Combining phonetic attributes using conditional random fields

Jeremy Morris and Eric Fosler-Lussier

Department of Computer Science and Engineering
The Ohio State University, Columbus, OH, USA
{morrijer,fosler}@cse.ohio-state.edu

Abstract

A Conditional Random Field is a mathematical model for sequences that is similar in many ways to a Hidden Markov Model, but is discriminative rather than generative in nature. In this paper, we explore the application of the CRF model to ASR processing of discriminative phonetic features by building a system that performs first-pass phonetic recognition using discriminatively trained phonetic features. With this system, we show that this CRF model achieves an accuracy level in a phone recognition task that is superior to a similarly trained HMM model.

Index Terms: speech recognition, conditional random fields.

1. Introduction

In recent years, attempts have been made to incorporate features determined from a speech signal via discriminative methods into the Automatic Speech Recognition (ASR) process. These attempts have led to systems like Tandem systems [1] where individual features are extracted from the speech signal using a neural network, then used as inputs into a traditional HMM system.

Models based around the framework of maximum entropy have shown successful results in various areas of discriminative modeling, including part of speech tagging [2], information extraction [3], and statistical language modeling [4]. In ASR, the maximum entropy principle has been used to discriminatively train Gaussian-based systems [5], combining phonetic landmark estimates in lattice rescoring [6], and for performing score combination on the acoustic, segmental and word levels [7].

More recently, the concept of maximum entropy modeling has been extended successfully to sequences in the form of Conditional Random Fields (CRFs). The CRF framework builds the discriminative maximum entropy model framework into a Markov model for classifying labels for observed sequences of data. CRFs have been successfully implemented as classifiers for tasks in part of speech tagging [8], parsing [9] and other areas. In addition, CRFs have been explored for building language models for ASR [10] and for phone classification [11].

The framework of CRF models provides an intuitive method for combining evidence from various sources within the data to determine the content of the utterance. CRFs present an interesting set of tools for analyzing spoken utterances, as they mirrors the familiar HMM models used in speech recognition while still providing a framework for discriminative evidence combination. In this paper we look at a method of using Conditional Random Fields to combine together phonetic attributes for phone recognition. In the following section, we provide an overview of Conditional Random Fields, including how their definition and training. Following this, we discuss how CRFs can be used to combine together discriminative phonetic features for recognition. Finally we give an overview of our experimental setup and a discussion of our results.

2. Conditional Random Fields

Since Conditional Random Fields are relatively new for ASR, we begin with a brief discussion of the model using notation and terminology taken from [8] and [9]. A Conditional Random Field is a discriminative model that attempts to model the posterior probability of a label sequence given a set of data presented to it. It is a constraint-based model, where different attributes of the data being modeled are chosen to constrain the resulting probability of the various labels that the data segment can receive. We might, for example, wish to indicate that sometimes the phone /d/ can be realized in a way where it loses its voicing, and might take on properties normally associated with the phone /t/. A CRF model of the data allows us to look for these attributes in the data and to learn appropriate weights for how often /d/s might be realized as devoiced, and then apply this information when attempting to determine the proper label for that data segment.

A CRF defines a posterior probability $P(\mathbf{y}|\mathbf{x})$ of a label sequence $\mathbf{y}$ for a given input sequence $\mathbf{x}$. For our purposes, the input sequence $\mathbf{x}$ corresponds to a series of frames of speech data, while the label sequence $\mathbf{y}$ is the phone label sequence assigned to the input sequence. Each frame of $\mathbf{x}$ is assigned one label in $\mathbf{y}$.

A CRF is described by a series of *state feature functions* $\mathbf{s}(y, \mathbf{x}, i)$ with corresponding weights λ and a series of *transition feature functions* $\mathbf{t}(y, y', \mathbf{x}, i)$ with corresponding weights μ. Here y and y' are labels, $\mathbf{x}$ is a sequence of observations, and i an index pointing to a position in the sequence $\mathbf{x}$.

A *state feature function* is only non-zero if the label y matches the label that the feature function is defined for at position i and the observation sequence $\mathbf{x}$ at position i shows evidence of a particular attribute that the feature function is defined for. For example, we might have a state feature function that is defined for the particular phone /t/ indicating that the phone is not voiced. This state feature function for devoicing could be defined as follows:

$$s(y, \mathbf{x}, i) = \begin{cases} 1, & \text{if } y_i = t \text{ and } voiced(x_i) = false \\ 0, & \text{otherwise} \end{cases}$$

This state feature function evaluates to a non-zero value only when the input label matches the label associated with the function (in this case /t/) and when input data sequence does not show evidence of voicing at position i. For illustration purposes, the output of this function is binary, but the definition of a CRF does not require feature functions to be binary; the voicing observation above could be replaced by a real-valued function. Most of the feature definitions in the literature use binary feature functions, but in this paper we make use of real-valued feature functions.

Transition feature functions are defined in a similar manner, but with a dependency on two labels (the previous label and the current label) rather than just the current label. The transition feature function evaluates to a non-zero value only when the labels in the sequence (y_t and y_{t-1}) match the labels defined for the transition function (y and y' respectively) and some attribute in the data exists. Other than the use of two successive labels instead of a single label, transition feature functions are defined in the same manner as state feature functions.

Given these feature functions, the form of the conditional probability of a label sequence $\mathbf{y}$ over the observed sequence $\mathbf{x}$ takes the form:

$$P(\mathbf{y}|\mathbf{x}) \propto \exp \sum_i (\sum_j \lambda_j s_j(y, \mathbf{x}, i) + \sum_k \mu_k t_k(y_{i-1}, y_i, \mathbf{x}, i)) \quad (1)$$

Following [9], we make this notation more uniform by collapsing the notation for state feature functions and transition feature functions together into one notation style for all feature functions:

$$f(\mathbf{y}, \mathbf{x}, i) = \begin{cases} s(y_i, \mathbf{x}, i), & \text{if } f \text{ is a state function} \\ t(y_{i-1}, y_i, \mathbf{x}, i), & \text{if } f \text{ is a transition function} \end{cases}$$

In addition, we merge the values of the associated μ and λ values together into a single weight vector λ. This allows the features from the observed data sequence $\mathbf{x}$ and the label sequence $\mathbf{y}$ to be represented as a *feature vector* defined as:

$$\mathbf{F}(\mathbf{y}, \mathbf{x}) = \sum_i \mathbf{f}(\mathbf{y}, \mathbf{x}, i) \quad (2)$$

which allows us to re-write equation (1) as:

$$P(\mathbf{y}|\mathbf{x}) = \frac{\exp(\lambda \cdot \mathbf{F}(\mathbf{y}, \mathbf{x}))}{Z(\mathbf{x})} \quad (3)$$

where

$$Z(\mathbf{x}) = \sum_{\mathbf{y}} \exp(\lambda \cdot \mathbf{F}(\mathbf{y}, \mathbf{x})) \quad (4)$$

Here the normalizing constant $Z(\mathbf{x})$ is dependent on the sum of all possible label combinations over the observed data sequence. Note that this means that if we only want to find the best sequence through the data and not the actual probability for a sequence, we merely have to find the sequence $\mathbf{y}$ that maximizes $\lambda \cdot \mathbf{F}(\mathbf{y}, \mathbf{x})$.

$$\arg\max_{\mathbf{y}} P(\mathbf{y}|\mathbf{x}) = \arg\max_{\mathbf{y}} \exp(\lambda \cdot \mathbf{F}(\mathbf{y}, \mathbf{x})) \quad (5)$$

As pointed out by [9], the feature vector $\mathbf{F}(\mathbf{y}, \mathbf{x})$ decomposes into a sum of terms for sequential labels, and the maximum value can be found through the application of the Viterbi algorithm.

Conditional Random Fields are trained by maximizing the log-likelihood of the the training set with respect to the model – the best model for the training data comes from the model that gives the highest likelihood for the training set. A great deal of investigation has gone into how conditional random field based models can be trained. In [12], various methods for computing these weights were examined and gradient-based methods were found to give good performance for maximizing the log-likelihood. We follow this recommendation and use LBFGS to find the zeros of the gradient.

Computing the gradient of the log-likelihood is not a trivial task, as it requires the computation of the normalizing constant $Z(\mathbf{x})$ at every iteration. Fortunately, there is a variant of the forward-backward algorithm that computes this efficiently (a discussion of this implementation can be found in [9]).

3. CRFs and Discriminative ASR

Viewing speech recognition as an evidence combination task, we can see how the constraint-based conditional random field framework might work in ASR. A discriminative classifier (such as an multi-layer perceptron) provides evidence from the speech signal of a particular attribute. At the highest level, this might just be an indication that a particular phone has been uttered, while at lower levels this may be an indication that a particular speech attribute (such as voicing, or place of articulation) exists in the frame. In the CRF framework, these bits of evidence from the speech signal are each weighed based on their relevance to the phone being hypothesized, adding their positive or negative weights based on whether they are good evidence either for or against that phone. The most probable phone can then be selected.

We can see that the CRF framework plays a role comparable to that of an HMM – it provides a Markov framework via the transition features for allowing hypotheses from previous timesteps to influence the selection of the current label. In this manner, we can implement a discriminative framework for ASR using a CRF in place of the traditional HMM. One possible advantage of this framework is that it makes no assumptions about the interdependence of the observed data. HMMs make an assumption that the observed data is independent given the labels, while CRFs require no such assumptions to be made.

For our initial exploration, we have chosen to extend prior work on phonological feature detection (e.g. [13]) to build a model that combines together phonetic attributes defined by the International Phonetic Association (IPA) phonetic chart and learned via a multi-layer perceptron. Each phone was broken down into a description of its component phonetic attributes as given in Table 1. Our goal for this experiment was to examine the effectiveness of using a CRF model to perform the labeling of a sequence of phones over an observed speech signal based on extractions of these common phonetic attributes from the speech signal.

Table 1: *Phonetic attributes extracted.*

attribute	possible output values
SONORITY	vowel, obstruent, sonorant, syllabic, silence
VOICE	voiced, unvoiced, n/a
MANNER	fric., stop, closure, flap, nasal, approx., nasalflap, n/a
PLACE	lab., dent., alveolar, pal., vel., glot., lat., rhotic, n/a
HEIGHT	high, mid, low, lowhigh, midhigh, n/a
FRONT	front, back, central, backfront, n/a
ROUND	round, nonround, roundnonround, nonroundround, n/a
TENSE	tense, lax, n/a

Our work differs from the work in [11] on phone classification in two major ways: First, our system uses phonetic attributes as input features, rather than the MFCCs. Second, our paper examines a system that performs a phone recognition task, rather than a phone classification task, allowing for the possibility of deletion and insertion errors in the results.

4. Experimental Setup

For these initial experiments we used the TIMIT acoustic phonetic speech corpus [14] for all of our training and testing. To extract phonetic features, we trained a set of neural networks using the ICSI QuickNet MLP neural network software [15]. These neural networks were trained on individual phonetic attributes (Table 1) derived from the phonetic transcriptions of TIMIT. Each phone in

the transcript was converted to its canonical attribute definition. The MLP networks were trained to produce a posterior probability over the values of each attribute class. 12th order PLP features, plus delta coefficients, were used as input to these MLPs.

As a baseline for comparison purposes, we compared phone-level accuracies of the system to the results given by a system built using the Tandem model described in [1]: the output of the MLP attribute detectors is used as input to a Gaussian-based HMM. For these experiments, the Tandem system was built using HTK [16] and trained on the a modified version of the outputs from our MLP system described above. A phone bigram language model built from the training set was used in the scoring of the results. As described in [1] we used the linear output of the MLPs with a KL transform applied to them to decorrelate the features, as this gave the best results for the HMM system.[1] We also show results of HTK systems trained on MFCCs and directly on the derived PLP features for comparison purposes. In addition, for all of the systems we used a reduced phoneme labeling for TIMIT of 39 possible outputs instead of the full 61 phone labels as described in [17]. All HTK triphone results are for 4 gaussian mixtures, while monophone results are for single gaussians.

To train our conditional random fields models, we used software derived from the Java CRF package on Sourceforge [18]. This package (and the code that we derived from it) uses a quasi-Newton LBFGS algorithm to perform the gradient minimization used to train the maximum entropy models. The training process as implemented was based on the work done in [9], using their version of the forward-backward algorithm to compute the gradient of the log-likelihood for minimization.

To build a CRF model with a structure that is comparable to the Tandem system, state feature functions are defined as the outputs of the QuickNet MLPs and an associated label, with one feature function per label/net output combination. In other words, if the outputs of the MLPs for a particular data segment x_i are given as the vector $\mathbf{O}$, then the feature function $s_j(y, \mathbf{x}, i)$ takes on the value of the attribute in $\mathbf{O}$ associated with function f_j when the label y matches the label at time i that we are evaluating and the value of 0 when the label y does not match this label. The model additionally has a bias state feature for each possible label. Defined in this manner, feature functions are not restricted to the values of 0 or 1, but instead are continuous values bounded by 0 and 1. In addition, unlike the HMM model, no decorrelation or linearization is performed on the features given to the CRF – the posteriors learned by the MLPs are fed directly into the CRF system.

Transition feature functions are implemented in a simple binary manner, with the transition function $t_j(y, y', \mathbf{x}, i)$ evaluating to 1 if a transition occurs between the two labels y and y' at the time from $i-1$ to i and evaluating to 0 if this transition does not occur. In addition, t_j evaluates to 0 when i is at time 0 (since no previous transitions occurred into time zero). Unlike the state feature functions, this is a hard 0 or 1 decision - either a transition occurs at this timeframe or it does not.

At testing time, we do not know whether a particular transition has occurred or not at a given frame, so all possible transitions are postulated (essentially giving every transition feature function a value of 1 for a given transition label pair). State feature functions use the values provided by the MLPs to determine their frame-level results. For decoding purposes, frame-level results are written to a lattice, where each possible phone value for a state is a node and the sum of the transition and state features are placed on the arcs (corresponding to the unnormalized log probabilities of the terms in Equation (5). These lattices are then decoded using the AT&T Finite State Toolkit [19] to give the best path.

[1] The Tandem system described in [1] uses phone posteriors, rather than the posteriors for the phonetic attributes we use here. Internal experiments within our group have shown comparable results for the Tandem system trained on phonetic attribute posteriors to one trained on phone posteriors.

5. Results & Discussion

Table 2 shows the comparisons between the different models on phone correctness and accuracy as given by HTK's HResults program. As we can see, the basic CRF model performs with better accuracy than the Tandem monophone-only system, and its accuracy begins to approach that of the Tandem triphone model. It is important to note, however, that the CRF was not trained on anything as complex as triphones – only on single phone labels. For the CRF, the simple transition model implemented here acts much like a combination of the duration model in an HMM and a language model over the phones. The CRF effectively has a single-state duration model built into its training process (i.e. the weight on a transition from a phone label to itself in the next frame), as well as a transition weight from every phone to every other possible phone. Also note that the Tandem and CRF models do not have a comparable number of parameters. The CRF model has one parameter for each state function and one parameter for each transition function, with roughly 4500 parameters for this model. By comparison, the triphone Tandem model has over two million parameters while the monophone Tandem model has over 28,000.

Table 2: *Phone accuracy comparisons.*

Model	Phone Accuracy	Phone Correct
Tandem (monophone)	61.48%	63.50%
Tandem (triphone)	66.69%	72.52%
HTK (triphone / PLPs)	60.08%	64.07%
HTK (triphone / MFCCs)	62.37%	70.82%
CRF (monophone)	65.23%	66.74%

Importantly, the model as described here implements only a very simplistic transition model – it does not make use of the observed data to decide directly if a transition has occurred or not. Instead, it makes use only of the overwhelming evidence that the state must have changed (because the *state* features point overwhelmingly to the conclusion that the current state cannot be the same as the previous state). The CRF as implemented here has many more deleted phones in its results than when compared to either the monophone or the triphone Tandem models – indicating that the CRF system is undergenerating its results. This gives weight to the idea that the system's performance relative to the Tandem triphone system may be because of the simple transition model used in implementation.

Besides the extra deletions, the CRF also showed fewer insertions and substitutions than the other models. We can analyze this effect by looking at the precision of the models, using a form that is used quite often in statistical natural language applications:

$$precision = \frac{totalcorrect}{correct + subs + inserts} \quad (6)$$

The precision gives us a measurement of how often we are right when we generate a phone hypothesis, rather than the percentage correct measure which shows how many of our hypotheses were right overall. Precision results for the various models are

shown in Table 3. Here we can see that when the CRF hypothesizes a phone, it is correct more often than any of the other models. This lends more evidence to the idea that the undergeneration that the CRF is performing is hurting its final result.

Table 3: *Phone Precision.*

Model	Precision
Tandem (monophone)	73.66%
Tandem (triphone)	73.44%
HTK (triphone / PLPs)	69.97%
HTK (triphone / MFCCs)	68.89%
CRF	77.66%

6. Conclusions and Future Work

These results show some promise for the idea of using Conditional Random Field models for ASR with discriminatively trained attributes. With a simple model, we achieve results that are comparable to similar HMMs. Even though the system was unable to achieve the results of a triphone-trained HMM system, we were able to achieve results that were superior to an HMM system that was trained on monophone labels which is the type of system that is most similar to the CRF system we built. Also, there has been not attempt to impose extra parameters that the HMM makes use of onto the CRF model like word transition penalties or a language model scaling factor. While parameters like these could be imposed on the model, we feel that adding more information to the learning process itself may be a better way of tuning this model. This is something we would like to explore further.

In addition, the real power of the CRF model comes from the ability to add features to the transition features – an aspect that we are currently not exploiting. Preliminary experiments expanding this model to include seemingly redundant features such as phone posteriors have shown some improved performance for the model. In the future, we also want to add more features to the state transitions to see if this improves performance. One idea is to add a boundary detector [20] that includes information about whether a boundary has been observed or not, which may help with the undergeneration problem observed above. We would also like to try adding more context-dependencies to the transitions – in other work involving CRFs adding transition features derived from attributes of the surrounding observed nodes has improved performance, and our preliminary experiments in this direction show improvements for this model as well.

7. Acknowledgments

The authors would like to thank Keith Johnson, Anton Rytting, and Yu Wang for useful discussions of this work and the International Computer Science Institute for providing the neural network software. This work was supported by NSF ITR grant IIS-0427413; the opinions and conclusions expressed in this work are those of the authors and not of any funding agency.

8. References

[1] H. Hermansky, D. Ellis, and S. Sharma, "Tandem connectionist feature stream extraction for conventional HMM systems", in *Proc. of the ICASSP*, 2000.

[2] A. Ratnaparkhi, "A maximum entropy model for part-of-speech tagging", in *Proc. of the EMNLP*, 1996.

[3] A. McCallum, D. Freitag, and F. Pereira, "Maximum entropy markov models for information extraction and segmentation", in *Proc. of the ICML*, 2000, pp. 591–598.

[4] R. Rosenfeld, *Adaptive statistical language modeling: A maximum entropy approach*, PhD thesis, Carnegie-Mellon University, 1994.

[5] W. Macherey and H. Ney, "A comparative study on maximum entropy and discriminative training for acoustic modeling in automatic speech recognition", in *Proc. of EuroSpeech*, 2003.

[6] M. Hasegawa-Johnson et. al, "Landmark-based speech recognition: Report of the 2004 johns hopkins summer workshop", in *ICASSP*, 2005.

[7] H. Yu and A. Waibel, "Integrating thumbnail features for speech recognition using conditional exponential models", in *Proc. of the ICASSP*, 2004, pp. 893–896.

[8] J. Lafferty, A. McCallum, and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data", in *Proc. of the ICML*, 2001.

[9] F. Sha and F. Pereira, "Shallow parsing with conditional random fields", in *Proc. of Human Language Technology, NAACL*, 2003.

[10] B. Roark, M. Saraclar, M. Collins, and M. Johnson, "Discriminative language modeling with conditional random fields and the perceptron algorithm", in *Proc. of ACL*, 2004, pp. 48–55.

[11] A. Gunawardana, M. Mahajan, A. Acero, and J. Platt, "Hidden conditional random fields for phone classification", in *Proc. of Interspeech*, 2005.

[12] R. Malouf, "A comparison of algorithms for maximum entropy parameter estimation", in *Proc. of the Sixth Conference on Natural Language Learning*, 2002, p. 4955.

[13] M. Rajamanohar and E. Fosler-Lussier, "An evaluation of hierarchical articulatory feature detectors", in *IEEE Automatic Speech Recognition and Understanding Workshop*, 2005.

[14] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, D. S. Pallett, and N. L. Dahlgren, "DARPA TIMIT acoustic phonetic continuous speech corpus cdrom", 1993.

[15] D. Johnson et al., "ICSI quicknet software package", `http://www.icsi.berkely.edu/Speech/qn.html`, 2004.

[16] S. Young, G. Evermann, T. Hain, D. Kershaw, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. Woodland, *The HTK Book*, Cambridge Unveristy Engineering Department, 2002, http://htk.eng.cam.ac.uk.

[17] K. Lee and H. Hon, "Speaker-independent phone recognition using hidden markov models", *IEEE Transacations on Acoustics, Speech and Signal Processing*, pp. 1641–1648, 1989.

[18] S. Sarawagi, "CRF package for java", `http://crf.sourceforge.net/`.

[19] F. Pereira M. Mohri and M. Riley, "AT&T finite-state machine library", `http://www.research.att.com/sw/tools/fsm/`.

[20] Y. Wang and E. Fosler-Lussier, "Integrating phonetic boundary discrimination explicitly into hmm systems", in *Proc. of Interspeech*, 2006.