Adjusted Viterbi Training. A proof of concept.

Jüri Lember and Alexey Koloydenko

January 20, 2004

1 Introduction

Motivated by applications of the Viterbi Training (VT) algorithm to estimate parameters of Hidden Markov Models in speech recognition (9, 19, 24, 25, 30, 31), natural language modelling (20), image analysis (16), bioinformatics (6, 21), and by connections with constrained vector quantization (4, 8), we *propose a new principled way to improve accuracy of the VT estimators while preserving the essential computational advantages of the baseline algorithm.*

Let θ_l be the emission parameters of an HMM with states $l \in S = \{1, ..., K\}$. The standard method to compute the maximum likelihood estimator of θ is the EM algorithm that in the HMM context is also known as the *Baum-Welch* or *forward-backward algorithm* (1, 2, 9, 11, 23, 24, 34). Since EM can in practice be computationally expensive, it is commonly replaced by *Viterbi Training*. VT effectively replaces the computationally costly expectation (E) step of EM by an appropriate maximization step that is computationally less intensive. An important example of successful and elaborate application of VT in industry is Philips speech recognition systems (19).

There are also variations of VT that use more than one best alignment, or several perturbations of the best alignment (20). The improvements that we explore are, however, of a different nature: Roughly, we increase estimation accuracy by means of analytic calculations and do not require computing more than one optimal alignment.

The message of our work is as follows: *If an application relies on the computational efficiency of VT, and in particular finds any of the efficient implementations of EM (e.g. (10, 33)) still too intensive, such an application is likely to benefit from our adjustment: The proposed accuracy improvement requires no extra point-wise processing of the data.*

Let us recall that VT is inferior to EM in terms of accuracy because the VT estimators need not be (local) maximum likelihood estimators (VT does not necessarily increase the likelihood), leading to bias and inconsistency (§2).

Given current parameter values, VT first finds a Viterbi *alignment* that is a sequence of hidden states maximizing the likelihood of the observed data. Observations assumed to have been emitted from state l, are regarded as an *i.i.d.* sample from P_l, the corresponding emission distribution. These observations produce $\hat{P}_l^n$, the empirical version of P_l, and ultimately $\hat{\mu}_l$, a maximum likelihood estimate of θ_l. $\hat{\mu}$ is then used to find an alignment in the next step, and so forth. It can be shown that in general this procedure

converges in finitely many steps; also, it is usually much faster than EM.

In speech recognition, the same training procedure was already described by L. Rabiner *et al.* in (12, 25) (see also (23, 24)) who considered his procedure as a variation of the *Lloyd algorithm* from vector quantization, and referred to it as *segmential K-means training*. The analogy with vector quantization is especially pronounced when the underlying chain is a sequence of *i.i.d.* variables in which case the observations are simply an *i.i.d.* sample from a mixture distribution (§3). For mixture models, Viterbi training was also described by R. Gray *et al.* in (4), where the training algorithm was considered in the vector quantization context under the name of *entropy constrained vector quantization (ECVQ)*. (See also (8) for more recent developments in this theory.) A better known name for VT in the mixture case is *Classification EM (CEM)*, (3, 7), stressing that instead of the mixture likelihood, CEM maximizes the *Classification Likelihood* (3, 7, 18). Also, for the uniform mixture of Gaussians with a common covariance matrix of the form $\sigma^2 I$ and unknown σ, VT, or CEM, is equivalent to the *k-means clustering* (3, 4, 7, 28).

Our ultimate goal is to alleviate the inconsistency of the VT estimators in the general HMM case while preserving the fast convergence and computational feasibility of the baseline VT algorithm. To this end, we note that θ^*, the true parameters, are asymptotically a fixed point of EM but not of VT §2, §3. In the multivariate mixture models, for example, EM is noticed to typically produce improved partitions when started with reasonable ones (7). This leads us to conjecture that the reason has to do with the fixed point property, and the effect might also be appreciable in the general HMM case. We therefore attempt to adjust VT in order to restore this property by studying the asymptotics of $\hat{P}_l^n$. Thus, we study the existence of Q_l $l \in S$

$$\hat{P}_l^n \Rightarrow Q_l, \quad l \in S \quad \text{a.s.}, \tag{1}$$

first in the general HMM context – §2, and then in the special case of mixture models – §3. If such limiting measures exist, then under certain continuity assumptions, the estimators $\hat{\mu}_l$ will converge to μ_l, where

$$\mu_l = \arg\max_{\theta_l} \int \ln f_l(x, \theta_l) Q_l(dx).$$

Taking into account the difference between μ_l and the true parameter, the appropriate adjustment of the Viterbi training can now be defined (§2).

However, the asymptotic behavior of $\hat{P}_l^n$ is not in general straightforward and its analysis requires an extension of the definition of Viterbi alignment *at infinitum* (15). With the infinite alignment one can prove the existence of the limiting measures Q_l (1), which is essential for the general definition of the adjusted Viterbi training.

To implement these ideas in practice, a closed form of Q_l (or $\hat{\mu}_l$) as a function of the true parameters is necessary. However, the measures Q_l depend on the transition as well as on the emission models, and computing Q_l can be very difficult. In the special case of mixture models §3, on the other hand, the measures Q_l are easier to find. Although *mixture models are not our goal*, we are in part motivated by the continuing interest of others in computational efficiency and accuracy of parameter estimation in mixture models (5, 17). In §3, we describe the adjusted Viterbi training (VA1) for the mixture case,

which we view, however, *only as a proof of concept*: VA1 *recovers the asymptotic fixed point property* and, since its adjustment function *does not depend on data*, each iteration of VA1 enjoys the *same order of computational complexity (in terms of the sample size) as VT*. Moreover, for commonly used mixtures, such as, for example, mixtures of multivariate normal distributions with unknown means and known covariances (Example 3.1), the adjustment function is available in a *closed form* requiring integration with the mixture densities. Depending on the dimension of the emission variates and on the number of components, and on the available computational resources, one can vary the accuracy of the adjustment. We reiterate that, unlike the computations of the E-step of EM, computations of the adjustment *do not involve evaluation and subsequent summation of the mixture density at individual data points.*

We first introduce these ideas for the case of known mixture weights and then extend them in §3.1 to the case of unknown weights.

To test our theory, in §5 we simulate a mixture of two univariate normal distributions with unit variance, unknown means, and unequal but comparable weights. The main goal of our simulations is to compare the performances of VT, VA1, and EM in terms of the accuracy, convergence, amount of computations per iteration, and the total amount of computations. The simulations are performed with different types of the initial guess, and with the weights assumed to be known, §5.1, and unknown, §5.2; the results (§5.3) are consistently in favor of VA1. Similar simulations have been performed for mixtures of multivariate Gaussians with known covariances, using stochastic approximations for the adjustement and leading to similar conclusions, but details (except for the discussion of Example 3.1) are omitted for conciseness.

In §4, we briefly propose VA2, a more advanced correction to VT, presently merely as a mathematical complement of our adjustment idea; we verify its fast convergence and high accuracy on the simulated data in §5, but its computationally feasible implementations would require more investigation. A concluding summary is given in §6.

2 General HMMs

Let Y be a Markov chain with a finite state space S. We assume Y to be irreducible and aperiodic with the transition matrix $P = (p_{ij})$ and the initial distribution π that is also the stationary distribution of Y. To every state $l \in S$ there corresponds an *emission distribution* P_l on $(\mathcal{X}, \mathcal{B})$, a separable metric space and the corresponding Borel σ-algebra. Let f_l, the density of P_l with respect to some reference measure λ (for instance, the Lebesgue measure), be known up to the parametrization $f_l(x; \theta_l)$. When Y is in state l, an observation according to $P_l(\theta^*)$ and independent of everything else is emitted, with $\theta^* = (\theta_1^*, \ldots, \theta_K^*)$ being the unknown true parameters.

Thus, for any $y = y_1, y_2, \ldots$, a realization of Y, there corresponds a sequence of independent random variables, $X_1, X_2, \ldots$, where X_n has distribution P_{y_n}. Note that we only observe $X = X_1, X_2, \ldots$ and the realization y is unknown (Y is hidden).

The distribution of X is completely determined by the chain parameters (P, π) and the emission distributions P_l, $l \in S$. The process X is also *mixing* and, therefore, ergodic. We now recall the Viterbi Alignment and Training.

Let $x_1, \ldots, x_n$ be first n observations on X. Let $\Lambda(q_1, \ldots, q_n; x_1, \ldots, x_n; \theta)$ be the likelihood function $\mathbf{P}(Y_i = q_i,\ i = 1, \ldots, n) \prod_{i=1}^n f_{q_i}(x_i; \theta_{q_i})$, $q_i \in S$.

The *Viterbi alignment* is any sequence of states $q_1, \ldots, q_n \in S$ that maximizes the likelihood of $x_1, \ldots, x_n$, θ being fixed. Thus, for a fixed θ, the Viterbi alignment is the maximum-likelihood estimator of *the realization of* $Y_1, \ldots, Y_n$ given $x_1, \ldots, x_n$. In the following, the Viterbi alignment will be referred to as the alignment. Since the alignment need not be unique, for each $n \geq 1$, let $\mathcal{V}$ denote the set of all state sequences resulting in the alignment:

$$\mathcal{V}(x_1, \ldots, x_n; \theta) = \{v \in S^n :\ \forall w \in S^n\ \Lambda(v; x_1, \ldots, x_n; \theta) \geq \Lambda(w; x_1, \ldots, x_n; \theta)\}. \tag{2}$$

Any map $v : \mathcal{X}^n \mapsto \mathcal{V}(x_1, \ldots, x_n; \theta)$ will also be called an alignment. Further, unless explicitly specified, v_θ will denote an arbitrary element of $\mathcal{V}(x_1, \ldots, x_n; \theta)$.

Viterbi Training

1.) Choose an initial value $\theta^0 = (\theta_1^0, \ldots, \theta_K^0)$.

2.) Given θ^j $j \geq 0$, compute the alignment

$$v_{\theta^j}(x_1, \ldots, x_n) = (v_1, \ldots, v_n)$$

and partition $x_1, \ldots, x_n$ into K subsamples, with x_k going to the l^{th} subsample if and only if $v_k = l$. Equivalently, evaluate K empirical measures defined in (3) below:

$$\hat{P}_l^n(A; \theta^j) := \frac{\sum_{i=1}^n I_{A \times l}(x_i, v_i)}{\sum_{i=1}^n I_l(v_i)}, \quad A \in \mathcal{B}, \quad l \in S, \tag{3}$$

where I_A stands for the indicator function of set A.

3.) For every subsample find the MLE given by:

$$\hat{\mu}_l(\theta^j) = \arg\max_{\theta_l \in \Theta_l} \int \ln f_l(x; \theta_l) \hat{P}_l^n(dx; \theta^j), \tag{4}$$

and take $\theta_l^{j+1} = \hat{\mu}_l(\theta^j), \quad l \in S$. If for some $l \in S$, $v_i \neq l$ for any $i = 1, \ldots, n$ (l^{th} subsample is empty), then the empirical measure $\hat{P}_l^n$ is formally undefined, in which case we take $\theta_l^{j+1} = \theta_l^j$. We omit this exceptional case in the ensuing discussion.

VT can be interpreted as follows. Suppose that at step j, $\theta^j = \theta^*$ and hence v_{θ^j} is obtained using the true parameters. The training is then based on the assumption that the alignment $v(x_1, \ldots, x_n) = (v_1, \ldots, v_n)$ is correct, i.e., $v_i = Y_i$, $i = 1, \ldots, n$. In this assumption were true, the empirical measures $\hat{P}_l^n(\theta^j)$, $l \in S$ would be obtained from the i.i.d. sample generated from $P_l(\theta^*)$, and the MLE $\hat{\mu}_l(\theta^*)$ would be the natural estimator to use. Clearly, under this assumption (and passing from $x_1, x_2, \ldots$ to $X_1, X_2, \ldots$), $\hat{P}_l^n(\theta^*) \Rightarrow P_l(\theta^*)$ a.s. and, provided that $\{f_l(\cdot; \theta) : \theta \in \Theta_l\}$ is a P_l-Glivenko-Cantelli class and Θ_l is equipped with some suitable metric, $\lim_{n \to \infty} \hat{\mu}_l(\theta^*) = \theta_l^*$ a.s. Hence, if n is sufficiently large, then $\hat{P}_l^n \approx P_l$ and $\theta_l^{j+1} = \hat{\mu}_l(\theta^*) \approx \theta_l^* = \theta_l^j$, $\forall l$ i.e. $\theta^j = \theta^*$ would be (approximately) a fixed point of the training algorithm.

A weak point of the previous argument is that the alignment in general is not correct even when the parameters used to find it are, i.e. generally $v_i \neq Y_i$. In particular, this implies that the empirical measures $\hat{P}_l^n(\theta^*)$ are not obtained from an i.i.d. sample taken from $P_l(\theta^*)$. Hence, we have no reason to believe that $\hat{P}_l^n(\theta^*) \Rightarrow P_l(\theta^*)$ a.s. and $\lim_{n\to\infty} \hat{\mu}_l(\theta^*) = \theta_l^*$ a.s. Moreover, we do not even know whether the sequences of empirical measures $\{\hat{P}_l^n(\theta^*)\}$ and MLE estimators $\{\hat{\mu}_l(\theta^*)\}$ converge (a.s.) at all.

In (15), we prove the existence of limiting probability measures $Q_l(\theta, \theta^*)$, $l \in S$, that depend on θ, the parameters used to find the alignment $v_\theta(x_1, \ldots, x_n)$, and on θ^*, the true parameters with which the random samples are emitted. Namely, these Q_l, $l \in S$ will be such that for every l

$$\hat{P}_l^n(\theta^*) \Rightarrow Q_l(\theta^*, \theta^*), \quad \text{a.s..} \tag{5}$$

Suppose also that the parameter space Θ_l is equipped with some metric. Then, under certain consistency assumptions on classes $\mathcal{F}_l = \{f_l(\cdot; \theta_l) : \theta_l \in \Theta_l\}$, the convergence

$$\lim_{n\to\infty} \hat{\mu}_l(\theta^*) = \mu_l(\theta^*, \theta^*) \quad \text{a.s.} \tag{6}$$

can be deduced from (5), where

$$\mu_l(\theta, \theta^*) \stackrel{\text{def}}{=} \arg\max_{\theta_l' \in \Theta_l} \int \ln f_l(x; \theta_l') Q_l(dx; \theta, \theta^*). \tag{7}$$

We also show that in general, for the baseline Viterbi training $Q_l(\theta^*, \theta^*) \neq P_l(\theta^*)$, implying $\mu_l(\theta^*, \theta^*) \neq \theta_l^*$. In an attempt to reduce the bias $\theta_l^* - \mu_l(\theta^*, \theta^*)$, we next propose the *adjusted Viterbi training*. Suppose (5) and (6) hold. Based on (7), we now consider the mapping

$$\mu_l(\theta) = \mu_l(\theta, \theta), \quad l = 1, \ldots, K. \tag{8}$$

Since this function is independent of the sample, we can define the following correction for the bias:

$$\Delta_l(\theta) = \theta_l - \mu_l(\theta), \quad l = 1, \ldots, K. \tag{9}$$

VA1 – Adjusted Viterbi Training

1.) Choose an initial value $\theta^0 = (\theta_1^0, \ldots, \theta_K^0)$.

2.) Given θ^j, perform the alignment and define K empirical measures $\hat{P}_l^n(\theta^j)$ as in (3).

3.) For every $\hat{P}_l^n$, find $\hat{\mu}_l(\theta^j)$ as in (4) and for each l, define $\theta_l^{j+1} = \hat{\mu}_l(\theta^j) + \Delta_l(\theta^j)$, where Δ_l is defined $\forall l \in S$ in (9).

Note that, as desired, for n sufficiently large, the adjusted training algorithm has θ^* as its (approximately) fixed point: Indeed, suppose $\theta^j = \theta^*$. From (6), $\hat{\mu}_l(\theta^j) = \hat{\mu}_l(\theta^*) \approx \mu_l(\theta^*) = \mu_l(\theta^j)$, for all $l \in S$. Hence,

$$\theta_l^{j+1} = \hat{\mu}_l(\theta^*) + \Delta_l(\theta^*) \approx \mu_l(\theta^*, \theta^*) + \Delta_l(\theta^*) = \theta_l^* = \theta^j, \quad l \in S. \tag{10}$$

3 Mixture models

In general, no closed form for the distribution $Q_l(\theta^*, \theta^*)$ in (5) is available. Therefore, the mapping (8) may be impossible to determine exactly and approximations of Q_l should be used for the adjustments of Viterbi training (§2). However, in the case of the mixture models, the distributions Q_l are straightforward to find and the adjusted Viterbi training can therefore be immediately given. In this model, Y, the underlying Markov chain, is a sequence of i.i.d. discrete random variables with the state space $S = \{1, \dots, K\}$ of *mixture components*. Thus, the transition probabilities are $p_{ij} = p_j$, $i, j \in S$, where p_j are mixture weights. To each component $l \in S$ there corresponds a probability distribution P_l with density $f_l = f_l(\cdot; \theta_l^*)$, where θ_i^* are the true parameters. Unless explicitly stated otherwise, the mixture weights p_l will be assumed to be known. Such a model produces observations $x_1, \dots, x_n$, that are regarded as an i.i.d. sample from the mixture distribution P with density

$$\sum_{i=1}^{K} p_i f_i = \sum_{i=1}^{K} p_i f_i(\cdot; \theta_i^*) = f(\cdot; \theta^*) = f. \tag{11}$$

For any set of parameters $\theta = (\theta_1, \dots \theta_K)$, the alignment v_θ can be obtained via a *Voronoi partition* $\mathcal{S}(\theta) = \{S_1(\theta), \dots, S_K(\theta)\}$, where

$$S_1(\theta) = \{x : p_1 f_1(x; \theta_1) \geq p_j f_j(x; \theta_j), \quad \forall j \in S\} \tag{12}$$

$$S_l(\theta) = \{x : p_l f_l(x; \theta_l) \geq p_j f_j(x; \theta_j), \quad \forall j \in S\} \backslash (S_1 \cup \dots \cup S_{l-1}), \quad l = 2, \dots, K. \tag{13}$$

Now, the alignment can be defined as follows: $v_\theta(x) = l$ if and only if $x \in S_l(\theta)$. In particular, given the Voronoi partition $\mathcal{S}(\theta) = \{S_1, \dots, S_l\}$, the empirical measures $\hat{P}_l^n$ (3) are

$$\hat{P}_l^n(A; \theta) = \frac{\sum_{i=1}^n I_{S_l(\theta) \cap A}(x_i)}{\sum_{i=1}^n I_{S_l(\theta)}(x_i)}, \quad A \in \mathcal{B}, \quad l \in S. \tag{14}$$

Thus, given the same partition, $\hat{\mu}_l(\theta)$ (4), the sub-sample MLE for component l, becomes

$$\hat{\mu}_l(\theta) = \arg\max_{\theta_l' \in \Theta_l} \int_{S_l(\theta)} \ln f_l(x; \theta_l') \hat{P}_n(dx), \tag{15}$$

where $\hat{P}_n$ is the ordinary empirical measure associated with the given random sample. The convergence (5) then follows immediately from (14). Indeed, for any θ, by virtue of the *Strong Law of Large Numbers* we have

$$\lim_{n \to \infty} \hat{P}_l^n(A; \theta) \overset{\text{a.s.}}{=} \frac{P(A \cap S_l(\theta); \theta^*)}{P(S_l(\theta); \theta^*)} = \frac{\int_{S_l(\theta) \cap A} f(x; \theta^*) d\lambda(x)}{\int_{S_l(\theta)} f(x; \theta^*) d\lambda(x)} = \frac{\sum_i p_i \int_{S_l(\theta) \cap A} f_i(x; \theta_i^*) d\lambda(x)}{\sum_i p_i \int_{S_l(\theta)} f_i(x; \theta_i^*) d\lambda(x)}.$$

Since $\mathcal{X}$ is separable, it follows that $\hat{P}_l^n \Rightarrow Q_l$ a.s., where

$$q_l(x; \theta, \theta^*) \propto f(x; \theta^*) I_{S_l(\theta)} = (\sum_i p_i f_i(x; \theta^*)) I_{S_l(\theta)}, \quad l = 1, \dots, K$$

are the densities of respective $Q_l(\theta, \theta^*)$'s.

Now it is clear that even when the partition $\mathcal{S}(\theta^*)$ is obtained using the true parameters θ^*, $Q_l(\theta^*, \theta^*)$, the limiting distribution (with density $q_l(x; \theta^*, \theta^*)$), can be different

from $P_l(\theta^*)$, the desired distribution (with density $f_l(x;\theta^*)$). Likewise, $\mu_l(\theta^*)$ (8) can be different from

$$\theta_l^* = \arg\max_{\theta_l' \in \Theta_l} \int \ln f_l(x;\theta_l') f_l(x;\theta_l^*) d\lambda(x).$$

In order to see this, note that (7) and (8) in the context of the mixture model specialize to

$$\mu_l(\theta,\theta^*) = \arg\max_{\theta_l' \in \Theta_l} \int_{S_l(\theta)} \ln f_l(x;\theta_l') f(x;\theta^*) d\lambda(x) \tag{16}$$

$$\mu_l(\theta) = \arg\max_{\theta_l' \in \Theta_l} \int_{S_l(\theta)} \ln f_l(x;\theta_l') \big(\sum_i p_i f_i(x;\theta_i)\big) d\lambda(x), \tag{17}$$

respectively. We also emphasize that Δ can be significant which justifies the adjustment.

Example 3.1 *Let*

$$f(x;\theta^*) = \frac{1}{K}\sum_{l=1}^{K} \phi(x;\theta_l^*),$$

where $\phi(x;\theta_l^*)$ is the density of the d-variate normal distribution with identity covariance matrix and vector of means $\theta_l^* \in \mathbb{R}^d = \Theta_l$ for $l = 1, 2, \ldots, K$. In this case, for each K-tuple of parameters $\theta = (\theta_1, \ldots, \theta_K)$, the decision-rule for the alignment is essentially as follows (disregarding possible ties): $v_\theta(x) = i$ if and only if $\|x - \theta_i\| \leq \min_j \|x - \theta_j\|$. Thus, the decision regions in this case correspond to the Voronoi partition in its original sense, justifying our generalization of this term. Now, it can be easily seen that for all $m = 1, \ldots, d$:

$$(\mu_l(\theta))_m = \frac{\sum_{i=1}^K \int_{S_l(\theta)} x_m \phi(x;\theta_i) dx_1 \cdots dx_d}{\sum_{i=1}^K \int_{S_l(\theta)} \phi(x;\theta_i) dx_1 \cdots dx_d}. \tag{18}$$

Although the functions μ_l are data independent, the exact integration in (18) can require intensive computations when d and K are large. If this becomes an issue, one may be interested in approximations of (18). Even when approximated, the adjustment can still asymptotically reduce the bias provided that the approximation error is smaller than Δ_l. In the context of the above example, one might think of the following approximations for $\Delta_l(\theta) = \theta_l - \mu_l(\theta)$:

1.) Approximate $\Big(\sum_l f_i(x;\theta_i)\Big) I_{S_l(\theta)}$ in (18) by $f_l(\theta_l, x) I_{S_l(\theta)}$, so

$$(\mu_l)_m \approx \frac{\int_{S_l(\theta)} x_m \phi(x;\theta_l) dx_1 \cdots dx_d}{\int_{S_l(\theta)} \phi(x;\theta_l) dx_1 \cdots dx_d}. \tag{19}$$

This approximation is based on the limiting case when the components are "infinitely" far from each other.

2.) If $K > d$, then some components are fully surrounded by others, namely, the partition cells corresponding to such "internal" components are bounded (Figure 1). It is

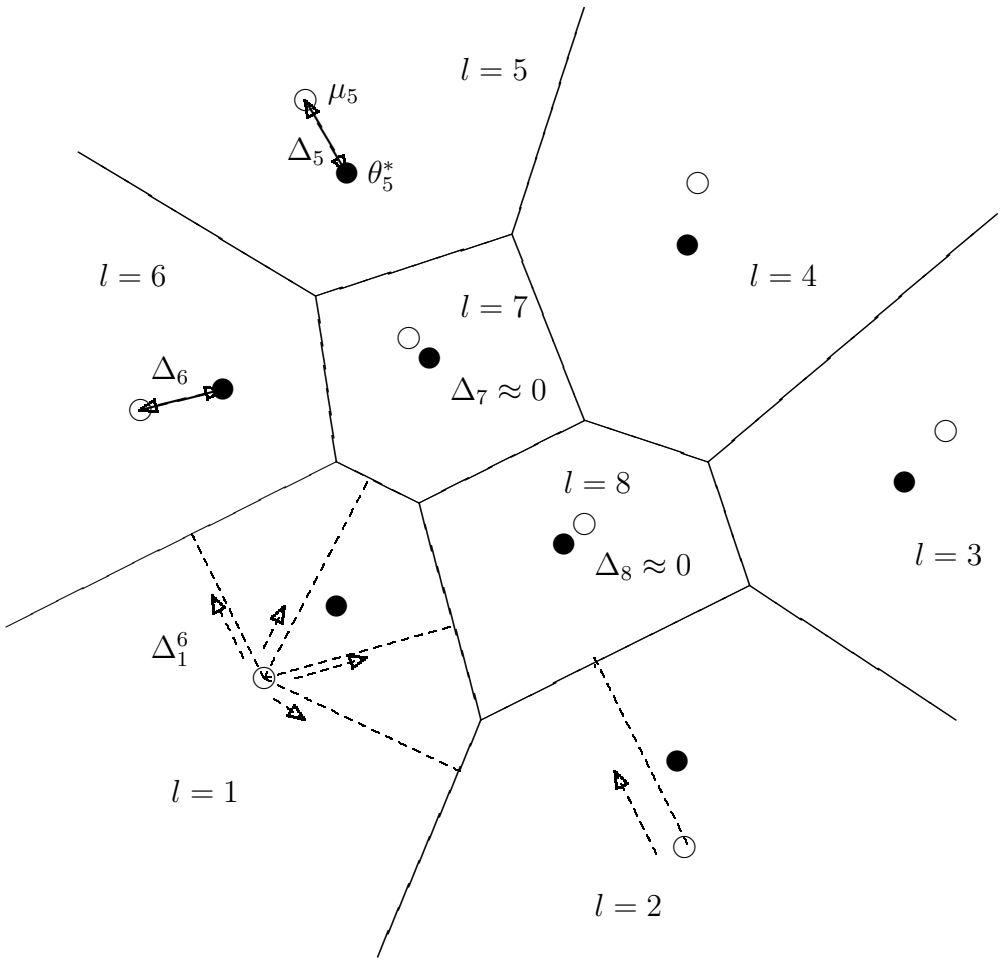

Fig. 1. An example of the Voronoi partition for multiple components. True parameters θ^* and corresponding $\mu(\theta^*)$ are marked with solid and transparent dots, respectively. For $l = 1$ component, $\Delta_l^j(\theta^*)$, the correction components, are indicated. For $l = 2$ component, the main direction of the correction is indicated. Neglecting the corrections for the estimators corresponding to the bounded Voronoi regions appears to give a reasonable approximation.

then conceivable that Δ_l's that correspond to the bounded cells are less significant than the others, in which case one might correct only the estimators of the internal components. This approach seems to be particularly appealing for speech recognition. In speech recognition, a phonem is often modelled by a mixture of Gaussians or Laplace densities (19). One significant difficulty in the acoustic-phonetic modelling is determining the boundaries of the phonemes (in the appropriate feature space). The boundaries depend mostly on the external, or "surface", components. If the mixture parameters are estimated by Viterbi training, then the external components tend to be too far from their means (see Figure 1), resulting in less accurate boundaries and an overall imprecision of the model estimation. Thus, correcting only the "surface" components might already significantly improve the entire acoustic-phonetic model.

3.) Every Voronoi cell is determined by several hyperplanes and every such hyperplane H^j gives rise to Δ_l^j, a component of Δ_l in the direction perpendicular to H^j and corresponding to the l^{th} term in the sum in (18). Thus, $\Delta_l = \sum_j \Delta_l^j$ (see, for example, $l = 1$ cell in Figure 1). It may be reasonable to find only the "main direction" of correction, i.e. the largest Δ_l^j for each l (see, for example, the $l = 2$ cell in Figure 1).

4.) The integrals (18) are very easy to compute by Monte-Carlo (or quasi Monte-Carlo) methods (26, 27, 29). This leads to the *stochastically adjusted Viterbi training (SAV1)* that modifies step 3 of VA1 as follows:

a) Generate a sample from $\sum_{l=1}^{K} f_l(\theta_l^j)$, say $y_1, \ldots, y_m$.

b) Using the sample $y_1, \ldots, y_m$, approximate Voronoi partition by $\hat{\mathcal{S}}(\theta^j)$, and $\mu_l(\theta^j)$ (18) - by the appropriate Monte-Carlo estimates $\hat{\mu}_l^{MC}(\theta^j)$

c) Estimate the correction

$$\hat{\Delta}_l^{MC}(\theta^j) := \theta_l^j - \hat{\mu}_l^{MC}(\theta^j), \quad l = 1, \ldots, K.$$

The additional sampling step in SAV1 obviously jeopardizes the computational attractiveness of VA1. However, there are many ways to control the the Monte-Carlo integration in order to keep the overall complexity of SAV1 considerably smaller than that of EM. A great advantage of SAV1 is that *it is easy to implement even in very complex settings (including that of HMM).* In (13), SAV1 was implemented for 2-dimensional Gaussian mixtures. The simulations showed that in terms of precision, SAV1 is comparable with VA1 and EM, and strongly outperforms VT. Moreover, in this 2-dimensional setting, SAV1 and VA1 outperform VT even in terms of the number of iterations.

Remark 3.2 *In Example 3.1, the decision regions correspond to the Voronoi partition in its original sense. Moreover, it is easy to see that in this particular case, the Viterbi training is none other than the well-known (generalized) Lloyd algorithm designed for finding vector quantizers, which in this case are also called* K-means *(see, e.g. (28)). In this case, the estimators obtained by the Viterbi training are empirical* K*-means. These latter estimators enjoy certain desirable properties, and in particular they are consistent with respect to the population* K*-means (22). However, they need not be consistent with respect*

to θ^*, our parameters of interest. In the mixture case, the Viterbi training can always be considered as the (generalized) Lloyd algorithm, and the estimators obtained by Viterbi training can be regarded as (generalized) empirical K-means (4). This observation links the study of Viterbi Training and related algorithms to the theory of vector quantization.

3.1 Unknown weights

We consider the case when the mixture weights p_l are unknown, which corresponds to the case of the unknown transition parameters (P, π) in the general HMM context.

The Voronoi partition depends on the weight-vector $p = (p_1, \dots, p_K)$ as well as on θ. Hence, $\mathcal{S}(\theta, p)$ and the vector p should be re-estimated at each step along with θ. Given a Voronoi partition $\mathcal{S} = \{S_1, \dots, S_K\}$, the simplest way to estimate the weights p_l is to take $p_l = \hat{P}_n(S_l)$, the empirical measure of S_l. Hence all the algorithms considered so far can be modified accordingly to include the weight estimation as in (20).

$$p_l^{j+1} = \hat{P}_n(S_l(\theta^j, p^j)), \quad l = 1, \dots, K. \tag{20}$$

Taking into account the asymptotics, it is easy to correct the estimators p^{j+1} as well. Indeed, suppose $\theta^j = \theta^*$ and $p^j = p$, i.e. $\mathcal{S}(\theta^j, p^j) = \mathcal{S}(\theta^*, p) = \mathcal{S}^*$. If $n \to \infty$, then

$$\hat{P}_n\Big(S_l(\theta^*, p)\Big) \overset{\text{a.s.}}{\to} P\Big(S_l(\theta^*, p)\Big) = \int_{S_l(\theta^*, p)} f(x; \theta^*) d\lambda = \sum_i p_i \int_{S_l(\theta^*, p)} f_i(x; \theta_i^*) d\lambda. \tag{21}$$

In general the latter differs from p_l. The difference is $p_l - P\big(S_l(\theta^*, p)\big)$. Hence, by analogy with (9), we can define the weight correction $D(\theta, p) = (D_1(\theta, p), \dots, D_K(\theta, p))$ as follows:

$$D_l(\theta, p) = p_l - \sum_i p_i \int_{S_l(\theta, p)} f_i(x; \theta_i) d\lambda, \tag{22}$$

which is also *data independent*. We now summarize the above by giving a formal definition of the adjusted Viterbi training with the weight correction. The Viterbi training with p unknown can be defined similarly.

VA1 with the weight correction

1.) Choose $\theta^0 = (\theta_1^0, \dots, \theta_K^0)$ and $p^0 = (p_1^0, \dots p_K^0)$

2.) Given $\theta^j = (\theta_1^j, \dots, \theta_K^j)$ and $p^j = (p_1^j, \dots p_K^j)$ define the Voronoi partition $\mathcal{S}(\theta^j, p^j) = \{S_1, \dots, S_K\}$ as in (12) and (13), and the empirical measures $\hat{P}_l^n(\theta^j, p^j)$ as in (14).

3.) Put $\theta^{j+1} = \hat{\mu}^j(\theta^j) + \Delta(\theta^j)$, where $\hat{\mu}^j$ is defined in (17).

4.) Put $p^{j+1} = \hat{P}_n(S_l(\theta^j, p^j)) + D(\theta^j, p^j)$.

4 VA2 – A more advanced adjustment

The adjusted Viterbi training is designed to asymptotically fix the true parameter θ^*, returning approximately the correct solution given this solution as the initial guess and

given an infinitely large data sample: VA1(θ^*) $\approx \theta^*$. VA2 goes further and attempts to maximally expand $\{\theta : \text{VA1}(\theta) \approx \theta^*\}$, the set of parameter values that are asymptotically mapped to the true ones, to $\{\theta : \text{VA2}(\theta) \approx \theta^*\}$. Specifically, if the algorithm ever arrives at $\mathcal{S}(\theta^*)$, the Voronoi partition corresponding to the true parameters θ^*, then we would like to coerce the adjusted estimates to return θ^*. Let us explain these ideas in more detail.

Let $\mathcal{S}^*$ stand for $\mathcal{S}(\theta^*)$, the true Voronoi partition (that also coincides with the Bayes decision boundary). The mapping $\theta \mapsto \mathcal{S}(\theta)$ is generally many-to-one, hence the set $\Theta(\mathcal{S}^*) = \{\theta : \mathcal{S}(\theta) = \mathcal{S}^*\}$ generally contains more than one element. (This also means that guessing $\mathcal{S}^*$, i.e. guessing any element from $\Theta(\mathcal{S}^*)$, is generally easier than guessing θ^*.) We now introduce VA2:

Note first that $\mu_l(\theta, \theta^*)$ in (16), as well as the estimate $\hat{\mu}_l(\theta)$ in (15), depends on θ through $\mathcal{S}(\theta)$ only. However, the correction $\Delta_l(\theta) = \theta_l - \mu_l(\theta, \theta)$ does depend on θ fully and hence would not generally work (in the sense of (10)) for an arbitrary $\theta^j \in \Theta(\mathcal{S}^*)$ unless $\theta^j = \theta^*$. We now attempt to improve the first type of adjustment that is based on adding $\Delta(\theta^j)$ to $\hat{\mu}(\theta^j)$. Namely, we propose the following iterative update for $l = 1, \ldots, K$: Next, define $\mu_{l,\Theta(\mathcal{S}(\theta^0))}(\theta)$ (as function of θ only) to be the restriction of $\mu_l(\theta^0, \theta)$ to $\Theta(\mathcal{S}(\theta^0))$, and write $\mu_{l,\theta^0}(\theta)$ in place of the more cumbersome $\mu_{l,\Theta(\mathcal{S}(\theta^0))}(\theta)$. Let

$$\theta_l^{j+1} = \begin{cases} \mu_{l,\theta^j}^{-1}(\hat{\mu}_l(\theta^j)), & \text{if a unique } \mu_{l,\theta^j}^{-1}(\hat{\mu}_l(\theta^j)) \quad \text{exists} \\ \hat{\mu}_l(\theta^j) + \Delta_l(\theta^j) & \text{otherwise.} \end{cases} \tag{23}$$

For any θ^j and θ^*, the event that $\hat{\mu}(\theta^j, \theta^*)$ belongs to the range of $\mu(\theta^j, \theta)$ as a function of $\theta \in \Theta(\mathcal{S}(\theta^j))$ is of zero probability, as Example 4.1 illustrates. Hence, the introduction of the individual inverses μ_{l,θ^j}^{-1} $l = 1, \ldots, K$ is essential, although still not always effective: In some mixture models (a mixture of normal distributions with unequal weights is one such example), for a fixed l, the event that $\hat{\mu}_l(\theta^j)$ belongs in the range of $\mu_l(\theta^j, \theta)$ (as function of $\theta \in \Theta(\mathcal{S}(\theta^j))$) need not occur with probability one for all θ^j and θ^*. This, and also the fact that the inverses in general need not have a closed form, or may require intensive computations, may reduce the attractiveness of the suggested method. Further discussion of the computational issues related to this method is outside the scope of this paper, except for mentioning the possibility of various, e.g. linear or quadratic, approximations of the above functions $\mu_{l,\theta}^{-1}$.

In order to better understand the meaning of the new adjustment, imagine that $\theta^j \in \Theta(\mathcal{S}^*)$. We would then expect for $l = 1, \ldots, K$:

$$\theta_l^{j+1} = \mu_{l,\theta^j}^{-1}(\hat{\mu}_l(\theta^j)) = \mu_{l,\theta^*}^{-1}(\hat{\mu}_l(\theta^*)) \approx \mu_{l,\theta^*}^{-1}(\mu_l(\theta^*, \theta^*)) = \theta_l^*.$$

The above argument, of course, also depends on the regularity of the above inverses at $\mu_l(\theta^*, \theta^*)$ $l = 1, \ldots, K$, and in this regard our experiments in §5 provide encouraging results for an important model similar to the model in the following example:

Example 4.1 *Let $f(x; \theta^*) = \frac{1}{2}\phi(x - \theta_1^*) + \frac{1}{2}\phi(x - \theta_2^*)$, where ϕ is the density of the standard normal distribution. In this case any Voronoi partition is specified by a single parameter $t = 0.5(\theta_1 + \theta_2)$ solving $\phi(t - \theta_1) = \phi(t - \theta_2)$ (ties are evidently inessential in this context). The true Voronoi partition corresponds to $t^* = 0.5(\theta_1^* + \theta_2^*)$. Given a Voronoi*

partition $\mathcal{S}(t(\theta))$, $\Theta(t) = \{(t-a), (t+a) : a \in \mathbb{R}^+\}$. *Hence, restricted to* $\Theta(t)$, *the function* $\mu_{\mathcal{S}(t)}(\theta) = (\mu_{1,\mathcal{S}(t)}(\theta), \mu_{2,\mathcal{S}(t)}(\theta))$ *depends on one parameter only: Let* a *be this parameter and define* $\mu_{\mathcal{S}(t)}(\theta(a)) = (\mu_1(a), \mu_2(a))$ *as follows:* $\mu_1(a) = -a(1 - 2\Phi(-a)) - 2\phi(-a) + t$, $\mu_2(a) = 2t - \mu_1(a)$, *where* Φ *is the distribution function of the standard normal distribution. After calculating* $\hat{\mu}_1 < \hat{\mu}_2$ *from the data, the inversion equations of* (23) *become*

$$t - [a(1 - 2\Phi(-a)) + 2\phi(a)] = \hat{\mu}_1, \quad t + [a(1 - 2\Phi(-a)) + 2\phi(a)] = \hat{\mu}_2. \tag{24}$$

Obviously (24) *has a (unique) solution if and only if* $\hat{\mu}_1$, $\hat{\mu}_2$ *are symmetric with respect to* t *and the probability of this latter event is clearly zero under the model. Thus, as suggested in* (23), *we consider the equations separately:*

$$a(1 - 2\Phi(-a)) + 2\phi(a) = t - \hat{\mu}_1 \tag{25}$$

$$a(1 - 2\Phi(-a)) + 2\phi(a) = \hat{\mu}_2 - t. \tag{26}$$

It can be shown that (25) *and* (26) *have unique solutions, let us denote the latter by* a_1 *and* a_2, *respectively. The points* $t - a_1$ *and* $t + a_2$ *will be now taken as the estimators of* θ_1^* *and* θ_2^* *for the next step of iterations.*

VA2

1.) Choose $\theta^0 = (\theta_1^0, \ldots, \theta_K^0)$.

2.) Given θ^j, find $\mathcal{S}(\theta^j)$ and define empirical measures $\hat{P}_l^n(\theta^j)$ as in (14).

3.) For every $\hat{P}_l^n$, find $\hat{\mu}_l(\theta^j, \theta^*)$ as in (15).

4.) Update θ^{j+1} in accordance with (23).

VA2 with p unknown can be defined by analogy with §3.1.

5 Simulation studies

In order to support our theory of adjusted Viterbi Training we simulate 1000 i.i.d. random samples of size 1000 according to the following mixture:

$$\frac{1}{\sqrt{2\pi}}(pe^{-\frac{(x-\theta_1)^2}{2}} + (1-p)e^{-\frac{(x-\theta_2)^2}{2}}).$$

The true parameters in our experiments are $\theta^* = (-2.5, 0)$ and $(p, 1-p) = (0.7, 0.3)$. The corresponding density is plotted in Figure 2. Note that for all such mixtures with $p > 0.5$ and $\theta_1 < \theta_2$, $\theta_2 - \theta_1 < \sqrt{2p/(1-p)}$ ($= 2.1602$ in our case) implies that the both means fall on one side of the decision boundary, which makes detection of the second component particularly difficult as is already becoming the case in our setting with $\theta_2^* - \theta_1^* = 2.5$.

Our main goal is to compare the performances of VT, VA1, and EM in terms of the accuracy, convergence, amount of computations per iteration, and the total amount of computations. We implement these algorithms in Matlab (32), providing a fair comparison of their computational intensities based on their execution times. Our code is available

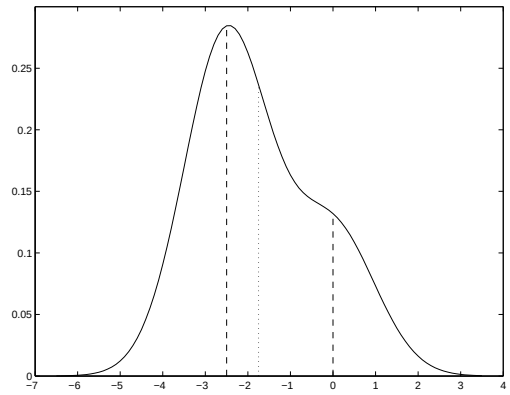

Fig. 2. $\frac{1}{\sqrt{2\pi}}(0.7e^{-\frac{(x+2.5)^2}{2}} + 0.3e^{-\frac{x^2}{2}})$. The dashed vertical lines indicate the means of the individual components, the dotted line marks the mean of the mixture.

for the reader's perusal (14) and is fully-optimized for speed in the case of VT and EM. Consequently, our simulations possibly only overestimate the execution times for VA1.

Additionally, we compare VA2 with the above algorithms by the accuracy and convergence. We use a numerical solver to compute the adjustment function of VA2 and presently make no effort to replace this by a computationally efficient approximation. Hence, we do not discuss the computational intensity of VA2 in this work.

In our experiments, the algorithms are instructed to terminate as soon as the L_2 distance between consecutive θ updates falls below 0.001. We also provide a high precision MLE computed with a built-in matlab optimization function. The cases of known and unknown weights (§3.1) are considered in §5.1 and §5.2, respectively. We report the following statistics for each of the algorithms in the form: average±one standard deviation.

- $\theta = (\theta_1, \theta_2)$ - the estimates of the means;
- p - the estimate of the weight of the first component;
- $||\theta - \theta^*||_{1,2}$ - L_1- and L_2-normed distances between θ and the true parameters;
- n - number of steps used by the algorithm;
- T - total time in milliseconds to execute the entire algorithm;
- t - time in milliseconds to execute one iteration of the algorithm;

5.1 Known weights

It is often the case in speech recognition models that the weights are assumed known.

First, consider $(-1, 2)$ as an "arbitrary" initial guess for θ. Table 1 presents the performance statistics based on the 1000 samples. The baseline Viterbi method terminates quickly (on average in 9.04 steps), outperformed only by VA2, but is the least accurate among the considered methods. As expected, VT also requires fewest computations: 0.2 ms per iteration and 1.85 ms total. Ranked from low to high, accuracies of VA1, VA2,

and EM appear similar, and are about three times superior to that of VT. In units of the VT execution time, EM compares to VA1 as 16.85:6.7 per iteration, and as 20.43:7.59 by the total execution times. In order to illustrate the asymptotic fixed point property, we

Table 1. "Arbitrary" initial guess.

	VT	VA1	VA2	EM	MEL
θ_1	-2.4869±0.0497	-2.4952±0.0500	-2.4959±0.0498	-2.4970±0.0456	-2.4973±0.0456
θ_2	0.2880±0.0732	0.0099±0.0917	0.0082±0.0916	0.0030±0.0757	0.0024±0.0757
$\|\|\theta - \theta^*\|\|_1$	0.3291±0.0844	0.1138±0.0681	0.1133±0.0678	0.0958±0.0562	0.0958±0.0562
$\|\|\theta - \theta^*\|\|_2$	0.2927±0.0727	0.0902±0.0537	0.0899±0.0536	0.0761±0.0451	0.0761±0.0451
n	9.04±1.55	10.49±1.61	7.84±1.59	11.20±0.42	N/A
t	0.20±0.05	1.34±0.19	39.24±1.56	3.37±0.07	N/A
T	1.85±0.55	14.04±2.95	308.57±68.63	37.79±1.56	N/A

Table 2. Correct initial guess.

	VT	VA1	VA2	EM	MLE
θ_1	-2.4904±0.0495	-2.4973±0.0488	-2.4973±0.0490	-2.4973±0.0455	-2.4973±0.0456
θ_2	0.2820±0.0729	0.0051±0.0880	0.0052±0.0892	0.0024±0.0753	0.0024±0.0757
$\|\|\theta - \theta^*\|\|_1$	0.3223±0.0829	0.1087±0.0661	0.1102±0.0664	0.0953±0.0561	0.0958±0.0562
$\|\|\theta - \theta^*\|\|_2$	0.2867±0.0721	0.0861±0.0523	0.0874±0.0525	0.0756±0.0450	0.0761±0.0451
n	5.56±1.72	5.06±1.57	4.73±1.55	5.69±1.29	N/A
t	0.22±0.02	1.37±0.05	42.23±0.94	3.42±0.08	N/A
T	1.21±0.31	6.91±2.05	199.52±65.67	19.39±4.28	N/A

initialize the algorithms to $(-2.5, 0)$, the true value of the parameters, see Table 2. In this case, as expected, the both types of adjustments exit noticeably faster than VT and EM, are comparable in accuracy to EM, and are about three times more accurate than VT. Unlike VA1, VA2 or EM, the baseline algorithm, as predicted, disturbs the correct initial guess, resulting in an appreciable bias. The times per iteration of VA1 and EM are similar to as before, and their total times are (in units of the VT time): 5.71 and 16.03, respectively.

In order to illustrate the idea of the second type of adjustment, we now initialize the algorithms to $(-3.1229, 0.8771)$, which produces the same decision boundary $t = -0.9111$ as $\theta^* = (-2.5, 0)$, the true values. Table 3 collects these results. Note that since VT and VA2 depend on the initial guess only via the decision boundary, they produce in this case exactly the same results (disregarding a small rounding error) as in the case of the correct initial guess (Table 2). As expected, VA2 now terminates significantly faster than its competitors, and accuracy-wise is only slightly superior to VA1 and slightly inferior to EM. The times per iteration of VA1 and EM are similar to as before, and their total times are 7.84 and 20.83, respectively.

5.2 Unknown weights

Assume now that the weights are unknown (§3.1) and hence need to be estimated along with the means. We use the same data and the same three types of conditions as in the case of known weights: Arbitrary initialization to $(-1, 2)$ (Table 4), initialization to the correct values $(-2.5, 0)$ (Table 5), and initialization to $(-3.1229, 0.8771)$, an arbitrary

point giving rise the correct inter-component boundary (Table 6). VT and the adjusted algorithms VA1 and VA2 in this case are implemented with the asymptotic correction (22). (The maximization in the high precision MLE is now performed in the three variables.)

Table 3. Correct decision boundary.

	VT	VA1	VA2	EM	MLE
θ_1	-2.4904±0.0495	-2.4954±0.0497	-2.4973±0.0490	-2.4971±0.0456	-2.4973±0.0456
θ_2	0.2820±0.0729	0.0094±0.0909	0.0052±0.0892	0.0030±0.0757	0.0024±0.0757
$\|\|\theta - \theta^*\|\|_1$	0.3223±0.0829	0.1131±0.0668	0.1102±0.0664	0.0958±0.0562	0.0958±0.0562
$\|\|\theta - \theta^*\|\|_2$	0.2867±0.0721	0.0897±0.0528	0.0874±0.0525	0.0761±0.0450	0.0761±0.0451
n	5.56±1.72	7.09±1.38	4.72±1.56	7.44±0.94	N/A
t	0.22±0.03	1.35±0.05	42.37±1.12	3.42±0.08	N/A
T	1.22±0.31	9.56±1.81	200.24±66.30	25.41±3.19	N/A

Table 4. Unknown weights. "Arbitrary" guess.

	VT	VA1	VA2	EM	MLE
p	0.747 ±0.031	0.703 ±0.028	0.702 ±0.028	0.700 ±0.024	0.699 ±0.024
θ_1	-2.4299 ±0.0753	-2.4919 ±0.0596	-2.4930 ±0.0594	-2.4976 ±0.0531	-2.4992 ±0.0532
θ_2	0.3944 ±0.1178	0.0194 ±0.1099	0.0173 ±0.1094	0.0070 ±0.0944	0.0039 ±0.0947
$\|\|\theta - \theta^*\|\|_1$	0.4775 ±0.1653	0.1382 ±0.0851	0.1372 ±0.0846	0.1179 ±0.0708	0.1179 ±0.0710
$\|\|\theta - \theta^*\|\|_2$	0.4058 ±0.1237	0.1084 ±0.0657	0.1076 ±0.0653	0.0931 ±0.0558	0.0931 ±0.0560
n	14.16 ±3.60	13.85 ±3.25	12.23 ±2.86	24.90 ±2.60	N/A
t	0.72 ±0.15	1.32 ±0.09	39.01 ±1.26	3.52 ±0.35	N/A
T	10.13 ±3.01	18.25 ±4.27	478.19 ±117.67	87.67 ±12.98	N/A

The adjusted algorithms now converge 1.7 (VA1) and 2 (VA2) times faster than EM, and, what is more remarkable, VA1 and VA2 converge even faster than VT. The per iteration times of VA1 and EM compare as about 1.8:4.8 for all the initializations, and the total times – as 1.8:8.7 (arbitrary guess), 1.3:6.67 (true values), and 1.73:7.64 (true boundary), all in units of the VT time. VA1 and VA2 are again at least three times more accurate than VT in θ estimation and about one standard deviation more accurate than VT in the weight estimation. They are also comparable in accuracy to EM.

5.3 Summary of the results

VA1 is consistently close in accuracy to EM which is always superior to Viterbi Training: Specifically, in estimating the means, the gain in accuracy is about three-fold as measured by L_1- and L_2-distances, and in estimating the weights, it is about one standard deviation.

VA1 always converges almost as fast as VT and noticeably (by 30% in the case of unknown weights) faster than EM.

When the weights are known, an iteration of VA1 is about six times longer than that of VT and is more than twice as fast as that of EM. By total execution, VA1 is at most eight times slower than VT and is more than two and a half times faster than EM.

When the weights are unknown, VA1 is at most twice slower than VT and more than two and a half times faster than EM, per iteration. It is also about 50% slower than than VT and more than four times faster than EM in total times.

Accuracy of VA2 is consistently between those of VA1 and EM, and VA2 additionally converges faster than VA1.

Table 5. Unknown weights. Correct guess.

	VT	VA1	VA2	EM	MLE
p	0.737 ±0.030	0.699 ±0.026	0.699 ±0.026	0.699 ±0.023	0.699 ±0.024
θ_1	-2.4526 ±0.0700	-2.4987 ±0.0555	-2.4987 ±0.0557	-2.4991 ±0.0522	-2.4992 ±0.0532
θ_2	0.3537 ±0.1114	0.0058 ±0.1007	0.0060 ±0.1021	0.0038 ±0.0925	0.0039 ±0.0947
$\|\|\theta - \theta^*\|\|_1$	0.4212 ±0.1467	0.1244 ±0.0782	0.1263 ±0.0782	0.1149 ±0.0701	0.1179 ±0.0710
$\|\|\theta - \theta^*\|\|_2$	0.3626 ±0.1146	0.0978 ±0.0607	0.0994 ±0.0607	0.0907 ±0.0553	0.0931 ±0.0560
n	8.53 ±3.47	6.01 ±2.44	6.27 ±2.40	11.89 ±4.24	N/A
t	0.74 ±0.04	1.36 ±0.05	41.01 ±1.24	3.53 ±0.06	N/A
T	6.27 ±2.42	8.13 ±3.19	257.35 ±99.70	41.84 ±14.81	N/A

Table 6. Unknown weights. Correct boundary.

	VT	VA1	VA2	EM	MLE
p	0.737 ±0.029	0.702 ±0.026	0.700 ±0.026	0.700 ±0.023	0.699 ±0.024
θ_1	-2.4517 ±0.0689	-2.4941 ±0.0573	-2.4972 ±0.0556	-2.4981 ±0.0526	-2.4992 ±0.0532
θ_2	0.3549 ±0.1096	0.0148 ±0.1050	0.0087 ±0.1024	0.0059 ±0.0930	0.0039 ±0.0947
$\|\|\theta - \theta^*\|\|_1$	0.4218 ±0.1459	0.1327 ±0.0779	0.1271 ±0.0780	0.1164 ±0.0694	0.1179 ±0.0710
$\|\|\theta - \theta^*\|\|_2$	0.3637 ±0.1132	0.1043 ±0.0606	0.0999 ±0.0606	0.0919 ±0.0548	0.0931 ±0.0560
n	8.16 ±3.40	7.74 ±2.58	6.54 ±2.22	12.98 ±4.22	N/A
t	0.74 ±0.04	1.34 ±0.04	41.12 ±1.74	3.52 ±0.05	N/A
T	5.97 ±2.36	10.32 ±3.35	268.96 ±91.44	45.59 ±14.69	N/A

6 Conclusion

We have considered the problem of parameter estimation of the emission distribution in Hidden Markov Models using the two most relevant estimation principles: Viterbi Training and MLE. We have identified the sources of bias, or inconsistency, in the VT algorithm, contrasting this with the EM algorithm that computes the generally consistent MLE: Trading the EM's accuracy for the VT's ease of computations, one in particular loses the asymptotic fixed point property. Namely, VT no longer fixes the true parameter values, not even asymptotically. We have proposed to restore this property, and consequently increase accuracy of Viterbi Training. Specifically, we have proposed two types of analytic adjustments to the baseline Viterbi Training algorithm, neither requiring additional point-wise processing of the data. In particular, *our correction functions are independent of the data size.* Our first adjustment, VA1, simply restores the asymptotic fixed point property, whereas the second one, VA2, additionally ensures that asymptotically the true parameters are returned as soon as the algorithm finds the true alignment (i.e. Voronoi partition). *To our knowledge, these kinds of consistency corrections for Viterbi Training have not been proposed elsewhere in the literature.*

This paper has also shown that in the case of mixture models (a special and important case of HMM), the VA1 correction is always available, either in a closed form or via integration that can be suitably approximated. We have also explained why providing the VA1 correction in the general HMM case is more challenging, and intend to present our general theory in a follow-up paper shortly.

This work has also presented evidence that, at least in the case of mixture models (a special and important case of HMM), the actual amount of extra computations of VA1 relative to Viterbi Training can be very reasonable. For this special case, we have pro-

vided simulation studies based on 1000 large random samples which illustrate the key features of the adjusted algorithms in contrast with baseline Viterbi Training and EM. In our simulations, VA1 demonstrates a significant increase of accuracy (three-fold and one standard deviation in estimating the mixture means and weights, respectively) relative to VT. In fact, the accuracy of VA1 is already comparable to that of EM. Computation-wise, VA1 in our studies is still several factors faster than EM.

Due to the more sophisticated nature of the VA2 correction, its computationally feasible implementations require more work.

Certainly, the final decision as to which algorithm to use remains application dependent.

References

[1] L. Baum and T. Petrie. Statistical inference for probabilistic functions of finite state Markov chains. *Ann. Math. Stat.*, 37:1554–1563, 1966.

[2] J. Bilmes. A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models. Technical Report 97–021, International Computer Science Institute, Berkeley, CA, USA, 1998.

[3] G. Celeux and G. Govaert. A Classification EM algorithm for clustering and two stochastic versions. *Computationals Statistics & Data Analysis*, 14(3):315–332, 1992.

[4] P. Chou, T. Lookbaugh, and R. Gray. Entropy-Constrained Vector Quantization. *IEEE Transaction on Acoustic Speech and Signal Processing*, 37(1):31–42, 1989.

[5] J. Dias and M. Wedel. An empirical comparison of EM, SEM and MCMC performance for problematic Gaussian mixture likelihoods. *Statistics and Computing*, 14:323–332, 2004.

[6] G. Ehret, P. Reichenbach, U. Schindler, C. Horvath, S. Fritz, M. Nabholz, and P. Bucher. DNA Binding Specificity of Different STAT Proteins. *The Journal of Biological Chemistry*, 276(9):6675–6688, 2001.

[7] C. Fraley and A. Raftery. Model-Based Clustering, Discriminant Analysis, and Density Estimation. *Journal of the American Statistical Association*, 97(458):611–631, June 2002.

[8] R. Gray, T. Linder, and J. Li. A Lagrangian formulation of Zador's entropy-constrained quantization theorem. *IEEE Transactions on Information Theory*, 48(3):695–707, 2000.

[9] X. Huang, Y. Ariki, and M. Jack. *Hidden Markov models for speech recognition.* Edinburgh University Press, Edinburgh, UK, 1990.

[10] W. Jank and J. Booth. Efficiency of monte carlo em and simulated maximum likelihood in two-stage hierarchical models. *Journal of Computational and Graphical Statistics*, 12(1):214–229, 2003.

[11] F. Jelinek. *Statistical methods for speech recognition*. The MIT Press, Cambridge, MA, USA, 2001.

[12] B.-H. Juang and L.R. Rabiner. The segmential K-means algorithm for estimating parameters of hidden Markov models. *IEEE Trans. Acoustics, Speech, Signal Proc.*, 38(9):1639–1641, 1990.

[13] R. Kolde. Estimating of mixture density parameters with adjusted Viterbi training (in Estonian). Bachelor Theses, Tartu University, 2005.

[14] A. Koloydenko and J. Lember. Matlab code for simulation studies of Adjusted Viterbi Training. Online at http://euridice.tue.nl/∼akoloide/VA, 2003.

[15] J. Lember and A. Koloydenko. The Theory of Adjusted Viterbi Training. Technical Report 2003:019, Eurandom, Eindhoven, The Netherlands, 2004. in preparation.

[16] J. Li, R. Gray, and R. Olshen. Multiresolution image classification by hierarchical modeling with two dimensional hidden Markov models. *IEEE Transactions on Information Theory*, 46(5):1826–1841, 2000.

[17] C. Lin, C. Chen, and W. Wu. Fuzzy clustering algorithm for latent class model. *Statistics and Computing*, 14:299–310, 2004.

[18] G. McLachlan and D. Peel. *Finite Mixture Models*. Probability and Statistics. Wiley, 2000.

[19] H. Ney, V. Steinbiss, R. Haeb-Umbach, B. Tran, and U. Essen. An overview of the Philips research system for large vocabulary continuous speech recognition. *International Journal of Pattern Recognition and Artificial Intelligence*, 8(1):33–70, 1994.

[20] F. Och and H. Ney. Improved Statistical Alignment Models. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, A digital archive of research papers in computational linguistics: http://acl.ldc.upenn.edu/P/P00/P00-1056.pdf, 2000.

[21] U. Ohler, H. Niemann, G. Liao, and G. Rubin. Joint modeling of DNA sequence and physical properties to improve eukaryotic promoter recognition. *Bioinformatics*, 17(Suppl. 1):S199–S206, 2001.

[22] D. Pollard. Strong consistency of k-means clustering. *Ann. Statist.*, 9(1):135–140, 1981.

[23] L. Rabiner. A tutorial on Hidden Markov models and selected applications in speech recognition. *Proc. IEEE*, 77(2):257–286, 1989.

[24] L. Rabiner and B. Juang. *Fundamentals of speech recognition*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993.

[25] L. Rabiner, J. Wilpon, and B. Juang. A segmential K-means training procedure for connected word recognition. *AT&T Tech. J.*, 64(3):21–40, 1986.

[26] B. D. Ripley. *Stochastic simulation*. John Wiley & Sons, Inc., 1987.

[27] C. Robert and G. Casella. *Monte Carlo Statistical Methods*. Springer, 1999.

[28] M. Sabine and R. Gray. Global convergence and empirical consistency of the Generalized Lloyd algorithm. *IEEE Transaction on Information Theory*, 32(2):148–155, 1986.

[29] I. M. Sobol. *Chislennye metody Monte-Karlo*. "Nauka", USSR, 1973.

[30] V. Steinbiss, H. Ney, X. Aubert, S. Besling, C. Dugast, U. Essen, D. Geller, R. Haeb-Umbach, R. Kneser, H. Meyer, M. Oerder, and B. Tran. The Philips research system for continuous-speech recognition. *Philips Journal of Research*, 49:317–352, 1995.

[31] N. Ström, L. Hetherington, T. Hazen, E. Sandness, and J. Glass. Acoustic Modeling Improvements in a Segment-Based Speech Recognizer. In *Proc. IEEE ASRU Workshop Keystone, CO, USA*, MIT Comp. Sci. and AI Lab., Spoken Language Systems http://www.sls.lcs.mit.edu/sls/publications/1999/asru99-strom.pdf, 1999.

[32] *Getting Started with Matlab*. Natick, MA, USA, 2004. http://www.mathworks.com/access/helpdesk/help/helpdesk.shtml.

[33] G.C.G. Wei and M.A. Tanner. A Monte Carlo Implementation of the EM Algorithm and the Poor Man's Data Augmentation Algorithms. *Journal of the American Statistical Association*, (85):699–704, 1990.

[34] S. Young. The Hidden Vector State Language Model. Technical Report CUED/F-INFENG/TR.467, Cambridge University, Cambridge, UK, 2003.