HIERARCHICAL CLUSTERING OF MIXTURE TYING USING A PARTIALLY OBSERVABLE MARKOV DECISION PROCESS

Michael Jonas
<mrisgin@eecs.tufts.edu>

James G. Schmolze
<schmolze@eecs.tufts.edu>

Computer Science Department
Tufts University
Medford, MA 02155

ABSTRACT

For over a decade, the Hidden Markov Model (HMM) has been the primary tool used for acoustic modeling in the field of speech recognition. In this paper we examine a more general approach using a Partially Observable Markov Decision Process (POMDP) to model the base phonetic unit. We introduce the concept of multiple phonetic context classes, one for each of the infinite possible contexts a phoneme can be in, and show how a POMDP can be used to represent such a model. Much the same way that tying mixtures at the state level across phonemes sharing linguistic properties is used to fill in gaps in the model space due to lack of data, the POMDP model can fill in additional gaps, in effect adding a second level of clustering driven by the data itself.

1. INTRODUCTION

The current state of the art speech recognition engines use context dependent phonetic units to model the acoustic signal using Hidden Markov Models (HMM). To achieve a uniform coverage among all models, specifically for those models where little or no training data exists, information among models is shared. This sharing can be done in various ways, with varying levels of complexity, but centers around the idea of sharing (tying) the mixture densities of models with similar linguistic features [1][2] .

In this paper we look at another way to boost a model's performance. By combining different contexts in the same model, we achieve a more accurate representation of each model. This representation can be seen as a second tier of clustering, on top of the clustering achieved by the aforementioned tying of mixture densities. We model this new representation using a Partially Observable Markov Decision Process (POMDP).

In section 2 we discuss the basics of a POMDP and its relationship to an HMM. In section 3 we apply our new model to represent a richer acoustic model. Several decoding strategies, centered around a modified Viterbi search, are discussed in section 4. We give experimental results in section 5 using the TIMIT data set and discuss further work in section 6.

2. POMDP

A Partially Observable Markov Decision Process can be seen as a generalized form of a Hidden Markov Model. It differs from an HMM in allowing multiple transitions between the same two states where each such transition represents an action–hence it's a "decision process"–and includes a reward at each state. Thus, each action has its own transition model between states, and an agent selects actions so as to maximize its expected total reward at each state.

Conversely, we can describe an HMM as a POMDP with only a single action–i.e. with no decision process–and no reward [3][4].

We formally define a POMDP as a model $\lambda = <S,D,a,b,r,\pi>$ where:

- S - a finite set of N states
- D - a finite set of K actions
- $a^d_{i,j}$ - state i to j transition probability with action d
- b^d_i - observation probability in state i with action d
- r^d_i - reward of being in state i with action d
- π - the initial state distribution

It is easily seen that if D in λ is a set containing a single action and r is zero for all states, our model represents an HMM since we can do away with any reference to actions d for a and b and can ignore r. We can therefore use a POMDP to represent a single phoneme, where each action, d, represents one of a possibly infinite number of contexts the phoneme can be in.

3. MODELING SPEECH USING POMDP'S

The base unit in speech is the phoneme. To better discriminate among models representing those phonemes, we add context. The more context we add, the better a model can discriminate between similar phonemes and the more training data we need. Current state of the art systems use triphones, a phoneme in context of its left and right phoneme. Quinphone systems also exist, but tend to be less popular as the size of the model space increases exponentially. Though mixture tying can fill some of the gaps caused by this growth, discriminability suffers. We alleviate this by adding a hierarchy to mixture tying.

As with an HMM, a POMDP's states represent the pronunciation task: beginning, middle, end of phoneme where the observed acoustics features are associated with each state. The randomness in the state transitions accounts for time stretching in the phoneme and the randomness in the observations accounts for the variability in pronunciations.

3.1. POMDP model of a phoneme

Our initial POMDP model consists of 3 states, with multiple transitions between consecutive states as well as self-loops. Transitions represent the various contexts in our model. Our POMDP will look something like this:

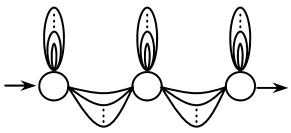

Figure 3.1: POMDP phonetic context model

Having multiple transitions between two distinct states, one per action (i.e. context), a POMDP represents the different contextual models the phoneme finds itself in. A phoneme in context will now be modeled in all its different context by a single POMDP, where each action represents the different context, carrying with it its own set of transition probabilities and mixture densities. This differs from an HMM model, where each context is modeled by its own HMM.

Initially we will constrain our POMDP model by requiring that the action taken as we enter the model remain the same throughout the model. This insures that we stay within the same context as we transition through the model. We will also relax this constraint by allowing actions belonging to a class of related context to be taken, thus enabling us to jump between models.

Additionally, our POMDP will model multiple *context classes*, such as monophone, biphone (left context), and triphone. In theory, a POMDP represents a phoneme in all possible contexts of all possible context classes. In practice, however, since it would be impossible to achieve such a model, we limit the number of context classes to those mentioned previously.

3.2. Training with EM

The training procedure for a POMDP is similar to that used in training a traditional HMM based model. The basic method is based on expectation maximization (EM). We train each context (monophone, biphone, triphone) independently on the same data using the general EM approach and combine them afterwards into a single, unified model, representing our POMDP. Combining the different models is simply done by labeling each transition and associated mixture density of every model by both its context and context class.

4. DECODING

We define a *cross context* phoneme as a tuple made up of context models that all share the same *partial context*. Partial context is defined by the following rules: a triphone and biphone share a partial context if they share the same left context and center phone and a monophone shares the same partial context with either a triphone or biphone by sharing the same center phone.

4.1. Modified Viterbi

Having a model that now contains not only all occurrences of a particular context class, but also multiple context classes, we modify our search to utilize this additional information. We use the Viterbi algorithm as a starting point and modify it to fit within our new model's additional complexity.

Several strategies have been explored in creating a modified Viterbi algorithm. The basic approach extends the search space to include all context classes by expanding all cross context phonemes in place of each original phoneme. Since under-training of models can be a problem, especially with small data sets, augmenting our search space by including all context class models can strengthen overall the ability to acoustically match the input signal correctly.

By initially combining all context classes and using a general weighting factor per context class to balance the impact of each class on our search we can extend our search space. This weighting factor represents a reward in our POMDP that we apply upon entering the model.

However, to achieve a more workable balance between models the weighting factors are normalized for each context model by estimates of context class weights for each model using the frequency of occurrences of that context. This is similar to the technique used for language

modeling by combining mixed classes of ngram models. We call this a Weighted Mixed Model Viterbi (WMM Viterbi).

A second and more robust method extends WMM Viterbi by relaxing the constraint on actions. We allow all actions belonging to the same cross context model to be taken at any time through the model. We call this a Cross-context Mixed Model Viterbi (CMM Viterbi).

4.2. Weighted Mixed Model Viterbi

In this approach we combine all models and individually weight each model by the frequency they appear in the training set and allow Viterbi to choose the best path through the entire lattice of possibilities. We relax the context rules during the search by matching up all partial context phones. This, in effect, will wild-card all monophones to match up with all triphones of the same center phone and all biphones to match up with those triphones whose other context they share. The idea is similar to that used in language modeling, whereby ngram probabilities are derived from a representative data set to strengthen word pairs or triples that appear more frequently.

To generate the relative weighting, we take a statistical summary of the training set relative to each context class, *c*, and generate a weight, w^c_m, for each phonetic context model, *m*, in that context class based on each model's frequency count, f^c_m. This weight is again applied as a percentage to the model upon entering its start state. Since small data sets tend to have many unseen models, we apply a lower threshold weight, L^c, for all unseen models. Additionally, to compute a model's weight, we take its frequency count and normalize it by a constant K^c where this constant is arbitrary and can differ for each context class. Finally, we apply an artificial upper bound, W^c, to each context class which represents a global weighting factor to each of the context classes, *c*, to balance the impact each class has on the entire search space. The weighting will discount, as a percentage, under-trained models as well as less discriminating but well-trained models. Constructing our equation yields:

$$w^c_m = L^c + \min(f^c_m / K^c , 1) * (W^c - L^c)$$

The *min* function in our equation insures that our weight remains within the bounds of L^c and W^c, since K^c can be smaller than f^c_m.

4.3. Cross-context Mixed Model Viterbi

A more sophisticated strategy in decoding our new models is to use the full power of the POMPD model by relaxing the constraint of enforcing the same context throughout the model. This, in effect, further increases the search space of our model by creating additional branches at each state. To insure that we don't degrade our model by washing it out with non-related information from other models, we carefully choose how to relax these constraints, choosing only contexts belonging to the same cross context phones.

Now, at each time step, we can choose between all cross context models and transition to any of them. In fact, we choose all possible transitions, within the limit of the beam pruning threshold of our general Viterbi search. We apply the same weighting factor, w^c_m, used in our WMM Viterbi, but this time we apply it at each state as we transition to another state, and perhaps another model. The weight is added into to the maximum likelihood score at each transition.

What we've added, in essence, is a set of "jump" transitions to our traditional HMM model. Along with skips and self-loops, our model can now jump to a constrained set of other models at each state. We add the ability to weight these jumps to allow for tuning of our model to the data.

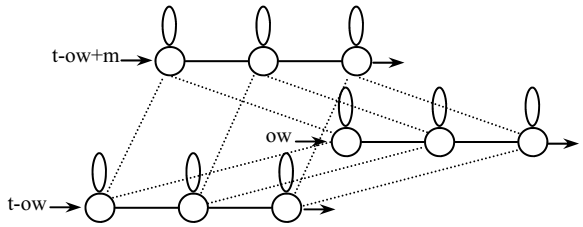

Figure 4.1: "jump" transitions for triphone "t-ow+m"

The jumps are zero probability transitions with the model weight, w^c_m, inserted in its place when the jump transition is taken.

5. EXPERIMENTS

We tested our new models using the TIMIT corpus, which is considered a small data set for continuous speech recognition systems. We used version 5.10 of the ISIP prototype speech recognition system from Mississippi State University to build our models and a modified version of their decoder (trace_projector) to decode them. To build language models, we used the CMU-Cambridge Statistical Language Modeling Toolkit version 2. Finally, for scoring we used the NIST Scoring Toolkit version 1.2.

5.1. TIMIT data set

The TIMIT training corpus is composed of 387 speakers with 3869 utterances of spoken English for a little over 3 hours of data. It uses 47 phonemes and a 4457 word

dictionary. The TIMIT testing corpus is composed of 14 speakers not from training, with 60 utterances of spoken English for a total of 3 minutes of data. We built a trigram language model from the TIMIT training data set, and augmented it with additional utterances from similar spoken English text found in Oregon Graduate Institute's National Cellular and Stories corpora.

5.2. Building models

We used 16 mixture training per state and a maximum likelihood decision tree-based state tying procedure. We built an initial set of non-mixture monophone models and created cross word models from those. The table below gives ISIP specific parameters used in building our models. Refer to the Aurora Working Group report of the ISIP baseline system [5] for details.

Training Corpus	no. states per model	beam width	state-tying threshold		
			split	merge	occup
TIMIT	3	2500	100	100	600

Table 5.1: ISIP model training parameters

5.3. Baseline experiments

For all decoding experiments, we initially found optimal parameter settings for our HMM models. All subsequent TIMIT decoding experiments used a word insertion penalty of -8, a language model scaling factor of 9, and state/model/word beam pruning of 600/500/500. The table below gives baseline numbers for TIMIT word recognition decoding results for each individual HMM model.

Corpus	WER	accuracy
TIMIT		
triphone	52.4	53.9
biphone	53.1	51.3
monophone	69.3	33.5

Table 5.2: HMM baseline results

5.4. Decoding experiments

We ran experiments for all three decoding strategies. Using our modified decoder, we combined our individually trained triphone, biphone, and monophone HMM models to create our POMDP model. For both WMM and CMM Viterbi, we found a set of weights, W^c, L^c and K^c, that yielded optimal accuracy with smallest possible word error rate.

Table 5.3 lists some optimal weights found for each of our decoding strategies.

Viterbi	L^c / W^c / K^c			WER	accur.
	tri	bi	mono		
WMM	0 / 5 / 90	25 / 100 / 90	0 / 10 / 90	50.5	54.9
CMM	0 / 5 / 90	45 / 100 / 90	0 / 10 / 90	48.8	55.3

Table 5.3: Decoding results

6. CONCLUSIONS

Preliminary results look promising. Table 5.3 shows that our models generally improve upon the baseline HMM systems. In both cases, the weights for triphones and monophones are relatively small, meaning that we are primarily using the biphone model in our weighted combination. We expect additional experimentation to find an optimal set of weights that more fully utilizes the higher context models to yield even better performance.

Our best performance comes from our CMM model. This is of great interest to us as it utilizes the full power of a POMDP model. By allowing cross context model jumps, we are able to explore a search space that has not been looked at in any previous acoustic modeling work with HMM's to date. Allowing cross model transitions, with proper weighting and a well constrained model set, could be an important discovery in acoustic modeling.

Further investigation is required to determine how best to weight each of our different context models to generate a more robust POMDP model. We plan to look into a fuller utilization of our POMDP model by relaxing the constraints on actions, allowing more jumps across the model space. Additionally, we want to look at the training process. A modified form of EM that incorporates the POMDP model as a whole in training is needed.

7. REFERENCES

[1] J.R. Deller, J.H.L. Hansen, J.G. Proakis, *Discrete-Time Processing of Speech*, Prentice Hall, Upper Saddle River, NJ, 1993.

[2] F. Jelinek, *Statistical Methods for Speech Recognition*, MIT Press, Cambridge, MA, 1997.

[3] H. Shatkay and L.P. Kaelbling, "Heading in the Right Direction," *Proceedings of the Fifteenth International Conference on Machine Learning*, 1998.

[4] L.P. Kaelbling, M.L. Littman, A.R. Cassandra, *Planning and Acting in Partially Observable Stochastic Domains*, Artificial Intelligence, Vol 101, 1998.

[5] N. Parihar and J. Picone, "DSR Front End LVCSR Evaluation - Baseline Recognition System Description," *Aurora Working Group*, July 27, 2001.

[6] J.C. Junqua and L. Vassallo, Context Modeling and Clustering in Continuous Speech Recognition, *4th International Conference on Spoken Language Processing*, 1996.