CRFs for ASR: Extending to Word Recognition

Jeremy Morris

05/16/2008
Outline

- Review of Background and Previous Work
- Word Recognition
- Pilot experiments
Background

- **Conditional Random Fields (CRFs)**
  - Discriminative probabilistic sequence model
  - Directly defines a posterior probability of a label sequence $Y$ given an input observation sequence $X$ - $P(Y|X)$
  - An extension of Maximum Entropy (MaxEnt) models to sequences
Conditional Random Fields

\[
P(Y \mid X) = \frac{\exp \sum_{k} \left( \sum_{i} \lambda_{i} s_{i}(x, y_{k}) + \sum_{j} \mu_{j} t_{j}(x, y_{k}, y_{k-1}) \right)}{Z(x)}
\]

- CRF extends maximum entropy models by adding weighted transition functions
  - Both types of functions can be defined to incorporate observed inputs
Conditional Random Fields

State functions help determine the identity of the state.

Transition functions add associations between transitions from one label to another.
Background: Previous Experiments

- **Goal:** Integrate outputs of speech attribute detectors together for recognition
  - e.g. Phone classifiers, phonological feature classifiers
- **Attribute detector outputs highly correlated**
  - Stop detector vs. phone classifier for /t/ or /d/
- **Build a CRF model and compare to a Tandem HMM built using the same features**
Feature functions built using the neural net output

- Each attribute/label combination gives one feature function
- Phone class: \( s_{/t/,/t/} \) or \( s_{/t/,/s/} \)
- Feature class: \( s_{/t/,\text{stop}} \) or \( s_{/t/,\text{dental}} \)
## Background: Previous Results

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRF (phone posteriors)</td>
<td>67.32%</td>
</tr>
<tr>
<td><strong>CRF (phone posteriors – realigned)</strong></td>
<td><strong>69.92%</strong>*</td>
</tr>
<tr>
<td>Tandem[3] 4mix (phones)</td>
<td>68.07%</td>
</tr>
<tr>
<td>Tandem[3] 16mix (phones)</td>
<td>69.34%</td>
</tr>
<tr>
<td>CRF (phono. fea. linear KL)</td>
<td>66.37%</td>
</tr>
<tr>
<td><strong>CRF (phono. fea. lin-KL – realigned)</strong></td>
<td><strong>68.99%</strong></td>
</tr>
<tr>
<td>Tandem[3] 4mix (phono fea.)</td>
<td>68.30%</td>
</tr>
<tr>
<td>Tandem[3] 16mix (phono fea.)</td>
<td>69.13%</td>
</tr>
<tr>
<td>CRF (phones+feas)</td>
<td>68.43%</td>
</tr>
<tr>
<td><strong>CRF (phones+feas – realigned)</strong></td>
<td><strong>70.63%</strong>*</td>
</tr>
<tr>
<td>Tandem[3] 16mix (phones+feas)</td>
<td>69.40%</td>
</tr>
</tbody>
</table>

* Significantly (p<0.05) better than comparable CRF monophone system
* Significantly (p<0.05) better than comparable Tandem 4mix triphone system
* Significantly (p<0.05) better than comparable Tandem 16mix triphone system
We now have CRF models that perform as well or better than HMM models for the task of phone recognition.

Problem: How do we extend this to *word recognition*?
Word Recognition

\[ P(W \mid X) \]

- Problem: For a given input signal \( X \), find the word string \( W \) that maximizes \( P(W \mid X) \)
- The CRF gives us an assignment over phone labels, not over word labels
Word Recognition

\[ P(W \mid X) = \sum_{\Phi} P(W \mid \Phi, X)P(\Phi \mid X) \]

- Problem: For a given input signal \( X \), find the word string \( W \) that maximizes \( P(W \mid X) \)
- The CRF gives us an assignment over phone labels, not over word labels
Word Recognition

\[ P(W \mid X) = \sum_{\Phi} P(W \mid \Phi, X) P(\Phi \mid X) \]

\[ \approx \sum_{\Phi} P(W \mid \Phi) P(\Phi \mid X) \]

- Assume that the word sequence is independent of the signal given the phone sequence (dictionary assumption)
Word Recognition

\[ P(W \mid X) \approx \sum_{\Phi} P(W \mid \Phi)P(\Phi \mid X) \]

- Another problem: CRF does not give \( P(\Phi \mid X) \)
  - \( \Phi \) here is a phone segment level assignment of phone labels
  - CRF gives related quantity – \( P(Q \mid X) \) where \( Q \) is the frame level assignment of phone labels
Word Recognition

- Frame level vs. Phone segment level
  - Mapping from frame level to phone level may not be deterministic
  - Example: The number “OH” with pronunciation /ow/
  - Consider this sequence of frame labels:
    - ow  ow  ow  ow  ow  ow  ow  ow  ow
  - How many separate utterances of the word “OH” does that sequence represent?
Frame level vs. Phone segment level

- This problem occurs because we’re using a single state to represent the phone /ow/
  - Phone either transitions to itself or transitions out to another phone
- What if we change this representation to a multi-state model?
  - This brings us closer to the HMM topology

```
  ow1  ow2  ow2  ow2  ow2  ow2  ow3  ow3
```

- Now we can see a single “OH” in this utterance
Word Recognition

\[ P(W \mid X) \approx \sum_{\Phi} P(W \mid \Phi)P(\Phi \mid X) \]

- Another problem: CRF does not give \( P(\Phi \mid X) \)
  - Multi-state model gives us a deterministic mapping of \( Q \rightarrow \Phi \)
    - Each frame-level assignment \( Q \) has exactly one segment level assignment associated with it
    - Potential problem – what if the multi-state model is inappropriate for the features we’ve chosen?
Word Recognition

\[ P(W | X) \approx \sum_{\Phi} P(W | \Phi) P(\Phi | X) \]

- What about \( P(W|\Phi) \)?
  - Non-deterministic across sequences of words
    - \( \Phi = / ah f eh r / \)
    - \( W = ? \) “a fair”? “affair”?  
    - The more words in the string, the more possible combinations can arise
    - Not easy to see how this could be computed directly or broken into smaller pieces for computation
Word Recognition

\[
P(W \mid X) \approx \sum_{\Phi} P(W \mid \Phi)P(\Phi \mid X)
\]

\[
= \sum_{\Phi} \frac{P(\Phi \mid W)P(W)}{P(\Phi)} P(\Phi \mid X)
\]

- Dumb thing first: Bayes Rule
  - P(W) – language model
  - P(\Phi \mid W) – dictionary model
  - P(\Phi) – prior probability of phone sequences
  - All of these can be built from data
Proposed Implementation

\[
P(W | X) \approx \sum_{\Phi} \frac{P(\Phi | W)P(W)}{P(\Phi)} P(\Phi | X)
\]

- CRF code produces a finite-state lattice of phone transitions
- Implement the first term as composition of finite state machines
- As an approximation, take the highest scoring word sequence (argmax) instead of performing the summation
Pilot Experiment: TIDIGITS

- First word recognition experiment – TIDIGITS recognition
  - Both isolated and strings of spoken digits, ZERO (or OH) to NINE
  - Male and female speakers

- Training set – 112 speakers total
  - Random selection of 11 speakers held out as development set
  - Remaining 101 speakers used for training as needed
Pilot Experiment: TIDIGITS

\[
P(W | X) \approx \sum_{\Phi} P(W | \Phi)P(\Phi | X)
\]

\[
= \sum_{\Phi} \frac{P(\Phi | W)P(W)}{P(\Phi)} P(\Phi | X)
\]

- Important characteristic of the DIGITS problem – a given phone sequence maps to a single word sequence
- \(P(W|\Phi)\) easy to implement as FSTs in this problem
Pilot Experiment: TIDIGITS

- Implementation
  - Created a composed dictionary and language model FST
    - No probabilistic weights applied to these FSTs – assumption of uniform probability of any digit sequence
  - Modified CRF code to allow composition of above FST with phone lattice
    - Results written to MLF file and scored using standard HTK tools
    - Results compared to HMM system trained on same features
Pilot Experiment: TIDIGITS

- Features
  - Choice of multi-state model for CRF may not be best fit with neural network posterior outputs
    - The neural network abstracts away distinctions among different parts of the phone across time (by design)
  - Phone Classification (Gunawardana et al., 2005)
    - Feature functions designed to take MFCCs, PLP or other traditional ASR inputs and use them in CRFs
    - Gives the equivalent of a single Gaussian per state model
    - Fairly easy to adapt to our CRFs
Pilot Experiment: TIDIGITS

- Labels
  - Unlike TIMIT, TIDIGITS files do not come with phone-level labels
  - To generate these labels for CRF training, weights derived from TIMIT were used to force align a state-level transcript
  - This label file was then used for training the CRF
## Pilot Experiment: Results

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>HMM (monophone, 1 Gaussian)</td>
<td>98.74%</td>
</tr>
<tr>
<td>HMM (triphone, 1 Gaussian)</td>
<td>98.45%</td>
</tr>
<tr>
<td>HMM (monophone, 32 Gaussians)</td>
<td>99.93%</td>
</tr>
<tr>
<td>HMM (triphone, 32 Gaussians)</td>
<td>99.82%</td>
</tr>
<tr>
<td>CRF</td>
<td>98.81%</td>
</tr>
</tbody>
</table>

- CRF Performance falls in line with the single Gaussian models
  - CRF with these features achieves ~63% accuracy on TIMIT phone task, compared to ~69% accuracy of triphone HMM, 32 models
  - These results may not be the best we can get for the CRF – still working on tuning the learning rate and trying various realignments
Features – Part II

- Tandem systems often concatenate phone posteriors with MFCCs or PLPs for recognition
  - We can incorporate those features here as well
  - This is closer to our original experiments, though we did not use the PLPs directly in the CRF before
  - These results show phone posteriors trained on TIMIT and applied to TIDIGITS – MLPs were not been retrained on TIDIGITS
  - Experiments are still running, but I have preliminary results
## Pilot Experiment: Results

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>HMM (monophone, 1 Gaussian)</td>
<td>XX</td>
</tr>
<tr>
<td>HMM (triphone, 1 Gaussian)</td>
<td>XX</td>
</tr>
<tr>
<td>HMM (monophone, 32 Gaussians)</td>
<td>XX</td>
</tr>
<tr>
<td>HMM (triphone, 32 Gaussians)</td>
<td>99.75%</td>
</tr>
<tr>
<td>CRF</td>
<td>98.99%</td>
</tr>
</tbody>
</table>

- CRF performance increases over just using raw PLPs, but not by much
  - HMM performance has a slight, but insignificant degradation from just using PLPs alone
  - As a comparison – for phone recognition with these features the HMM achieves 71.5% accuracy, the CRF achieves 72% accuracy
- Again – results have not had full tuning. I strongly suspect that in this case the learning rate for the CRF is not well tuned, but these are preliminary numbers
Pilot Experiment: What Next?

- Continue gathering results on TIDIGITS trials
  - Experiments currently running examining different features, as well as the use of transition feature functions
  - Consider ways of getting that missing information to bring the results closer to parity with 32 Gaussian HMMs (e.g. more features)

- Work on the $P(W|\Phi)$ model
  - Computing probabilities – best way to get $P(\Phi)$?
  - Building and applying LM FSTs to an interesting test

- Move to a more interesting data set
  - WSJ 5K words is my current thought in this regard
Discussion
References

Background – Discriminative Models

- Directly model the association between the observed features and labels for those features
  - e.g. neural networks, maximum entropy models
  - Attempt to model boundaries between competing classes

- Probabilistic discriminative models
  - Give conditional probabilities instead of hard class decisions
  - Find the class \( y \) that maximizes \( P(y|x) \) for observed features \( x \)
Background – Sequential Models

- Used to classify sequences of data
  - HMMs the most common example
  - Find the most probable sequence of class labels
- Class labels depend not only on observed features, but on surrounding labels as well
  - Must determine *transitions* as well as *state* labels
Background – Sequential Models

- Sample Sequence Model - HMM
Conditional Random Fields

- A probabilistic, discriminative classification model for sequences
  - Based on the idea of Maximum Entropy Models (Logistic Regression models) expanded to sequences
Conditional Random Fields

- Probabilistic sequence model
Conditional Random Fields

- Probabilistic sequence model
  - Label sequence $Y$ has a Markov structure
  - Observed sequence $X$ may have any structure
Conditional Random Fields

- Extends the idea of maxent models to sequences
  - Label sequence $Y$ has a Markov structure
  - Observed sequence $X$ may have any structure

State functions help determine the identity of the state
Conditional Random Fields

- Extends the idea of maxent models to sequences
  - Label sequence \( Y \) has a Markov structure
  - Observed sequence \( X \) may have any structure

State functions help determine the identity of the state
Transition functions add associations between transitions from one label to another
Maximum Entropy Models

- Probabilistic, discriminative classifiers
  - Compute the conditional probability of a class $y$ given an observation $x$ – $P(y|x)$
  - Build up this conditional probability using the principle of maximum entropy
    - In the absence of evidence, assume a uniform probability for any given class
    - As we gain evidence (e.g. through training data), modify the model such that it supports the evidence we have seen but keeps a uniform probability for unseen hypotheses
Maximum Entropy Example

Suppose we have a bin of candies, each with an associated label (A, B, C, or D)

- Each candy has multiple colors in its wrapper
- Each candy is assigned a label randomly based on some distribution over wrapper colors

* Example inspired by Adam Berger’s Tutorial on Maximum Entropy
For any candy with a red label pulled from the bin:

- $P(A|\text{red})+P(B|\text{red})+P(C|\text{red})+P(D|\text{red}) = 1$
- Infinite number of distributions exist that fit this constraint
- The distribution that fits with the idea of maximum entropy is:
  - $P(A|\text{red})=0.25$
  - $P(B|\text{red})=0.25$
  - $P(C|\text{red})=0.25$
  - $P(D|\text{red})=0.25$
Maximum Entropy Example

Now suppose we add some evidence to our model

- We note that 80% of all candies with red labels are either labeled A or B
  - $P(A|\text{red}) + P(B|\text{red}) = 0.8$

- The updated model that reflects this would be:
  - $P(A|\text{red}) = 0.4$
  - $P(B|\text{red}) = 0.4$
  - $P(C|\text{red}) = 0.1$
  - $P(D|\text{red}) = 0.1$

- As we make more observations and find more constraints, the model gets more complex
Maximum Entropy Models

- “Evidence” is given to the MaxEnt model through the use of feature functions
  - Feature functions provide a numerical value given an observation
  - Weights on these feature functions determine how much a particular feature contributes to a choice of label
    - In the candy example, feature functions might be built around the existence or non-existence of a particular color in the wrapper
    - In NLP applications, feature functions are often built around words or spelling features in the text
Maximum Entropy Models

\[ P(y | x) = \frac{\exp \sum_i \lambda_i s_i(x, y)}{\sum_k \exp \sum_i \lambda_i s_i(x, y_k)} \]

- The maxent model for \( k \) competing classes
- Each feature function \( s(x,y) \) is defined in terms of the input observation \((x)\) and the associated label \((y)\)
- Each feature function has an associated weight \((\lambda)\)
Maximum Entropy – Feature Funcs.

- Feature functions for a maxent model associate a label and an observation
  - For the candy example, feature functions might be based on labels and wrapper colors
  - In an NLP application, feature functions might be based on labels (e.g. POS tags) and words in the text
Maximum Entropy – Feature Funcs.

\[ s(y, x) = \begin{cases} 
1 & \text{iff } (y = \text{NOUN}, x = "dog") \\
0 & \text{otherwise}
\end{cases} \]

- Example: MaxEnt POS tagging
  - Associates a tag (NOUN) with a word in the text ("dog")
  - This function evaluates to 1 only when both occur in combination
    - At training time, both tag and word are known
    - At evaluation time, we evaluate for all possible classes and find the class with highest probability
Maximum Entropy – Feature Funcs.

\[ s_1(y, x) = \begin{cases} 
1 & \text{iff } (y = \text{NOUN}, x = \text{"dog"}) \\
0 & \text{otherwise}
\end{cases} \]

\[ s_2(y, x) = \begin{cases} 
1 & \text{iff } (y = \text{VERB}, x = \text{"dog"}) \\
0 & \text{otherwise}
\end{cases} \]

- These two feature functions would never fire simultaneously
  - Each would have its own lambda-weight for evaluation
MaxEnt models do not make assumptions about the independence of features

Depending on the application, feature functions can benefit from context

\[ s_1(y, X, n) = \begin{cases} \frac{1}{n} & \text{iff } (y = \text{NOUN}, x_n = "dog", x_{n-1} = "my") \\ 0 & \text{otherwise} \end{cases} \]
Maximum Entropy – Feature Funcs.

- Other feature functions possible beyond simple word/tag association
  - Does the word have a particular prefix?
  - Does the word have a particular suffix?
  - Is the word capitalized?
  - Does the word contain punctuation?
- Ability to integrate many complex but sparse observations is a strength of maxent models.
Conditional Random Fields

- Feature functions defined as for maxent models
  - Label/observation pairs for state feature functions
  - Label/label/observation triples for transition feature functions
    - Often transition feature functions are left as “bias features” – label/label pairs that ignore the attributes of the observation
Conditional Random Fields

\[ s(y, y', x) = \begin{cases} 1 & \text{iff } (y = \text{NOUN}, y' = \text{DET}, x = \text{"dog"}) \\ 0 & \text{otherwise} \end{cases} \]

- Example: CRF POS tagging
  - Associates a tag (NOUN) with a word in the text ("dog") AND with a tag for the prior word (DET)
  - This function evaluates to 1 only when all three occur in combination
    - At training time, both tag and word are known
    - At evaluation time, we evaluate for all possible tag sequences and find the sequence with highest probability (Viterbi decoding)
Conditional Random Fields

- Example – POS tagging (Lafferty, 2001)
  - State feature functions defined as word/label pairs
  - Transition feature functions defined as label/label pairs
  - Achieved results comparable to an HMM with the same features

<table>
<thead>
<tr>
<th>Model</th>
<th>Error</th>
<th>OOV error</th>
</tr>
</thead>
<tbody>
<tr>
<td>HMM</td>
<td>5.69%</td>
<td>45.99%</td>
</tr>
<tr>
<td>CRF</td>
<td>5.55%</td>
<td>48.05%</td>
</tr>
</tbody>
</table>
Example – POS tagging (Lafferty, 2001)

- Adding more complex and sparse features improved the CRF performance
- Capitalization?
- Suffixes? (-iy, -ing, -ogy, -ed, etc.)
- Contains a hyphen?

<table>
<thead>
<tr>
<th>Model</th>
<th>Error</th>
<th>OOV error</th>
</tr>
</thead>
<tbody>
<tr>
<td>HMM</td>
<td>5.69%</td>
<td>45.99%</td>
</tr>
<tr>
<td>CRF</td>
<td>5.55%</td>
<td>48.05%</td>
</tr>
<tr>
<td>CRF+</td>
<td>4.27%</td>
<td>23.76%</td>
</tr>
</tbody>
</table>
Conditional Random Fields

- Based on the framework of Markov Random Fields
Conditional Random Fields

- Based on the framework of Markov Random Fields
  - A CRF iff the graph of the label sequence is an MRF when conditioned on a set of input observations (Lafferty et al., 2001)
Conditional Random Fields

- Based on the framework of Markov Random Fields
  - A CRF iff the graph of the label sequence is an MRF when conditioned on the input observations

State functions help determine the identity of the state
Conditional Random Fields

Based on the framework of Markov Random Fields

- A CRF iff the graph of the label sequence is an MRF when conditioned on the input observations

State functions help determine the identity of the state

Transition functions add associations between transitions from one label to another
Conditional Random Fields

\[
P(Y \mid X) = \frac{\exp \sum_k (\sum_i \lambda_i s_i (x, y_k) + \sum_j \mu_j t_j (x, y_k, y_{k-1}))}{Z(x)}
\]

- CRF defined by a weighted sum of state and transition functions
  - Both types of functions can be defined to incorporate observed inputs
  - Weights are trained by maximizing the likelihood function via gradient descent methods
SLaTe Experiments - Setup

- CRF code
  - Built on the Java CRF toolkit from Sourceforge
  - Performs maximum log-likelihood training
  - Uses Limited Memory BGFS algorithm to perform minimization of the log-likelihood gradient
SLaTe Experiments

- Implemented CRF models on data from phonetic attribute detectors
  - Performed phone recognition
  - Compared results to Tandem/HMM system on same data

- Experimental Data
  - TIMIT corpus of read speech
SLaTe Experiments - Attributes

- Attribute Detectors
  - ICSI QuickNet Neural Networks

- Two different types of attributes
  - Phonological feature detectors
    - Place, Manner, Voicing, Vowel Height, Backness, etc.
    - N-ary features in eight different classes
    - Posterior outputs -- \( P(\text{Place}=\text{dental} | X) \)
  - Phone detectors
    - Neural networks output based on the phone labels
  - Trained using PLP 12+deltas
Experimental Setup

- Baseline system for comparison
  - Tandem/HMM baseline (Hermansky et al., 2000)
  - Use outputs from neural networks as inputs to gaussian-based HMM system
  - Built using HTK HMM toolkit

- Linear inputs
  - Better performance for Tandem with linear outputs from neural network
  - Decorrelated using a Karhunen-Loeve (KL) transform
Background: Previous Experiments

- Speech Attributes
  - Phonological feature attributes
    - Detector outputs describe phonetic features of a speech signal
      - Place, Manner, Voicing, Vowel Height, Backness, etc.
    - A phone is described with a vector of feature values
  - Phone class attributes
    - Detector outputs describe the phone label associated with a portion of the speech signal
      - /t/, /d/, /aa/, etc.
### Initial Results (Morris & Fosler-Lussier, 06)

<table>
<thead>
<tr>
<th>Model</th>
<th>Params</th>
<th>Phone Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tandem [1] (phones)</td>
<td>20,000+</td>
<td>60.82%</td>
</tr>
<tr>
<td>Tandem [3] (phones) 4mix</td>
<td>420,000+</td>
<td>68.07%*</td>
</tr>
<tr>
<td>CRF [1] (phones)</td>
<td>5280</td>
<td>67.32%*</td>
</tr>
<tr>
<td>Tandem [1] (feas)</td>
<td>14,000+</td>
<td>61.85%</td>
</tr>
<tr>
<td>Tandem [3] (feas) 4mix</td>
<td>360,000+</td>
<td>68.30%*</td>
</tr>
<tr>
<td>CRF [1] (feas)</td>
<td>4464</td>
<td>65.45%*</td>
</tr>
<tr>
<td>Tandem [1] (phones/feas)</td>
<td>34,000+</td>
<td>61.72%</td>
</tr>
<tr>
<td>Tandem [3] (phones/feas) 4mix</td>
<td>774,000+</td>
<td>68.46%</td>
</tr>
<tr>
<td>CRF (phones/feas)</td>
<td>7392</td>
<td>68.43%*</td>
</tr>
</tbody>
</table>

* Significantly (p<0.05) better than comparable Tandem monophone system
* Significantly (p<0.05) better than comparable CRF monophone system
Feature Combinations

- CRF model supposedly robust to highly correlated features
  - Makes no assumptions about feature independence
- Tested this claim with combinations of correlated features
  - Phone class outputs + Phono. Feature outputs
  - Posterior outputs + transformed linear outputs
- Also tested whether linear, decorrelated outputs improve CRF performance
### Feature Combinations - Results

<table>
<thead>
<tr>
<th>Model</th>
<th>Phone Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRF (phone posteriors)</td>
<td>67.32%</td>
</tr>
<tr>
<td>CRF (phone linear KL)</td>
<td>66.80%</td>
</tr>
<tr>
<td><strong>CRF (phone post+linear KL)</strong></td>
<td><strong>68.13%</strong></td>
</tr>
<tr>
<td>CRF (phono. feature post.)</td>
<td>65.45%</td>
</tr>
<tr>
<td>CRF (phono. feature linear KL)</td>
<td>66.37%</td>
</tr>
<tr>
<td><strong>CRF (phono. feature post+linear KL)</strong></td>
<td><strong>67.36%</strong></td>
</tr>
</tbody>
</table>

* Significantly (p<0.05) better than comparable posterior or linear KL systems
Viterbi Realignment

- Hypothesis: CRF results obtained by using only pre-defined boundaries
  - HMM allows “boundaries” to shift during training
  - Basic CRF training process does not
- Modify training to allow for better boundaries
  - Train CRF with fixed boundaries
  - Force align training labels using CRF
  - Adapt CRF weights using new boundaries
Conclusions

- Using correlated features in the CRF model did not degrade performance
  - Extra features improved performance for the CRF model across the board
- Viterbi realignment training significantly improved CRF results
  - Improvement did not occur when best HMM-aligned transcript was used for training
Current Work - Crandem Systems

- Idea – use the CRF model to generate features for an HMM
  - Similar to the Tandem HMM systems, replacing the neural network outputs with CRF outputs
  - Preliminary phone-recognition experiments show promise
    - Preliminary attempts to incorporate CRF features at the word level are less promising
Future Work

- Recently implemented stochastic gradient training for CRFs
  - Faster training, improved results
- Work currently being done to extend the model to word recognition
- Also examining the use of transition functions that use the observation data
  - Crandem system does this with improved results for phone recognition