *mixture* of experts.

4.2 Expert Tracking using HMMs

We explained in the previous section how expert tracking can be implemented using expert sequence priors. In this section we specify ES-priors using hidden Markov models (HMMs). The advantage of using HMMs is that the complexity of the resulting expert tracking procedure can be read off directly from the structure of the HMM. We first give a short overview of the particular kind of HMMs that we use throughout this chapter. We then show how HMMs can be used to specify ES-priors. As illustrations we render the ES-priors that we obtained for Bayesian mixtures and elementwise mixtures in the previous sections as HMMs. We conclude by giving the forward algorithm for our particular kind of HMMs. In Section 4.3 we provide an overview of ES-priors and their defining HMMs that are found in the literature.

[1]These mixtures are sometimes just called mixtures, or predictive mixtures. We use the term elementwise mixtures both for descriptive clarity and to avoid confusion with Bayesian mixtures.

4.2.1 Hidden Markov Models Overview

Hidden Markov models (HMMs) are a well-known tool for specifying probability distributions on sequences with temporal structure. Furthermore, these distributions are very appealing algorithmically: many important probabilities can be computed efficiently for HMMs. These properties make HMMs ideal models of expert sequences: ES-priors. For an introduction to HMMs, see [66]. We require a slightly more general notion that incorporates silent states and forecasting systems as explained below.

We define our HMMs on a generic set of outcomes $\mathcal{O}$ to avoid confusion in later sections, where we use HMMs in two different contexts. First in Section 4.2.2, we use HMMs to define ES-priors, and instantiate $\mathcal{O}$ with the set of experts Ξ. Then in Section 4.2.4 we modify the HMM that defines the ES-prior to incorporate the experts' predictions, whereupon $\mathcal{O}$ is instantiated with the set of observable outcomes $\mathcal{X}$.

Definition 4.2.1. Let $\mathcal{O}$ be a finite set of outcomes. We call a quintuple

$$\mathbb{A} = \left\langle Q, Q_{\mathrm{p}}, \mathrm{P}_\circ, \mathrm{P}, \left\langle P_q \right\rangle_{q \in Q_{\mathrm{p}}} \right\rangle$$

a *hidden Markov model* on $\mathcal{O}$ if Q is a countable set, $Q_{\mathrm{p}} \subseteq Q$, $\mathrm{P}_\circ \in \triangle(Q)$, $\mathrm{P} : Q \to \triangle(Q)$ and P_q is an $\mathcal{O}$-forecasting system for each $q \in Q_{\mathrm{p}}$.

Terminology and Notation We call the elements of Q *states*. We call the states in Q_{p} *productive* and the other states *silent*. We call $\mathrm{P}_\circ$ the *initial distribution*, let I denote its support (i.e. $I := \left\{q \in Q \mid \mathrm{P}_\circ(q) > 0\right\}$) and call I the set of *initial states*. We call P the *stochastic transition function*. We let S_q denote the support of $\mathrm{P}(q)$, and call each $q' \in S_q$ a *direct successor* of q. We abbreviate $\mathrm{P}(q)(q')$ to $\mathrm{P}(q \to q')$. A finite or infinite sequence of states $q^\lambda \in Q^{\leq \infty}$ is called a *branch* through $\mathbb{A}$. A branch q^λ is called a *run* if either $\lambda = 0$ (so $q^\lambda = \epsilon$), or $q_1 \in I$ and $q_{i+1} \in S_{q_i}$ for all $1 \leq i < \lambda$. A finite run $q^n \neq \epsilon$ is called *a run to* q_n. For each branch q^λ, we denote by q^λ_{p} its subsequence of productive states. We denote the elements of q^λ_{p} by q^{p}_1, q^{p}_2 etc. We call an HMM *continuous* if q^∞_{p} is infinite for each infinite run q^∞.

Restriction In this chapter we only work with continuous HMMs. This restriction is necessary for the following to be well-defined.

Definition 4.2.2. An HMM $\mathbb{A}$ defines the following distribution on sequences of states. $\pi_{\mathbb{A}}(\epsilon) := 1$, and for $\lambda \geq 1$

$$\pi_{\mathbb{A}}(q^\lambda) := \mathrm{P}_\circ(q_1) \prod_{i=1}^{\lambda-1} \mathrm{P}(q_i \to q_{i+1}).$$

Then via the PFSs, $\mathbb{A}$ induces the joint distribution $P_{\mathbb{A}}$ on runs and sequences of outcomes. Let $o^n \in \mathcal{O}^n$ be a sequence of outcomes and let $q^\lambda \neq \epsilon$ be a run with at least n productive states, then

$$P_{\mathbb{A}}(o^n, q^\lambda) := \pi_{\mathbb{A}}(q^\lambda) \prod_{i=1}^{n} P_{q_i^{\mathrm{P}}}(o_i | o^{i-1}).$$

The value of $P_{\mathbb{A}}$ at arguments o^n, q^λ that do not fulfil the condition above is determined by the additivity axiom of probability.

Generative Perspective The corresponding generative viewpoint is the following. Begin by sampling an initial state q_1 from the initial distribution $\mathrm{P}_\circ$. Then iteratively sample a direct successor q_{i+1} from $\mathrm{P}(q_i)$. Whenever a productive state q_i is sampled, say the n^{th}, also sample an outcome o_n from the forecasting system P_{q_i} given all previously sampled outcomes o^{n-1}.

The Importance of Silent States Silent states can always be eliminated. Let q' be a silent state and let $R_{q'} := \{q \mid q' \in S_q\}$ be the set of states that have q' as their direct successor. Now by connecting each state $q \in R_{q'}$ to each state $q'' \in S_{q'}$ with transition probability $\mathrm{P}(q \to q')\,\mathrm{P}(q' \to q'')$ and removing q' we preserve the induced distribution on Q^∞. Now if $|R_{q'}| = 1$ or $|S_{q'}| = 1$ then q' deserves this treatment. Otherwise, the number of successors has increased, since $|R_{q'}| \cdot |S_{q'}| \geq |R_{q'}| + |S_{q'}|$, and the increase is quadratic in the worst case. Thus, silent states are important to keep our HMMs small: they can be viewed as shared common subexpressions. It is important to keep HMMs small, since the size of an HMM is directly related to the running time of standard algorithms that operate on it. These algorithms are described in the next section.

Algorithms

There are three classical tasks associated with hidden Markov models [66]. To give the complexity of algorithms for these tasks we need to specify the input size. Here we consider the case where Q is finite. The infinite case will be covered in Section 4.2.5. Let $m := |Q|$ be the number of states and $e := \sum_{q \in Q} |S_q|$ be the number of transitions with nonzero probability. The three tasks are:

1. Computing the marginal probability $P(o^n)$ of the data o^n. This task is performed by the forward algorithm. This is a dynamic programming algorithm with time complexity $O(ne)$ and space requirement $O(m)$.

2. MAP estimation: computing a sequence of states q^λ with maximal posterior weight $P(q^\lambda | o^n)$. Note that $\lambda \geq n$. This task is solved using the Viterbi algorithm, again a dynamic programming algorithm with time complexity $O(\lambda e)$ and space complexity $O(\lambda m)$.

3. Parameter estimation. Instead of a single probabilistic transition function P, one often considers a collection of transition functions $\langle \mathrm{P}_\theta \mid \theta \in \Theta \rangle$ indexed by a set of parameters Θ. In this case one often wants to find the parameter θ for which the HMM using transition function P_θ achieves highest likelihood $P(o^n|\theta)$ of the data o^n.

   This task is solved using the Baum-Welch algorithm. This is an iterative improvement algorithm (in fact an instance of Expectation Maximisation (EM)) built atop the forward algorithm (and a related dynamic programming algorithm called the backward algorithm).

Since we apply HMMs to sequential prediction, we are mainly concerned with Task 1 and occasionally with Task 2. Task 3 is outside the scope of this study.

We note that the forward and backward algorithms actually compute more information than just the marginal probability $P(o^n)$. They compute $P(q_i^{\mathrm{p}}, o^i)$ (forward) and $P(o^n|q_i^{\mathrm{p}}, o^i)$ (backward) for each $i = 1, \ldots, n$. The forward algorithm can be computed incrementally, and can thus be used for on-line prediction. Forward-backward can be used together to compute $P(q_i^{\mathrm{p}}|o^n)$ for $i = 1, \ldots, n$, a useful tool in data analysis.

Finally, we note that these algorithms are defined e.g. in [66] for HMMs without silent states and with simple distributions on outcomes instead of forecasting systems. All these algorithms can be adapted straightforwardly to our general case. We formulate the forward algorithm for general HMMs in Section 4.2.5 as an example.

4.2.2 HMMs as ES-Priors

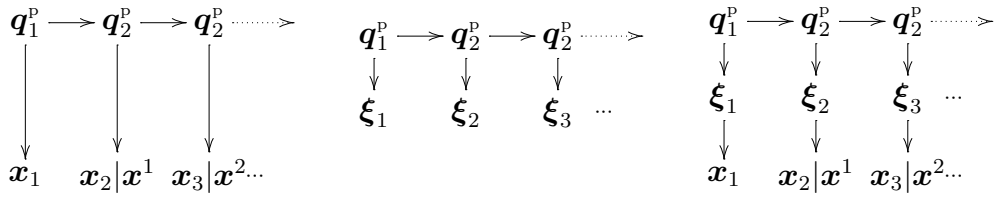
In applications HMMs are often used to model data. This is a good idea whenever there are local temporal correlations between outcomes. A graphical model depicting this approach is displayed in Figure 4.2(a).

Here we take a different approach; we use HMMs as ES-priors, that is, to specify temporal correlations between the performance of our experts. Thus instead of concrete observations our HMMs will produce sequences of experts, that are never actually observed. Figure 4.2(b). illustrates this approach.

Using HMMs as priors allows us to use the standard algorithms of Section 4.2.1 to answer questions about the prior. For example, we can use the forward algorithm to compute the prior probability of the sequence of one hundred experts that issues the first expert at all odd time-points and the second expert at all even moments.

Of course, we are often interested in questions about the data rather than about the prior. In Section 4.2.4 we show how joints based on HMM priors (Figure 4.2(c)) can be transformed into ordinary HMMs (Figure 4.2(a)) with at most a $|\Xi|$-fold increase in size, allowing us to use the standard algorithms of Section 4.2.1 not only for the experts, but for the data as well, with the same

Figure 4.2 HMMs. q_i^{p}, $\boldsymbol{\xi}_i$ and $\boldsymbol{x}_i$ are the i^{th} productive state, expert and observation.

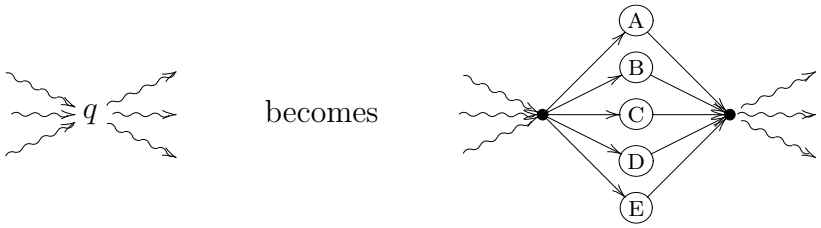


(a) Standard use of HMM (b) HMM ES-prior (c) Application to data

increase in complexity. This is the best we can generally hope for, as we now need to integrate over all possible expert sequences instead of considering only a single one. Here we first consider properties of HMMs that represent ES-priors.

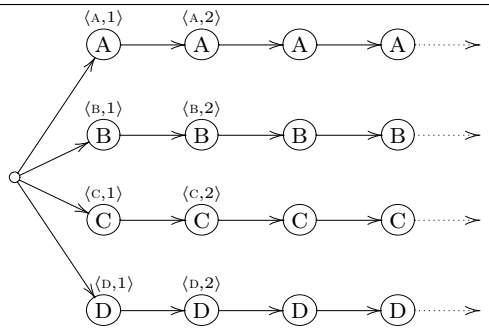
Restriction HMM priors "generate", or define the distribution on, sequences of experts. But contrary to the data, which are observed, no concrete sequence of experts is realised. This means that we cannot condition the distribution on experts in a productive state q_n^{p} on the sequence of previously produced experts ξ^{n-1}. In other words, we can only use an HMM on Ξ as an ES-prior if the forecasting systems in its states are simply distributions, so that all dependencies between consecutive experts are carried by the state. This is necessary to avoid having to sum over all (exponentially many) possible expert sequences.

Deterministic Under the restriction above, but in the presence of silent states, we can make any HMM deterministic in the sense that each forecasting system assigns probability one to a single outcome. We just replace each productive state $q \in Q_{\mathrm{p}}$ by the following gadget:



In the left diagram, the state q has distribution P_q on outcomes $\mathcal{O} = \{\mathrm{A}, \ldots, \mathrm{E}\}$. In the right diagram, the leftmost silent state has transition probability $P_q(o)$ to a state that deterministically outputs outcome o. We often make the functional relationship explicit and call $\langle Q, Q_{\mathrm{p}}, \mathrm{P}_{\mathrm{o}}, \mathrm{P}, \Lambda \rangle$ a *deterministic HMM* on $\mathcal{O}$ if $\Lambda : Q_{\mathrm{p}} \to \mathcal{O}$. Here we slightly abuse notation; the last component of a (general) HMM assigns a *PFS* to each productive state, while the last component of a deterministic HMM assigns an *outcome* to each productive states.

Figure 4.3 Combination of four experts using a standard Bayesian mixture.



Sequential prediction using a general HMM or its deterministic counterpart costs the same amount of work: the $|\mathcal{O}|$-fold increase in the number of states is compensated by the $|\mathcal{O}|$-fold reduction in the number of outcomes that need to be considered per state.

Diagrams Deterministic HMMs can be graphically represented by pictures. In general, we draw a node N_q for each state q. We draw a small black dot, e.g. •, for a silent state, and an ellipse labelled $\Lambda(q)$, e.g. (D), for a productive state. We draw an arrow from N_q to $N_{q'}$ if q' is a direct successor of q. We often reify the initial distribution $\mathrm{P}_\circ$ by including a virtual node, drawn as an open circle, e.g. ∘, with an outgoing arrow to N_q for each initial state $q \in I$. The transition probability $P(q \to q')$ is not displayed in the graph.

4.2.3 Examples

We are now ready to give the deterministic HMMs that correspond to the ES-priors of our earlier examples from Section 4.1.1: Bayesian mixtures and elementwise mixtures with fixed parameters.

Example 11 (HMM for Bayesian Mixtures)**.** The Bayesian mixture ES-prior π_{bayes} as introduced in Example 9 represents the hypothesis that a single expert predicts best for all sample sizes. A simple deterministic HMM that generates the prior π_{bayes} is given by $\mathbb{A}_{\text{bayes}} = \langle Q, Q_{\text{p}}, \mathrm{P}, \mathrm{P}_\circ, \Xi, \Lambda \rangle$, where

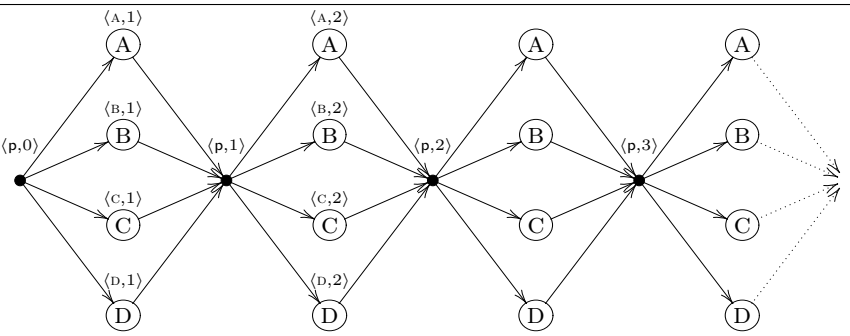
$$Q = Q_{\text{p}} = \Xi \times \mathbb{Z}^+ \qquad \mathrm{P}\left(\langle \xi, n \rangle \to \langle \xi, n+1 \rangle\right) = 1 \tag{4.6a}$$

$$\Lambda(\xi, n) = \xi \qquad \mathrm{P}_\circ(\xi, 1) = w(\xi) \tag{4.6b}$$

The diagram of (4.6) is displayed in Figure 4.3. From the picture of the HMM it is clear that it computes the Bayesian mixture. Hence, using (4.4), the loss of the HMM with prior w is bounded for all x^n by

$$-\log P_{\mathbb{A}_{\text{bayes}}}(x^n) + \log P_\xi(x^n) \leq -\log w(\xi) \qquad \text{for all experts } \xi. \tag{4.7}$$

Figure 4.4 Combination of four experts using a fixed elementwise mixture



In particular this bound holds for $\hat{\xi} = \arg\max_\xi P_\xi(x^n)$, so we predict as well as the single best expert with *constant* overhead. Also $P_{\mathbb{A}_{\text{bayes}}}(x^n)$ can obviously be computed in $O(n\,|\Xi|)$ using its definition (4.4). We show in Section 4.2.5 that computing it using the HMM prior above gives the same running time $O(n\,|\Xi|)$, a perfect match. ♢

Example 12 (HMM for Elementwise Mixtures)**.** We now present the deterministic HMM $\mathbb{A}_{\text{mix},\alpha}$ that implements the ES-prior $\pi_{\text{mix},\alpha}$ of Example 10. Its diagram is displayed in Figure 4.4. The HMM has a single silent state per outcome, and its transition probabilities are the mixture weights α. Formally, $\mathbb{A}_{\text{mix},\alpha}$ is given using $Q = Q_{\text{s}} \cup Q_{\text{p}}$ by

$$Q_{\text{s}} = \{\mathsf{p}\} \times \mathbb{N} \quad Q_{\text{p}} = \Xi \times \mathbb{Z}^+ \quad \text{P}_\circ(\mathsf{p}, 0) = 1 \quad \Lambda(\xi, n) = \xi \tag{4.8a}$$

$$\text{P}\begin{pmatrix} \langle \mathsf{p}, n\rangle \to \langle \xi, n+1\rangle \\ \langle \xi, n\rangle \to \langle \mathsf{p}, n\rangle \end{pmatrix} = \begin{pmatrix} \alpha(\xi) \\ 1 \end{pmatrix} \tag{4.8b}$$

The vector-style definition of P is shorthand for one P per line. We show in Section 4.2.5 that this HMM allows us to compute $P_{\mathbb{A}_{\text{mix},\alpha}}(x^n)$ in time $O(n\,|\Xi|)$. ♢

4.2.4 The HMM for Data

We obtain our model for the data (Figure 4.2(c)) by composing an HMM prior on Ξ^∞ with a PFS P_ξ for each expert $\xi \in \Xi$. We now show that the resulting marginal distribution on data can be implemented by a single HMM on $\mathcal{X}$ (Figure 4.2(a)) *with the same number of states as the HMM prior*. Let P_ξ be an $\mathcal{X}$-forecasting system for each $\xi \in \Xi$, and let the ES-prior $\pi_{\mathbb{A}}$ be given by the deterministic HMM $\mathbb{A} = \langle Q, Q_{\text{p}}, \text{P}_\circ, \text{P}, \Lambda\rangle$ on Ξ. Then the marginal distribution of the data (see (4.1)) is given by

$$P_{\mathbb{A}}(x^n) = \sum_{\xi^n} \pi_{\mathbb{A}}(\xi^n) \prod_{i=1}^{n} P_{\xi_i}(x_i|x^{i-1}).$$

The HMM $\mathbb{X} := \left\langle Q, Q_{\text{p}}, \text{P}_\circ, \text{P}, \left\langle P_{\Lambda(q)}\right\rangle_{q\in Q_{\text{p}}}\right\rangle$ on $\mathcal{X}$ induces the same marginal distribution (see Definition 4.2.2). That is, $P_{\mathbb{X}}(x^n) = P_{\mathbb{A}}(x^n)$. Moreover, $\mathbb{X}$ contains

only the forecasting systems that also exist in $\mathbb{A}$ and it retains the structure of $\mathbb{A}$. In particular this means that the HMM algorithms of Section 4.2.1 have the *same* running time on the prior $\mathbb{A}$ as on the marginal $\mathbb{X}$.

4.2.5 The Forward Algorithm and Sequential Prediction

We claimed in Section 4.2.1 that the standard HMM algorithms could easily be extended to our HMMs with silent states and forecasting systems. In this section we give the main example: the forward algorithm. We will also show how it can be applied to sequential prediction. Recall that the forward algorithm computes the marginal probability $P(x^n)$ for fixed x^n. On the other hand, sequential prediction means predicting the next *observation* $\boldsymbol{x}_{n+1}$ for given data x^n, i.e. computing its distribution. For this it suffices to predict the next *expert* $\boldsymbol{\xi}_{n+1}$; we then simply predict $\boldsymbol{x}_{n+1}$ by averaging the expert's predictions accordingly: $P(x_{n+1}|x^n) = \mathrm{E}[P_{\boldsymbol{\xi}_{n+1}}(x_{n+1}|x^n)]$.

We first describe the preprocessing step called *unfolding* and introduce notation for nodes. We then give the forward algorithm, prove its correctness and analyse its running time and space requirement. The forward algorithm can be used for prediction with expert advice. We conclude by outlining the difficulty of adapting the Viterbi algorithm for MAP estimation to the expert setting.

Unfolding Every HMM can be transformed into an equivalent HMM in which each productive state is involved in the production of a unique outcome. The single node in Figure 4.6(a) is involved in the production of $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots$ In its unfolding Figure 4.6(b) the i^{th} node is only involved in producing $\boldsymbol{x}_i$. Figures 4.6(c) and 4.6(d) show HMMs that unfold to the Bayesian mixture shown in Figure 4.3 and the elementwise mixture shown in Figure 4.4. In full generality, fix an HMM $\mathbb{A}$. The *unfolding* of $\mathbb{A}$ is the HMM

$$\mathbb{A}^{\mathrm{u}} := \left\langle Q^{\mathrm{u}}, Q^{\mathrm{u}}_{\mathrm{p}}, \mathrm{P}^{\mathrm{u}}_{\circ}, \mathrm{P}^{\mathrm{u}}, \left\langle P^{\mathrm{u}}_{q} \right\rangle_{q \in Q^{\mathrm{u}}} \right\rangle,$$

where the states and productive states are given by:

$$Q^{\mathrm{u}} := \left\{ \langle q_\lambda, n\rangle \mid q^\lambda \text{ is a run through } \mathbb{A} \right\}, \quad \text{where } n = \left| q^\lambda_{\mathrm{p}} \right| \tag{4.9a}$$

$$Q^{\mathrm{u}}_{\mathrm{p}} := Q^{\mathrm{u}} \cap (Q_{\mathrm{p}} \times \mathbb{N}) \tag{4.9b}$$

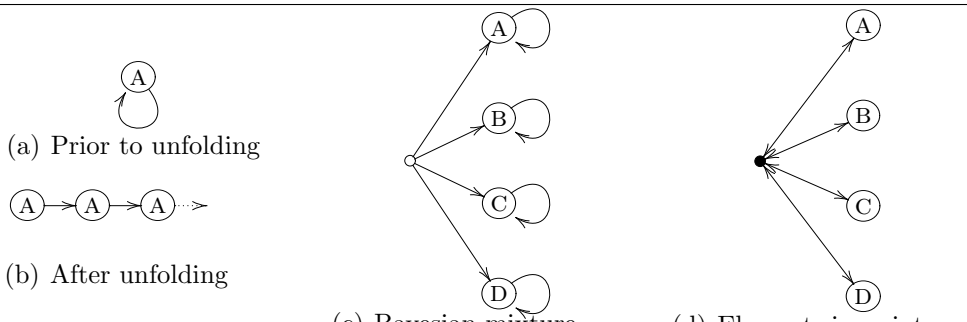
and the initial probability, transition function and forecasting systems are:

$$\mathrm{P}^{\mathrm{u}}_{\circ}\left(\langle q, 0\rangle\right) := \mathrm{P}_{\circ}(q) \tag{4.9c}$$

$$\mathrm{P}^{\mathrm{u}}\begin{pmatrix} \langle q, n\rangle \to \langle q', n+1\rangle \\ \langle q, n\rangle \to \langle q', n\rangle \end{pmatrix} := \begin{pmatrix} \mathrm{P}(q \to q') \\ \mathrm{P}(q \to q') \end{pmatrix} \tag{4.9d}$$

$$P^{\mathrm{u}}_{\langle q,n\rangle} := P_q \tag{4.9e}$$

Figure 4.5 Unfolding example



First observe that unfolding preserves the marginal: $P_{\mathbb{A}}(o^n) = P_{\mathbb{A}^{\mathrm{u}}}(o^n)$. Second, unfolding is an idempotent operation: $(\mathbb{A}^{\mathrm{u}})^{\mathrm{u}}$ is isomorphic to $\mathbb{A}^{\mathrm{u}}$. Third, unfolding renders the set of states infinite, but for each n it preserves the number of states reachable in exactly n steps.

Order The states in an unfolded HMM have earlier-later structure. Fix $q, q' \in Q^{\mathrm{u}}$. We write $q < q'$ iff there is a run to q' through q. We call $<$ the *natural order* on Q^{u}. Obviously $<$ is a partial order, furthermore it is the transitive closure of the reverse direct successor relation. It is well-founded, allowing us to perform induction on states, an essential ingredient in the forward algorithm (Algorithm 4.1) and its correctness proof (Theorem 4.2.3).

Interval Notation We introduce interval notation to address subsets of states of unfolded HMMs, as illustrated by Figure 4.6. Our notation associates each productive state with the sample size at which it produces its outcome, while the silent states fall in between. We use intervals with borders in $\mathbb{N}$. The interval contains the border $i \in \mathbb{N}$ iff the addressed set of states includes the states where the i^{th} observation is produced.

$$Q^{\mathrm{u}}_{[n,m)} := Q^{\mathrm{u}} \cap (Q \times [n,m)) \qquad Q^{\mathrm{u}}_{[n,m]} := Q^{\mathrm{u}}_{[n,m)} \cup Q^{\mathrm{u}}_{\{m\}} \tag{4.10a}$$

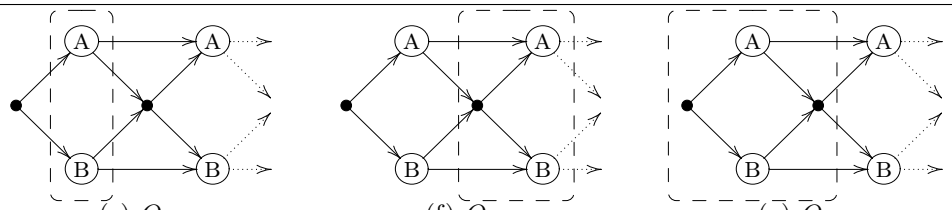
$$Q^{\mathrm{u}}_{\{n\}} := Q^{\mathrm{u}} \cap (Q_{\mathrm{p}} \times \{n\}) \qquad Q^{\mathrm{u}}_{(n,m)} := Q^{\mathrm{u}}_{[n,m)} \setminus Q^{\mathrm{u}}_{\{n\}} \tag{4.10b}$$

$$Q^{\mathrm{u}}_{(n,m]} := Q^{\mathrm{u}}_{[n,m]} \setminus Q^{\mathrm{u}}_{\{n\}} \tag{4.10c}$$

Fix $n > 0$, then $Q^{\mathrm{u}}_{\{n\}}$ is a non-empty $<$-anti-chain (i.e. its states are pairwise $<$-incomparable). Furthermore $Q^{\mathrm{u}}_{(n,n+1)}$ is empty iff $Q^{\mathrm{u}}_{\{n+1\}} = \bigcup_{q \in Q^{\mathrm{u}}_{\{n\}}} S_q$, in other words, if there are no silent states between sample sizes n and $n+1$.

The Forward Algorithm Fix an unfolded deterministic HMM prior $\mathbb{A} = \langle Q, Q_{\mathrm{p}}, \mathrm{P}_{\circ}, \mathrm{P}, \Lambda \rangle$ on Ξ, and an $\mathcal{X}$-PFS P_ξ for each expert $\xi \in \Xi$. The input consists of a sequence x^∞ that arrives sequentially. Then the forward algorithm for sequential prediction on models with silent states can be rendered as follows.

Figure 4.6 Interval notation



(e) $Q_{\{1\}}$ (f) $Q_{(1,2]}$ (g) $Q_{(0,2)}$

Analysis Consider a state $q \in Q$, say $q \in Q_{[n,n+1)}$. Initially, $q \notin \mathrm{dom}(w)$. Then at some point $w(q) \leftarrow \mathrm{P}_\circ(q)$. This happens either in the second line because $q \in I$ or in FORWARD PROPAGATION because $q \in S_u$ for some u (in this case $\mathrm{P}_\circ(q) = 0$). Then $w(q)$ accumulates weight as its direct predecessors are processed in FORWARD PROPAGATION. At some point all its predecessors have been processed. If q is productive we call its weight at this point (that is, just before LOSS UPDATE) $\mathrm{Alg}(\mathbb{A}, x^{n-1}, q)$. Finally, FORWARD PROPAGATION removes q from the domain of w, never to be considered again. We call the weight of q (silent or productive) just before removal $\mathrm{Alg}(\mathbb{A}, x^n, q)$.

Note that we associate *two* weights with each productive state $q \in Q_{\{n\}}$: the weight $\mathrm{Alg}(\mathbb{A}, x^{n-1}, q)$ is calculated *before* outcome n is observed, while on the other hand $\mathrm{Alg}(\mathbb{A}, x^n, q)$ denotes the weight *after* the loss update incorporates outcome n.

Theorem 4.2.3. *Fix an HMM prior $\mathbb{A}$, $n \in \mathbb{N}$ and $q \in Q_{[n,n+1]}$, then*

$$\mathrm{Alg}(\mathbb{A}, x^n, q) = P_{\mathbb{A}}(x^n, q).$$

Note that the theorem applies twice to productive states.

Proof. By $<$-induction on states. Let $q \in Q_{(n,n+1]}$, and suppose that the theorem holds for all $q' < q$. Let $B_q = \left\{q' \mid \mathrm{P}(q' \to q) > 0\right\}$ be the set of direct predecessors of q. Observe that $B_q \subseteq Q_{[n,n+1)}$. The weight that is accumulated by FORWARD PROPAGATION(n) onto q is:

$$\begin{aligned}\mathrm{Alg}(\mathbb{A}, x^n, q) &= \mathrm{P}_\circ(q) + \sum_{q' \in B_q} \mathrm{P}(q' \to q)\, \mathrm{Alg}(\mathbb{A}, x^n, q') \\ &= \mathrm{P}_\circ(q) + \sum_{q' \in B_q} \mathrm{P}(q' \to q) P_{\mathbb{A}}(x^n, q') \;=\; P_{\mathbb{A}}(x^n, q).\end{aligned}$$

The second equality follows from the induction hypothesis. Additionally if $q \in Q_{\{n\}}$ is productive, say $\Lambda(q) = \xi$, then after LOSS UPDATE(n) its weight is:

$$\begin{aligned}\mathrm{Alg}(\mathbb{A}, x^n, q) &= P_\xi(x_n | x^{n-1})\, \mathrm{Alg}(\mathbb{A}, x^{n-1}, q) \\ &= P_\xi(x_n | x^{n-1}) P_{\mathbb{A}}(x^{n-1}, q) \;=\; P_{\mathbb{A}}(x^n, q).\end{aligned}$$

□

The second inequality holds by induction on n, and the third by Definition 4.2.2.

Complexity We are now able to sharpen the complexity results as listed in Section 4.2.1, and extend them to infinite HMMs. Fix $\mathbb{A}$, $n \in \mathbb{N}$. The forward algorithm processes each state in $Q_{[0,n)}$ once, and at that point this state's weight is distributed over its successors. Thus, the running time is proportional to $\sum_{q \in Q_{[0,n)}} \left|S_q\right|$. The forward algorithm keeps $\left|\mathrm{dom}(w)\right|$ many weights. But at each sample size n, $\mathrm{dom}(w) \subseteq Q_{[n,n+1]}$. Therefore the space needed is at most proportional to $\max_{m<n} \left|Q_{[m,m+1]}\right|$. For both Bayes (Example 11) and elementwise mixtures (Example 12) one may read from the figures that $\sum_{q \in Q_{[n,n+1)}} \left|S_q\right|$ and $\left|Q_{[n,n+1)}\right|$ are $O(|\Xi|)$, so we indeed get the claimed running time $O(n\,|\Xi|)$ and space requirement $O(|\Xi|)$.

MAP Estimation The forward algorithm described above computes the probability of the data, that is

$$P(x^n) \;=\; \sum_{q^\lambda : q_\lambda \in Q_{\{n\}}} P(x^n, q^\lambda).$$

Instead of the entire sum, we are sometimes interested in the sequence of states q^λ that contributes most to it:

$$\arg\max_{q^\lambda} P(x^n, q^\lambda) \;=\; \arg\max_{q^\lambda} P(x^n|q^\lambda)\pi(q^\lambda).$$

The Viterbi algorithm [66] is used to compute the most likely sequence of states for HMMs. It can be easily adapted to handle silent states. However, we may also write

$$P(x^n) \;=\; \sum_{\xi^n} P(x^n, \xi^n),$$

and wonder about the sequence of experts ξ^n that contributes most. This problem is harder because in general, a single sequence of experts can be generated by many different sequences of states. This is unrelated to the presence of the silent states, but due to different states producing the same expert simultaneously (i.e. in the same $Q_{\{n\}}$). So we cannot use the Viterbi algorithm as it is. The Viterbi algorithm can be extended to compute the MAP expert sequence for general HMMs, but the resulting running time explodes. Still, the MAP ξ^n can be sometimes be obtained efficiently by exploiting the structure of the HMM at hand. The first example is the unambiguous HMMs. A deterministic HMM is *ambiguous* if it has two runs that agree on the sequence of *experts* produced, but not on the sequence of *productive states*. The straightforward extension of the Viterbi algorithm works for unambiguous HMMs. The second important example is the (ambiguous) switch HMM that we introduce in Section 4.4.1. We show how to compute its MAP expert sequence in Section 4.4.1.

4.3 Zoology

Perhaps the simplest way to predict using a number of experts is to pick one of them and mirror her predictions exactly. Beyond this "fixed expert model", we have considered two methods of combining experts so far, namely taking Bayesian mixtures, and taking elementwise mixtures as described in Section 4.2.3. Figure 4.1 shows these and a number of other, more sophisticated methods that fit in our framework. The arrows indicate which methods are generalised by which other methods. They have been partitioned in groups that can be computed in the same amount of time using HMMs.

We have presented two examples so far, the Bayesian mixture and the elementwise mixture with fixed coefficients (Examples 11 and 12). The latter model is parameterised. Choosing a fixed value for the parameter beforehand is often difficult. The first model we discuss learns the optimal parameter value on-line, at the cost of only a small additional loss. We then proceed to discuss a number of important existing expert models.

4.3.1 Universal Elementwise Mixtures

A distribution is "universal" for a family of distributions if it incurs small additional loss compared to the best member of the family. A standard Bayesian mixture constitutes the simplest example. It is universal for the fixed expert model, where the unknown parameter is the used expert. In (4.7) we showed that the additional loss is at most $\log|\Xi|$ for the uniform prior.

In Example 12 we described elementwise mixtures with fixed coefficients as ES-priors. Prior knowledge about the mixture coefficients is often unavailable. We now expand this model to learn the optimal mixture coefficients from the data. To this end we place a prior distribution w on the space of mixture weights $\triangle(\Xi)$. Using (4.5) we obtain the following marginal distribution:

$$P_{\text{umix}}(x^n) = \int_{\triangle(\Xi)} P_{\text{mix},\alpha}(x^n) w(\alpha)\,\mathrm{d}\alpha = \int_{\triangle(\Xi)} \sum_{\xi^n} P_{\xi^n}(x^n)\pi_{\text{mix},\alpha}(\xi^n) w(\alpha)\,\mathrm{d}\alpha$$
$$= \sum_{\xi^n} P_{\xi^n}(x^n)\pi_{\text{umix}}(\xi^n), \quad \text{where} \quad \pi_{\text{umix}}(\xi^n) = \int_{\triangle(\Xi)} \pi_{\text{mix},\alpha}(\xi^n) w(\alpha)\,\mathrm{d}\alpha. \quad (4.11)$$

Thus P_{umix} is the ES-joint with ES-prior π_{umix}. This applies more generally: parameters α can be integrated out of an ES-prior regardless of which experts are used, since the expert predictions $P_{\xi^n}(x^n)$ do not depend on α.

We will proceed to calculate a loss bound for the universal elementwise mixture model, showing that it really is universal. After that we will describe how it can be implemented as a HMM.

A Loss Bound

In this section we relate the loss of a universal elementwise mixture with the loss obtained by the maximum likelihood elementwise mixture. While mixture models occur regularly in the statistical literature, we are not aware of any appearance in universal prediction. Therefore, to the best of our knowledge, the following simple loss bound is new. Our goal is to obtain a bound in terms of properties of the prior. A difficulty here is that there are many expert sequences exhibiting mixture frequencies close to the maximum likelihood mixture weights, so that each individual expert sequence contributes relatively little to the total probability (4.11). The following theorem is a general tool to deal with such situations.

Theorem 4.3.1. *Let π, ρ be ES-priors s.t. ρ is zero whenever π is. Then for all x^n, reading $0/0 = 0$,*

$$-\log \frac{P_\pi(x^n)}{P_\rho(x^n)} \;\leq\; E_{P_\rho}\left[-\log \frac{\pi(\boldsymbol{\xi}^n)}{\rho(\boldsymbol{\xi}^n)}\middle| x^n\right] \;\leq\; -\log \max_{\xi^n} \frac{\pi(\xi^n)}{\rho(\xi^n)}.$$

Proof. For non-negative $a_1, \ldots a_m$ and $b_1, \ldots b_m$:

$$\left(\sum_{i=1}^m a_i\right) \log \frac{\sum_{i=1}^m a_i}{\sum_{i=1}^m b_i} \;\leq\; \sum_{i=1}^m a_i \log \frac{a_i}{b_i} \;\leq\; \left(\sum_{i=1}^m a_i\right) \max_i \log \frac{a_i}{b_i}. \tag{4.12}$$

The first inequality is the log sum inequality [25, Theorem 2.7.1]. The second inequality is a simple overestimation. We now apply (4.12) substituting $m \mapsto |\Xi^n|$, $a_{\xi^n} \mapsto P_\rho(x^n, \xi^n)$ and $b_{\xi^n} \mapsto P_\pi(x^n, \xi^n)$ and divide by $\sum_{i=1}^m a_i$ to complete the proof. □

Using this theorem, we obtain a loss bound for universal elementwise mixtures that can be computed prior to observation and without reference to the experts' PFSs.

Corollary 4.3.2. *Let P_{umix} be the universal elementwise mixture model defined using the $(\frac{1}{2}, \ldots, \frac{1}{2})$-Dirichlet prior (that is, Jeffreys' prior) as the prior $w(\alpha)$ in (4.11). Let $\hat{\alpha}(x^n)$ maximise the likelihood $P_{\text{mix},\alpha}(x^n)$ w.r.t. α. Then for all x^n the additional loss incurred by the universal elementwise mixture is bounded thus*

$$-\log P_{\text{umix}}(x^n) + \log P_{\text{mix},\hat{\alpha}(x^n)}(x^n) \;\leq\; \frac{|\Xi| - 1}{2} \log \frac{n}{\pi} + c$$

for a fixed constant c.

Proof. By Theorem 4.3.1

$$-\log P_{\text{umix}}(x^n) + \log P_{\text{mix},\hat{\alpha}(x^n)}(x^n) \;\leq\; \max_{\xi^n} \left(-\log \pi_{\text{umix}}(\xi^n) + \log \pi_{\text{mix},\hat{\alpha}(x^n)}(\xi^n)\right). \tag{4.13}$$

We now bound the right hand side. Let $\hat{\alpha}(\xi^n)$ maximise $\pi_{\mathrm{mix},\alpha}(\xi^n)$ w.r.t. α. Then for all x^n and ξ^n

$$\pi_{\mathrm{mix},\hat{\alpha}(x^n)}(\xi^n) \;\leq\; \pi_{\mathrm{mix},\hat{\alpha}(\xi^n)}(\xi^n). \tag{4.14}$$

For the $\left(\frac{1}{2},\ldots,\frac{1}{2}\right)$-Dirichlet prior, for all ξ^n

$$-\log \pi_{\mathrm{umix}}(\xi^n) + \log \pi_{\mathrm{mix},\hat{\alpha}(\xi^n)}(\xi^n) \;\leq\; \frac{|\Xi|-1}{2}\log\frac{n}{\pi} + c$$

for some fixed constant c (see e.g. [100]) Combination with (4.14) and (4.13) completes the proof. □

Since the overhead incurred as a penalty for not knowing the optimal parameter $\hat{\alpha}$ in advance is only logarithmic, we find that P_{umix} is strongly universal for the fixed elementwise mixtures.

HMM

While universal elementwise mixtures can be described using the ES-prior π_{umix} defined in (4.11), unfortunately any HMM that computes it needs a state for each possible count vector, and is therefore huge if the number of experts is large. The HMM $\mathbb{A}_{\mathrm{umix}}$ for an arbitrary number of experts using the $\left(\frac{1}{2},\ldots,\frac{1}{2}\right)$-Dirichlet prior is given using $Q = Q_{\mathrm{s}} \cup Q_{\mathrm{p}}$ by

$$Q_{\mathrm{s}} = \mathbb{N}^{\Xi} \quad Q_{\mathrm{p}} = \mathbb{N}^{\Xi} \times \Xi \quad \mathrm{P}_\circ(\mathbf{0}) = 1 \quad \Lambda(\vec{n},\xi) = \xi \tag{4.15}$$

$$\mathrm{P}\begin{pmatrix} \langle \vec{n} \rangle \to \langle \vec{n},\xi \rangle \\ \langle \vec{n},\xi \rangle \to \langle \vec{n} + \mathbf{1}_\xi \rangle \end{pmatrix} = \begin{pmatrix} \frac{1/2+n_\xi}{|\Xi|/2+\sum_\xi n_\xi} \\ 1 \end{pmatrix} \tag{4.16}$$

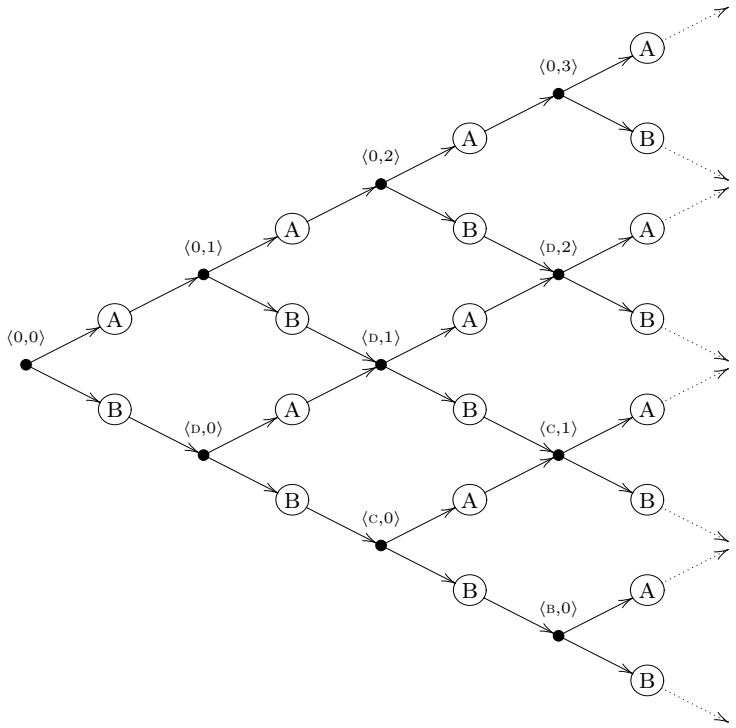
We write $\mathbb{N}^{\Xi}$ for the set of assignments of counts to experts; $\mathbf{0}$ for the all zero assignment, and $\mathbf{1}_\xi$ marks one count for expert ξ. We show the diagram of $\mathbb{A}_{\mathrm{umix}}$ for the practical limit of two experts in Figure 4.7. In this case, the forward algorithm has running time $O(n^2)$. Each productive state in Figure 4.7 corresponds to a vector of two counts (n_1, n_2) that sum to the sample size n, with the interpretation that of the n experts, the first was used n_1 times while the second was used n_2 times. These counts are a sufficient statistic for the multinomial model: per (4.5b) and (4.11) the probability of the next expert only depends on the counts, and these probabilities are exactly the successor probabilities of the silent states (4.16).

Other priors on α are possible. In particular, when all mass is placed on a single value of α, we retrieve the elementwise mixture with fixed coefficients.

4.3.2 Fixed Share

The first publication that considers a scenario where the best predicting expert may change with the sample size is Herbster and Warmuth's paper on *tracking*

Figure 4.7 Combination of two experts using a universal elementwise mixture



the best expert [45, 46]. They partition the data of size n into m segments, where each segment is associated with an expert, and give algorithms to predict almost as well as the best *partition* where the best expert is selected per segment. They give two algorithms called fixed share and dynamic share. The second algorithm does not fit in our framework; furthermore its motivation applies only to loss functions other than log-loss. We focus on fixed share, which is in fact identical to our algorithm applied to the HMM depicted in Figure 4.8, where all arcs *into* the silent states have fixed probability $\alpha \in [0, 1]$ and all arcs *from* the silent states have some fixed distribution w on Ξ.[2] The same algorithm is also described as an instance of the Aggregating Algorithm in [95]. Fixed share reduces to fixed elementwise mixtures by setting $\alpha = 1$ and to Bayesian mixtures by setting $\alpha = 0$. Formally:

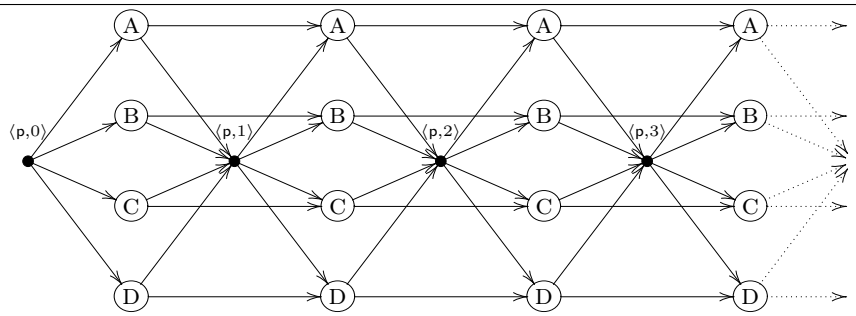
$$\begin{aligned} Q &= \Xi \times \mathbb{Z}^+ \cup \{\mathsf{p}\} \times \mathbb{N} & \mathrm{P}_\circ(\mathsf{p}, 0) &= 1 \\ Q_{\mathrm{P}} &= \Xi \times \mathbb{Z}^+ & \Lambda(\xi, n) &= \xi \end{aligned} \tag{4.17a}$$

$$\mathrm{P}\begin{pmatrix} \langle \mathsf{p}, n\rangle \to \langle \xi, n+1\rangle \\ \langle \xi, n\rangle \to \langle \mathsf{p}, n\rangle \\ \langle \xi, n\rangle \to \langle \xi, n+1\rangle \end{pmatrix} = \begin{pmatrix} w(\xi) \\ \alpha \\ 1-\alpha \end{pmatrix} \tag{4.17b}$$

Each productive state represents that a particular expert is used at a certain

[2] This is actually a slight generalisation: the original algorithm uses a uniform $w(\xi) = 1/|\Xi|$.

Figure 4.8 Combination of four experts using the fixed share algorithm



sample size. Once a transition to a silent state is made, all history is forgotten and a new expert is chosen according to w.[3]

Let $\hat{L}$ denote the loss achieved by the best partition, with switching rate $\alpha^* := m/(n-1)$. Let $L_{\mathrm{fs},\alpha}$ denote the loss of fixed share with uniform w and parameter α. Herbster and Warmuth prove[4]

$$L_{\mathrm{fs},\alpha} - \hat{L} \;\leq\; (n-1)H(\alpha^*,\alpha) + (m-1)\log(|\Xi|-1) + \log|\Xi|\,,$$

which we for brevity loosen slightly to

$$L_{\mathrm{fs},\alpha} - \hat{L} \;\leq\; nH(\alpha^*,\alpha) + m\log|\Xi|\,. \tag{4.18}$$

Here $H(\alpha^*,\alpha) = -\alpha^*\log\alpha - (1-\alpha^*)\log(1-\alpha)$ is the cross entropy. The best loss guarantee is obtained for $\alpha = \alpha^*$, in which case the cross entropy reduces to the binary entropy $H(\alpha)$. A drawback of the method is that the optimal value of α has to be known in advance in order to minimise the loss. In Sections Section 4.3.3 and Section 4.4 we describe a number of generalisations of fixed share that avoid this problem.

4.3.3 Universal Share

Volf and Willems describe universal share (they call it *the switching method*) [94], which is very similar to a probabilistic version of Herbster and Warmuth's fixed share algorithm, except that they put a prior on the unknown parameter, with the result that their algorithm adaptively learns the optimal value during prediction.

In [15], Bousquet shows that the overhead for not knowing the optimal parameter value is equal to the overhead of a Bernoulli universal distribution. Let $L_{\mathrm{fs},\alpha} = -\log P_{\mathrm{fs},\alpha}(x^n)$ denote the loss achieved by the fixed share algorithm with parameter α on data x^n, and let $L_{\mathrm{us}} = -\log P_{\mathrm{us}}(x^n)$ denote the

[3] Contrary to the original fixed share, we allow switching to the same expert. In the HMM framework this is necessary to achieve running-time $O(n\,|\Xi|)$. Under uniform w, non-reflexive switching with fixed rate α can be simulated by reflexive switching with fixed rate $\beta = \frac{\alpha|\Xi|}{|\Xi|-1}$ (provided $\beta \leq 1$). For non-uniform w, the rate becomes expert-dependent.

[4] This bound can be obtained for the fixed share HMM using the previous footnote.

loss of universal share, where $P_{\text{us}}(x^n) = \int P_{\text{fs},\alpha}(x^n) w(\alpha)\,\mathrm{d}\alpha$ with Jeffreys' prior $w(\alpha) = \alpha^{-1/2}(1-\alpha)^{-1/2}/\pi$ on $[0,1]$. Then

$$L_{\text{us}} - \min_{\alpha} L_{\text{fs},\alpha} \leq 1 + \tfrac{1}{2}\log n. \tag{4.19}$$

Thus P_{us} is universal for the model $\bigl\{P_{\text{fs},\alpha} \mid \alpha \in [0,1]\bigr\}$ that consists of all ES-joints where the ES-priors are distributions with a fixed switching rate.

Universal share requires quadratic running time $O(n^2\,|\Xi|)$, restricting its use to moderately small data sets.

In [63], Monteleoni and Jaakkola place a discrete prior on the parameter that divides its mass over $\sqrt{n}$ well-chosen points, in a setting where the ultimate sample size n is known beforehand. This way they still manage to achieve (4.19) up to a constant, while reducing the running time to $O(n\sqrt{n}\,|\Xi|)$. In [16], Bousquet and Warmuth describe yet another generalisation of expert tracking; they derive good loss bounds in the situation where the best experts for each section in the partition are drawn from a small pool.

The HMM for universal share with the $\left(\frac{1}{2},\frac{1}{2}\right)$-Dirichlet prior on the switching rate α is displayed in Figure 4.9. It is formally specified (using $Q = Q_{\text{s}} \cup Q_{\text{p}}$) by:

$$\begin{aligned} Q_{\text{s}} &= \{\mathsf{p},\mathsf{q}\} \times \bigl\{\langle m,n\rangle \in \mathbb{N}^2 \mid m \leq n\bigr\} \\ Q_{\text{p}} &= \quad \Xi \quad \times \bigl\{\langle m,n\rangle \in \mathbb{N}^2 \mid m < n\bigr\} \end{aligned} \tag{4.20a}$$

$$\Lambda(\xi,m,n) = \xi \qquad \mathrm{P}_\circ(\mathsf{p},0,0) = 1 \tag{4.20b}$$

$$\mathrm{P}\begin{pmatrix} \langle \mathsf{p},m,n\rangle \to \langle \xi,m,n+1\rangle \\ \langle \mathsf{q},m,n\rangle \to \langle \mathsf{p},m+1,n\rangle \\ \langle \xi,m,n\rangle \to \langle \mathsf{q},m,n\rangle \\ \langle \xi,m,n\rangle \to \langle \xi,m,n+1\rangle \end{pmatrix} = \begin{pmatrix} w(\xi) \\ 1 \\ (m+\frac{1}{2})/\,n \\ (n-m-\frac{1}{2})/\,n \end{pmatrix} \tag{4.20c}$$

Each productive state $\langle \xi,n,m\rangle$ represents the fact that at sample size n expert ξ is used, while there have been m switches in the past. Note that the last two lines of (4.20c) are subtly different from the corresponding topmost line of (4.16). In a sample of size n there are n possible positions to use a given expert, while there are only $n-1$ possible switch positions.

The presence of the switch count in the state is the new ingredient compared to fixed share. It allows us to adapt the switching probability to the data, but it also renders the number of states quadratic. We discuss reducing the number of states without sacrificing much performance in Section 4.5.1.

4.3.4 Overconfident Experts

In [95], Vovk considers overconfident experts. In this scenario, there is a single unknown best expert, except that this expert sometimes makes wild (overcategorical) predictions. We assume that the rate at which this happens is a

Figure 4.9 Combination of four experts using universal share

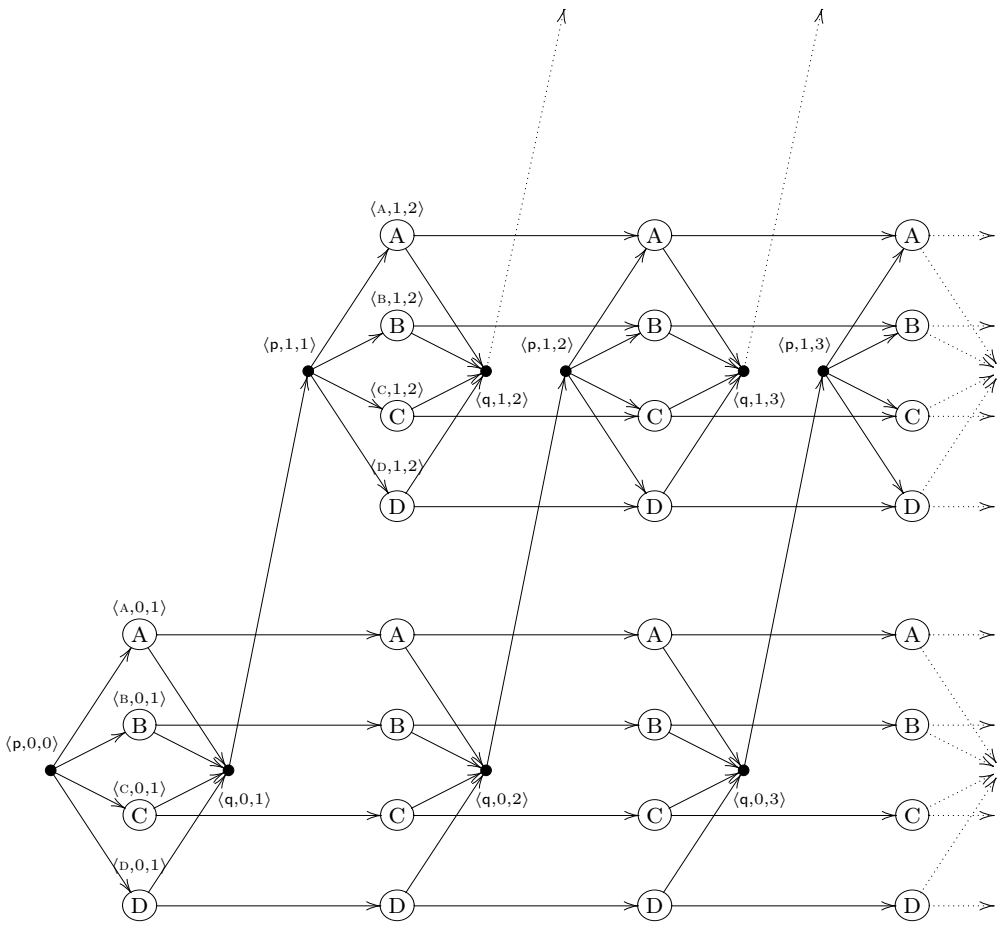
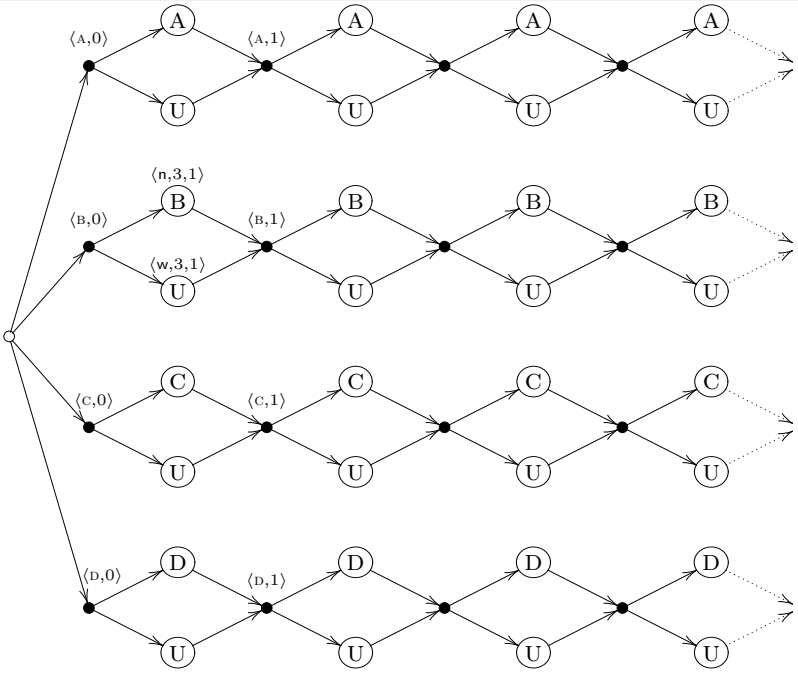


Figure 4.10 Combination of four overconfident experts



known constant α. The overconfident expert model is an attempt to mitigate the wild predictions using an additional "safe" expert $\text{U} \in \Xi$, who always issues the uniform distribution on $\mathcal{X}$ (which we assume to be finite for simplicity here). Using $Q = Q_\text{s} \cup Q_\text{p}$, it is formally specified by:

$$\begin{aligned} Q_\text{s} &= \Xi \times \mathbb{N} & \Lambda(\mathsf{n}, \xi, n) &= \xi \quad \text{P}_\circ(\xi, 0) = w(\xi) \\ Q_\text{p} &= \{\mathsf{n}, \mathsf{w}\} \times \Xi \times \mathbb{Z}^+ & \Lambda(\mathsf{w}, \xi, n) &= u \end{aligned} \tag{4.21a}$$

$$\text{P} \begin{pmatrix} \langle \xi, n \rangle \to \langle \mathsf{n}, \xi, n+1 \rangle \\ \langle \xi, n \rangle \to \langle \mathsf{w}, \xi, n+1 \rangle \\ \langle \mathsf{n}, \xi, n \rangle \to \langle \xi, n \rangle \\ \langle \mathsf{w}, \xi, n \rangle \to \langle \xi, n \rangle \end{pmatrix} = \begin{pmatrix} 1-\alpha \\ \alpha \\ 1 \\ 1 \end{pmatrix} \tag{4.21b}$$

Each productive state corresponds to the idea that a certain expert is best, and additionally whether the current outcome is normal or wild.

Fix data x^n. Let $\hat{\xi}^n$ be the expert sequence that maximises the likelihood $P_{\xi^n}(x^n)$ among all expert sequences ξ^n that switch between a single expert and U. To derive our loss bound, we underestimate the marginal probability $P_{\text{oce},\alpha}(x^n)$ for the HMM defined above, by dropping all terms except the one for $\hat{\xi}^n$.

$$P_{\text{oce},\alpha}(x^n) \quad = \sum_{\xi^n \in \Xi^n} \pi_{\text{oce},\alpha}(\xi^n) P_{\xi^n}(x^n) \quad \geq \quad \pi_{\text{oce},\alpha}(\hat{\xi}^n) P_{\hat{\xi}^n}(x^n). \tag{4.22}$$

(This first step is also used in the bounds for the two new models in Section 4.4.) Let α^* denote the frequency of occurrence of U in $\hat{\xi}^n$, let ξ_best be the other expert

that occurs in ξ^n, and let $\hat{L} = -\log P_{\hat{\xi}^n}(x^n)$. We can now bound our worst-case additional loss:

$$-\log P_{\mathrm{oce},\hat{\alpha}}(x^n) - \hat{L} \leq -\log \pi_{\mathrm{oce},\alpha}(\hat{\xi}^n) = -\log w(\xi_{\mathrm{best}}) + nH(\alpha^*, \alpha).$$

Again H denotes the cross entropy. From a coding perspective, after first specifying the best expert ξ_{best} and a binary sequence representing $\hat{\xi}^n$, we can then use $\hat{\xi}^n$ to encode the actual observations with optimal efficiency.

The optimal misprediction rate α is usually not known in advance, so we can again learn it from data by placing a prior on it and integrating over this prior. This comes at the cost of an additional loss of $\frac{1}{2}\log n + c$ bits for some constant c (which is ≤ 1 for two experts), and as will be shown in the next subsection, can be implemented using a quadratic time algorithm.

Recursive Combination

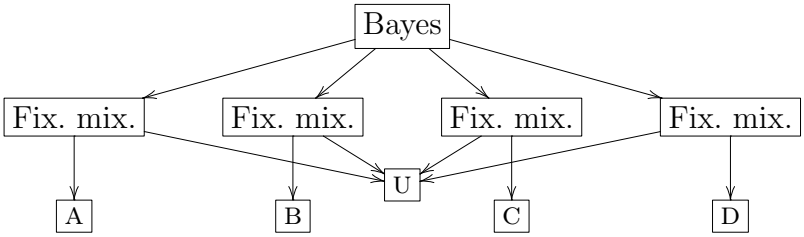
In Figure 4.10 one may recognise two simpler HMMs: it is in fact just a Bayesian combination of a set of fixed elementwise mixtures with some parameter α, one for each expert. Thus two models for combining expert predictions, the Bayesian model and fixed elementwise mixtures, have been recursively combined into a single new model. This view is illustrated in Figure 4.11.

More generally, any method to combine the predictions of multiple experts into a single new prediction strategy, can itself be considered an expert. We can apply our method recursively to this new "meta-expert"; the running time of the recursive combination is only the *sum* of the running times of all the component predictors. For example, if all used individual expert models can be evaluated in quadratic time, then the full recursive combination also has quadratic running time, *even though it may be impossible to specify using an HMM of quadratic size.*

Although a recursive combination to implement overconfident experts may save some work, the same running time may be achieved by implementing the HMM depicted in Figure 4.10 directly. However, we can also obtain efficient generalisations of the overconfident expert model, by replacing any combinator by a more sophisticated one. For example, rather than a fixed elementwise mixture, we could use a universal elementwise mixture for each expert, so that the error frequency is learned from data. Or, if we suspect that an expert may not only make incidental slip-ups, but actually become completely untrustworthy for longer stretches of time, we may even use a fixed or universal share model.

One may also consider that the fundamental idea behind the overconfident expert model is to combine each expert with a uniform predictor using a misprediction model. In the example in Figure 4.11, this idea is used to "smooth" the expert predictions, which are then used at the top level in a Bayesian combination. However, the model that is used at the top level is completely orthogonal to the model used to smooth expert predictions; we can safeguard against overconfident experts not only in Bayesian combinations but also in other models such as

Figure 4.11 Implementing overconfident experts with recursive combinations.



the switch distribution or the run-length model, which are described in the next section.

4.4 New Models to Switch between Experts

So far we have considered two models for switching between experts: fixed share and its generalisation, universal share. While fixed share is an extremely efficient algorithm, it requires that the frequency of switching between experts is estimated a priori, which can be hard in practice. Moreover, we may have prior knowledge about how the switching probability will change over time, but unless we know the ultimate sample size in advance, we may be forced to accept a linear overhead compared to the best parameter value. Universal share overcomes this problem by marginalising over the unknown parameter, but has quadratic running time.

The first model considered in this section, called the switch distribution, avoids both problems. It is parameterless and has essentially the same running time as fixed share. It also achieves a loss bound competitive to that of universal share. Moreover, for a bounded number of switches the bound has even better asymptotics.

The second model is called the run-length model because it uses a run-length code (c.f. [62]) as an ES-prior. This may be useful because, while both fixed and universal share model the distance between switches with a geometric distribution, the real distribution on these distances may be different. This is the case if, for example, the switches are highly clustered. This additional expressive power comes at the cost of quadratic running time, but we discuss a special case where this may be reduced to linear. We compare advantages and drawbacks of the run-length model compared to the switch distribution.

4.4.1 Switch Distribution

The switch distribution is a new model for combining expert predictions. Like fixed share, it is intended for settings where the best predicting expert is expected to change as a function of the sample size, but it has two major innovations. First, we let the probability of switching to a different expert decrease with the

sample size. This allows us to derive a loss bound close to that of the fixed share algorithm, without the need to tune any parameters.[5] Second, the switch distribution has a special provision to ensure that in the case where the number of switches remains bounded, the incurred loss overhead is $O(1)$.

The switch distribution is the subject of the next chapter, which addresses a long standing open problem in statistical model selection known as the "AIC vs BIC dilemma". Some criteria for model selection, such as AIC, are efficient when applied to sequential prediction of future outcomes, while other criteria, such as BIC, are "consistent": with probability one, the model that contains the data generating distribution is selected given enough data. Using the switch distribution, these two goals (truth finding vs prediction) can be reconciled. Refer to the paper for more information.

Here we disregard such applications and treat the switch distribution like the other models for combining expert predictions. We describe an HMM that corresponds to the switch distribution; this illuminates the relationship between the switch distribution and the fixed share algorithm which it in fact generalises.

The equivalence between the original definition of the switch distribution and the HMM is not trivial, so we give a formal proof. The size of the HMM is such that calculation of $P(x^n)$ requires only $O(n\,|\Xi|)$ steps.

We provide a loss bound for the switch distribution in Section 4.4.1. Then in Section 4.4.1 we show how the sequence of experts that has maximum a posteriori probability can be computed. This problem is difficult for general HMMs, but the structure of the HMM for the switch distribution allows for an efficient algorithm in this case.

Switch HMM

Let σ^∞ and τ^∞ be sequences of distributions on $\{0,1\}$ which we call the *switch probabilities* and the *stabilisation probabilities*. The switch HMM $\mathbb{A}_{\text{sw}}$, displayed

[5] The idea of decreasing the switch probability as $1/(n+1)$, which has not previously been published, was independently conceived by Mark Herbster and the authors.

in Figure 4.12, is defined below using $Q = Q_{\mathrm{s}} \cup Q_{\mathrm{p}}$:

$$\begin{aligned} Q_{\mathrm{s}} &= \{\mathsf{p}, \mathsf{p_s}, \mathsf{p_u}\} \times \mathbb{N} \qquad \mathrm{P_o}(\mathsf{p}, 0) = 1 \quad \Lambda(\mathsf{s}, \xi, n) = \xi \\ Q_{\mathrm{p}} &= \{\mathsf{s}, \mathsf{u}\} \times \Xi \times \mathbb{Z}^+ \qquad\qquad\quad \Lambda(\mathsf{u}, \xi, n) = \xi \end{aligned} \tag{4.23a}$$

$$\mathrm{P} \begin{pmatrix} \langle \mathsf{p}, n\rangle \to \langle \mathsf{p_u}, n\rangle \\ \langle \mathsf{p}, n\rangle \to \langle \mathsf{p_s}, n\rangle \\ \langle \mathsf{p_u}, n\rangle \to \langle \mathsf{u}, \xi, n+1\rangle \\ \langle \mathsf{p_s}, n\rangle \to \langle \mathsf{s}, \xi, n+1\rangle \\ \langle \mathsf{s}, \xi, n\rangle \to \langle \mathsf{s}, \xi, n+1\rangle \\ \langle \mathsf{u}, \xi, n\rangle \to \langle \mathsf{u}, \xi, n+1\rangle \\ \langle \mathsf{u}, \xi, n\rangle \to \langle \mathsf{p}, n\rangle \end{pmatrix} = \begin{pmatrix} \tau_n(0) \\ \tau_n(1) \\ w(\xi) \\ w(\xi) \\ 1 \\ \sigma_n(0) \\ \sigma_n(1) \end{pmatrix} \tag{4.23b}$$

This HMM contains two "expert bands". Consider a productive state $\langle \mathsf{u}, \xi, n\rangle$ in the bottom band, which we call the *unstable* band, from a generative viewpoint. Two things can happen. With probability $\sigma_n(0)$ the process continues horizontally to $\langle \mathsf{u}, \xi, n+1\rangle$ and the story repeats. We say that *no switch occurs*. With probability $\sigma_n(1)$ the process continues to the silent state $\langle \mathsf{p}, n\rangle$ directly to the right. We say that *a switch occurs*. Then a new choice has to be made. With probability $\tau_n(0)$ the process continues rightward to $\langle \mathsf{p_u}, n\rangle$ and then branches out to some productive state $\langle \mathsf{u}, \xi', n+1\rangle$ (possibly $\xi = \xi'$), and the story repeats. With probability $\tau_n(1)$ the process continues to $\langle \mathsf{p_s}, n\rangle$ in the top band, called the *stable* band. Also here it branches out to some productive state $\langle \mathsf{s}, \xi', n+1\rangle$. But from this point onward there are no choices anymore; expert ξ' is produced forever. We say that the process has *stabilised*.

By choosing $\tau_n(1) = 0$ and $\sigma_n(1) = \theta$ for all n we essentially remove the stable band and arrive at fixed share with parameter θ. The presence of the stable band enables us to improve the loss bound of fixed share in the particular case that the number of switches is bounded; in that case, the stable band allows us to remove the dependency of the loss bound on n altogether. We will use the particular choice $\tau_n(0) = \theta$ for all n, and $\sigma_n(1) = \pi_{\mathrm{T}}(\boldsymbol{Z} = n | \boldsymbol{Z} \geq n)$ for some fixed value θ and an arbitrary distribution π_{T} on $\mathbb{N}$. This allows us to relate the switch HMM to the parametric representation that we present next.

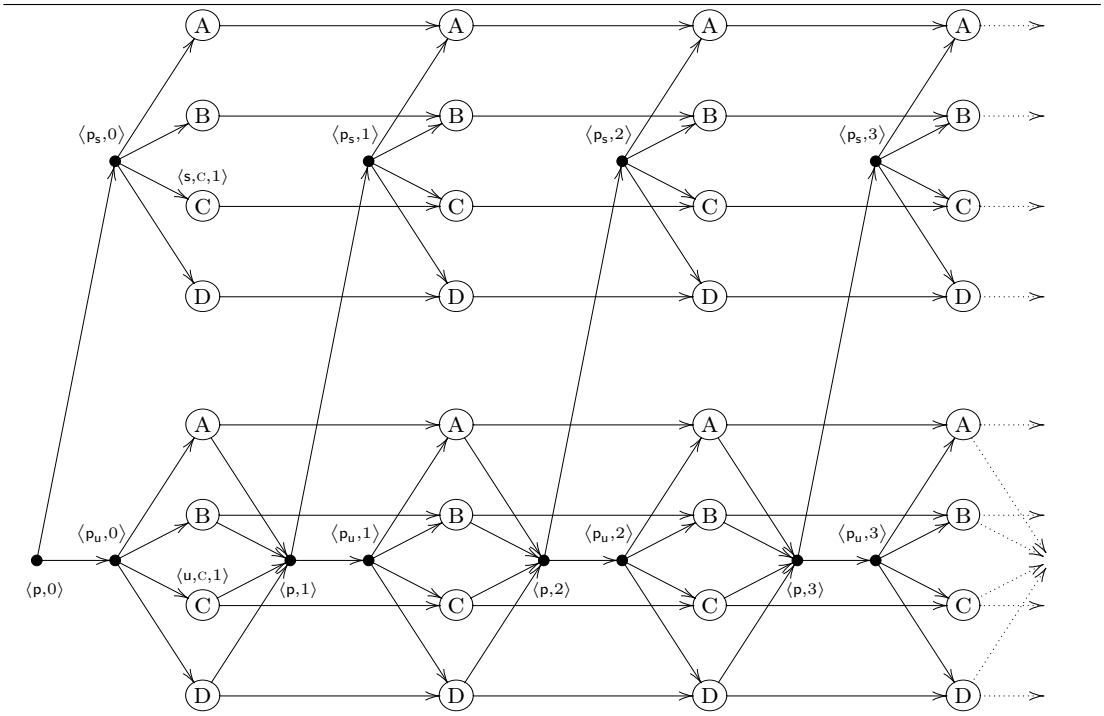
Switch Distribution

In Chapter 5, we give a parametric definition of the switch distribution, and provide an algorithm that computes it efficiently, i.e. in time $O(n\,|\Xi|)$, where n is the sample size and $|\Xi|$ is the number of considered experts. Here we show that this algorithm can really be interpreted as the forward algorithm applied to the switch HMM of Section 4.4.1.

Definition 4.4.1. We first define the countable set of *switch parameters*

$$\Theta_{\mathrm{sw}} := \left\{ \langle t^m, k^m \rangle \mid m \geq 1, k \in \Xi^m, t \in \mathbb{N}^m \text{ and } 0 = t_1 < t_2 < t_3 \ldots \right\}.$$

Figure 4.12 Combination of four experts using the switch distribution



The *switch prior* is the discrete distribution on switch parameters given by

$$\pi_{\text{sw}}(t^m, k^m) := \pi_{\text{M}}(m)\pi_{\text{K}}(k_1)\prod_{i=2}^{m}\pi_{\text{T}}(t_i|t_i > t_{i-1})\pi_{\text{K}}(k_i),$$

where π_{M} is geometric with rate θ, π_{T} and π_{K} are arbitrary distributions on $\mathbb{N}$ and Ξ. We define the mapping $\xi : \Theta_{\text{sw}} \to \Xi^\infty$ that interprets switch parameters as sequences of experts by

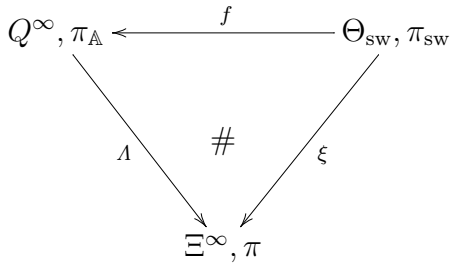
$$\xi(t^m, k^m) := k_1^{[t_2-t_1]} \frown k_2^{[t_3-t_2]} \frown \ldots \frown k_{m-1}^{[t_m-t_{m-1}]} \frown k_m^{[\infty]},$$

where $k^{[\lambda]}$ is the sequence consisting of λ repetitions of k. This mapping is not 1-1: infinitely many switch parameters map to the same infinite sequence, since k_i and k_{i+1} may coincide. The *switch distribution* P_{sw} is the ES-joint based on the ES-prior that is obtained by composing π_{sw} with ξ.

Equivalence

In this section we show that the HMM prior $\pi_{\mathbb{A}}$ and the switch prior π_{sw} define the same ES-prior. During this section, it is convenient to regard $\pi_{\mathbb{A}}$ as a distribution on sequences of states, allowing us to differentiate between distinct sequences of states that map to the same sequence of experts. The function $\Lambda : Q^\infty \to \Xi^\infty$, that we call *trace*, explicitly performs this mapping; $\Lambda(q^\infty)(i) := \Lambda(q_i^{\text{p}})$. We cannot

Figure 4.13 Commutativity diagram



relate π_{sw} to $\pi_{\mathbb{A}}$ directly as they are carried by different sets (switch parameters vs state sequences), but need to consider the distribution that both induce on sequences of experts via ξ and Λ. Formally:

Definition 4.4.2. If $f : \Theta \to \Gamma$ is a random variable and P is a distribution on Θ, then we write $f(P)$ to denote the distribution on Γ that is induced by f.

Below we will show that $\Lambda(\pi_{\mathbb{A}}) = \xi(\pi_{\text{sw}})$, i.e. that π_{sw} and $\pi_{\mathbb{A}}$ induce the same distribution on the expert sequences Ξ^∞ via the trace Λ and the expert-sequence mapping ξ. Our argument will have the structure outlined in Figure 4.13. Instead of proving the claim directly, we create a random variable $f : \Theta_{\text{sw}} \to Q^\infty$ mapping switch parameters into runs. Via f, we can view Θ_{sw} as a reparameterisation of Q^∞. We then show that the diagram commutes, that is, $\pi_{\mathbb{A}} = f(\pi_{\text{sw}})$ and $\Lambda \circ f = \xi$. This shows that $\Lambda(\pi_{\mathbb{A}}) = \Lambda(f(\pi_{\text{sw}})) = \xi(\pi_{\text{sw}})$ as required.

Proposition 4.4.3. *Let $\mathbb{A}$ be the HMM as defined in Section 4.4.1, and π_{sw}, ξ and Λ as above. If $w = \pi_{\text{K}}$ then*

$$\xi(\pi_{\text{sw}}) = \Lambda(\pi_{\mathbb{A}}).$$

Proof. Recall (4.23) that

$$Q = \{\mathsf{s}, \mathsf{u}\} \times \Xi \times \mathbb{Z}^+ \quad \cup \quad \{\mathsf{p}, \mathsf{p_s}, \mathsf{p_u}\} \times \mathbb{N}.$$

We define the random variable $f : \Theta_{\text{sw}} \to Q^\infty$ by

$$\begin{aligned} f(t^m, k^m) &:= \langle \mathsf{p}, 0 \rangle \frown u_1 \frown u_2 \frown \ldots \frown u_{m-1} \frown s, \qquad \text{where} \\ u_i &:= \left\langle \langle \mathsf{p_u}, t_i \rangle, \langle \mathsf{u}, k_i, t_i + 1 \rangle, \langle \mathsf{u}, k_i, t_i + 2 \rangle, \ldots, \langle \mathsf{u}, k_i, t_{i+1} \rangle, \langle \mathsf{p}, t_{i+1} \rangle \right\rangle \\ s &:= \left\langle \langle \mathsf{p_s}, t_m \rangle, \langle \mathsf{s}, k_m, t_m + 1 \rangle, \langle \mathsf{s}, k_m, t_m + 2 \rangle, \ldots \right\rangle. \end{aligned}$$

We now show that $\Lambda \circ f = \xi$ and $f(\pi_{\text{sw}}) = \pi_{\mathbb{A}}$, from which the theorem follows directly. Fix $p = \langle t^m, k^m \rangle \in \Theta_{\text{sw}}$. Since the trace of a concatenation equals the

concatenation of the traces,

$$\begin{aligned}\Lambda \circ f(p) &= \Lambda(u_1) \frown \Lambda(u_2) \frown \ldots \frown \Lambda(u_{m-1}) \frown \Lambda(s) \\ &= k_1^{[t_2-t_1]} \frown k_2^{[t_3-t_2]} \frown \ldots \frown k_2^{[t_m-t_{m-1}]} \frown k_m^{[\infty]} = \xi(p).\end{aligned}$$

which establishes the first part. Second, we need to show that $\pi_{\mathbb{A}}$ and $f(\pi_{\text{sw}})$ assign the same probability to all events. Since π_{sw} has countable support, so has $f(\pi_{\text{sw}})$. By construction f is injective, so the preimage of $f(p)$ equals $\{p\}$, and hence $f(\pi_{\text{sw}})(\{f(p)\}) = \pi_{\text{sw}}(p)$. Therefore it suffices to show that $\pi_{\mathbb{A}}(\{f(p)\}) = \pi_{\text{sw}}(p)$ for all $p \in \Theta_{\text{sw}}$. Let $q^\infty = f(p)$, and define u_i and s for this p as above. Then

$$\pi_{\mathbb{A}}(q^\infty) = \pi_{\mathbb{A}}(\langle \mathsf{p}, 0 \rangle) \left(\prod_{i=1}^{m-1} \pi_{\mathbb{A}}(u_i | u^{i-1}) \right) \pi_{\mathbb{A}}(s | u^{m-1})$$

Note that

$$\begin{aligned}\pi_{\mathbb{A}}(s|u^{m-1}) &= (1-\theta)\pi_{\text{K}}(k_i) \\ \pi_{\mathbb{A}}(u_i|u^{i-1}) &= \theta \pi_{\text{K}}(k_i) \left(\prod_{j=t_i+1}^{t_{i+1}-1} \pi_{\text{T}}(\boldsymbol{Z} > j | \boldsymbol{Z} \geq j) \right) \pi_{\text{T}}(\boldsymbol{Z} = t_{i+1} | \boldsymbol{Z} \geq t_{i+1}).\end{aligned}$$

The product above telescopes, so that

$$\pi_{\mathbb{A}}(u_i|u^{i-1}) = \theta \pi_{\text{K}}(k_i) \pi_{\text{T}}(\boldsymbol{Z} = t_{i+1} | \boldsymbol{Z} \geq t_{i+1}).$$

We obtain

$$\begin{aligned}\pi_{\mathbb{A}}(q^\infty) &= 1 \cdot \theta^{m-1} \left(\prod_{i=1}^{m-1} \pi_{\text{K}}(k_i) \pi_{\text{T}}(t_{i+1} | t_{i+1} > t_i) \right) (1-\theta) \pi_{\text{K}}(k_m) \\ &= \theta^{m-1}(1-\theta)\pi_{\text{K}}(k_1) \prod_{i=2}^{m} \pi_{\text{K}}(k_i) \pi_{\text{T}}(t_i | t_i > t_{i-1}) \\ &= \pi_{\text{sw}}(p),\end{aligned}$$

under the assumption that π_{M} is geometric with parameter θ. □

A Loss Bound

We derive a loss bound of the same type as the bound for the fixed share algorithm (see Section 4.3.2).

Theorem 4.4.4. *Fix data x^n. Let $\hat{\theta} = \langle t^m, k^m \rangle$ maximise the likelihood $P_{\xi(\hat{\theta})}(x^n)$ among all switch parameters of length m. Let $\pi_{\text{M}}(n) = 2^{-n}$, $\pi_{\text{T}}(n) = 1/(n(n+1))$*

and π_{K} *be uniform. Then the loss overhead* $-\log P_{\text{sw}}(x^n) + \log P_{\xi(\hat{\theta})}(x^n)$ *of the switch distribution is bounded by*

$$m + m \log |\Xi| + \log \binom{t_m + 1}{m} + \log(m!).$$

Proof. We have

$$\begin{aligned} & -\log P_{\text{sw}}(x^n) + \log P_{\xi(\hat{\theta})}(x^n) \\ \leq\ & -\log \pi_{\text{sw}}(\hat{\theta}) \\ =\ & -\log \left(\pi_{\text{M}}(m) \pi_{\text{K}}(k_1) \prod_{i=2}^{m} \pi_{\text{T}}(t_i | t_i > t_{i-1}) \pi_{\text{K}}(k_i) \right) \\ =\ & -\log \pi_{\text{M}}(m) + \sum_{i=1}^{m} -\log \pi_{\text{K}}(k_i) + \sum_{i=2}^{m} -\log \pi_{\text{T}}(t_i | t_i > t_i - 1). \end{aligned} \tag{4.24}$$

The considered prior $\pi_{\text{T}}(n) = 1/(n(n+1))$ satisfies

$$\pi_{\text{T}}(t_i | t_i > t_{i-1}) = \frac{\pi_{\text{T}}(t_i)}{\sum_{i=t_{i-1}+1}^{\infty} \pi_{\text{T}}(i)} = \frac{1/(t_i(t_i+1))}{\sum_{i=t_{i-1}+1}^{\infty} \frac{1}{i} - \frac{1}{i+1}} = \frac{t_{i-1}+1}{t_i(t_i+1)}.$$

If we substitute this in the last term of (4.24), the sum telescopes and we are left with

$$\underbrace{-\log(t_1+1)}_{=\,0} + \log(t_m+1) + \sum_{i=2}^{m} \log t_i. \tag{4.25}$$

If we fix t_m, this expression is maximised if $t_2, \ldots, t_{m-1}$ take on the values $t_m - m + 2, \ldots, t_m - 1$, so that (4.25) becomes

$$\sum_{i=t_m-m+2}^{t_m+1} \log i = \log \left(\frac{(t_m+1)!}{(t_m-m+1)!} \right) = \log \binom{t_m+1}{m} + \log(m!).$$

The theorem follows if we also instantiate π_{M} and π_{K} in (4.24). □

Note that this loss bound is a function of the index of the last switch t_m rather than of the sample size n; this means that in the important scenario where the number of switches remains bounded in n, the loss compared to the best partition is $O(1)$.

The bound can be tightened slightly by using the fact that we allow for switching to the same expert, as also remarked in Footnote 3 on page 96. If we take this into account, the $m \log |\Xi|$ term can be reduced to $m \log(|\Xi| - 1)$. If we take this into account, the bound compares quite favourably with the loss bound for the fixed share algorithm (see Section 4.3.2). We now investigate how much

worse the above guarantees are compared to those of fixed share. The overhead of fixed share (4.18) is bounded from above by $nH(\alpha) + m\log(|\Xi| - 1)$. We first underestimate this worst-case loss by substituting the optimal value $\alpha = m/n$, and rewrite

$$nH(\alpha) \;\geq\; nH(m/n) \;\geq\; \log \binom{n}{m}.$$

Second we overestimate the loss of the switch distribution by substituting the worst case $t_m = n - 1$. We then find the maximal difference between the two bounds to be

$$\left(m + m\log(|\Xi| - 1) + \log\binom{n}{m} + \log(m!)\right) - \left(\log\binom{n}{m} + m\log(|\Xi| - 1)\right)$$
$$= m + \log(m!) \;\leq\; m + m\log m. \quad (4.26)$$

Thus using the switch distribution instead of fixed share lowers the guarantee by at most $m + m\log m$ bits, which is significant only if the number of switches is relatively large. On the flip side, using the switch distribution does not require any prior knowledge about any parameters. This is a big advantage in a setting where we desire to maintain the bound sequentially. This is impossible with the fixed share algorithm in case the optimal value of α varies with n.

MAP Estimation

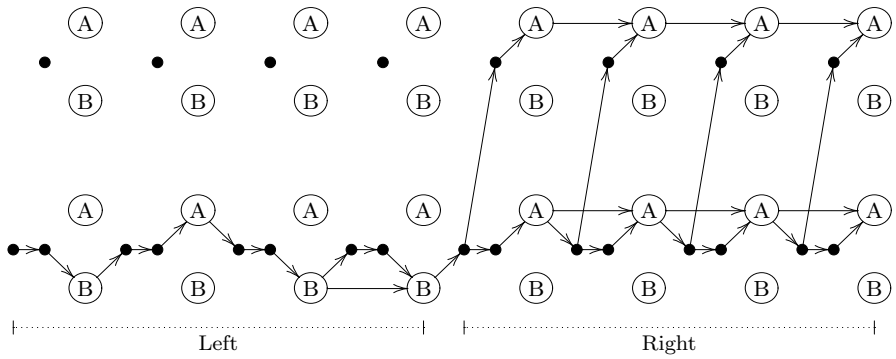
The particular nature of the switch distribution allows us to perform MAP estimation efficiently. The MAP sequence of experts is:

$$\arg\max_{\xi^n} P(x^n, \xi^n).$$

We observed in Section 4.2.5 that Viterbi can be used on unambiguous HMMs. However, the switch HMM is ambiguous, since a single sequence of experts is produced by multiple sequences of states. Still, it turns out that for the switch HMM we can jointly consider all these sequences of states efficiently. Consider for example the expert sequence ABAABBBB. The sequences of states that produce this expert sequence are exactly the runs through the pruned HMM shown in Figure 4.14. Runs through this HMM can be decomposed in two parts, as indicated in the bottom of the figure. In the right part a single expert is repeated, in our case expert D. The left part is contained in the unstable (lower) band. To compute the MAP sequence we proceed as follows. We iterate over the possible places of the transition from left to right, and then optimise the left and right segments independently.

In the remainder we first compute the probability of the MAP expert sequence instead of the sequence itself. We then show how to compute the MAP sequence from the fallout of the probability computation.

Figure 4.14 MAP estimation for the switch distribution. The sequences of states that can be obtained by following the arrows are exactly those that produce expert sequence ABAABBBB.



To optimise both parts, we define two functions L and R.

$$L_i := \max_{\xi^i} P(x^i, \xi^i, \langle \mathsf{p}, i \rangle) \tag{4.27}$$

$$R_i(\xi) := P(x^n, \xi_i = \ldots = \xi_n = \xi | x^{i-1}, \langle \mathsf{p}, i-1 \rangle) \tag{4.28}$$

Thus L_i is the probability of the MAP expert sequence of length i. The requirement $\langle \mathsf{p}, i \rangle$ forces all sequences of states that realise it to remain in the unstable band. $R_i(\xi)$ is the probability of the tail $x_i, \ldots, x_n$ when expert ξ is used for all outcomes, starting in state $\langle \mathsf{p}, i-1 \rangle$. Combining L and R, we have

$$\max_{\xi^n} P(x^n, \xi^n) = \max_{i \in [n], \xi} L_{i-1} R_i(\xi).$$

Recurrence L_i and R_i can efficiently be computed using the following recurrence relations. First we define auxiliary quantities

$$L'_i(\xi) := \max_{\xi^i} P(x^i, \xi^i, \langle \mathsf{u}, \xi, i \rangle) \tag{4.29}$$

$$R'_i(\xi) := P(x^n, \xi_i = \ldots = \xi_n = \xi | x^{i-1}, \langle \mathsf{u}, \xi, i \rangle) \tag{4.30}$$

Observe that the requirement $\langle \mathsf{u}, \xi, i \rangle$ forces $\xi_i = \xi$. First, $L'_i(\xi)$ is the MAP probability for length i under the constraint that the last expert used is ξ. Second, $R'_i(\xi)$ is the MAP probability of the tail $x_i, \ldots, x_n$ under the constraint that the same expert is used all the time. Using these quantities, we have (using the $\gamma_{(\cdot)}$ transition probabilities shown in (4.34))

$$L_i = \max_{\xi} L'_i(\xi) \gamma_1 \qquad R_i(\xi) = \gamma_2 R'_i(\xi) + \gamma_3 P_\xi(x^n | x^{i-1}). \tag{4.31}$$

For $L'_i(\xi)$ and $R'_i(\xi)$ we have the following recurrences:

$$L_{i+1}(\xi) = P_\xi(x_{i+1}|x^i) \max\left\{L'_i(\xi)(\gamma_4 + \gamma_1\gamma_5), L_i\gamma_5\right\} \tag{4.32}$$
$$R'_i(\xi) = P_\xi(x_i|x^{i-1})\left(\gamma_1 R_{i+1}(\xi) + \gamma_4 R'_{i+1}(\xi)\right). \tag{4.33}$$

The recurrence for L has border case $L_0 = 1$. The recurrence for R has border case $R_n = 1$.

$$\begin{aligned}
\gamma_1 &= \mathrm{P}\left(\langle \mathsf{u}, \xi, i\rangle \to \langle \mathsf{p}, i\rangle\right) \\
\gamma_2 &= \mathrm{P}\left(\langle \mathsf{p}, i-1\rangle \to \langle \mathsf{p}_\mathsf{u}, i-1\rangle \to \langle \mathsf{u}, \xi, i\rangle\right) \\
\gamma_3 &= \mathrm{P}\left(\langle \mathsf{p}, i-1\rangle \to \langle \mathsf{p}_\mathsf{s}, i-1\rangle \to \langle \mathsf{s}, \xi, i\rangle\right) \\
\gamma_4 &= \mathrm{P}\left(\langle \mathsf{u}, \xi, i\rangle \to \langle \mathsf{u}, \xi, i+1\rangle\right) \\
\gamma_5 &= \mathrm{P}\left(\langle \mathsf{p}, i\rangle \to \langle \mathsf{p}_\mathsf{u}, i\rangle \to \langle \mathsf{u}, \xi, i+1\rangle\right)
\end{aligned} \tag{4.34}$$

Complexity A single recurrence step of L_i costs $O(|\Xi|)$ due to the maximisation. All other recurrence steps take $O(1)$. Hence both L_i and $L'_i(\xi)$ can be computed recursively for all $i = 1, \ldots, n$ and $\xi \in \Xi$ in time $O(n\,|\Xi|)$, while each of $R_i, R'_i(\xi)$ and $P_\xi(x^n|x^{i-1})$ can be computed recursively for all $i = n, \ldots, 1$ and $\xi \in \Xi$ in time $O(n\,|\Xi|)$ as well. Thus the MAP probability can be computed in time $O(n\,|\Xi|)$. Storing all intermediate values costs $O(n\,|\Xi|)$ space as well.

The MAP Expert Sequence As usual in Dynamic Programming, we can retrieve the final solution — the MAP expert sequence — from these intermediate values. We redo the computation, and each time that a maximum is computed we record the expert that achieves it. The experts thus computed form the MAP sequence.

4.4.2 Run-length Model

Run-length codes have been used extensively in the context of data compression, see e.g. [62]. Rather than applying run length codes directly to the observations, we reinterpret the corresponding probability distributions as ES-priors, because they may constitute good models for the distances between consecutive switches.

The run length model is especially useful if the switches are clustered, in the sense that some blocks in the expert sequence contain relatively few switches, while other blocks contain many. The fixed share algorithm remains oblivious to such properties, as its predictions of the expert sequence are based on a Bernoulli model: the probability of switching remains the same, regardless of the index of the previous switch. Essentially the same limitation also applies to the universal share algorithm, whose switching probability normally converges as the sample size increases. The switch distribution is efficient when the switches are clustered toward the beginning of the sample: its switching probability decreases

in the sample size. However, this may be unrealistic and may introduce a new unnecessary loss overhead.

The run-length model is based on the assumption that the *intervals* between successive switches are independently distributed according to some distribution π_{T}. After the universal share model and the switch distribution, this is a third generalisation of the fixed share algorithm, which is recovered by taking a geometric distribution for π_{T}. As may be deduced from the defining HMM, which is given below, we require quadratic running time $O(n^2 |\Xi|)$ to evaluate the run-length model in general.

Run-length HMM

Let $\mathbb{S} := \big\{\langle m, n\rangle \in \mathbb{N}^2 \mid m < n\big\}$, and let π_{T} be a distribution on $\mathbb{Z}^+$. The specification of the run-length HMM is given using $Q = Q_{\text{s}} \cup Q_{\text{p}}$ by:

$$\begin{aligned} Q_{\text{s}} &= \{\mathsf{q}\} \times \mathbb{S} \;\cup\; \{\mathsf{p}\} \times \mathbb{N} & \Lambda(\xi, m, n) &= \xi \\ Q_{\text{p}} &= \Xi \times \mathbb{S} & \mathrm{P}_\circ(\mathsf{p}, 0) &= 1 \end{aligned} \tag{4.35a}$$

$$\mathrm{P}\begin{pmatrix} \langle \mathsf{p}, n\rangle \to \langle \xi, n, n+1\rangle \\ \langle \xi, m, n\rangle \to \langle \xi, m, n+1\rangle \\ \langle \xi, m, n\rangle \to \langle \mathsf{q}, m, n\rangle \\ \langle \mathsf{q}, m, n\rangle \to \langle \mathsf{p}, n\rangle \end{pmatrix} = \begin{pmatrix} w(\xi) \\ \pi_{\text{T}}(\boldsymbol{Z} > n | \boldsymbol{Z} \geq n) \\ \pi_{\text{T}}(\boldsymbol{Z} = n | \boldsymbol{Z} \geq n) \\ 1 \end{pmatrix} \tag{4.35b}$$

A Loss Bound

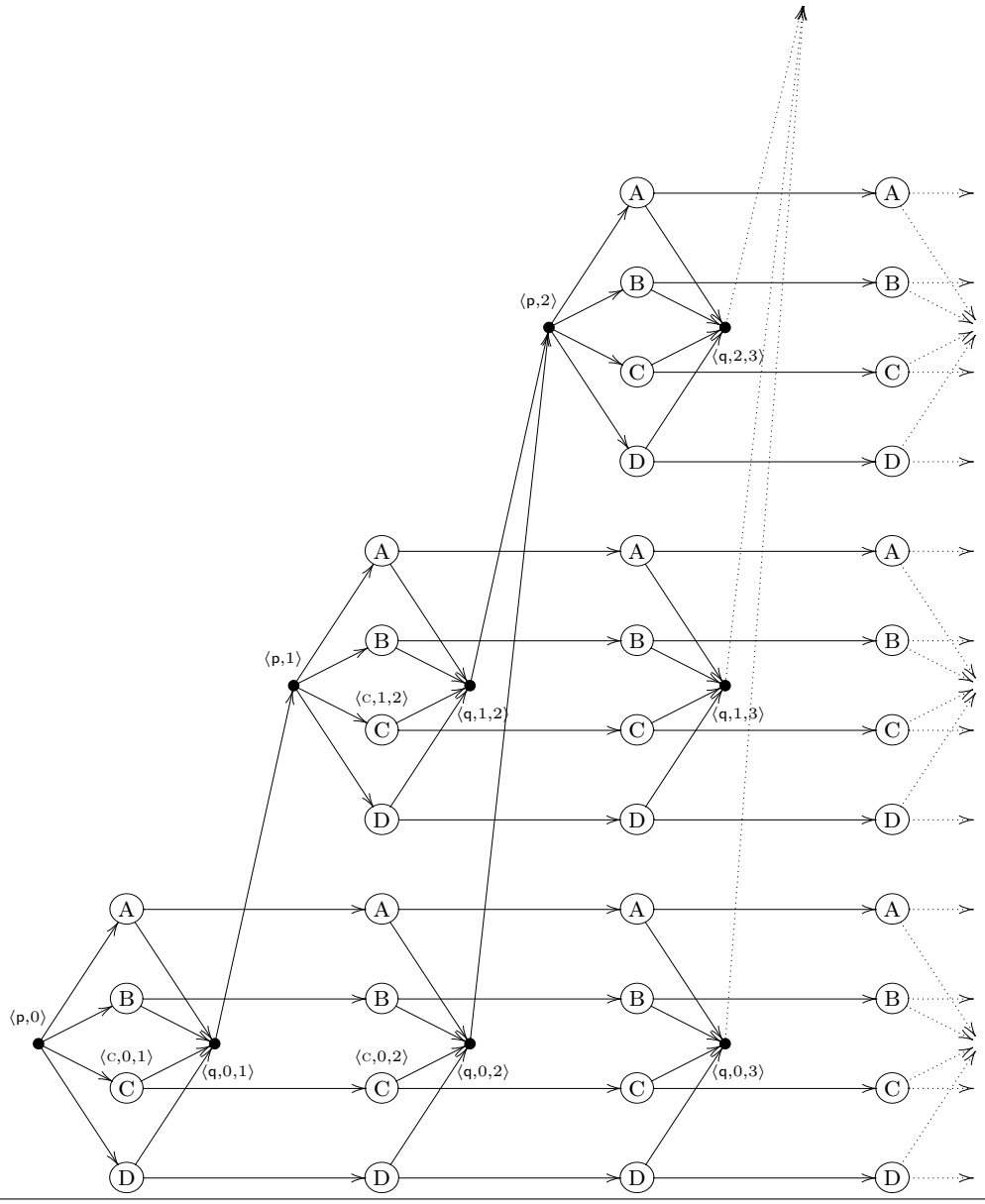
Fix an expert sequence ξ^n with m blocks. For $i = 1, \ldots, m$, let δ_i and k_i denote the length and expert of block i. From the definition of the HMM above, we obtain that $\pi_{\text{rl}}(\xi^n)$ equals

$$\sum_{i=1}^{m} -\log w(k_i) + \sum_{i=1}^{m-1} -\log \pi_{\text{T}}(\boldsymbol{Z} = \delta_i) - \log \pi_{\text{T}}(\boldsymbol{Z} \geq \delta_m).$$

Theorem 4.4.5. *Fix data x^n. Let ξ^n maximise the likelihood $P_{\xi^n}(x^n)$ among all expert sequences with m blocks. Let w be the uniform distribution on experts, and let π_{T} be log-convex. Then the loss overhead is bounded thus*

$$-\log P_{\text{rl}}(x^n) + \log P_{\xi^n}(x^n) \;\leq\; m\left(\log |\Xi| - \log \pi_{\text{T}}\left(\tfrac{n}{m}\right)\right).$$

Figure 4.15 HMM for the run-length model



Proof. Let δ_i denote the length of block i. We overestimate

$$\begin{aligned} &-\log P_{\text{rl}}(x^n) + \log P_{\xi^n}(x^n) \;\leq\; -\log \pi_{\text{rl}}(\xi^n) \\ &= m \log |\Xi| + \sum_{i=1}^{m-1} -\log \pi_{\text{T}}(\boldsymbol{Z} = \delta_i) - \log \pi_{\text{T}}(\boldsymbol{Z} \geq \delta_m) \\ &\leq m \log |\Xi| + \sum_{i=1}^{m} -\log \pi_{\text{T}}(\delta_i). \end{aligned} \tag{4.36}$$

Since $-\log \pi_{\text{T}}$ is concave, by Jensen's inequality we have

$$\sum_{i=1}^{m} \frac{-\log \pi_{\text{T}}(\delta_i)}{m} \leq -\log \pi_{\text{T}} \left(\sum_{i=1}^{m} \frac{\delta_i}{m} \right) = -\log \pi_{\text{T}} \left(\frac{n}{m} \right).$$

In other words, the block lengths δ_i are all equal in the worst case. Plugging this into (4.36) we obtain the theorem. □

Finite Support

We have seen that the run-length model reduces to fixed share if the prior on switch distances π_{T} is geometric, so that it can be evaluated in linear time in that case. We also obtain a linear time algorithm when π_{T} has finite support, because then only a constant number of states can receive positive weight at any sample size. For this reason it can be advantageous to choose a π_{T} with finite support, even if one expects that arbitrarily long distances between consecutive switches may occur. Expert sequences with such longer distances between switches can still be represented with a truncated π_{T} using a sequence of switches from and to the same expert. This way, long runs of the same expert receive exponentially small, but positive, probability.

4.4.3 Comparison

We have discussed two models for switching: the recent switch distribution and the new run-length model. It is natural to wonder which model to apply. One possibility is to compare asymptotic loss bounds. To compare the bounds given by Theorems 4.4.4 and 4.4.5, we substitute $t_m + 1 = n$ in the bound for the switch distribution, and use a prior π_{T} for the run-length model that satisfies $-\log \pi_{\text{T}}(n) \leq \log n + 2 \log \log(n+1) + 3$ (for instance an Elias code [32]). The next step is to determine which bound is better depending on how fast m grows as a function of n. It only makes sense to consider m non-decreasing in n.

Theorem 4.4.6. *The loss bound of the switch distribution (with $t_n = n$) is asymptotically lower than that of the run-length model (with π_{T} as above) if*

$m = o\big((\log n)^2\big)$, *and asymptotically higher if* $m = \Omega\big((\log n)^2\big)$.[6]

Proof sketch. After eliminating terms common to both loss bounds, it remains to compare

$$m + m \log m \quad \text{to} \quad 2m \log \log \left(\frac{n}{m} + 1 \right) + 3.$$

If m is bounded, the left hand side is clearly lower for sufficiently large n. Otherwise we may divide by m, exponentiate, simplify, and compare

$$m \quad \text{to} \quad (\log n - \log m)^2,$$

from which the theorem follows directly. □

For finite samples, the switch distribution can be used in case the switches are expected to occur early on average, or if the running time is paramount. Otherwise the run-length model is preferable.

4.5 Extensions

The approach described in Sections 4.1 and 4.2 allows efficient evaluation of expert models that can be defined using small HMMs. It is natural to look for additional efficient models for combining experts that cannot be expressed as small HMMs in this way.

In this section we describe a number of such extensions to the model as described above. In Section 4.5.1 we outline different methods for approximate, but faster, evaluation of large HMMs. The idea behind Section 4.3.4 is to treat a *combination* of experts as a single expert, and subject it to "meta" expert combination. Then in Section 4.5.2 we outline a possible generalisation of the considered class of HMMs, allowing the ES-prior to depend on observed data. Finally we propose an alternative to MAP expert sequence estimation that is efficiently computable for general HMMs.

4.5.1 Fast Approximations

For some applications, suitable ES-priors do not admit a description in the form of a small HMM. Under such circumstances we might require an exponential amount of time to compute quantities such as the predictive distribution on the next expert (4.3). For example, although the size of the HMM required to describe the elementwise mixtures of Section 4.3.1 grows only polynomially in n, this is still not feasible in practice. Consider that the transition probabilities at sample size n must depend on the number of times that each expert has occurred previously.

[6] Let $f, g : \mathbb{N} \to \mathbb{N}$. We say $f = o(g)$ if $\lim_{n\to\infty} f(n)/g(n) = 0$. We say $f = \Omega(g)$ if $\exists c > 0 \exists n_0 \forall n \geq n_0 : f(n) \geq cg(n)$.

The number of states required to represent this information must therefore be at least $\binom{n+k-1}{k-1}$, where k is the number of experts. For five experts and $n = 100$, we already require more than four million states! In the special case of mixtures, various methods exist to efficiently find good parameter values, such as expectation maximisation, see e.g. [59] and Li and Barron's approach [55]. Here we describe a few general methods to speed up expert sequence calculations.

Discretisation

The simplest way to reduce the running time of Algorithm 4.1 is to reduce the number of states of the input HMM, either by simply omitting states or by identifying states with similar futures. This is especially useful for HMMs where the number of states grows in n, e.g. the HMMs where the parameter of a Bernoulli source is learned: the HMM for universal elementwise mixtures of Figure 4.7 and the HMM for universal share of Figure 4.9. At each sample size n, these HMMs contain states for count vectors $(0, n), (1, n-1), \ldots, (n, 0)$. In [63] Monteleoni and Jaakkola manage to reduce the number of states to $\sqrt{n}$ when the sample size n is known in advance. We conjecture that it is possible to achieve the same loss bound by joining ranges of well-chosen states into roughly $\sqrt{n}$ super-states, and adapting the transition probabilities accordingly.

Trimming

Another straightforward way to reduce the running time of Algorithm 4.1 is by run-time modification of the HMM. We call this *trimming*. The idea is to drop low probability transitions from one sample size to the next. For example, consider the HMM for elementwise mixtures of two experts, Figure 4.7. The number of transitions grows linearly in n, but depending on the details of the application, the probability mass may concentrate on a subset that represents mixture coefficients close to the optimal value. A speedup can then be achieved by always retaining only the smallest set of transitions that are reached with probability p, for some value of p which is reasonably close to one. The lost probability mass can be recovered by renormalisation.

The ML Conditioning Trick

A more drastic approach to reducing the running time can be applied whenever the ES-prior assigns positive probability to all expert sequences. Consider the desired marginal probability (4.2) which is equal to:

$$P(x^n) = \sum_{\xi^n \in \Xi^n} \pi(\xi^n) P(x^n \mid \xi^n). \tag{4.37}$$

In this expression, the sequence of experts ξ^n can be interpreted as a parameter. While we would ideally compute the Bayes marginal distribution, which means

integrating out the parameter under the ES-prior, it may be easier to compute a point estimator for ξ^n instead. Such an estimator $\xi(x^n)$ can then be used to find a lower bound on the marginal probability:

$$\pi(\xi(x^n))P(x^n \mid \xi(x^n)) \quad \leq \quad P(x^n). \tag{4.38}$$

The first estimator that suggests itself is the Bayesian maximum a-posteriori:

$$\xi_{\text{map}}(x^n) := \arg\max_{\xi^n \in \Xi^n} \pi(\xi^n) P(x^n \mid \xi^n).$$

In Section 4.2.5 we explain that this estimator is generally hard to compute for ambiguous HMMs, and for unambiguous HMMs it is as hard as evaluating the marginal (4.37). One estimator that is much easier to compute is the maximum likelihood (ML) estimator, which disregards the ES-prior π altogether:

$$\xi_{\text{ml}}(x^n) := \arg\max_{\xi^n \in \Xi^n} P(x^n \mid \xi^n).$$

The ML estimator may correspond to a much smaller term in (4.37) than the MAP estimator, but it has the advantage that it is extremely easy to compute. In fact, letting $\hat{\xi}^n := \xi_{\text{ml}}(x^n)$, each expert $\hat{\xi}_i$ is a function of only the corresponding outcome x_i. Thus, calculation of the ML estimator is cheap. Furthermore, if the goal is not to find a lower bound, but to predict the outcomes x^n with as much confidence as possible, we can make an even better use of the estimator if we use it sequentially. Provided that $P(x^n) > 0$, we can approximate:

$$\begin{aligned} P(x^n) = \prod_{i=1}^{n} P(x_i|x^{i-1}) &= \prod_{i=1}^{n} \sum_{\xi_i \in \Xi} P(\xi_i|x^{i-1}) P_{\xi_i}(x_i|x^{i-1}) \\ &\approx \prod_{i=1}^{n} \sum_{\xi_i \in \Xi} \pi(\xi_i|\hat{\xi}^{i-1}) P_{\xi_i}(x_i|x^{i-1}) =: \tilde{P}(x^n). \end{aligned} \tag{4.39}$$

This approximation improves the running time if the conditional distribution $\pi(\xi_n|\xi^{n-1})$ can be computed more efficiently than $P(\xi_n|x^{n-1})$, as is often the case.

Example 13. As can be seen in Figure 4.1, the running time of the universal elementwise mixture model (cf. Section 4.3.1) is $O(n^{|\Xi|})$, which is prohibitive in practice, even for small Ξ. We apply the above approximation. For simplicity we impose the uniform prior density $w(\alpha) = 1$ on the mixture coefficients. We use the generalisation of Laplace's Rule of Succession to multiple experts, which states:

$$\pi_{\text{ue}}(\xi_{n+1}|\xi^n) \;=\; \int_{\triangle(\Xi)} \alpha(\xi_{n+1}) w(\alpha|\xi^n)\, \mathrm{d}\alpha \;=\; \frac{\left|\left\{j \leq n \mid \xi_j = \xi_{n+1}\right\}\right| + 1}{n + |\Xi|}. \tag{4.40}$$

Substitution in (4.39) yields the following predictive distribution:

$$\begin{aligned}\tilde{P}(x_{n+1}|x^n) &= \sum_{\xi_{n+1}\in\Xi} \pi(\xi_{n+1} \mid \hat{\xi}^n) P_{\xi_{n+1}}(x_{n+1} \mid x^n) \\ &= \sum_{\xi_{n+1}} \frac{|\{j \le n \mid \hat{\xi}_j(x^n) = \xi_{n+1}\}| + 1}{n + |\Xi|} P_{\xi_{n+1}}(x_{n+1}|x^n).\end{aligned} \tag{4.41}$$

By keeping track of the number of occurrences of each expert in the ML sequence, this expression can easily be evaluated in time proportional to the number of experts, so that $\tilde{P}(x^n)$ can be computed in the ideal time $O(n\,|\Xi|)$ (which is a lower bound because one has to consider all experts at all sample sizes). ♢

The difference between $P(x^n)$ and $\tilde{P}(x^n)$ is difficult to analyse in general, but the approximation does have two encouraging properties. First, the lower bound (4.38) on the marginal probability, instantiated for the ML estimator, also provides a lower bound on $\tilde{P}$. We have

$$\tilde{P}(x^n) \quad \geq \quad \prod_{i=1}^{n} \pi(\hat{\xi}_i \mid \hat{\xi}^{i-1}) P_{\hat{\xi}_i}(x_i \mid x^{i-1}) \;=\; \pi(\hat{\xi}^n) P(x^n \mid \hat{\xi}^n).$$

To see why the approximation gives higher probability than the bound, consider that the bound corresponds to a defective distribution, unlike $\tilde{P}$.

Second, the following information processing argument shows that even in circumstances where the approximation of the posterior $\tilde{P}(\xi_i \mid x^{i-1})$ is poor, the approximation of the predictive distribution $\tilde{P}(x_i \mid x^{i-1})$ might be acceptable.

Lemma 4.5.1. *Let π, ρ be ES-priors. Then for all $n \in \mathbb{N}$,*

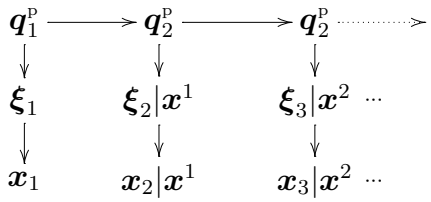
$$D\left(P_\rho(\boldsymbol{x}^n) \| P_\pi(\boldsymbol{x}^n)\right) \;\leq\; D\left(\rho(\boldsymbol{\xi}^n) \| \pi(\boldsymbol{\xi}^n)\right).$$

Proof. The claim follows from taking an expectation of Theorem 4.3.1 under P_ρ:

$$E_{P_\rho}\left[-\log \frac{P_\pi(\boldsymbol{x}^n)}{P_\rho(\boldsymbol{x}^n)}\right] \;\leq\; E_{P_\rho} E_{P_\rho}\left[-\log \frac{\pi(\boldsymbol{\xi}^n)}{\rho(\boldsymbol{\xi}^n)}\middle| \boldsymbol{x}^n\right] \;=\; E_{P_\rho}\left[-\log \frac{\pi(\boldsymbol{\xi}^n)}{\rho(\boldsymbol{\xi}^n)}\right]. \quad \square$$

We apply this lemma to the predictive distribution on the single next outcome after observing a sequence x^n. Setting π to $P_\pi(\boldsymbol{\xi}_{n+1}|\xi(x^n))$ and ρ to $P_\pi(\boldsymbol{\xi}_{n+1}|x^n)$, we find that the divergence between the predictive distribution on the next *outcome* and its approximation, is at most equal to the divergence between the posterior distribution on the next *expert* and its approximation. In other words, approximation errors in the posterior tend to cancel each other out during prediction.

Figure 4.16 Conditioning ES-prior on past observations for free



4.5.2 Data-Dependent Priors

To motivate ES-priors we used the slogan *we do not understand the data*. When we discussed using HMMs as ES-priors we imposed the restriction that for each state the associated Ξ-PFS was independent of the previously produced experts. Indeed, conditioning on the *expert history* increases the running time dramatically as all possible histories must be considered. However, conditioning on the *past observations* can be done *at no additional cost*, as the data are *observed*. The resulting HMM is shown in Figure 4.16. We consider this technical possibility a curiosity, as it clearly violates our slogan. Of course it is equally feasible to condition on some function of the data. An interesting case is obtained by conditioning on the vector of losses (cumulative or incremental) incurred by the experts. This way we maintain ignorance about the data, while extending expressive power: the resulting ES-joints are generally not decomposable into an ES-prior and expert PFSs. An example is the Variable Share algorithm introduced in [46].

4.5.3 An Alternative to MAP Data Analysis

Sometimes we have data x^n that we want to analyse. One way to do this is by computing the MAP sequence of experts. Unfortunately, we do not know how to compute the MAP sequence for general HMMs. We propose the following alternative way to gain in sight into the data. The forward and backward algorithm compute $P(x^i, q_i^{\mathrm{p}})$ and $P(x^n|q_i^{\mathrm{p}}, x^i)$. Recall that q_i^{p} is the productive state that is used at time i. From these we can compute the a-posteriori probability $P(q_i^{\mathrm{p}}|x^n)$ of each productive state q_i^{p}. That is, the posterior probability taking the entire future into account. This is a standard way to analyse data in the HMM literature. [66] To arrive at a conclusion about experts, we simply project the posterior on states down to obtain the posterior probability $P(\xi_i|x^n)$ of each expert $\xi \in \Xi$ at each time $i = 1, \ldots, n$. This gives us a sequence of mixture weights over the experts that we can, for example, plot as a $\Xi \times n$ grid of gray shades. On the one hand this gives us mixtures, a richer representation than just single experts. On the other hand we lose temporal correlations, as we treat each time instance separately.

4.6 Conclusion

In prediction with expert advice, the goal is to formulate prediction strategies that perform as well as the best possible expert (combination). Expert predictions can be combined by taking a weighted mixture at every sample size. The best combination generally evolves over time. In this chapter we introduced expert sequence priors (ES-priors), which are probability distributions over infinite sequences of experts, to model the trajectory followed by the best expert combination. Prediction with expert advice then amounts to marginalising the joint distribution constructed from the chosen ES-prior and the experts' predictions.

We employed hidden Markov models (HMMs) to specify ES-priors. HMMs' explicit notion of current state and state-to-state evolution naturally fit the temporal correlations we seek to model. For reasons of efficiency we use HMMs with silent states. The standard algorithms for HMMs (Forward, Backward, Viterbi and Baum-Welch) can be used to answer questions about the ES-prior as well as the induced distribution on data. The running time of the forward algorithm can be read off directly from the graphical representation of the HMM.

Our approach allows unification of many existing expert models, including mixture models and fixed share. We gave their defining HMMs and recovered the best known running times. We also introduced two new parameterless generalisations of fixed share. The first, called the switch distribution, was recently introduced to improve model selection performance. We rendered its parametric definition as a small HMM, which shows how it can be evaluated in linear time. The second, called the run-length model, uses a run-length code in a novel way, namely as an ES-prior. This model has quadratic running time. We compared the loss bounds of the two models asymptotically, and showed that the run-length model is preferred if the number of switches grows like $(\log n)^2$ or faster, while the switch distribution is preferred if it grows slower. We provided graphical representations and loss bounds for all considered models.

Finally we described a number of extensions of the ES-prior/HMM approach, including approximating methods for large HMMs, as well as recursive combinations of expert models.

Algorithm 4.1 Forward($\mathbb{A}$). Fix an unfolded deterministic HMM prior $\mathbb{A} = \langle Q, Q_{\mathrm{p}}, \mathrm{P}_\circ, \mathrm{P}, \Lambda \rangle$ on Ξ, and an $\mathcal{X}$-PFS P_ξ for each expert $\xi \in \Xi$. The input consists of a sequence x^ω that arrives sequentially.

Declare the weight map (partial function) $w : Q \to [0,1]$.
$w(v) \leftarrow \mathrm{P}_\circ(v)$ **for all** v s.t. $\mathrm{P}_\circ(v) > 0$.
$\triangleright$ $\mathrm{dom}(w) = I$
for $n = 1, 2, \ldots$ **do**
  Forward Propagation(n)
  Predict next expert: $P(\boldsymbol{\xi}_n = \xi | x^{n-1}) = \dfrac{\sum_{v \in Q_{\{n\}} : \Lambda(v) = \xi} w(v)}{\sum_{v \in Q_{\{n\}}} w(v)}$.
  Loss Update(n)
  Report probability of data: $P(x^n) = \sum_{v \in Q_{\{n\}}} w(v)$.
end for

Procedure Forward Propagation(n):
while $\mathrm{dom}(w) \neq Q_{\{n\}}$ **do**
  $\triangleright$ $\mathrm{dom}(w) \subseteq Q_{[n-1,n]}$
  Pick a $<$-minimal state u in $\mathrm{dom}(w) \setminus Q_{\{n\}}$.
  $\triangleright$ $u \in Q_{[n-1,n)}$
  for $v \in S_u$ **do**
    $\triangleright$ $v \in Q_{(n-1,n]}$
    $w(v) \leftarrow 0$ **if** $v \notin \mathrm{dom}(w)$.
    $w(v) \leftarrow w(v) + w(u)\,\mathrm{P}(u \to v)$.
  end for
  Remove u from the domain of w.
end for

Procedure Loss Update(n):
for $v \in Q_{\{n\}}$ **do**
  $\triangleright$ $v \in Q_{\mathrm{p}}$
  $w(v) \leftarrow w(v) P_{\Lambda(v)}(x_n | x^{n-1})$.
end for