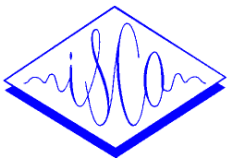

6th International Conference on Spoken Language Processing (ICSLP 2000)
Beijing, China
October 16-20, 2000

EFFICIENT MIXED-ORDER HIDDEN MARKOV MODEL INFERENCE

Ludwig Schwardt, Johan du Preez

Department of Electrical and Electronic Engineering,
University of Stellenbosch, Private Bag X1, Matieland 7602, South Africa
E-mail: schwardt@ing.sun.ac.za

ABSTRACT

Recent studies have shown that high-order hidden Markov models (HMMs) are feasible and useful for spoken language processing. This paper extends the fixed-order versions to ergodic mixed-order HMMs, which allow the modelling of variable-length contexts with significantly less parameters. A novel training procedure automatically infers the number of states and the topology of the HMM from the training set, based on information-theoretic criteria. This is done by incorporating only high-order contexts with sufficient support in the data. The mixed-order training algorithm is faster than fixed-order methods, with similar classification performance in language identification tasks.

1. INTRODUCTION

Hidden Markov models (HMMs) [1] find widespread use in the speech community. Its success can be ascribed to its compact representation of temporal dependencies and the existence of efficient algorithms for training and scoring data.

Despite a recent spate of new HMM variants and extensions [2], the most common HMM in speech processing is still a hand-crafted first-order model with a structure dictated by domain knowledge. Natural speech, on the other hand, is known to contain significant high-order contextual information, but initial approaches to exploit it proved intractable [1].

Du Preez [3] recently showed that high-order HMMs can be used efficiently by converting them to equivalent first-order HMMs that represent context by states rather than links. This led to the FIT algorithm that incrementally expands the HMM order during Expectation-Maximisation (EM) training to alleviate problems with local minima. It can only train fixed-order HMMs, however, which leads to an over-abundance of model parameters if the true source is mixed-order in nature. This can lead to poorer optimisation performance.

This study introduces a novel approach to ergodic HMM training that results in inherently mixed-order models with variable-length contexts and significantly less parameters than those trained by FIT. It automatically infers the topology of the HMM from the training set, based on information-theoretic criteria. This is done by incorporating only high-order contexts with sufficient support in the data. The resulting training algorithm is faster than FIT, with similar classification performance on previously unseen test data.

The proposed method uses the fact that the output pdfs shared between the states of an HMM can be viewed as ”soft” symbols, while the states themselves represent the contexts in which these symbols occur. The method allows mixed-order dependencies between the output pdfs by replacing the traditional transition probability estimation in the M-step of Viterbi reestimation training with a flexible Markov structure inference procedure. The reestimation of the output pdfs is left unchanged.

The structure inference starts off with the most likely symbol (or output pdf) sequence obtained by the Viterbi algorithm in the previous E-step of the EM algorithm. The soft symbol contexts are explicitly modelled by an alternative representation of an ergodic Markov chain, the prediction suffix tree (PST) of Ron *et al* [4].

This context tree is constructed recursively and efficiently from the current pdf index sequence, using the smallest encoding context tree (SECT) algorithm. This minimises the combined description length of the given symbol sequence and the sufficient statistics of the context occurrences, as stored in the PST. This is done in the spirit of the minimum message length (MML) [5] and minimum description length (MDL) [6] approaches. It prevents the tree from expanding to contexts that are judged not well supported by the data, and thereby leads to a sparse mixed-order representation.

Experimental results on spoken language identification (LID) show that the mixed-order algorithm results in significantly smaller language models that train faster, with similar classification performance to fixed-order models trained with the FIT algorithm. More details on this application can be found in a companion paper [7].

The next section gives an overview of the Markov structure inference algorithm. Section 3 extends this algorithm to HMMs. LID results are presented in Section 4, with conclusions and further improvements discussed in Section 5.

2. INFERRING MARKOV CHAINS

A Markov chain (MC) models sequences of symbols by describing their generation in terms of a probabilistic finite-state machine. It consists of a collection of states and a set of transitions between pairs of states. Each transition has an associated probability a_{ij} which represents the probability that the next state will be j given that the current state is i. Each state also has an associated output symbol. The Markov chain generates strings by jumping from state to state according to the transition probabilities and emitting the symbol associated with the current state at each time instant.

The general MC representation is well suited to the generation and scoring of symbol sequences, but is difficult to infer from

training data because of its highly variant structure. Certain Markov chain subclasses, however, are easier to infer and can still provide a rich representation, such as the probabilistic suffix automata (PSA) of [4] (named variable length Markov models in [2] and closely related to the variable length Markov chains (VLMC) of [8]). Models in this subclass have states modelling well-defined symbol substrings or *contexts* of variable length, and have a natural representation as *context trees* [9]. This tree structure leads to efficient inference algorithms. The models in the PSA subclass are ergodic, which implies that every state can be reached from any other state. This represents stationary string constraints well, making it suitable to model global language structure.

The smallest encoding context tree (SECT) algorithm uses the tree structure associated with PSAs, known as the prediction suffix tree [4], and grows the tree to represent the mixed-order contexts found in the training data. The growth is regulated by the minimum description length (MDL) principle [6], which searches for a compact model that still describes the training data well. The following sections provide a short overview of these two components. More details of the SECT (or MMM) algorithm can be found in [10].

2.1. Prediction Suffix Trees

Prediction suffix trees (PSTs) were first studied in the field of universal data compression [9] and subsequently extended to Markov model inference [4]. It consists of a basic suffix tree with probability distributions on each node. A similar approach can be found in [8].

A context (suffix) tree is a data structure that stores symbol strings s from an alphabet of K symbols as nodes in a tree graph. Each node can have up to K edges emanating from it, and each edge is uniquely labelled by a single symbol. The string s^n associated with a specific node n is obtained by concatenating the edge labels as the tree is traversed from that node upwards to the root. The root node is associated with the empty string e.

The PST extends the context tree by placing a *next symbol probability distribution*

$$a(k \,|\, s^n) = P\left(r_t = k \,\middle|\, \text{suffix}\left[r_1^{t-1}\right] = s^n\right) \qquad (1)$$

at each node n. This gives the probability that a specific symbol k follows the context substring s^n associated with the node n in a string $r_1^\infty = r_1 r_2 \cdots r_{t-1} r_t \cdots$ generated by the model. This forms the predictive component of the PST, without which it would merely provide storage for the context strings.

A PST can generate or score symbol strings in a similar fashion to Markov chains. The probability that the PST M generates the string $r_1^T = r_1 r_2 \ldots r_T$ of length T is given by

$$P(r_1^T \,|\, M) = \prod_{t=1}^{T} a\left(r_t \,|\, s_{t-1}\right), \qquad (2)$$

where $s_0 = e$, and for $1 \le t \le T-1$, s_t is the string labelling the deepest node reached by traversing M from the root along the labels $r_t r_{t-1} \ldots r_1$. The string s_t therefore represents the longest significant context or history influencing the probability of the next symbol r_{t+1}. Outside of this context, the next symbol is statistically independent of all other symbols, which allows $P(r_1^T \,|\, M)$ to be factorised in a product form. This conditional independence property is shared with Markov chains.

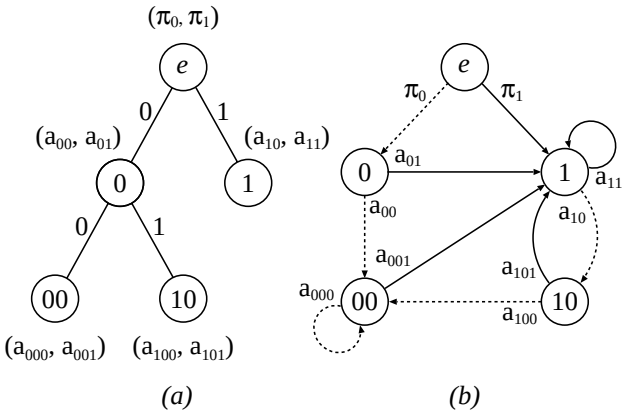

Figure 1: (a) A mixed-order prediction suffix tree on the alphabet $\{0, 1\}$. The prediction probabilities of the symbols '0' and '1', respectively, are shown in parentheses next to each node. The model has a maximum order of two. (b) An equivalent (ergodic) Markov chain with five states. The states are labelled by their associated contexts. Bold edges indicate transitions emitting the symbol '1', while dashed edges correspond to the symbol '0' being output. The transition probabilities are shown on the edges and follow standard Markov model notation.

An example of a PST on the binary alphabet $\{0, 1\}$ is shown in Figure 1, together with its equivalent Markov chain. This PST generates the string 10010 with probability $\pi_1 \times a_{10} \times a_{100} \times a_{001} \times a_{10}$, where the contexts used for the prediction are $s_0 = e$, $s_1 = 1$, $s_2 = 10$, $s_3 = 00$ and $s_4 = 1$.

The main strength of the PST is its amenability to inference from training strings, because of its direct focus on context strings and its tree structure. When it is necessary to score or generate symbol sequences, it can be converted to an equivalent Markov chain [4]. The leaf nodes of the PST generally become the persistent states of the MC, while the internal nodes become transient states.

2.2. Encoding The Tree

A typical PST inference algorithm starts out with a training string and builds up a context tree in a top-down fashion. Each node stores the counts of symbols following the associated context string in the data. Longer contexts will necessarily be sparsely present in a finite string, providing a natural way to stop the growth of the tree. Such a tree would be too specialised on the training set statistics.

The SECT algorithm uses a more principled approach based on an information-theoretic version of Occam's Razor, the minimum description length (MDL) principle [6]. For a certain data set and a set of models $M = \{m_1, m_2, \ldots\}$ that try to explain the data, MDL selects the model that results in the shortest combined description of the model $H(m_i)$ and the data in terms of the model $H(data \,|\, m_i)$, as measured in bits. The encoding process assigns a code length of $H(x) = -\log_2 P(x)$ bits to an event x with probability $P(x)$, in accordance with Shannon's informa-

tion theory. MDL therefore has a direct correspondence to the Bayesian maximum a posteriori (MAP) approach, as the MAP model m_{MAP} also minimises the description length:

$$\begin{aligned} m_{MAP} &= \arg\max_i P(m_i \mid data) \\ &= \arg\min_i [-\log_2(P(m_i)P(data \mid m_i))] \\ &= \arg\min_i [H(m_i) + H(data \mid m_i)] \end{aligned} \quad (3)$$

The data string is encoded by $-\log_2 P(data \mid \mathrm{PST})$ bits. The description length of the PST is determined heuristically by accumulating the number of bits required to store the next symbol counts and node structure. The bit calculations are simplified by the fact that all statistics are in the form of integers. The bit allocation in effect constructs a prior on the class of PSTs. The prior is kept conservative by requiring that enough information is stored to enable the decoding of the training sequence from the bit strings of the PST and encoded data (although the purpose of the inference is classification and not compression).

The conditional independence between contexts in different subtrees in the PST allows the MDL principle to be applied locally in a recursive tree building algorithm. A node will only be added if it results in a reduction of the total description length. This will be the case if the next symbol distribution $p = a(\cdot \mid rs^n)$ of the child context differs substantially from that of its parent context, $q = a(\cdot \mid s^n)$, as measured by the Kullback-Leibler divergence $D(p \,\|\, q)$ [4]. The tree is less likely to expand as the support for a specific context decreases.

The end result is a highly efficient procedure that infers a PST from a symbol string and converts it to an equivalent (ergodic) Markov chain.

3. EXTENSION TO HMMS

The hidden Markov model (HMM) is an extension of the basic Markov chain that allows the symbols associated with each state to be "soft". Each symbol has an associated probability density function (pdf) $f(x \mid k)$ defined on an observation or feature space, typically d-dimensional Euclidean space $\mathbb{R}^d$ for continuous HMMs or a set of vector codewords for discrete HMMs. The "softness" results when the symbol pdfs overlap, causing a many-to-one mapping between symbols and features and effectively hiding the state and symbol labels associated with each feature.

The standard HMM training methods are based on the expectation-maximisation (EM) algorithm [11], a two-step iterative procedure well suited to so-called incomplete problems. During the E-step, the hidden information (i.e. the state labels) is estimated from the data and the current model estimate, and in the M-step this information is used to obtain new estimates of the model parameters. These two steps are alternated repeatedly, and the model estimate is guaranteed to monotonically converge to a local optimum.

The Viterbi reestimation procedure [1] is an EM technique which uses the Viterbi algorithm to obtain the most likely state sequence as the E-step. During the M-step, counts of the state transitions in this sequence are used to reestimate the transition probabilities of the HMM. The HMM topology is specified a priori by the user and is unchanged by the reestimation, except for the links that are discarded when their transition probabilities become zero.

It is straightforward to incorporate the SECT algorithm in Viterbi HMM training. The most likely symbol sequence is first obtained from the state sequence by replacing each state with its associated symbol (or output pdf index). The SECT algorithm then derives a new ergodic Markov chain structure from this sequence, which replaces the current HMM states and links. The new HMM topology represents the contexts of the soft symbols as it was observed in the training data during the previous iteration of the EM algorithm. The reestimation of the output pdfs and transition probabilities are unaffected by the structural inference.

This leads to an ergodic HMM training algorithm that automatically determines the link structure that is well supported by the training data. The method also automatically incorporates state tying of both output pdfs and transition probabilities where appropriate.

The transition probabilities are independently reestimated from the next symbol counts stored in the PST. The current version uses the standard maximum likelihood estimates (relative frequencies), but any other suitable estimator can be substituted, such as Dirichlet priors, Good-Turing smoothed probabilities [12] and Brand's entropic prior [13]. The latter estimate looks especially promising to enhance the search for HMM structure.

The new M-step can be interpreted as maximum a posteriori (MAP) estimation instead of the usual maximum likelihood version. The SECT inference maximises the posterior probability of the HMM structure in the process of minimising the total description length (Eq. 3). The model prior is determined by the PST description length.

As in the case of high-order HMMs, a direct attempt to find the full model structure can lead to poor optimisation performance, due to excessive degrees of freedom. A FIT-like approach trains successively higher-order HMMs by initialising the next higher-order model with a trained HMM of the previous order. The Markov order is limited by constraining the maximum depth of the context tree during inference.

4. PERFORMANCE

The performance of the mixed-order HMM training procedure was investigated on a spoken language identification (LID) task. More details on the LID experiments can be found in a companion paper [7].

The experiments were performed on the CALLFRIEND speech corpus, a large untranscribed database of conversational telephone speech. The database size promotes the use of longer contexts. Classification was done between all twelve languages in the corpus. The models were trained on the full training and development set (about 15 hours of speech per language) and evaluated on the 30s CALLFRIEND segments of the official NIST 1996 evaluation set.

The results are shown in Table 1. The FIT-trained mixed-order models with SECT inference (MFn) are smaller and train faster than their fixed-order counterparts (Fn), for similar classification performance. This is because the fixed-order FIT training fully expands the HMM to the next order, thereby increasing the link count by a factor M, before discarding most of the links during subsequent training. The FIT-trained MF3 also outperformed the directly trained MFx, due to the ability of incremental training to

Model	Training time (hr/language)	Model size (links)	Test set error (%)
X1	0.5	585	48.2
F2	1.6	2640	37.8
MF2	**1.4**	**2313**	**38.5**
F3	5.8	9074	33.9
MF3	**2.0**	**4313**	**35.0**
MFx	**3.2**	**11280**	**38.1**

Table 1: LID results obtained on the CALLFRIEND corpus. X1 is a standard first-order model. The Fn models were trained incrementally via FIT, with n indicating the fixed model order. The mixed-order MFn models were trained incrementally with SECT inference, where n indicates the maximum order, while MFx was trained directly with no order limiting.

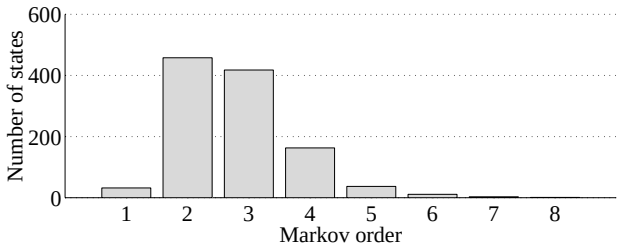

Figure 2: Distribution of states in direct mixed-order model MFx according to associated order (context length). The decline with order indicates a sparse mixed-order model.

better direct the search through model space.

The mixed-order training selectively expands the HMM where the training data supports it. This allows the modelling of sparse high-order structures, typically identifiable with common sound "phrases" in the language. Fixed-order training becomes infeasible above about fifth order, while the large training set supports up to eighth-order contexts with direct mixed-order models (MFx). As can be seen in Figure 2, the vast majority of contexts had a length of two or three, confirming the sparse mixed-order behaviour. Also interesting to note is the mismatch of standard first-order modelling to this task.

5. CONCLUSIONS

By incorporating the SECT algorithm for ergodic Markov chain inference into HMM training, we are able to infer HMM topology from the training data. The resulting ergodic HMM is inherently mixed-order, allowing more efficient language modelling compared to fixed-order models. This leads to faster training times and smaller models for comparable classification performance on language identification tasks.

Several areas of improvement are apparent. The most likely symbol sequence does not incorporate the uncertainty and overlap associated with the symbol (output) pdfs. A remedy would be the use of Baum-Welch reestimation [1] instead of Viterbi reestimation, which allows "soft" probabilistic counts for each symbol. The inference framework can also be expanded to train context-specific and duration-specific HMMs as in [3] and to admit discriminative training.

6. REFERENCES

1. Deller, J. R., Proakis, J. G., and Hansen, J. H. L., *Discrete-Time Processing of Speech Signals*, Macmillan Publishing Company, Englewood Cliffs, New Jersey, USA, 1993.
2. Bengio, Y., "Markovian models for sequential data," *Neural Computing Surveys*, vol. 2, pp. 129–162, 1999.
3. Du Preez, J. A. and Weber, D. M., "Automatic language recognition using high-order HMMs," in *Proceedings of the International Conference on Spoken Language Processing*, Sydney, Australia, 1998, vol. 2, pp. 117–120.
4. Ron, D., Singer, Y., and Tishby, N., "The power of amnesia: Learning probabilistic automata with variable memory length," *Machine Learning*, vol. 25, no. 2/3, pp. 117–149, 1996.
5. Oliver, J. J. and Hand, D., "Introduction to minimum encoding inference," Tech. Rep. TR4-94, Statistics Dept., Open University, Sept. 1994.
6. Rissanen, J., *Stochastic Complexity in Statistical Inquiry*, World Scientific, Singapore, 1989.
7. Schwardt, L. C. and Du Preez, J. A., "Automatic language identification using mixed-order HMMs and untranscribed corpora," in *Proceedings of the International Conference on Spoken Language Processing*, Beijing, China, 2000.
8. Bühlmann, P. and Wyner, A. J., "Variable length Markov chains," *Annals of Statistics*, vol. 27, pp. 480–513, 1999.
9. Rissanen, J., "A universal data compression system," *IEEE Trans. Info. Theory*, vol. 29, no. 5, pp. 656–664, 1983.
10. Schwardt, L. C. and Du Preez, J. A., "Modelling temporal structure with prediction suffix trees," in *Proceedings of the IAPR PRASA Conference*, Stellenbosch, South Africa, 1999.
11. Dempster, A. P., Laird, N. M., and Rubin, D. B., "Maximum likelihood from incomplete data via the EM algorithm," *J. Royal Stat. Soc., B*, vol. 39, no. 1, pp. 1–38, 1977.
12. Chen, S. F. and Goodman, J., "An empirical study of smoothing techniques for language modelling," Tech. Rep. TR-10-98, Harvard University, Cambridge, Mass., Aug. 1998.
13. Brand, M., "Structure learning in conditional probability models via an entropic prior and parameter extinction," *Neural Computation*, vol. 11, no. 5, pp. 1155–1182, July 1999.