EXEMPLAR-BASED SPEECH RECOGNITION IN A RESCORING APPROACH

Georg Heigold, Google, USA
Joint work with Patrick Nguyen, Mitch Weintraub, Vincent Vanhoucke
Outline

- Motivation & Objectives
- Tools: Conditional Random Fields, Dynamic Time Warping, Distributed Models, ...
- Scaling it up... & Analysis of Results
- Summary
Motivation

- Today's speech recognition systems based on hidden Markov models (HMM)
- Potential limitation: "conditional frame synchronous independence"
- Possible solution: HMMs with richer topology
- Here: $k$NN/non-parametric approach
Challenges

- Exemplar-based approaches require large amounts of data and computing power:
  - Store/access data: distributed memory
  - Process (all) training data: distributed computing
- Coverage ↔ context/efficiency
- Massive but noisy data
Objectives

- Investigate word templates in the domain of massive, noisy data
- Within re-scoring framework based on CRFs
Data

Voice Search
• Search by voice: “How heavy is a rhinoceros?”

YouTube
• Audio transcriptions of videos
• Transcripts: confidence-filtered captions uploaded by users

<table>
<thead>
<tr>
<th></th>
<th>[h]</th>
<th>#Utt.</th>
<th>#Words</th>
<th>Manual transcriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voice Search</td>
<td>3k</td>
<td>3.3M</td>
<td>11.2M</td>
<td>70%</td>
</tr>
<tr>
<td>YouTube</td>
<td>4k</td>
<td>4.4M</td>
<td>40M</td>
<td>0%</td>
</tr>
</tbody>
</table>
Hypothesis Space

- Sequence of feature vectors $X = x_1, \ldots, x_T$
- Hypothesis = sequence of words with segmentation

\[
\begin{align*}
t_0 &= 0 && t_1 && t_2 = T
\end{align*}
\]

\[
\Omega = [w_1, t_0 = 0, t_1], [w_2, t_1, t_2], \ldots, [w_N, t_{N-1}, t_N = T]
\]

- Assume word-segmentations from first pass
Model

Segmental Conditional Random Field

\[ p(\Omega | X) = \exp(\lambda \sum_{n} f ([w_{n-1}, t_{n-2}, t_{n-1}]; [w_n, t_{n-1}, t_n], X)) / Z \]

- Features \( f = f_1, f_2, \ldots \) (find good ones)
- Weights \( \lambda = \lambda_1, \lambda_2, \ldots \) (estimate)
- Normalization constant \( Z \)
- Marginalize over segmentations (only training)

\[ p(W | X) = \sum_{\Omega \in W} p(\Omega | X) \]

Training

Criterion: Conditional Maximum Likelihood

\[ F(\lambda) = \log p_{\lambda}(W | X) \]

- Including \( l1 \)-regularization \(-C_1 \| \lambda \|_1 \) (sparsity) and \( l2 \)-regularization \(-C_2 \| \lambda \|_2^2 \)

- Optimization problem: \( \max_\lambda F(\lambda) \)

- Optimization by L-BFGS or Rprop

- Manual or automatic transcripts used as truth for supervised training
Rescoring

Re-scored word sequence = word sequence associated with $\hat{\Omega} = \arg\max_{\Omega} p(\Omega|X)$
Transducer-Based Representation

- Hypothesis space limited word lattice from first pass

- Features: \( f(h; [w, t, t'], x_i') \)

- Standard lattice-/transducer-based training algorithms can be used

Features: An Example

- Acoustic and language model scores from first-pass GMM/HMM (two features / weights)

- Why should we use them?
  - “Guaranteed” baseline performance at no additional cost
  - Backoff for words with little or no data
  - Add complementary but imperfect information without building full, stand-alone system
Dynamic Time Warping (DTW)

- “$k$-nearest neighbors for speech recognition”
- Metric: DTW distance $DTW(X, Y)$
- DTW distance: Euclidean distance between two sequences of vectors $X = x_1, \ldots, x_T$, $Y = y_1, \ldots, y_S$
- Use dynamic programming
- Literature: Dirk Van Compernolle, etc.
“1 feature / word”

- Hypothesis \( w, X \rightarrow Y \), templates \( Y \)
- \( kNN_v(X) \): \( k \)-nearest templates to \( X \) associated with word \( v \)

\[
f_v(w, X) = \frac{\delta(v, w)}{|kNN_v(X)|} \sum_{Y \in kNN_v(X)} DTW(X, Y)
\]

average distance between \( X \) and \( k \)-nearest templates \( Y \)

- One feature and weight per word, one active feature per word hypothesis
Templates

- Templates: instances of feature vector sequences representing a word
- Here: PLPs including HDA (and CMLLR)
- Extract from training data using forced alignment
- Ignore templates not in lattice or silence
- Imperfect because:
  - Incorrect word boundaries: 10-20%
  - Incorrect word labeling: 10-20%
  - Worse for short words like 'a', 'the',...
“1 feature / template”

- Hypothesis $w, X \rightarrow Y$, templates $Y$, scaling factor $\beta$

$$f_Y(w, X) = \exp(-\beta \, DTW(X, Y))$$

- Reduce complexity by considering word-dependent subsets of templates, e.g., templates assigned to $w$

- One feature / weight per template

- Non-linearity needed for arbitrary, non-quadratic decision boundaries
“1 feature / template”

• Properties:
  – Doesn't assume correct labeling of templates
  – Learn relevance/complementarity of each template
  – Is sparse representation

• Similar to SVMs with Gaussian kernel, in particular if using margin-based MMI
“1 feature / word” vs. “1 feature / template”

<table>
<thead>
<tr>
<th>Features</th>
<th>WER [%]</th>
<th>Voice Search</th>
<th>YouTube</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMLM</td>
<td></td>
<td>14.7</td>
<td>57.0</td>
</tr>
<tr>
<td>+ “1 feature / word”</td>
<td></td>
<td>14.3</td>
<td>56.7</td>
</tr>
<tr>
<td>+ “1 feature / template”</td>
<td></td>
<td>14.1</td>
<td>55.9</td>
</tr>
</tbody>
</table>
Adding More Context

- (Hopefully) better modeling by relaxing frame independence assumption
- More structured search space → more efficient search
- So far: acoustic unit = context
- Context may be: + preceding word, + left/right phones, + speaker information, etc.
- But: number of contexts ↔ coverage
Bigram Word Templates (YouTube)

- More templates don't help and are inefficient
- Short filler words with little context dominate ‘the’, ‘to’, ‘and’, ‘a’, ‘of’, ‘that’, ‘is’, ‘in’, ‘it’ make up 30% of words
- Consider word template in context of preceding word

<table>
<thead>
<tr>
<th>Features</th>
<th>Context</th>
<th>WER [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMLM</td>
<td>N/A</td>
<td>57.0</td>
</tr>
<tr>
<td>+ “1 feature / word”</td>
<td>unigram</td>
<td>55.9</td>
</tr>
<tr>
<td></td>
<td>bigram</td>
<td>55.0</td>
</tr>
</tbody>
</table>

- Gain from bigram discriminative LM: ~0.2%
Distributed Templates / DTW

# Scalability

<table>
<thead>
<tr>
<th></th>
<th>#Templates [M]</th>
<th>Audio [h]</th>
<th>Memory [GB]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phone</td>
<td>0.5</td>
<td>30</td>
<td>1</td>
</tr>
<tr>
<td>Triphone</td>
<td>25</td>
<td>1,500</td>
<td>45</td>
</tr>
<tr>
<td>Word</td>
<td>10</td>
<td>1,000</td>
<td>30</td>
</tr>
<tr>
<td>Word / bigram</td>
<td>20</td>
<td>2,000</td>
<td>60</td>
</tr>
<tr>
<td>Debugging</td>
<td>20</td>
<td>2,000</td>
<td>500</td>
</tr>
</tbody>
</table>

- Computation time and WER decrease from top to bottom
Sparsity

- Impose sparsity by $l_1$-regularization (cf. template selection)
- Active word templates similar to support vectors in SVMs
- Inactive templates don't need to be processed in decoding

<table>
<thead>
<tr>
<th>Active templates</th>
<th>Standalone</th>
<th>With AMLM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voice Search</td>
<td>&gt;90%</td>
<td>&lt;1%</td>
</tr>
<tr>
<td>YouTube</td>
<td>&gt;90%</td>
<td>1%</td>
</tr>
</tbody>
</table>
Data Sharpening

- Standard method for outlier detection, smoothing
- Replace original vector $x$ aligned with some HMM state by average over $k$-nearest feature vectors aligned to same HMM state
- But: breaks long-span acoustic context if on frame-level
# Data Sharpening (YouTube)

<table>
<thead>
<tr>
<th>Setup</th>
<th>WER [%]</th>
<th>Data sharpening</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>No</td>
</tr>
<tr>
<td>kNN, with oracle(^1)</td>
<td>26.1</td>
<td>20.4</td>
</tr>
<tr>
<td>kNN, all(^2)</td>
<td>62.4</td>
<td>59.5</td>
</tr>
<tr>
<td>AMLM + word templates(^3)</td>
<td>56.4</td>
<td>55.9</td>
</tr>
<tr>
<td>AMLM + bigram word templates(^3)</td>
<td>56.3</td>
<td>55.0</td>
</tr>
</tbody>
</table>

1. Classification limited to reference word with hypothesis in lattice
2. Ditto but including all reference words
3. Re-scoring on top of first-pass
DTW vs. HMM Scores

- Replace DTW by HMM scores for check
- Voice Search, triphone templates

<table>
<thead>
<tr>
<th></th>
<th>AMLM</th>
<th>+ HMM scores</th>
<th>+ DTW scores</th>
</tr>
</thead>
<tbody>
<tr>
<td>WER [%]</td>
<td>14.7</td>
<td>14.2</td>
<td>14.0</td>
</tr>
</tbody>
</table>

Summary

- Experiments for large-scale, exemplar-based speech recognition

  Up to 20 M word templates = 2,000 h waveforms = 60 GB data

- Additional context helps, data sharpening also helps...

- Only small fraction (say, 1%) of all templates needed → efficient decoding

- Modest gains: hard but realistic data conditions? unsupervised training? estimation?