Towards LVCSR System based on Structured SVMs

Austin Zhang and Mark Gales

7 November 2011

Cambridge University Engineering Department
SVMs for Continuous Speech Recognition

- SVMs generate boundaries between classes
  - For isolated digit recognition, no problem (10 classes)
    sequence kernels: map to fixed length
  - Continuous speech? too many classes! (e.g., 6 digits $\Rightarrow 10^6$ classes)

- Simplest approach:

  1. Using HMM-based Segmentation
  2. For each segment, isolated classification

Problem: Restricted to one fixed segmentation

- Alternatively, incorporate structures into SVM classes – *Structured SVM*!
Structured SVMs for CSR

• Model whole utterance – observations $O$, word sequence $w$
  - joint features $\phi(O, w)$.

• Want to learn a linear decoding rule: \[
\arg \max_w \alpha^T \phi(O, w)
\]

• Parameters $\alpha$ is learned by

\[
\min_{\alpha, \xi} \frac{1}{2} \|\alpha\|^2 + \frac{C}{n} \sum_{i=1}^{n} \xi_i
\]

s.t. for every utterance $i$, for all the competing labels $w_*^{(i)}$

\[
\alpha^T \phi(O^{(i)}, w_{\text{ref}}^{(i)}) - \alpha^T \phi(O^{(i)}, w_*^{(i)}) \geq \mathcal{L}(w_{\text{ref}}^{(i)}, w_*^{(i)}) - \xi_i
\]

correct pair competing pair loss/margin
Structured SVM \equiv \text{Large Margin Log Linear Model}

- Example of log-linear models

\[ P(w|O; \alpha) = \frac{1}{Z} \exp(\alpha^T \phi(O, w)) \]

- Margin: minimum distance between correct \( w_{\text{ref}} \) and competing \( w \)

\[
\min_{w \neq w_{\text{ref}}} \left\{ \log \left( \frac{P(w_{\text{ref}}|O; \alpha)}{P(w|O; \alpha)} \right) \right\}
\]

- Gaussian Prior \( \mathcal{N}(\alpha; 0, C\mathbf{I}) \) + Large Margin \( \Leftrightarrow \) Structured SVM

\[
\frac{1}{2} ||\alpha||_2^2 + C \sum_{i=1}^{n} \left[ \max_{w \neq w_{\text{ref}}} \left\{ \mathcal{L}(w, w_{\text{ref}}^{(i)}) - \log \left( \frac{\exp(\alpha^T \phi(O^{(i)}, w_{\text{ref}}^{(i)})}{\exp(\alpha^T \phi(O^{(i)}, w))} \right) \right\} \right] +
\]
Structured SVM ≡ Large Margin Log Linear Model

- **LM Log Linear Model: Convex Optimization**

\[
\frac{1}{2} \|\alpha\|^2 + C \sum_{r=1}^{n} \left[ -\alpha^T \phi(O^{(i)}, w^{(i)}_{\text{ref}}) + \max_{w \neq w_{\text{ref}}} \left\{ \mathcal{L}(w, w^{(i)}_{\text{ref}}) + \alpha^T \phi(O^{(i)}, w) \right\} \right]
\]

- **Training Issues**
  - If differentiable objective, standard quasi-Newton methods, e.g., LBFGS.
  - However, non-smooth objective

- **Existing approaches**
  - Sub-gradient, Extragradient algorithms
  - **Cutting-plane algorithm** (Bundle methods) → Structured SVM Training
Structured SVM Training

- Training $\alpha$, Maximize the Margin

Subject To: score of correct pair $w_{ref}$ ≥ all competing $w$

$$\min_{\alpha, \xi} \frac{1}{2}||\alpha||^2$$

S.T.

Sample 1

```
O^{(1)}, w_{ref}^{(1)}
```

$$\alpha^T \phi(\text{"Hello World"}) \geq \alpha^T \phi(\text{"Hello Dog"}) + 1,$$

$$\alpha^T \phi(\text{"Hello World"}) \geq \alpha^T \phi(\text{"Hey World"}) + 1,$$

a lot competing

$$\vdots$$

Sample n

```
O^{(n)}, w_{ref}^{(n)}
```

$$\alpha^T \phi(\text{"See you"}) \geq \alpha^T \phi(\text{"Say you"}) + 1,$$

$$\alpha^T \phi(\text{"See you"}) \geq \alpha^T \phi(\text{"Sayonara"}) + 1,$$

a lot competing

- Replace “0-1 loss” as $\mathcal{L}(w_{ref}, w)$

- Introduce slack variable $\xi_i$ for linearly nonseparable cases
Structured SVM Training

- Training $\alpha$ in QP (for simplicity ignore $\theta$):

$$\min_{\alpha, \xi} \frac{1}{2}||\alpha||^2 + \frac{C}{n} \sum_{i=1}^{n} \xi_i$$

s.t. for every utterance $i$, for ALL the competing labels $w^{(i)}$

$$\alpha^T \phi(O^{(i)}, w^{(i)}_{ref}) - \alpha^T \phi(O^{(i)}, w^{(i)}_{*}) \geq \mathcal{L}(w^{(i)}_{ref}, w^{(i)}_{*}) - \xi_i$$

- Unconstraint form

$$\frac{1}{2}||\alpha||^2 + C \sum_{r=1}^{n} \left[ - \alpha^T \phi(O^{(i)}, w^{(i)}_{ref}) + \max_{w \neq w^{(i)}_{ref}} \left\{ \mathcal{L}(w, w^{(i)}_{ref}) + \alpha^T \phi(O^{(i)}, w) \right\} \right] +$$

---

Cambridge University
Engineering Department
Toshiba Presentation
Structured SVM Training

• Training $\alpha$ in QP (for simplicity ignore $\theta$):

$$\min_{\alpha, \xi} \frac{1}{2}||\alpha||^2 + \frac{C}{n} \sum_{i=1}^{n} \xi_i$$

s.t. for every utterance $i$, for ALL the competing labels $w^{(i)}$

$$\alpha^T \phi(O^{(i)}, w^{(i)}_{\text{ref}}) - \alpha^T \phi(O^{(i)}, w^{(i)}_*) \geq \mathcal{L}(w^{(i)}_{\text{ref}}, w^{(i)}_*) - \xi_i$$

Correct pair

Competing pair

Loss controlled margin

• Handling exponential numbers of constraints - Cutting Plane Algorithm:
Structured SVM Training

• \((n\text{-slack})\) Cutting Plane Algorithm

\[
\text{repeat} \\
\text{for } i=1, \ldots, n \text{ do} \\
\quad ① \text{ constraint set } \leftarrow \text{Generate a new constraint (one } w(i), \text{ one constraint)} \\
\quad ② \alpha \leftarrow \text{Solving QP with current constraint set} \\
\text{end for} \\
\text{until no new constraints}
\]

• Note: Add \(n\) constraints every iteration!
Issues of Extending to LVCSR

• Joint feature space very large – context-dependent features
  ✔ Parameter tying

• Too many constraints! – Add $n$ constraints every iteration
  ⇒ in Aurora 4, after 20 iteration, 50,000 constraints!
  ✔ 1-Slack Cutting plane algorithm

• Converge too slow – How to speed up
  ✔ Incorporating Prior
  ✔ Parallization
  ✔ Caching and Pruning
I. Handle Large Joint Feature Space
Review Joint Feature Space

- How to build joint features $\phi(O, w)$
  
  ✔ handle continuous observations:
  
  sequence kernels $\Rightarrow$ generative model to extract features model compensation
  
  ✔ handle structured labels:
  
  introduce segmentation $\theta$, concatenate structurally
Structured SVMs for LVCSR

Joint Feature Space

• Issues: $M$ context-dependent phone models $\Rightarrow M^2$ dimension in $\phi(O, w; \theta)$
  
  – “matched” context features only
  
  $\begin{bmatrix}
  \log P(o; \lambda^{a-a+a}) \\
  \vdots \\
  \log P(o; \lambda^{a-x+b}) \\
  \vdots \\
  \log P(o; \lambda^{z-z+z})
  \end{bmatrix}
  \begin{bmatrix}
  \vdots \\
  \vdots \\
  \vdots \\
  \vdots \\
  \vdots
  \end{bmatrix}
  \rightarrow
  \begin{bmatrix}
  \log P(o; \lambda^{a-a+c}) \\
  \vdots \\
  \log P(o; \lambda^{a-b+c}) \\
  \vdots \\
  \log P(o; \lambda^{a-z+c})
  \end{bmatrix}
  \begin{bmatrix}
  \vdots \\
  \vdots \\
  \vdots \\
  \vdots \\
  \vdots
  \end{bmatrix}
  \Rightarrow
  M: \text{# of triphones}
  M_1: \text{# of mono phones}

  – model level parameter tying (using phonetic decision tree)

  $\phi(O, w; \theta)$
  \begin{bmatrix}
  \vdots \\
  \vdots \\
  \vdots \\
  \vdots \\
  \vdots
  \end{bmatrix} +
  \begin{bmatrix}
  \vdots \\
  \vdots \\
  \vdots \\
  \vdots \\
  \vdots
  \end{bmatrix}
  \rightarrow
  \begin{bmatrix}
  \vdots \\
  \vdots \\
  \vdots \\
  \vdots \\
  \vdots
  \end{bmatrix} +
  \begin{bmatrix}
  \vdots \\
  \vdots \\
  \vdots \\
  \vdots \\
  \vdots
  \end{bmatrix}
  \Rightarrow
  \begin{bmatrix}
  \vdots \\
  \vdots \\
  \vdots \\
  \vdots \\
  \vdots
  \end{bmatrix}
  \begin{bmatrix}
  \vdots \\
  \vdots \\
  \vdots \\
  \vdots \\
  \vdots
  \end{bmatrix}
  M_1M_2 + 1

  e.g., in Aurora 4, we use $M_1 = M_2 = 47$
Structured SVMs for LVCSR

\[ \alpha^T \phi(O^{(i)}, w^{(i)}_{ref}) - \alpha^T \phi(O^{(i)}, w^{(i)}_*) \geq \mathcal{L}(w^{(i)}_{ref}, w^{(i)}_*) - \xi_i \quad \forall \ i = 1 \ldots n, w^{(i)}_* \]

\( n \)-slack algorithm: Add \( n \) constraints every iteration

II. Handle “Too many constraints”

– Pruning the constraints heavily? \( \Rightarrow \) lose accuracy

– Using 1-slack Formulation
1-slack V.S. n-slack

- $n$-slack Formulation (Review):
  \[
  \min_{\alpha, \xi} \frac{1}{2} ||\alpha||^2 + C \sum_{i=1}^{n} \xi_i \quad \text{s.t. for every utterance } i, \text{ for all } w_{*}^{(i)} \ldots
  \]

- 1-slack Formulation: ($\xi = \sum_{i=1}^{n} \xi_i$)
  \[
  \min_{\alpha, \xi} \frac{1}{2} ||\alpha||^2 + \frac{C}{n} \xi \quad \text{s.t. for all sets } \{w_{*}^{(i)}\}_{i=1}^{n}
  \]
  \[
  \sum_{i=1}^{n} \alpha^T \phi(O^{(i)}, w_{\text{ref}}^{(i)}) - \sum_{i=1}^{n} \alpha^T \phi(O^{(i)}, w_{*}^{(i)}) \geq \sum_{i=1}^{n} \mathcal{L}(w_{\text{ref}}^{(i)}, w_{*}^{(i)}) - \xi
  \]

- **Equivalence**: theoretically, same solution with $n$-slack
1-slack V.S. $n$-slack

- (1-slack) Cutting Plane Algorithm

\[
\begin{align*}
\text{repeat} & \\
\text{for } i=1, \ldots, n & \text{ do} \\
\quad 1 & \text{ Generate a new } w_\ast^{(i)} \text{ (one set } \left\{w_\ast^{(i)}\right\}_{i=1}^n \text{, one constraint)} \\
\text{end for} & \\
\quad 2 & \text{ } \alpha \leftarrow \text{Solving QP with current constraint set} \\
\text{until } & \text{no new constraints}
\end{align*}
\]

- Fewer active constraints!
  - every iteration only add one constraint
  - iterations is bounded by $\frac{C}{\varepsilon}$
  - e.g., in Aurora 2, $n$-slack produce 629 active constraints
    1-slack finally produce 24 active constraints
    the QP of 1-slack in every iteration are much smaller!
III. Speed Up for LVCSR

✔ Modifying Prior
✔ Parallization
✔ Caching
Incorporating Modified Prior

- Original structured SVMs using zero mean, $C$ variance priori:

$$\log P(\alpha) = \log \mathcal{N}(\alpha; 0, CI) \propto -\frac{1}{2} ||\alpha||^2 \Rightarrow \mu_\alpha = \begin{bmatrix} 0 \\ \vdots \end{bmatrix}$$

- One proper mean of prior – yields the standard HMM system

$$\arg \max_w \mu_\alpha^T \phi(O, w) = \arg \max_w \log \left( P(O|w)^{\frac{1}{16}} P(w) \right) \Rightarrow \mu_\alpha = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 16 \end{bmatrix}$$

- $\mathcal{N}(\alpha; \mu_\alpha, \Sigma_\alpha) \Rightarrow$ New optimisation:

$$\min_{\alpha, \xi} \frac{1}{2} (\alpha - \mu_\alpha)^T \Sigma_\alpha^{-1} (\alpha - \mu_\alpha) + \frac{C}{n} \xi \quad \text{s.t. } \ldots$$
Incorporating Modified Prior

• To use cutting plane algorithm
  
  - transformed $\tilde{\alpha} = (\alpha - \mu_\alpha) \sim \mathcal{N}(0, CI)$
  
  - modified $\tilde{\mathcal{L}}(w, w^{(i)}_{\text{ref}}) = \mu_\alpha^T \left( \phi(O^{(i)}, w) - \phi(O^{(i)}, w^{(i)}_{\text{ref}}) \right) + \underbrace{\mathcal{L}(w, w^{(i)}_{\text{ref}})}_{\text{transcription loss}}$
    
    \hspace{2cm} + \underbrace{\mathcal{L}(w, w^{(i)}_{\text{ref}})}_{\text{acoustic and language loss}}$

• Same optimisation form, for transformed $\tilde{\alpha}$:

  \[
  \min_{\tilde{\alpha}, \xi} \frac{1}{2} \| \tilde{\alpha} \|^2 + \frac{C}{n} \xi \quad \text{s.t.} \quad \cdots \cdots \geq \tilde{\mathcal{L}}(w, w^{(i)}_{\text{ref}}) \cdots
  \]

• If $\mu_\alpha$ (mean of prior) is good enough, we can use small $C$ (variance)
  
  - bound of iterations is $\propto \frac{C}{\varepsilon} \Rightarrow$ converge quicker
III. Speed Up for LVCSR

✔ Incorporating Prior
✔ Parallization
✔ Caching
Parallization

• Search for most competing labels (most expensive part)

for \( i = 1, \ldots, n \) do
  \( 1. \) Generate a new \( w_{*}^{(i)} \) ← \( \max_{w \in \mathcal{L}^{\text{den}}} \left\{ \mathcal{L}(w_{\text{ref}}, w) + \alpha^{T} \phi(O^{(i)}, w) \right\} \)
end for

– score-marked lattice:

• Parallization
  – for \( n \)-slack, update \( \alpha \) after every new constraint, \( w_{*}^{(i)} \) have to be modified to batch update \( \Rightarrow \) lose performance
  – for \( 1 \)-slack, update \( \alpha \) after every new constraint, \( \left\{ w_{*}^{(i)} \right\}_{i=1}^{n} \) easy to parallel, up to \( n \) threads
III. Speed Up for LVCSR

✔ Incorporating Prior
✔ Parallization
✔ Caching
Caching

• Need to compute $\phi(O^{(i)}, w^{(i)}_*)$ from lattices for all constraints.
  
  – $n$-slack: $w^{(i)}_*$ must differ for all iterations $\Rightarrow$ no point to cache
  – 1-slack: $w^{(i)}_*$ in previous iteration may be repeated $\Rightarrow$ useful to cache

• For example, in 1-slack, $w^{(i)}_*$ repeated

  Iter1 : \[
  \begin{cases}
  w^{(1)}_* = \text{The cat sat on mat} \\
  w^{(n)}_* = \text{Hello World}
  \end{cases}
  \]

  Iter2 : \[
  \begin{cases}
  w^{(1)}_* = \text{The rat sat on mat} \\
  w^{(n)}_* = \text{Hello Word}
  \end{cases}
  \]

  Iter3 : \[
  \begin{cases}
  w^{(1)}_* = \text{The rat sat on mat} \\
  w^{(n)}_* = \text{Hello World}
  \end{cases}
  \]

  Iter4 : \[
  \begin{cases}
  w^{(1)}_* = \text{The rat sat on mat} \\
  w^{(n)}_* = \text{Hey World}
  \end{cases}
  \]

• For every utterance $i$, caching 10 most recently used $\phi(O^{(i)}, w^{(i)}_*)$
  
  – First search $\phi(O^{(i)}, w^{(i)}_*)$ in cache, if fail then search lattice
Other Practical Issues
Practical Issues

• Restart QP from previous solution
  In every iteration, solve QP from previous solution as starting point, instead of from scratch.

• Restart Training
  – gradient-based algorithm, from previous solution + accumulator
  – cutting plane algorithm, from previous solution + ALL previous constraints

• Pruning
  If constraints added in early iteration are not active for 50 iterations, removed
Overall Framework

- Lattice-based framework:

- Loop 2 is parallelized training of original structured SVM
AURORA 2 Results

- Noise corrupted continuous digit task. SNRs from 0dB–20dB.

<table>
<thead>
<tr>
<th>Model</th>
<th>Set A</th>
<th>Set B</th>
<th>Set C</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>HMM</td>
<td>9.8</td>
<td>9.1</td>
<td>9.5</td>
<td>9.5</td>
</tr>
<tr>
<td>LLM (CML)</td>
<td>8.1</td>
<td>7.7</td>
<td>8.3</td>
<td>8.1</td>
</tr>
<tr>
<td>SSVM (n-slack)</td>
<td>7.6</td>
<td>7.2</td>
<td>8.0</td>
<td>7.5</td>
</tr>
<tr>
<td>SSVM (1-slack)</td>
<td>7.6</td>
<td>7.3</td>
<td>7.9</td>
<td>7.5</td>
</tr>
<tr>
<td>SSVM (prior,1-slack)</td>
<td>7.5</td>
<td>7.1</td>
<td>7.9</td>
<td>7.4</td>
</tr>
</tbody>
</table>

Table 1: All the features are appended log likelihood features $\phi_a^0$

- Gain over VTS-compensated HMMs: **22%**

- 1-slack V.S. $n$-slack: similar result, **fewer memory required**

- Using General Gaussian prior: similar result, **converge much quicker**
Structured SVMs for LVCSR

**AURORA 4 Results**

- Noise corrupted medium to large vocabulary task.

<table>
<thead>
<tr>
<th>Model</th>
<th>Criterion</th>
<th>Test Set WER (%)</th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>HMM</td>
<td>ML</td>
<td>7.1</td>
<td>15.3</td>
</tr>
<tr>
<td>LLM</td>
<td>CML</td>
<td>7.2</td>
<td>14.7</td>
</tr>
<tr>
<td></td>
<td>MPE</td>
<td>7.7</td>
<td>14.4</td>
</tr>
<tr>
<td>SSVM</td>
<td>LM (priori, 1-slack)</td>
<td>7.5</td>
<td>14.3</td>
</tr>
</tbody>
</table>

Table 2: All the features are appended log likelihood features $\phi^0_a$

- Gain over VTS-compensated HMMs: 6%

- n-slack algorithm cannot be applied, too many constraints!

- Converge too slow without a proper priori
Conclusion

- Structured SVMs ⇔ Large Margin Log Linear Models

- Extending structured SVMs to LVCSR
  - Parameter tying for context-dependent joint feature spaces
  - 1-Slack cutting plane algorithm
  - Modifying prior
  - Parallization
  - Caching and Pruning
  - Restart training capability

- Future Work
  - optimal segmentation for LVCSR
  - Kernelization