Bayesian methods for HMM speech recognition with limited training data

Darryl W Purnell, Elizabeth C Botha

Department of Electrical, Electronic and Computer Engineering
University of Pretoria, 0002, South Africa.
botha@ee.up.ac.za

Abstract

This paper presents a Bayesian approach to learning for HMMs in speech recognition. The implementation of Bayesian learning for HMMs in speech recognition is discussed, including the requirement of maintaining the original HMM constraints, choice of prior and utterance recognition. This work shows that the Bayesian learning approach can be successfully applied to complex models when the amount of training data is small.

1. Introduction

Bayesian methods can be used for the inference of parameter values in a model given the data. We begin here by introducing the Bayesian learning framework and discuss past work both within the field of speech recognition and neural networks. This work was inspired by the work of Neal [1], who proposed a Markov chain Monte Carlo based Bayesian learning procedure for neural networks. The remainder of this section will summarize the relevant Bayesian theory used in this article. For a more complete introductory text on Bayesian statistics, the reader is referred to Box and Tiao [2] and DeGroot [3].

1.1. Bayes' theorem

The speech data is given as a vector $\mathbf{O} = (\mathbf{o}_1, \ldots, \mathbf{o}_n)$ of n observations (each $\mathbf{o}_i$ is itself a vector of dimension D), with probability distribution $P(\mathbf{O}|\theta)$, which depends on the k parameters $\theta' = (\theta_1, \ldots, \theta_k)$. The parameter vector θ has the probability distribution $P(\theta)$. Given the observed data, the conditional distribution of θ is

$$P(\theta|\mathbf{O}) = \frac{P(\mathbf{O}|\theta)P(\theta)}{P(\mathbf{O})}. \quad (1)$$

The denominator in Eq. (1), $P(\mathbf{O})$, is a normalizing factor, which ensures that the integral of $P(\theta|\mathbf{O})$ is equal to one.

Equation (1) is referred to as Bayes' theorem. The distribution $P(\theta)$, is called the *prior* distribution and expresses what is known about the model parameters before any data is observed. The *posterior* distribution $P(\theta|\mathbf{O})$ tells us what is known about the model parameters, given that data has been observed.

1.2. Bayesian learning and prediction

The result of Bayesian learning is a probability distribution, the posterior $P(\theta|\mathbf{O})$, which expresses our beliefs of how likely individual parameters values are. This is crucial, as it allows learning to be performed using probability theory.

In a Bayesian approach to HMM parameter estimation and recognition, the objective is to find a predictive distribution for an unknown utterance, given the associated observations, as well as the training observations. Let the observations for the ith example be written as $\mathbf{O}_i$, with n training examples $(\mathbf{O}_1, \ldots, \mathbf{O}_n)$.

In a Bayesian framework, when we wish to classify a unknown input, we need to calculate the following probability,

$$P(\mathbf{O}_{unknown}|\mathbf{O}_1^{(i)}, \ldots, \mathbf{O}_n^{(i)}) = \int P(\mathbf{O}_{unknown}|\theta)P(\theta|\mathbf{O}_1^{(i)}, \ldots, \mathbf{O}_n^{(i)})d\theta, \quad (2)$$

where i is the class and $\mathbf{O}_{unknown}$ is the unknown observation. The classifier decision is the class resulting in the highest value of Eq. (2), i.e.

$$C(\mathbf{O}_{unknown}) = \underset{j}{argmax}\ P(\mathbf{O}_{unknown}|\mathbf{O}_1^{(j)}, \ldots, \mathbf{O}_n^{(j)}), \quad (3)$$

where $C(\mathbf{O}_{unknown})$ is the classifiers decision for the unknown observation.

Unfortunately, due to the nature of the incomplete data problem caused by the underlying hidden processes of an HMM, the evaluation of Eq. (2) is non-trivial. Gaussian approximations of the posterior have been used to make the evaluation of Eq. (2) simpler [4, 5].

We can, however, use Monte Carlo methods to obtain a better approximation of Eq. (2). Monte Carlo methods make no assumption concerning the form of the distribution, as done in the above approximations. In theory, they can approximate Eq. (2) for complex distributions with multiple modes, as well as distributions for which the dominant contribution of the integral results from areas in parameter space which are not near a mode. Markov chain Monte Carlo methods will therefore be used in this implementation.

As mentioned, we want to evaluate Eq. (2), which is the expectation of the function $P(\mathbf{O}_{unknown}|\theta)$ with respect to the posterior distribution $P(\theta|\mathbf{O}_1, \ldots, \mathbf{O}_n)$. Such expectations can be estimated using Monte Carlo methods, by summing $P(\mathbf{O}_{unknown}|\theta_j)$ using N samples generated from the posterior distribution ($j = \{1, \ldots, N\}$), i.e.

$$P(\mathbf{O}_{unknown}|\mathbf{O}_1, \ldots, \mathbf{O}_n) \approx \sum_{j=1}^{N} P(\mathbf{O}_{unknown}|\theta_j), \quad (4)$$

where the samples $\theta_1, \ldots, \theta_N$ are generated by a process such that the distribution of the samples is that of the posterior.

1.3. Hierarchical models

The Hidden Markov models whose parameters we have to sample are complex and their parameters are numerous. It is therefore useful to specify the joint distribution of some of these parameters in terms of a common hyperparameter which has a prior distribution of its own. This is known as a *hierarchical model*.

The prior distribution of the parameters $P(\theta)$ can then be written in terms of the hyperparameters as follows (assuming independence):

$$P(\theta) = \int P(\gamma) \prod_{i=1}^{D} P(\theta_i|\gamma) d\gamma \tag{5}$$

where $P(\gamma)$ is the prior distribution of the hyperparameter γ and $P(\theta_i|\gamma)$ is the prior distribution of the parameter θ_i given the hyperparameter γ.

A hierarchical model, if well formulated, can be considerably more intelligible than using a direct prior distribution. We can also in this way, incorporate vague heuristic information into the prior, as will be done in Section 2.

2. Implementation of Bayesian HMM learning

2.1. HMM constraints

The Gaussian mixture density function and the HMM place constraints on some of the parameters. As we will use a second order technique for sampling, namely the leapfrog algorithm, we cannot use transformations on the parameters to maintain the constraints, as is done in some gradient-based methods [6].

A constrained version of the sampling algorithm was therefore implemented, in which the HMM constraints are applied. It is too complex to implement a constrained algorithm which maintains the constraints for the transition probabilities and mixture weights (i.e. $\sum_j a_{ij} = 1$ and $\sum_i c_i = 1$), without using transformations. These parameters are therefore treated as being fixed and only the posterior distribution of the means and variances of the HMM Gaussian mixtures are therefore sampled.

2.2. HMM prior and hyperparameters

The prior distribution for the HMM Gaussian mixture mean and variances was chosen to be a normal-Wishart distribution. The normal-Wishart prior is as follows,

$$\begin{aligned} &g_{gaussian}(r_{jk}, \mu_{jk}|n_{jk}, \nu_{jk}, m_{jk}, \tau_{jk}) = \\ &(2\pi)^{-\frac{D}{2}} |\nu_{jk} r_{jk}|^{\frac{1}{2}} e^{-\frac{1}{2}\nu_{jk}(\mu_{jk}-m_{jk})^T r_{jk}(\mu_{jk}-m_{jk})} \\ &c|\tau_{jk}|^{-\frac{n_{jk}}{2}} |r_{jk}|^{\frac{(n_{jk}-D-1)}{2}} e^{-\frac{1}{2}tr(r_{jk}\tau_{jk}^{-1})}, \end{aligned} \tag{6}$$

where $(n_{jk}, \nu_{jk}, m_{jk}, \tau_{jk})$ are the prior distribution parameters associated with mixture k of state j. The value c is a normalizing constant.

It can be easily shown that choosing the parameters m_{jk} and τ_{jk} in Eq. (6) to be

$$m = \mu' \tag{7}$$

$$\tau = (n - D)\Sigma', \tag{8}$$

results in the mode of the prior being at the point $\mu = \mu'$ and $\Sigma = \Sigma'$. The parameters n_{jk} and ν_{jk} determine the degree to which the prior is peaked about its mode.

Although not necessary, one can reduce the number of variables in the prior by expressing the parameters n_{jk} and ν_{jk} in terms of a common parameter C_{jk},

$$n_{jk} = C_{jk} + D \tag{9}$$

$$\nu_{jk} = C_{jk} + 1, \tag{10}$$

with $C_{jk} > 0$.

The parameters C_{jk}, m_{jk} and τ_{jk} are now given their own prior distributions. These distributions and their parameters must be chosen in a meaningful way, such that the hyperparameters contain *a-priori* knowledge (albeit vague). For reasons beyond the scope of this paper (but which can be found in [7]), the prior parameters of a given state in the HMM have been given common distributions and hyperparameters.

The prior mean m_{jk} of state j is given a normal distribution with mean ω_j and standard deviation ς_j, i.e.

$$P(m_{jk}) = \mathcal{N}(\omega_j, \varsigma_j), \quad k = 1, \ldots, M, \tag{11}$$

where M is the number of mixtures.

Given that we wish to keep the hyperparameter distributions as intuitive as possible, we have chosen to represent the prior mode Σ' in Eq. (8) with a gamma distribution [3] with parameters ϕ_j and ψ_j. The distribution of the prior parameter τ_{jk} of state j can, using Eq. (8), therefore be written as

$$P(\tau_{jk}) = (n_{jk} - D)\mathcal{G}(\phi_j, \psi_j), \quad k = 1, \ldots, M. \tag{12}$$

Note that from Eq. (7) the prior mean m_{jk} is the mode of the prior with respect to the mean parameters. The distribution for the prior parameter C_{jk} is also chosen to be a gamma distribution [3] with parameters C_m and C_v, i.e.

$$P(C_{jk}^{(i)}) = \mathcal{G}(C_m, C_v), \quad k = 1, \ldots, M. \tag{13}$$

for every state $j = 1, \ldots, N$ in every HMM $i = 1, \ldots, N_h$.

2.2.1. Determining hyperparameters

The ML estimate of the parameters, which is used as the starting point for Bayesian learning, is used to estimate the hyperparameters ω_j, ς_j (prior mean m_{jk}) and ϕ_j, ψ_j (prior parameter τ_{jk}). The sample mean and variance of the mixture means for state j of the ML estimate are reasonable estimates for the parameters ϕ_j and ψ_j. Likewise, the sample mean and variance of the mixture variances for state j of the ML estimate are reasonable estimates for the mean and variance of distribution $\mathcal{G}(\phi_j, \psi_j)$.

The hyperparameters C_m and C_v are determined by the user and express our trust in the ML estimate. Large values of C_m will result in prior distributions which are peaked around the mode of the posterior, while small values of C_m will result in a relatively non-informative prior distribution.

2.3. Refreshing the hyperparameters

As a result of using hyperparameters, we have to evaluate Eq. (5). This is accomplished by Gibbs sampling [1] of the hyperparameters after each transition (i.e. before leapfrog integration) of the stochastic dynamics or hybrid Monte Carlo algorithm, which results in the numerical integration of Eq. (5). The hyperparameters are easily sampled, as they are independent of other parameters and hyperparameters, and are generated by known distribution forms (normal and Gamma). Standard techniques for generating normal and Gamma distributed variates are used.

2.4. Implementation of stochastic dynamics method

We can write the posterior in the form of a "potential energy" function, giving:

$$E(\theta) = -log[P(\theta)] - \sum_{i=1}^{n} log[P(\mathbf{O}_i|\theta)]. \qquad (14)$$

The implementation of the stochastic dynamics method (SDM) [1, 8] is as follows:

1. Refresh momenta (defined in SDM) and prior parameters (m_{jk}, τ_{jk} and C_{jk}) using Gibbs sampling [1].
2. Starting with the current set of parameters and momenta, perform L leapfrog steps to generate a new set of HMM parameters (r_{jk} and μ_{jk}). Here we require $\frac{\partial E(t)}{\partial \theta}$.
3. Keep current configuration, repeat steps 1 through 3 until N samples of the parameters are generated.

A segmental approach is used, in which we use the best state sequence ($\max_{q} P(\mathbf{O}, \mathbf{q}|\theta)$), as opposed to the sum of all possible sequences ($P(\mathbf{O}|\theta) = \sum_{\text{all } \mathbf{q}} P(\mathbf{O}, \mathbf{q}|\theta)$).

2.5. Recognition

Implementation of the decision rule (Eq. (3)) is somewhat straightforward when applied to discrete (or label-based) phoneme or word classification: use Eq. (4) to numerically determine $P_j(\mathbf{O}_{unknown}|\mathbf{O}_1, \ldots, \mathbf{O}_n)$ for each class (phoneme or word).

The implementation thereof for continuous speech recognition is not a trivial task. Here, a class is a string of phonemes or words and there can be many possible combinations. The direct implementation of Eq. (4), although relatively simple to implement, would not be viable in terms of computational complexity in this case.

To overcome this, we have formulated a simple, yet reasonable approximation [7]. This approximation has the advantage that it is only approximately N_m times slower than a standard HMM recognizer, where N_m is the generated sample size.

3. Experiments

The goal of this section is to experimentally determine the utility of the Bayesian learning approach described in this paper. The hypothesis that will be tested here, is that "Bayesian learning will improve performance markedly in situations where little data is available for training purposes". In situations where sufficient training data is available, the posterior will be peaked sharply about the MAP point and there will therefore be little advantage in using Bayesian learning under such conditions. Two datasets, SUNSpeech and TIMIT, are used.

Unless otherwise stated, the stochastic dynamics method was run for 100 iterations, i.e. 100 samples were generated. The last N_m samples were used for recognition tasks. For example, a recognition experiment using $N_m = 30$ will therefore use the last 30 samples, i.e. samples 71 through to 100.

Bayesian learning is not a deterministic process and this causes a certain variance in performance of the resultant systems. Each experiment was repeated 10 times, so as to provide an indication of the variance in accuracy that results. Error bars are given for each result, indicating the minimum, mean and maximum accuracy for a given configuration.

The phoneme recognition accuracy of the system is reported, where accuracy is defined as:

$$Accuracy = \frac{Phones - Subs - Dels - Ins}{Phones}, \qquad (15)$$

where $Phones$ is the number of phones in the correct transcription, $Subs$ the number of substitutions, $Dels$ the number of deletions and Ins the number of insertions. Error rates reported are $100\% - Accuracy$. The speech signal is blocked into frames of length 16ms, with overlap of 6ms. Thirteen Mel-frequency cepstral coefficients (MFCCs), along with their first and second order differentials are used. Unless otherwise stated, each phone is represented by a simple left-to-right, 3 state, 5 mixture HMM.

3.1. SUNSpeech

The SUNSpeech database [9] was compiled by the Department of Electrical and Electronic Engineering of the University of Stellenbosch to contain phonetically labelled speech in both English and Afrikaans. Sixty sentences comprising four sentence sets were chosen to exhibit the diversity of phonemes in the two languages. A total of 59 phonetic categories were used to segment both the Afrikaans and the English speech. In this work, only the Afrikaans subset of the SUNSpeech database was used. Table 1 summarizes the SUNSpeech Afrikaans training and testing sets as used in our experiments.

Table 1: Details of SUNSpeech training and testing sets used

Description	Label	Speakers	Duration (minutes)
Training set	A	39	22
Training subset	A_S	8	5.5
SI test set		15	13

Table 2 presents the baseline Afrikaans test set accuracy for the two training sets available, namely the full Afrikaans training set A and the reduced training set A_S. Note that for the small training set A_S, the less complex 3 state, 5 mixture HMM performs best.

Table 3 gives the results obtained for the SUNSpeech dataset when using the Bayesian learning algorithm described in this paper. Note that here, the 10 mixture model always performs best. This is as opposed to the ML estimate, where the 10 mixture system performed worse that the 5 mixture system when the small training set is used (Table 2).

Whether one can justify using the 90 sample system, which is 9 times slower than the 10 sample system and approximately 90 times slower than that of the equivalent standard HMM system, is currently unlikely.

Table 2: Baseline system accuracy for SUNSpeech Afrikaans test set

Training set	Mixtures	
	5	10
Afrikaans train (A)	48.6	**51.5**
Afrikaans adapt (A_S)	**42.5**	41.2

Table 3: Summary of the results obtained using Bayesian learning with the SUNSpeech dataset, for $L = 100$, $\epsilon = 0.001$.

Training set	Configuration			Sample size		
	Mix.	C_m	C_v	10	50	90
A_S	5	15	4	45.2	45.8	46.1
A_S	10	5	4	**46.5**	**46.9**	**47.0**
A	5	10	4	50.5	50.3	50.3
A	10	10	4	**53.7**	**54.0**	**54.1**

3.2. TIMIT

The TIMIT database was also used for evaluation purposes. So as to simulate a scenario where little training data is available, a small gender independent training subset of the recommended TIMIT training set has been used. This small training set was created by randomly selecting two male and two female speakers from each of the eight dialect regions in the TIMIT dataset. The duration of the small TIMIT training set is 16.1 minutes. Following convention, we recognize the standard 39-phone set.

The full TIMIT testing set has been used for all experiments presented in this section. The baseline system test set performance for the 3 state, 5 mixture HMM when using the small TIMIT training set is 52.6%.

Figure 1 shows the effect of the sample size N_m on the performance of the Bayesian learning system. The performance for a sample size of 1 ($N_m = 1$) is in general better than that of the Baseline ML system. This improvement is primarily due to the regularization effect of the chosen prior and associated hyperparameters. By far the greatest improvements resulted when using 10 or fewer samples. Using 10 samples results in an average system accuracy of 54.1%.

4. Conclusion

This article introduced an implementation of Bayesian learning and its usage within a continuous speech recognition framework. Markov chain Monte Carlo methods are used to sample the posterior distribution. Implementation specifics such as maintaining the HMM constraints, and issues regarding the prior distributions were also discussed.

Section 3 experimentally evaluated the Bayesian learning procedure introduced in this article. The SUNSpeech and TIMIT databases were used for this purpose. Significant improvements in system accuracy were attained for the scenarios created.

The results show that there is much promise in the usage of the proposed Bayesian learning procedure. Although considerably more computationally expensive, we believe, however, that the improved performance warrants the additional computational expense.

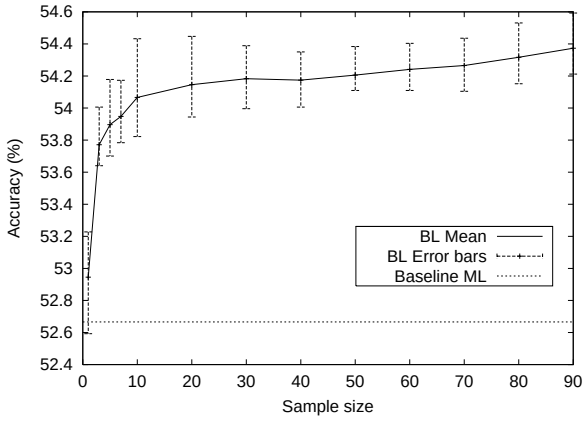

Figure 1: Performance of the Bayesian system versus the size of the sample used to numerically integrate with respect to the posterior.

5. Acknowledgments

The authors would like to thank the Mellon Foundation and the National Research Foundation of South Africa for their support of this work. The authors would also like to thank the University of Stellenbosch for the use of their SUNSpeech database.

6. References

[1] R.M. Neal, *Bayesian Learning for Neural Networks*, Springer, New York, 1996.

[2] G.E.P. Box and G.C. Tiao, *Bayesian Inference in Statistical Analysis*, John Wiley and Sons, New York, 1973,1992.

[3] M.H. DeGroot, *Optimal Statistical Decisions*, McGraw-Hill, New York, 1970.

[4] David J.C. Mackay, *Bayesian Methods for Adaptive Models*, Ph.D. thesis, California Institute of Technology, Pasadena, California, December 1991.

[5] Q. Huo, H. Jiang, and C. H. Lee, "A Bayesian predictive classification approach to robust speech recognition," in *Proc. ICASSP '97*, Munich, Germany, Apr. 1997, pp. 1547 – 1550.

[6] W. Chou, B.-H. Juang, and C.-H. Lee, "Segmental GPD training of HMM based speech recognizer," in *Proc. ICASSP '92*, San Francisco, 1992, IEEE, pp. 473–476.

[7] D.W. Purnell and E.C. Botha, "Bayesian methods for sparse training data HMM-based speech recognition," *Pattern Recognition*, submitted.

[8] H.C. Andersen, "Molecular dynamics simulations at constant pressure and/or temperature," *Journal of Chemical Physics*, vol. 72, pp. 2384–2393, 1980.

[9] T. Waardenburg, J.A. du Preez, and M.W. Coetzer, "The automatic recognition of stop consonants using hidden Markov models," in *Proc. ICASSP '92*, San Francisco, CA, Mar. 1992.