Overview

• Dependency Modelling in Speech Recognition:
  – latent variables
  – exponential family

• Augmented Statistical Models
  – Gaussian mixture models and hidden Markov models

• Support Vector Machines
  – Generative Kernels
  – maximum margin training

• Preliminary LVCSR experiments
Dependency Modelling

• Speech data is dynamic - observations are not of a fixed length

• Dependency modelling essential part of speech recognition:

\[ p(o_1, \ldots, o_T; \lambda) = p(o_1; \lambda)p(o_2|o_1; \lambda) \ldots p(o_T|o_1, \ldots, o_{T-1}; \lambda) \]

  – impractical to directly model in this form
  – make extensive use of conditional independence

• Two possible forms of conditional independence used:
  – observed variables
  – latent (unobserved) variables

• Even given dependency (form of Bayesian Network):
  – need to determine how dependencies interact
Bayesian networks

Yield conditional-independence assumptions

- round node: continuous variable;
- square node: discrete variable;
- shaded node: observable;
- no arrow: conditional independence.

Examples:

1. Factor Analysis:
   \[ p(o_t | x_t) = \mathcal{N}(o_t; C_t x_t + \mu_t^{(o)}, \Sigma_t^{(o)}) \]

2. Gaussian Mixture Model:
   \[ p(o_t | \omega_t = n) = \mathcal{N}(o_t; \mu_n, \Sigma_n) \]
Hidden Markov Model - A Dynamic Bayesian Network

(a) Standard HMM phone topology

(b) HMM Dynamic Bayesian Network

- Notation for DBNs:
  - circles - continuous variables
  - shaded - observed variables
  - squares - discrete variables
  - non-shaded - unobserved variables

- Observations conditionally independent of other observations given state.
- States conditionally independent of other states given previous states,
- Poor model of the speech process - piecewise constant state-space.
Dependency Modelling using Latent Variables

Switching linear dynamical system:
- discrete and continuous state-spaces
- observations conditionally independent given continuous and discrete state;
- exponential growth of paths, $O(N_s^T)$
  $\Rightarrow$ approximate inference required.

Multiple data stream DBN:
- e.g. factorial HMM/mixed memory model;
- asynchronous data common:
  - speech and video/noise;
  - speech and brain activation patterns.
- observation depends on state of both streams
Frames from phrase:
SHOW THE GRIDLEY’S . . .

Legend
- True
- HMM
- SLDS

- Unfortunately doesn’t currently classify better than an HMM!
Adaptive Training

- Observations conditionally independent:
  - state that generated the observation
  - continuous latent variable(s) $s$

- Latent variable:
  - represents the speaker/environment
  - various forms CMN/CVN/VTLN

- One powerful form is Speaker Adaptive Training using constrained MLLR

\[
p(O; \lambda) = \sum_{\theta \in \Theta} \int_{\mathcal{R}^n} \left( \prod_{t=1}^{T} P(\theta_t | \theta_{t-1}) |A| p(Ao_t + b | \theta_t; \lambda) \right) p(A, b | \lambda) dA db
\]

- ML/MAP estimation commonly used for $A, b$
- exact Bayesian inference intractable (at the moment)
- used in many state-of-the-art speech recognition systems
Dependency Modelling using Observed variables

- Commonly use member (or mixture) of the exponential family

\[
p(O; \alpha) = \frac{1}{\tau} h(O) \exp(\alpha^T T(O))
\]

- \(h(O)\) is the reference distribution
- \(\alpha\) are the natural parameters
- \(\tau\) is the normalisation term
- the function \(T(O)\) is a sufficient statistic.

- Hard to determine the appropriate form of statistics (\(T(O)\)) to use ...
Sufficient Statistic Example

- For the one-dimensional observation sequences $O = o_1, \ldots, o_T$ extract:
  
  - $T_1(O) = \sum_{t=2}^{T} o_t$; $T_2(O) = \sum_{t=2}^{T} o_{t-1}$
  - $T_3(O) = \sum_{t=2}^{T} o_t o_{t-1}$; $T_4(O) = \sum_{t=2}^{T} o_t^2$; $T_5(O) = \sum_{t=2}^{T} o_{t-1}^2$

- Probability (given the first observation) by

  \[
  p(o_2, \ldots, o_T | o_1; \alpha) = \exp \left( \sum_{i=1}^{5} \alpha_i T_i(O) \right) / \tau
  \]

  - $\alpha$ and $\tau$ directly found from the joint distribution of $\{o_t, o_{t-1}\}$

  \[
  \mu = \frac{1}{T-1} \begin{bmatrix} T_1(O) \\ T_2(O) \end{bmatrix}; \quad \Sigma = \frac{1}{T-1} \begin{bmatrix} T_4(O) & T_3(O) \\ T_3(O) & T_5(O) \end{bmatrix} - \mu \mu'
  \]

  - has the form of a single component single-state buried Markov model
Constrained Exponential Family

- Could hypothesise all possible dependencies and prune
  - discriminative pruning found to be useful (buried Markov models)
  - impractical for wide range (and lengths) of dependencies
- Consider constrained form of statistics
  - local exponential approximation to the reference distribution
  - $\rho^{th}$-order differential form considered (related to Taylor-series)
- Distribution has two parts
  - reference distribution defines latent variables
  - local exponential model defines statistics ($T(O)$)
- Slightly more general form is the augmented statistical model
  - train all the parameters (including the reference, base, distribution)
Augmented Statistical Models

- Augmented statistical models (related to fibre bundles)

\[
p(O; \lambda, \alpha) = \frac{1}{\tau} \tilde{p}(O; \lambda) \exp \left( \alpha' \begin{bmatrix} \nabla_{\lambda} \log(\tilde{p}(O; \lambda)) \\ \frac{1}{2!} \text{vec} (\nabla^2_{\lambda} \log(\tilde{p}(O; \lambda))) \\ \vdots \\ \frac{1}{\rho!} \text{vec} (\nabla^\rho_{\lambda} \log(\tilde{p}(O; \lambda))) \end{bmatrix} \right)
\]

- Two sets of parameters
  - \( \lambda \) - parameters of base distribution (\( \tilde{p}(O; \lambda) \))
  - \( \alpha \) - natural parameters of local exponential model

- Normalisation term \( \tau \) ensures that

\[
\int_{\mathbb{R}^n} p(O; \lambda, \alpha) dO = 1; \quad p(O; \lambda, \alpha) = \bar{p}(O; \lambda, \alpha)/\tau
\]

- can be very complex to estimate
Augmented Gaussian Mixture Model

• Use a GMM as the base distribution: \( \tilde{p}(o; \lambda) = \sum_{m=1}^{M} c_m N(o; \mu_m, \Sigma_m) \)
  - considering only the first derivatives of the means

\[
p(o; \lambda, \alpha) = \frac{1}{\tau} \sum_{m=1}^{M} c_m N(o; \mu_m, \Sigma_m) \exp \left( \sum_{n=1}^{M} P(n|o; \lambda) \alpha_n^T \Sigma_n^{-1}(o - \mu_n) \right)
\]

• Simple two component one-dimensional example:
Augmented Gaussian Mixture Model Example

- Maximum likelihood training of A-GMM on symmetric log-normal data

- 2-component base-distribution (poor model of data)
- A-GMM better model of distribution (log-likelihood -1.45 vs -1.59 GMM)
- approx. symmetry obtained without symmetry in parameters!
Augmented Hidden Markov Model

- For an HMM: \( \tilde{p}(O; \lambda) = \sum_{\theta \in \Theta} \left\{ \prod_{t=1}^{T} a_{\theta_{t-1}\theta_{t}} \left( \sum_{m \in \theta_{t}} c_{m} \mathcal{N}(o_{t}; \mu_{m}, \Sigma_{m}) \right) \right\} \)
  - The form of the statistics when an HMM used as the base distribution

\[
\nabla_{\mu_{jm}} \log \tilde{p}(O; \lambda) = \sum_{t=1}^{T} \gamma_{jm}(t) \Sigma_{jm}^{-1} (o_{t} - \mu_{jm})
\]

\[
\gamma_{jm}(t) = P(\theta_{t} = \{s_{j}, m\}|O; \lambda), \ \theta_{t} \text{ is the state/component pairing at time } t
\]

- An example higher order derivative has the form

\[
\nabla_{\mu_{in}} \nabla'_{\mu_{jm}} \log (\tilde{p}(O; \lambda)) =
\sum_{t=1}^{T} \sum_{\tau=1}^{T} \left\{ (\gamma_{\{jm, in\}}(t, \tau) - \gamma_{jm}(t) \gamma_{in}(\tau)) \Sigma_{in}^{-1} (o_{\tau} - \mu_{in})(o_{t} - \mu_{jm})' \Sigma_{jm}^{-1} \right\}
\]

where \( \gamma_{\{jm, in\}}(t, \tau) \) is the joint state/component posterior.
Augmented Model Dependencies

• If the base distribution is a mixture of members of the exponential family

\[
\tilde{p}(\mathbf{O}; \boldsymbol{\lambda}) = \prod_{t=1}^{T} \sum_{m=1}^{M} c_m \exp \left( \sum_{j=1}^{J} \lambda^{(m)}_j T^{(m)}_j (\mathbf{o}_t) \right) / \tau^{(m)}
\]

– consider a first order differential

\[
\frac{\partial}{\partial \lambda^{(n)}_k} \log (\tilde{p}(\mathbf{O}; \boldsymbol{\lambda})) = \sum_{t=1}^{T} P(n|\mathbf{o}_t; \boldsymbol{\lambda}) \left( T^{(n)}_k (\mathbf{o}_t) - \frac{\partial}{\partial \lambda^{(n)}_k} \log (\tau^{(m)}) \right)
\]

• Augmented models of this form

  – keep independence assumptions of the base distribution
  – remove conditional independence assumptions of the base model
    - the local exponential model depend on a posterior ...

• Same applies for dynamic models such as HMMs
Augmented Model Summary

- Extension to standard forms of statistical model
- Consists of two parts:
  - base distribution determines the latent variables
  - local exponential distribution augments base distribution
- Base distribution:
  - standard form of statistical model
  - examples considered Gaussian mixture models and hidden Markov models
- Local exponential distribution:
  - currently based on $\rho^{th}$-order differential form
  - gives additional dependencies not present in base distribution
- Normalisation term may be highly complex to calculate
  - maximum likelihood training may be very awkward
SVMs are a maximum margin, binary, classifier:

- related to minimising generalisation error;
- unique solution (compare to neural networks);
- may be kernelised - training/classification a function of dot-product ($x_i . x_j$).

Successfully applied to many tasks - how to apply to speech?
**Support Vector Machine Training**

- For non-linearly separable data a soft margin classifier is used: minimise

\[ \tau(w, \xi) = \frac{1}{2}||w||^2 + C \sum_{i=1}^{n} \xi_i \]

subject to \( y_i (\langle w, x_i \rangle + b) \geq 1 - \xi_i, \quad \xi_i \geq 0 \)

- two terms: \( \frac{k}{\text{margin}^2} \) and error rate bound \((C\) balances importance)

- The dual is commonly optimised (based only on \( \alpha^{\text{svm}} \))

\[
\hat{\alpha}^{\text{svm}} = \max_{\alpha^{\text{svm}}} \left\{ \sum_{i=1}^{n} \alpha_i^{\text{svm}} - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i^{\text{svm}} \alpha_j^{\text{svm}} y_i y_j (x_i, x_j) \right\}
\]

subject to \( 0 \leq \alpha_i^{\text{svm}} \leq C, \quad \sum_{i=1}^{m} \alpha_i^{\text{svm}} y_i = 0, \quad y_i \in \{-1, 1\} \) indicates the class.

\[ w = \sum_{i=1}^{n} \alpha_i^{\text{svm}} y_i x_i \]
The “Kernel Trick”

- SVM decision boundary linear in the feature-space
  - may be made non-linear using a non-linear mapping \( \phi() \) e.g.

\[
\phi \left( \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \right) = \begin{bmatrix} x_1^2 \\ \sqrt{2}x_1x_2 \\ x_2^2 \end{bmatrix}, \quad K(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle
\]

- Efficiently implemented using a Kernel: \( K(x_i, x_j) = (x_i \cdot x_j)^2 \)
SVMs, Generative Kernels and Maximum Margin Statistical Models

Handling Speech data

- Speech data has **inherent variability** in the number of samples:

<table>
<thead>
<tr>
<th>The</th>
<th>cat</th>
<th>sat</th>
<th>on</th>
<th>the</th>
<th>mat</th>
</tr>
</thead>
<tbody>
<tr>
<td>1200 frames</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( O_1 = { o_1, \ldots, o_{1200} } )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>The</th>
<th>cat</th>
<th>sat</th>
<th>on</th>
<th>the</th>
<th>mat</th>
</tr>
</thead>
<tbody>
<tr>
<td>900 frames</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( O_2 = { o_1, \ldots, o_{900} } )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Kernels can be used to map from variable to fixed length data.

- **Generative models** are an obvious candidate:
  - HMMs and GMMs handle variable length data
  - view as “mapping” sequence to a single dimension (log-likelihood)

\[
\phi (O; \lambda) = \frac{1}{T} \log (p(O; \lambda)) = \frac{1}{T} \sum_{t=1}^{T} \log p(o_t; \lambda)
\]
Generative Kernels

- SVMs can handle large dimensional data robustly:
  - higher dimensions - data more separable;
  - how to increase dimensionality?

- Have a generative model for each class: parameters $\omega_1: \lambda^{(1)}$ and $\omega_2: \lambda^{(2)}$

- Use a score-space:
  - add derivatives with respect to the model parameters
  - example is a log-likelihood ratio plus first derivative score-space:

$$\phi^{11}(O; \lambda) = \frac{1}{T} \left[ \log \left( p(O; \lambda^{(1)}) \right) - \log \left( p(O; \lambda^{(2)}) \right) \right]$$

$$\nabla_{\lambda^{(1)}} \log \left( p(O; \lambda^{(1)}) \right)$$

$$- \nabla_{\lambda^{(2)}} \log \left( p(O; \lambda^{(2)}) \right)$$

- dimensionality of feature-space: $1 + \text{parameters } \lambda^{(1)} + \text{parameters } \lambda^{(2)}$
Score-Space Metrics

- SVM training involves a distance from the decision boundary
  - need to determine appropriate distance metric

- Choose a maximally non-committal metric

\[ K(O_i, O_j; \lambda) = \phi(O_i; \lambda)'G^{-1}\phi(O_j; \lambda) \]

where \( O_i \) and \( O_j \) are sequences of length \( T_i \) and \( T_j \) respectively, and

\[ G = \mathcal{E} \{ (\phi(O; \lambda) - \mu_\phi)(\phi(O; \lambda) - \mu_\phi)' \} \]

where \( \mu_\phi = \mathcal{E} \{ \phi(O; \lambda) \} \).

- In practice \( G \) is usually set to be a diagonal matrix
Augmented Model Training

- Only consider simplified two-class problem

- Bayes’ decision rule for binary case (prior $P(\omega_1)$ and $P(\omega_2)$):

  $$
  \frac{P(\omega_1)\tau^{(2)}p(O; \lambda^{(1)}, \alpha^{(1)})}{P(\omega_2)\tau^{(1)}p(O; \lambda^{(2)}, \alpha^{(2)})} \begin{cases} 
  \omega_1 > 1; & \omega_2 < 1 \end{cases} \frac{1}{T} \log \left( \frac{p(O; \lambda^{(1)}, \alpha^{(1)})}{p(O; \lambda^{(2)}, \alpha^{(2)})} \right) + b \begin{cases} 
  \omega_1 > \omega_2 \geq 0 
  
  - b = \frac{1}{T} \log \left( \frac{P(\omega_1)\tau^{(2)}}{P(\omega_2)\tau^{(1)}} \right) - \text{no need to explicitly calculate } \tau 
  
  - Can express decision rule as the following scalar product

  $$
  \begin{bmatrix} w \\
  w_0 \end{bmatrix} \begin{bmatrix} \phi(O; \lambda) \\
  1 \end{bmatrix} \begin{cases} 
  \omega_1 > 1; & \omega_2 < 1 \end{cases} \begin{cases} 
  \begin{cases} 
  \omega_1 > \omega_2 \geq 0 

  \end{cases} 

  - form of score-space and linear decision boundary

  - SVM good choice as possibly high dimensional score-space
Augmented Model Training - Binary Case (cont)

- **Score-space** is given by (first order derivatives)

\[
\phi(O; \lambda) = \frac{1}{T} \left[ \log \left( p(O; \lambda^{(1)}) \right) - \log \left( p(O; \lambda^{(2)}) \right) \right]
\begin{bmatrix}
\nabla_{\lambda^{(1)}} \log \left( p(O; \lambda^{(1)}) \right) \\
\n\nabla_{\lambda^{(2)}} \log \left( p(O; \lambda^{(2)}) \right)
\end{bmatrix}
\]

- this is the generative kernel \( \phi^{11}(O; \lambda) \)
- only a function of the base-distribution parameters \( \lambda \)

- **Linear decision boundary** given by

\[
w' = \begin{bmatrix} 1 & \alpha^{(1)'} & \alpha^{(2)'} \end{bmatrix}'
\]

- only a function of the exponential model parameters \( \alpha \)

- **Bias** is represented by \( w_0 \)
- depends on both \( \alpha \) and \( \lambda \)
Estimating Model Parameters

- Two sets of parameters to be estimated using training data \( \{O_1, \ldots, O_n\} \):
  - generative models (Kernel) \( \lambda = \{\lambda^{(1)}, \lambda^{(2)}\} \)
  - SVM (Lagrange multipliers) \( \alpha_{\text{svm}} = \{\alpha_{\text{svm}}^1, \ldots, \alpha_{\text{svm}}^n\} \)
  - direction of decision boundary \( y_i \in \{-1, 1\} \) label of training example

\[
\mathbf{w} = \sum_{i=1}^{n} \alpha_{i,\text{svm}}^* y_i \mathbf{G}^{-1} \phi(O_i; \lambda)
\]

- SVM parameters trained using maximum margin training (to find \( \alpha_{\text{svm}}^* \))
- Kernel parameters may be estimated using:
  - maximum likelihood (ML) training;
  - discriminative training (e.g. maximum mutual information)
  - maximum margin (MM) training.
SVMs and Class Posteriors

- Common objection to SVMs - no probabilistic interpretation
  - use of additional sigmoidal mapping/relevance vector machines

- Generative kernels - distance from the decision boundary is the posterior ratio

\[
\frac{1}{w_1} \left( \begin{bmatrix} w \\ w_0 \end{bmatrix} \phi(O; \lambda) \right) = \frac{1}{T} \log \left( \frac{P(\omega_1|O)}{P(\omega_2|O)} \right)
\]

- \( w_1 \) is required to ensure first element of \( w \) is 1
- augmented version of the kernel PDF becomes the class-conditional PDF

- Decision boundary also yields the exponential natural parameters

\[
\begin{bmatrix} 1 \\ \alpha^{(1)} \\
\alpha^{(2)} \end{bmatrix} = \frac{1}{w_1} w = \frac{1}{w_1} \sum_{i=1}^{n} \alpha_i^{\text{svm}} y_i G^{-1} \phi(o_i; \lambda)
\]
Maximum Margin Kernel Estimation

- Using maximum margin training to estimate Kernel appealing:
  - optimising $\alpha^{svm}$ yields local exponential parameters
  - optimising $\lambda$ yields parameters of the base distribution

- Modified version of the standard SVM dual used:

$$\{\hat{\alpha}^{svm}, \hat{\lambda}\} = \arg\max_{\alpha^{svm}} \min_{\lambda} \left\{ \sum_{i=1}^{n} \alpha^{svm}_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha^{svm}_i \alpha^{svm}_j y_i y_j K(O_i, O_j; \lambda) \right\}$$

- Iterative optimisation required:
  - given values of $\lambda$ perform standard SVM training
  - given values of $\alpha^{svm}$ perform gradient descent optimisation of $\lambda$
Maximum Margin Training (detail)

- Training procedure used:
  1. Initialise parameters, $\lambda_0$, of generative model using MLE
  2. Train SVM to locate initial support vectors, $\alpha_{0}\text{svm}$
  3. Calculate initial value of objective function, $W(0) = W(\lambda_0, \alpha_{0}\text{svm})$
  4. For each iteration $k$:
     - (A) $\lambda_k = \arg \min_{\lambda} W(\lambda; \alpha_{k-1}\text{svm})$
     - (B) $\alpha_{k}\text{svm} = \arg \max_{\alpha_{\text{svm}}} W(\alpha_{\text{svm}}; \lambda_k)$
     - Recalculate objective function, $W(k) = W(\lambda_k, \alpha_{k}\text{svm})$
     Repeat until convergence: $|W(k) - W(k-1)| < \epsilon$

- (A) is a gradient descent scheme involving backing-off
  - back-off required to ensure that KKT conditions still satisfied
- (B) is standard SVM training
Maximun Margin Example

- Artificial example training class-conditional Gaussian with LLR score-space:

$$\phi(o; \lambda) = \left[ \log \left( \bar{p}(o; \lambda^{(1)}) \right) - \log \left( \bar{p}(o; \lambda^{(2)}) \right) \right]$$

- Decision boundary closer to Bayes’ decision boundary (dotted line)
  - can also be obtained by optimising $\alpha_{\text{svm}}$ using $\phi^{11}(O; \lambda)$ score-space ...
Exponential Family Base Distribution

- For a single component example the form of the augmented model is

\[
p(o; \lambda, \alpha) = \frac{1}{\tau} \exp (\lambda' T(o)) \exp (\alpha' T(o)) = \frac{1}{\tau} \exp ((\alpha + \lambda)' T(o))
\]

- still a member of the exponential family

- Using SVM training with generative kernel

\[
\phi(o; \lambda) = \begin{bmatrix}
\log (\tilde{p}(o; \lambda^{(1)})) - \log (\tilde{p}(o; \lambda^{(2)})) \\
T(o) \\
-T(o)
\end{bmatrix}
\]

- will yield a maximum margin estimate of the exponential model
- not true when using a model with latent variables
Valid Statistical Model?

- For a valid statistical model $\tau$ must be bounded:
  - for Gaussian covariance matrix must be positive-definite
- This places restrictions on the values of $\alpha$
- Consider the simplest single-dimension, Gaussian base distribution
  - score-space is LLR and first derivatives of mean and variance
  - the augmented model is also Gaussian with effective variance

$$\sigma^2 = \frac{\tilde{\sigma}^4}{\tilde{\sigma}^2 - \alpha}$$

if $\alpha \geq \tilde{\sigma}^2$ then the variance is negative!

- In practice this has not been an issue with the models examined here ...
Deterding Dataset

- Data from 11 vowels in British English in context of h*d
  - steady state portions partitioned into 6 Hamming window segments
  - linear prediction analysis to yield 10 log area parameters
  - static 10-dimensional feature vector for training/testing

- Corpus consists of
  - 48 training examples per vowel (total of 528 examples)
  - 42 test examples per vowel (total of 462 examples)

- Multi-class problem handled using set of 1-vs-1 SVM classifiers
  - single pair ties resolved using pair classifier decision
  - multiple ties resolved using the GMM classifier
## Deterding Data Experiments

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Num. Comp.</th>
<th>Training (%)</th>
<th>Test (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>initial</td>
<td>final</td>
</tr>
<tr>
<td>GMM</td>
<td>1</td>
<td>40.0</td>
<td>55.8</td>
</tr>
<tr>
<td>GMM</td>
<td>2</td>
<td>27.7</td>
<td>45.2</td>
</tr>
<tr>
<td>SVM (LLR)</td>
<td>1</td>
<td>38.1</td>
<td>1.9</td>
</tr>
<tr>
<td>SVM (LLR)</td>
<td>2</td>
<td>26.3</td>
<td>0.8</td>
</tr>
<tr>
<td>SVM (LLR + $\nabla \mu$)</td>
<td>1</td>
<td>10.6</td>
<td>1.0</td>
</tr>
</tbody>
</table>

- Maximum margin training of kernel (base distribution)
  - initial - performance using ML values for $\lambda$
  - final - performance using MM values for $\lambda$

- Use of maximum margin training improved performance
  - but overtraining clear with maximum margin training
SVMs and LVCSR

- SVMs are inherently binary:
  - speech recognition has a vast number of possible classes;
  - how to map to a simple binary problem?

- **Use pruned confusion networks:**
  - use standard HMM decoder to generate word lattice;
  - generate confusion networks (CN) from word lattice
    * gives posterior for each arc being correct;
  - prune CN to a maximum of two arcs (based on posteriors).
Incorporating Posterior Information

• Useful to incorporate arc log-posterior \((\mathcal{F}(\omega_1), \mathcal{F}(\omega_1))\) into decision process
  – posterior contains e.g. N-gram LM, cross-word context acoustic information

• Two simple approaches:
  – combination of two as independent sources (\(\beta\) empirically set)
    
    \[
    \frac{1}{T} \log \left( \frac{p(O; \lambda^{(1)}, \alpha^{(1)})}{p(O; \lambda^{(2)}, \alpha^{(2)})} \right) + b + \beta (\mathcal{F}(\omega_1) - \mathcal{F}(\omega_2)) \begin{cases} \omega_1 > 0 \\ \omega_2 < 0 \end{cases}
    \]

  – incorporate posterior into score-space (\(\beta\) obtained from decision boundary)
    
    \[
    \phi^{cn}(O; \lambda) = \begin{bmatrix} \mathcal{F}(\omega_1) - \mathcal{F}(\omega_2) \\ \phi(O; \lambda) \\ 1 \end{bmatrix}
    \]

• Incorporating in score-space requires consistency between train/test posteriors
**LVCSR Experimental Setup**

- HMMs trained on 400 hours of conversational telephone speech (fsh2004sub):
  - standard CUHTK CTS frontend (CMN/CVN/VTLN/HLDA)
  - state-clustered triphones (~6000 states, ~28 components/state);
  - maximum likelihood training
- Confusion networks generated for fsh2004sub:
  - bigram language model trained on fsh2004sub
- Perform 8-fold cross-validation on 400 hours training data:
  - matched training and test conditions
  - ML-trained Gaussian mixture model (first derivatives) score-space
  - posteriors “biased” as HMMs trained on “test” data
- Evaluation on held-out data (eva103)
  - 6 hours of test data
  - decoded using either LVCSR bigram or trigram
  - baseline using confusion network decoding
8-Fold Cross-Validation LVCSR Results

<table>
<thead>
<tr>
<th>Word Pair (examples)</th>
<th>Training</th>
<th>CN post.</th>
<th># Components</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>A/THE (8533)</td>
<td>ML</td>
<td>79.8</td>
<td>58.3</td>
</tr>
<tr>
<td></td>
<td>SVM $\phi^{ll}()$ + $\beta$ CN SVM $\phi^{cn}()$</td>
<td></td>
<td>61.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>79.8</td>
<td>79.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>80.4</td>
</tr>
<tr>
<td>CAN/CAN’T (3761)</td>
<td>ML</td>
<td>78.5</td>
<td>81.7</td>
</tr>
<tr>
<td></td>
<td>SVM $\phi^{ll}()$ + $\beta$ CN SVM $\phi^{cn}()$</td>
<td></td>
<td>84.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>78.5</td>
<td>88.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>89.0</td>
</tr>
<tr>
<td>KNOW/NO (4475)</td>
<td>ML</td>
<td>83.1</td>
<td>68.4</td>
</tr>
<tr>
<td></td>
<td>SVM $\phi^{ll}()$ + $\beta$ CN SVM $\phi^{cn}()$</td>
<td></td>
<td>72.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>83.1</td>
<td>84.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>85.7</td>
</tr>
</tbody>
</table>

- Posterior score-space best approach, maximum margin distributions useful.
Evaluation Data LVCSR Results

- Baseline performance using Viterbi and Confusion Network decoding

<table>
<thead>
<tr>
<th>Decoding</th>
<th>Language Model</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>bigram</td>
</tr>
<tr>
<td>Viterbi</td>
<td>34.4</td>
</tr>
<tr>
<td>Confusion Network</td>
<td>33.9</td>
</tr>
</tbody>
</table>

- Rescore common confusion pairs using 4-component and $\phi^{11}() + \beta$CN

<table>
<thead>
<tr>
<th>SVM Rescoring</th>
<th>#corrected/#pairs (% corrected)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>bigram LM</td>
</tr>
<tr>
<td>9 SVMs</td>
<td>44/1401 (3.1%)</td>
</tr>
<tr>
<td>15 SVMs</td>
<td>55/2116 (2.6%)</td>
</tr>
</tbody>
</table>

- $\beta$ roughly set - error rate relatively insensitive to exact value
- less than 3% of 76157 hypothesised words rescored - more SVMs required!
Summary

• Dependency modelling for speech recognition
  – use of latent variables
  – use of sufficient statistics from the data

• Augmented statistical models
  – allows simple combination of latent variables and sufficient statistics
  – use of constrained exponential model to define statistics

• Support vector machines
  – use of generative kernels for dynamic data
  – maximum margin training of augmented statistical models

• Preliminary results of a large vocabulary speech recognition task
  – SVMs/Augmented models possibly useful for speech recognition