Deep Neural SVM for Speech Recognition

Shi-Xiong Zhang, Chaojun Liu, Kaisheng Yao and Yifan Gong

Microsoft, Redmond

April 21, 2015, ICASSP
Outline

**Motivation**
- Neural Networks vs Support Vector Machines

**Deep Neural SVM**
- Max-Margin Training (Frame-level & Sequence-level)
- Decoding

**Experiments**
Motivation
Support Vector Machines (binary / multiclass / sequence)

SVMs generate linear boundaries in a feature space $x \mapsto \phi(x)$

$$y = w^T \phi(x)$$
Support Vector Machines (binary / multiclass / sequence)

SVMs generate linear boundaries in a feature space $x \mapsto \phi(x)$

$$y = w^T \phi(x) = \sum_i \alpha_i k(x, x_i)$$
Neural Networks

- Too many local minima
- Tend to overfit
- Fixed model size
- Deep?

Support Vector Machines

- Convex
- Max Margin
- Automatic model size (support vectors)

\[ h(x) = \sum \alpha_i k(x, x_i) \]
Deep Neural SVM
Softmax layer of DNN

Output of softmax

\[ P(s_t|o_t) = \frac{\exp (\mathbf{w}_{s_t}^T \mathbf{h}_t)}{\sum_{s_t=1}^{N} \exp (\mathbf{w}_{s_t}^T \mathbf{h}_t)} \]

Normalization term is independent of states

\[ \text{arg max } \log P(s|o_t) = \text{arg max } \mathbf{w}_s^T \mathbf{h}_t \]
**Softmax in DNN**

$$\text{arg max}_s \log P(s|o_t) = \text{arg max}_s \mathbf{w}_s^T h_t$$

**Classification in SVM**

$$\text{arg max}_s \mathbf{w}_s^T \phi(o_t)$$
The architecture of Deep Neural SVMs

DNN → Multiclass SVM → Sequence SVM → time sequence
Frame-level Max-Margin Training
1\textsuperscript{st} Step: training last layer

Maximize the Margin

S.T. score of ground true $s_t \geq$ all competing $\bar{s}_t$
1st Step: training last layer  $\equiv$  Multiclass SVM

Maximize the Margin

S.T.  \( \text{score of ground true } s_t \geq \text{ all competing } \bar{s}_t \)

\[
\min_{w_s, \xi_t} \frac{1}{2} \sum_{s=1}^{N} \|w_s\|_2^2 + C \sum_{t=1}^{T} \xi_t^2 \\
s.t. \text{ for every training frame } t = 1, \ldots, T, \text{ for every competing states } \bar{s}_t \in \{1, \ldots, N\} : \quad w_{s_t}^T h_t - w_{\bar{s}_t}^T h_t \geq 1 - \xi_t, \quad \bar{s}_t \neq s_t
\]
2\textsuperscript{nd} Step: training previous layers

\[ \frac{\partial F}{\partial n} \]

\[
\begin{align*}
\min_{w_s, \xi_t} & \quad \frac{1}{2} \sum_{s=1}^{N} \|w_s\|^2_2 + C \sum_{t=1}^{T} \xi_t^2 \\
\text{s.t.} & \quad \text{for every training frame } t = 1, \ldots, T, \\
& \quad \text{for every competing states } \bar{s}_t \in \{1, \ldots, N\} : \\
& \quad w_{s_t}^T h_t - w_{\bar{s}_t}^T h_t \geq 1 - \xi_t, \quad \bar{s}_t \neq s_t
\end{align*}
\]
2nd Step: training previous layers

\[
\frac{\partial F}{\partial n} = \left( \frac{\partial F}{\partial h_t} \right)^T \left( \frac{\partial h_t}{\partial n} \right)
\]

Important! Same as DNN 😊
2nd Step: training previous layers

Important!

Only the support vectors have gradient! 😊
Sequence-level Max-Margin Training

DNN

Sequence SVM

time sequence
1\textsuperscript{st} Step: training last layer

\[ \log P(S_{1:T} | O_{1:T}) \]

reference state sequence \( S \)

Margin

competing state sequence \( \overline{S} \)
1st Step: training last layer

\[ \mathcal{F} = \min_S \left\{ \log \frac{P(S|O)}{P(S)} \right\} = \min_S \left\{ \log \frac{P(O|S)P(S)}{P(O|\bar{S})P(S)} \right\} \]

- **reference state sequence** \( S \)
- **competing state sequence** \( \bar{S} \)

\( \log P(S_{1:T}|O_{1:T}) \)
1st Step: training last layer

\[ \mathcal{F} = \min_{\mathcal{S}} \left\{ \log \frac{P(\mathcal{S}|\mathcal{O})}{P(\mathcal{O}|\mathcal{S})} \right\} = \min_{\mathcal{S}} \left\{ \log \frac{P(\mathcal{O}|\mathcal{S})P(\mathcal{S})}{P(\mathcal{O}|\mathcal{S})P(\mathcal{S})} \right\} \]

\[ \sum_{t=1}^{T} w_{st}^T h_t - \log P(s_t) + \log P(s_t | s_{t-1}) \]
1st Step: training last layer

\[ F = \min_{\mathbf{s}} \left\{ \log \frac{P(\mathbf{s}|\mathbf{O})}{P(\mathbf{O}|\mathbf{s})} \right\} = \min_{\mathbf{s}} \left\{ \log \frac{P(\mathbf{O}|\mathbf{S})P(\mathbf{S})}{P(\mathbf{O}|\mathbf{S})P(\mathbf{S})} \right\} \]

\[ \sum_{t=1}^{T} \mathbf{w}_{st}^T \mathbf{h}_t - \log P(s_t) + \log P(s_t|s_{t-1}) \]
1st Step: training last layer \( \equiv \) Struct SVM

\[
\begin{align*}
\mathbf{w}_j^T \mathbf{h}_t \\
\mathbf{w}_1^T \mathbf{h}_t
\end{align*}
\]

Each path defines a feature space \( \phi(O, S) \)

\[
\begin{align*}
\mathbf{w}_{ij} \log P(s_t | s_{t-1})
\end{align*}
\]
2\textsuperscript{nd} Step: training previous layers

\[
F = \min_S \left\{ \log \frac{P(S|O)}{P(S|\bar{O})} \right\} = \min_S \left\{ \log \frac{P(O|S)P(S)}{P(O|\bar{S})P(\bar{S})} \right\}
\]
2nd Step: training previous layers

\[
\frac{\partial F}{\partial n} = \left( \frac{\partial F}{\partial h_t} \right)^T \left( \frac{\partial h_t}{\partial n} \right)
\]

Important! Only the support vectors have gradient! 😊
Decoding
Decoding of Deep Neural SVMs

\[ \mathbf{w}_j^T \mathbf{h}_t \]

\[ \mathbf{w}_2^T \mathbf{h}_t \]

\[ \mathbf{w}_1^T \mathbf{h}_t \]

states

\[ w_{ij} \log a_{ij} \]

time
Experiments
## Experiments

**TIMIT: Continuous Phone Recognition (3 states for each 61 monophones)**

<table>
<thead>
<tr>
<th>GMM</th>
<th>DNN</th>
<th>DNSVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>ML</td>
<td>CE</td>
<td>Frame Max Margin</td>
</tr>
<tr>
<td>31.0%</td>
<td>22.9%</td>
<td>22.0%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sequence Max Margin</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

8% improvement over DNN
<table>
<thead>
<tr>
<th>DNSVM (frame-level)</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Last layer only</td>
<td>22.03%</td>
<td></td>
</tr>
<tr>
<td>+previous layers</td>
<td>21.95%</td>
<td></td>
</tr>
</tbody>
</table>

- Most of gains come from last layer

<table>
<thead>
<tr>
<th>DNSVM (Sequence-level)</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>acoustic only</td>
<td>21.38%</td>
<td></td>
</tr>
<tr>
<td>joint learn with LM</td>
<td>21.04%</td>
<td></td>
</tr>
</tbody>
</table>

- Joint learn with LM yields 1.6% improvement
Conclusion

Deep Neural SVM

NN $\Rightarrow$ Deep NN
$\downarrow$ $\downarrow$
SVM $\Rightarrow$ Deep SVM

Deep Neural SVMs for ASR
Conclusion

Deep Neural SVM

NN $\rightarrow$ Deep NN

$\downarrow$  $\downarrow$

SVM $\rightarrow$ Deep SVM

Deep Neural SVMs for ASR

Future work: Deep Neural SVM

Q&A
Complementary
Support Vector Machines (binary / multiclass / sequence)

Kernel function

\[ w^T \phi(x) = \sum_i \alpha_i k(x, x_i) \]
The architecture of Deep Neural SVM
Frame-level max-margin training (previous layers)

\[
\frac{\partial F}{\partial n} = \frac{\partial}{\partial h_t} F = \frac{1}{2} \left[ \sum_{t=1}^{T} \left( 1 + \max_{s_t} w_{s_t}^T h_t - w_{s_t}^T h_t \right)^2 \right] + C
\]

\[
F = \frac{1}{2} \left\| w \right\|_2^2 + C \sum_{t=1}^{T} \left[ 1 + \max_{s_t} w_{s_t}^T h_t - w_{s_t}^T h_t \right]^2
\]
Frame-level max-margin training (previous layers)

\[
\frac{\partial F}{\partial n} = \left( \frac{\partial F}{\partial h_t} \right)^T \left( \frac{\partial h_t}{\partial n} \right)
\]

Important! Same as DNN 😊

\[
F = \frac{1}{2} \|w\|^2 + C \sum_{t=1}^{T} \left[ 1 + \max_{\bar{s}_t} w_{\bar{s}_t}^T h_t - w_{\bar{s}_t}^T h_t \right]^2
\]
Frame-level max-margin training (previous layers)

\[
\begin{align*}
\frac{\partial \mathcal{F}}{\partial n} &= \left( \frac{\partial \mathcal{F}}{\partial h_t} \right)^T \begin{pmatrix} \frac{\partial h_t}{\partial n} \end{pmatrix} \\
\mathcal{F} &= \frac{1}{2} \| \mathbf{w} \|_2^2 + C \sum_{t=1}^T \left[ 1 + \max_{\tilde{s}_t} \mathbf{w}_{\tilde{s}_t}^T h_t - \mathbf{w}_{\tilde{s}_t}^T h_t \right]^2
\end{align*}
\]

Important! Same as DNN 😎
Frame-level max-margin training (previous layers)

\[
\frac{\partial F}{\partial n} = \left( \frac{\partial F}{\partial h_t} \right)^T \left( \frac{\partial h_t}{\partial n} \right)
\]

Important! Same as DNN 😊

\[
F = \frac{1}{2} \|w\|^2_2 + C \sum_{t=1}^{T} \left[ 1 + \max_{\tilde{s}_t} w_{\tilde{s}_t}^T h_t - w_{\tilde{s}_t}^T h_t \right]^2
\]

Only the support vectors have gradient! 😊
Frame-level max-margin training (previous layers)

\[ F = \frac{1}{2} \|w\|_2^2 + C \sum_{t=1}^{T} \left[ 1 + \max_{s_t} w_{s_t}^T h_t - w_{s_t}^T h_t \right]^2 \]

\[ \frac{\partial F}{\partial n} = \left( \frac{\partial F}{\partial h_t} \right)^T \left( \frac{\partial h_t}{\partial n} \right) \]

Important! Same as DNN
Frame-level max-margin training (previous layers)

\[
F = \frac{1}{2} \|w\|_2^2 + C \sum_{t=1}^{T} \left[ 1 + \max_{s_t} w_{s_t}^T h_t - w_{s_t}^T h_t \right]_+^2
\]

\[
\frac{\partial F}{\partial n} = \left( \frac{\partial F}{\partial h_t} \right)^T \left( \frac{\partial h_t}{\partial n} \right)
\]

Key: Same as DNN

\[
\frac{\partial F_{IMM}}{\partial h_t} = 2C \left[ 1 + w_{s_t}^T h_t - w_{s_t}^T h_t \right]_+ (w_{s_t} - w_{s_t})
\]
2\textsuperscript{nd} Step: training previous layers

\[
\frac{\partial F}{\partial h_t} = \left( \frac{\partial F}{\partial h_t} \right)^T \left( \frac{\partial h_t}{\partial n} \right)
\]

Important!

Same as DNN 😊

\[
\frac{\partial F}{\partial n} = 2C \sum_{t=1}^{T} [1 + \mathbf{w}^T_{st} h_t - \mathbf{w}^T_{st} h_t]_+ (\mathbf{w}_{\bar{s}t} - \mathbf{w}_{st})
\]

Only the support vectors have gradient! 😊
Sequence-level max-margin training (last layer)

Maximize the Margin

S.T. score of reference $S_{1:T} \geq$ all competing $\tilde{S}_{1:T}$
2nd Step: training previous layers

$$\frac{\partial F}{\partial n} = \left( \frac{\partial F}{\partial h_t} \right)^T \left( \frac{\partial h_t}{\partial n} \right)$$

Important! Same as DNN 😊

$$\frac{\partial F}{\partial h_t} = 2C \sum_{t=1}^{T} [\mathcal{L} + \text{PathScore}_\bar{s} - \text{PathScore}_s]_+ (w_{\bar{s}_t} - w_{s_t})$$

Only the support vectors have gradient! 😊
Sequence-level max-margin training (last layer) $\equiv$ Struct SVM

$$\mathcal{F} = \min_{S \neq \bar{S}} \left\{ \log \frac{P(S|O)}{P(\bar{S}|O)} \right\} = \min_{S \neq \bar{S}} \left\{ \log \frac{P(O|S)P(S)}{P(O|\bar{S})P(\bar{S})} \right\}$$

$$\sum_{t=1}^{T} w_{st}^T h_t - \log P(s_t) + \log P(s_t|s_{t-1})$$

$$w^T \phi(O, S) = \sum_{t=1}^{T} \begin{bmatrix} w_1 & \vdots & w_N \end{bmatrix}^T \begin{bmatrix} \delta(s_t = 1) h_t \\ \vdots \\ \delta(s_t = N) h_t \end{bmatrix} + \begin{bmatrix} \log P(s_t) \\ \log P(s_t|s_{t-1}) \end{bmatrix}$$