Sequence Kernels for Speaker and Speech Recognition

Mark Gales - work with Martin Layton, Chris Longworth, Federico Flego

8 July 2009

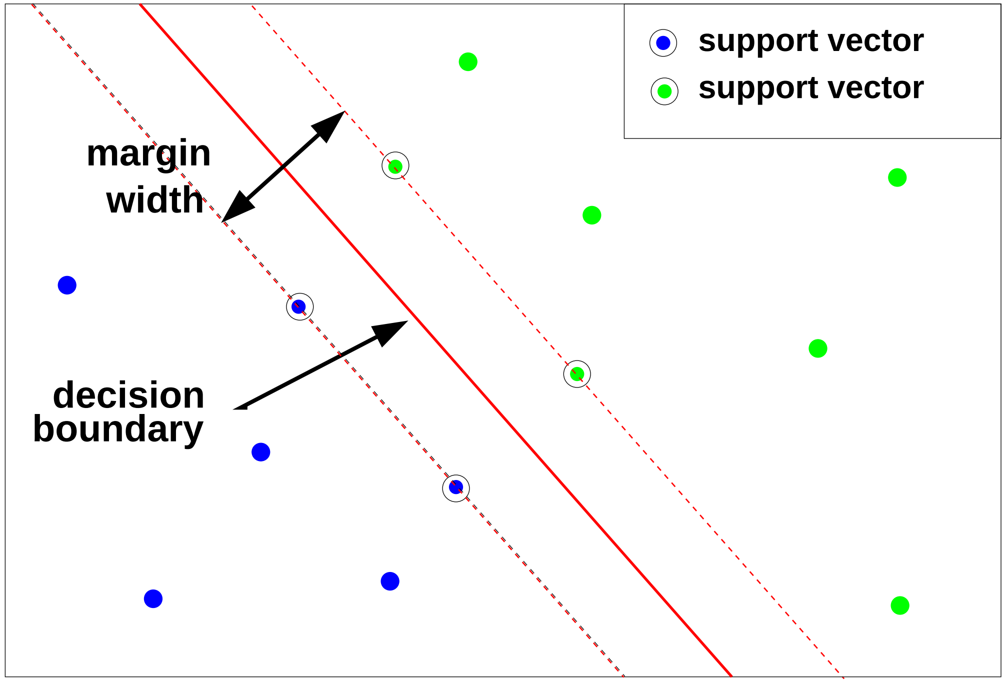
Cambridge University Engineering Department

JHU Workshop 2009

Overview

- Support Vector Machines and kernels
  - "static" kernels
  - text-independent speaker verification
- Sequence (dynamic) kernels
  - discrete-observation kernels
  - distributional kernels
  - generative kernels and scores
- Kernels and Score-Spaces for Speech Recognition
  - dependency modelling in speech recognition
  - parametric models
  - non-parametric models
- Noise Robust Speech Recognition

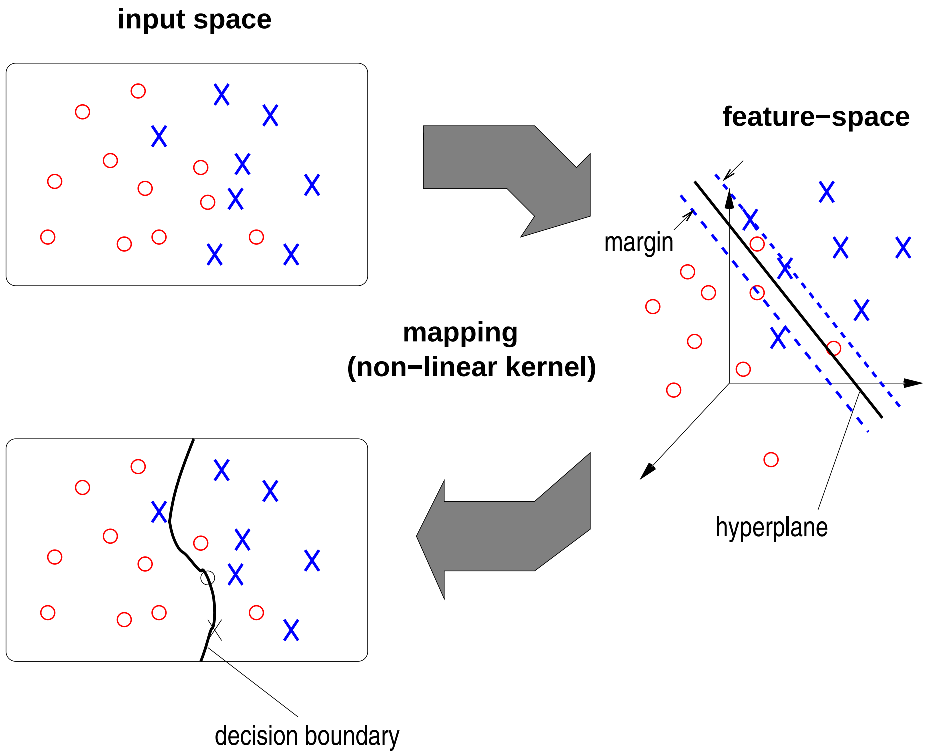
Support Vector Machines



- SVMs are a maximum margin, binary, classifier [1]:
  - related to minimising generalisation error;
  - unique solution (compare to neural networks);
  - use kernels: training/classification function of inner-product $< \boldsymbol{x}_i, \boldsymbol{x}_j >$.
- Can be applied to speech - use a kernel to map variable data to a fixed length.

The "Kernel Trick"

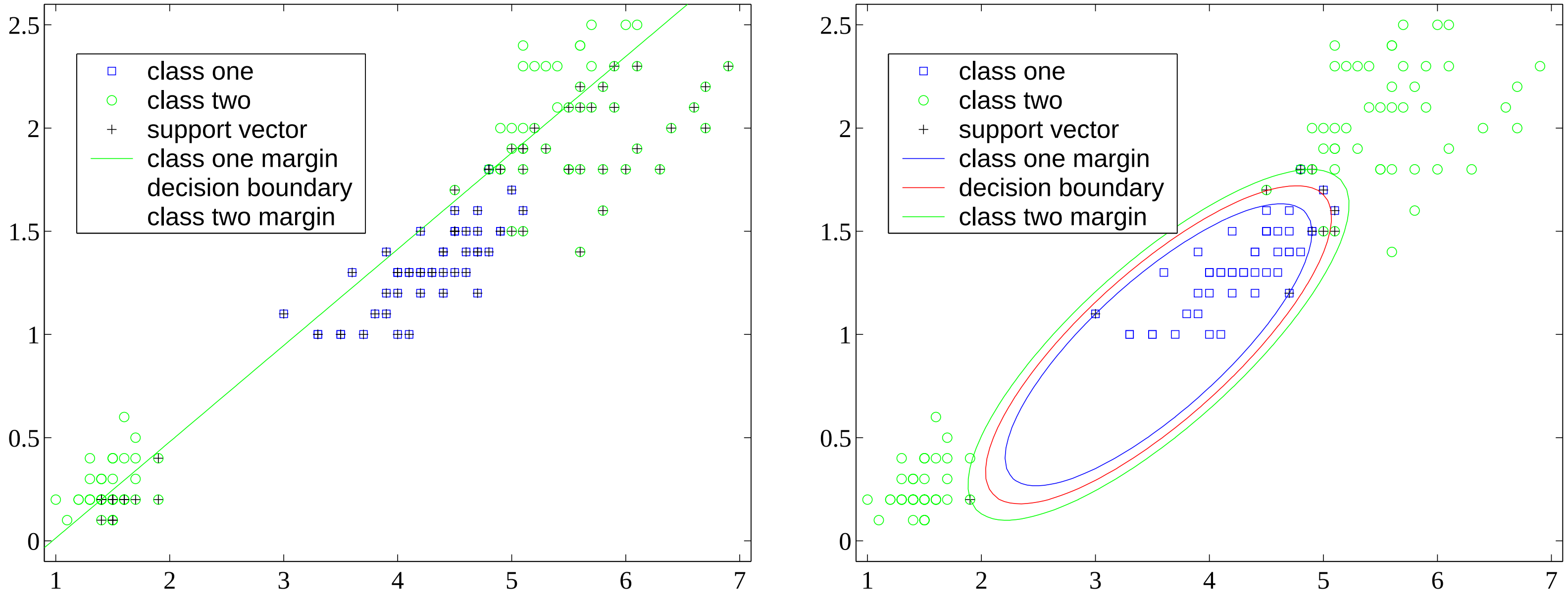
- General concept indicated below
  - a range of standard static kernels described and used in literature



- linear:
  $$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle$$
- polynomial, order d:
  $$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle + 1)^d$$
- Gaussian, width σ:
  $$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{||\boldsymbol{x}_i - \boldsymbol{x}_j||^2}{2\sigma^2}\right)$$

- Linear/non-linear transformations of fixed-length observations

Second-Order Polynomial Kernel

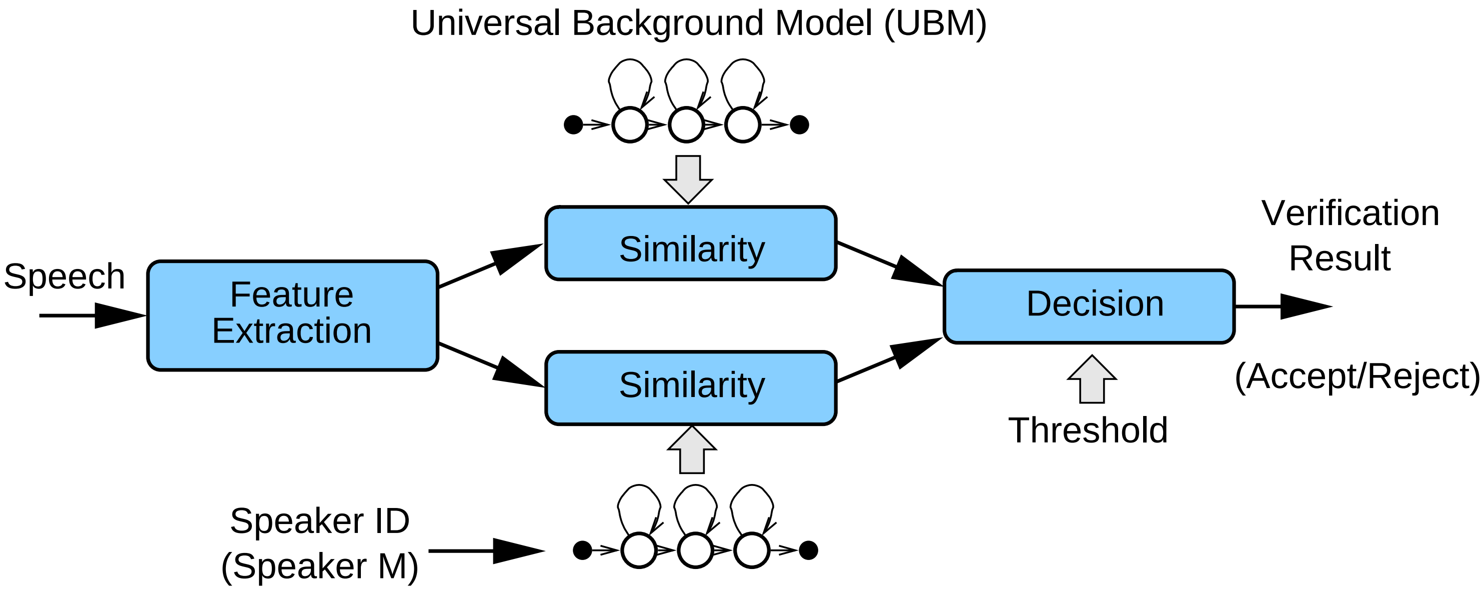


- SVM decision boundary linear in the feature-space
  - may be made non-linear using a non-linear mapping $\phi()$ e.g.

$$\phi\left(\left[\begin{array}{c} x_1 \\ x_2 \end{array}\right]\right) = \left[\begin{array}{c} x_1^2 \\ \sqrt{2}x_1x_2 \\ x_2^2 \end{array}\right], \quad K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \boldsymbol{\phi}(\boldsymbol{x}_i), \boldsymbol{\phi}(\boldsymbol{x}_j) \rangle$$

- Efficiently implemented using a Kernel: $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i.\boldsymbol{x}_j)^2$

Speaker Verification with SVMs



- GMM-based text-independent speaker verification common form used:

$$p(\boldsymbol{o};\boldsymbol{\lambda}) = \sum_{m=1}^{M} c_m \mathcal{N}(\boldsymbol{o};\boldsymbol{\mu}^{(m)},\boldsymbol{\Sigma}^{(m)})$$

  - compares likelihood from speaker model and general model (UBM)
  - how to integrate SVMs into the process [2]

Sequence Kernels

Sequence Kernel

- Sequence kernels are a class of kernel that handles sequence data
  - also applied in a range of biological applications, text processing, speech
  - in this talk a these kernels will be partitioned into three classes
- Discrete-observation kernels
  - appropriate for text data
  - string kernels simplest form
- Distributional kernels
  - distances between distributions trained on sequences
- Generative kernels:
  - parametric form: use the parameters of the generative model
  - derivative form: use the derivatives with respect to the model parameters

String Kernel

- For speech and text processing input space has variable dimension:
  - use a kernel to map from variable to a fixed length;
  - string kernels are an example for text [3].
- Consider the words `cat`, `cart`, `bar` and a character string kernel

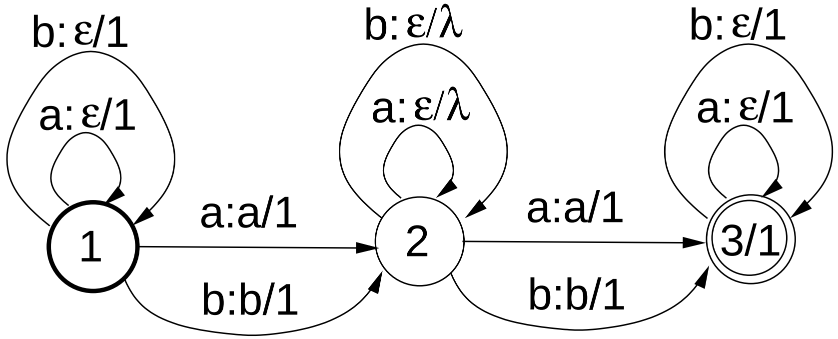
	c-a	c-t	c-r	a-r	r-t	b-a	b-r
$\phi(\texttt{cat})$	1	λ	0	0	0	0	0
$\phi(\texttt{cart})$	1	λ^2	λ	1	1	0	0
$\phi(\texttt{bar})$	0	0	0	1	0	1	λ

$$K(\texttt{cat},\texttt{cart}) = 1 + \lambda^3, \quad K(\texttt{cat},\texttt{bar}) = 0, \quad K(\texttt{cart},\texttt{bar}) = 1$$

- Successfully applied to various text classification tasks:
  - how to make process efficient (and more general)?

Rational Kernels

- Rational kernels [4] encompass various standard feature-spaces and kernels:
  - bag-of-words and N-gram counts, gappy N-grams (string Kernel),
- A transducer, T, for the string kernel (gappy bigram) (vocab $\{\mathtt{a}, \mathtt{b}\}$)



The kernel is: $K(\boldsymbol{O}_i, \boldsymbol{O}_j) = w\left[\boldsymbol{O}_i \circ (T \circ T^{-1}) \circ \boldsymbol{O}_j\right]$

- This form can also handle uncertainty in decoding:
  - lattices can be used rather than the 1-best output $(\boldsymbol{O}_i)$.
- Can also be applied for continuous data kernels [5].

Distributional Kernels

- General family of kernel that operates on distances between distributions
  - using the available estimate a distribution given the sequence

$$\boldsymbol{\lambda}^{(i)} = \underset{\boldsymbol{\lambda}}{\operatorname{argmax}} \left\{ \log(p(\mathbf{O}_i; \boldsymbol{\lambda})) \right\}$$

- Forms of kernel normally based (f_i distribution with parameters $\boldsymbol{\lambda}^{(i)}$)
  - Kullback-Leibler divergence:

$$\mathcal{KL}(f_i || f_j) = \int f_i(\mathbf{O}) \log \left(\frac{f_i(\mathbf{O})}{f_j(\mathbf{O})} \right) d\mathbf{O}$$

  - Bhattacharyya affinity measure:

$$\mathcal{B}(f_i || f_j) = \int \sqrt{f_i(\mathbf{O}) f_j(\mathbf{O})}\, d\mathbf{O}$$

GMM Mean-Supervector Kernel

- GMM-mean supervector derived from a range of approximations [6]
  - use symmetric KL-divergence: $\mathcal{KL}(f_i||f_j) + \mathcal{KL}(f_j||f_i)$
  - use matched pair KL-divergence approximation
  - GMM distributions only differ in terms of the means
  - use polarisation identity
- Form of kernel is

$$K(\mathbf{O}_i, \mathbf{O}_j; \boldsymbol{\lambda}) = \sum_{m=1}^{M} c_m \boldsymbol{\mu}^{(im)\mathsf{T}} \boldsymbol{\Sigma}^{(m)\text{-}1} \boldsymbol{\mu}^{(jm)}$$

  - $\boldsymbol{\mu}^{(im)}$ is the mean (ML or MAP) for component m using sequence $\mathbf{O}_i$
- Used in a range of speaker verification applications
  - BUT required to explicitly operate in feature-space

Generative Kernels

- Generative kernels are based on generative models (GMMs/HMMs):

$$K(\mathbf{O}_i, \mathbf{O}_j; \boldsymbol{\lambda}) = \boldsymbol{\phi}(\mathbf{O}_i; \boldsymbol{\lambda})^{\mathsf{T}} \mathbf{G}^{-1} \boldsymbol{\phi}(\mathbf{O}_j; \boldsymbol{\lambda})$$

  - $\boldsymbol{\phi}(\mathbf{O}; \boldsymbol{\lambda})$ is the score-space for $\mathbf{O}$ using parameters $\boldsymbol{\lambda}$
  - $\mathbf{G}$ is the appropriate metric for the score-space

- Parametric generative kernels use scores of the following form [7]

$$\boldsymbol{\phi}(\mathbf{O}; \boldsymbol{\lambda}) = \underset{\boldsymbol{\lambda}}{\operatorname{argmax}} \left\{ \log(p(\mathbf{O}; \boldsymbol{\lambda})) \right\}$$

  - possible to concatenate parameters of competing GMMs $\boldsymbol{\lambda} = \left\{ \boldsymbol{\lambda}^{(i)}, \boldsymbol{\lambda}^{(j)} \right\}$
  - using the appropriate metric, this is the GMM-supervector kernel

- Also possible to use different parameters derived from sequences.

  - MLLR transform kernel [8]/Cluster adaptive training kernel [9]

Derivative Generative Kernels

- An alternative score-space can be defined using

$$\boldsymbol{\phi}\left(\mathbf{O};\boldsymbol{\lambda}\right) = \boldsymbol{\nabla}_{\boldsymbol{\lambda}} \log\left(p(\mathbf{O};\boldsymbol{\lambda})\right)$$

  – using just the "UBM" same as the Fisher kernel [10]
  – can be trained on unsupervised data

- Possible to extend this using competing models: log-likelihood ratio score-space

$$\boldsymbol{\phi}(\mathbf{O};\boldsymbol{\lambda}) = \left[\begin{array}{c} \log\left(p(\mathbf{O};\boldsymbol{\lambda}^{(i)})\right) - \log\left(p(\mathbf{O};\boldsymbol{\lambda}^{(j)})\right) \\ \boldsymbol{\nabla}_{\lambda^{(i)}} \log\left(p(\mathbf{O};\boldsymbol{\lambda}^{(i)})\right) \\ -\boldsymbol{\nabla}_{\lambda^{(j)}} \log\left(p(\mathbf{O};\boldsymbol{\lambda}^{(j)})\right) \end{array}\right]$$

  – "speaker"-specific models used
  – include log-likelihood ratio in score-space
  – higher-order derivatives also possible

Derivative versus Parametric Generative Kernels

- Parametric kernels and derivative kernels are closely related [11]
- Consider gradient based optimisation

$$\boldsymbol{\lambda}^{n+1} = \boldsymbol{\lambda}^{n} + \eta\, \boldsymbol{\nabla} \log(p(\mathbf{O};\boldsymbol{\lambda}))|_{\boldsymbol{\lambda}^{n}}$$

forms become the same when:

  - learning rate η independent of $\mathbf{O}$
  - stationary kernel used: $K(\mathbf{O}_i, \mathbf{O}_j) = \mathcal{F}(\boldsymbol{\phi}(\mathbf{O}_i) - \boldsymbol{\phi}(\mathbf{O}_j))$

- Both used for speaker verification [12, 6]
  - when forms are not identical, they can be beneficially combined
- BUT derivative kernels more flexible
  - higher-order derivatives can be used
  - score-space also related to other kernels, e.g. marginalised count kernel [13]

Form of Metric

- The exact form of the metric is important
  - standard form is a maximally non-committal metric

$$\boldsymbol{\mu}_{\mathrm{g}} = \mathcal{E}\left\{\boldsymbol{\phi}(\mathbf{O};\boldsymbol{\lambda})\right\}; \quad \mathbf{G} = \boldsymbol{\Sigma}_{\mathrm{g}} = \mathcal{E}\left\{(\boldsymbol{\phi}(\mathbf{O};\boldsymbol{\lambda}) - \boldsymbol{\mu}_{\mathrm{g}})(\boldsymbol{\phi}(\mathbf{O};\boldsymbol{\lambda}) - \boldsymbol{\mu}_{\mathrm{g}})^{\mathsf{T}}\right\}$$

  - empirical approximation based on training data is often used
  - equal "weight" given to all dimensions
  - Fisher kernel with ML-trained models $\mathbf{G}$ Fisher Information Matrix
- Metric can be used for session normalisation in verification/classification
  - nuisance attribute projection: project out dimensions [14]
  - within class covariance normalisation [15] - average within class covariance

Speech Recognition

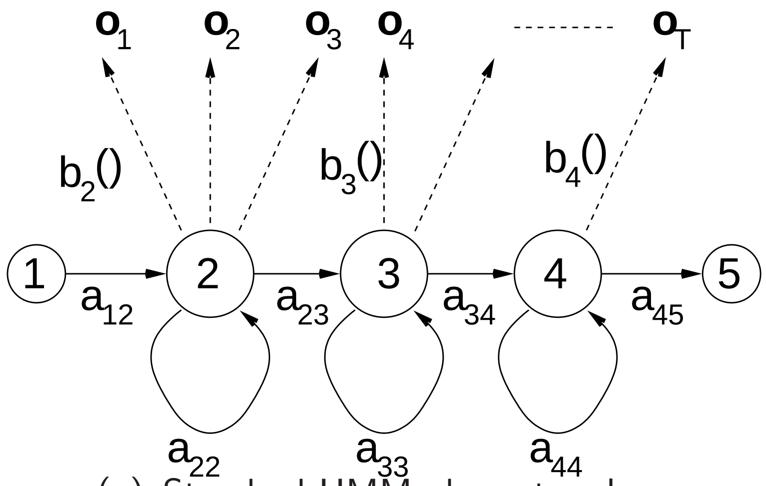
Dependency Modelling for Speech Recognition

- Sequence kernels for text-independent speaker verification used GMMs
  - for ASR interested modelling inter-frame dependencies
- Dependency modelling essential part of modelling sequence data:

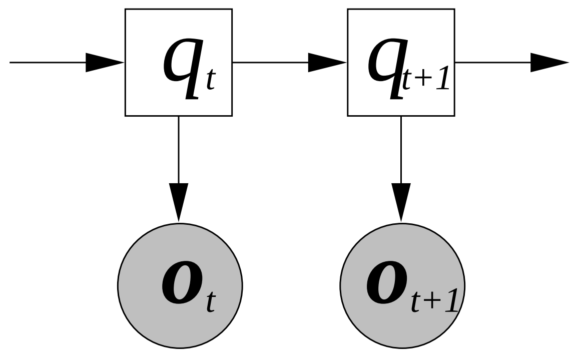
$$p(\boldsymbol{o}_1, \ldots, \mathbf{o}_T; \boldsymbol{\lambda}) = p(\boldsymbol{o}_1; \boldsymbol{\lambda}) p(\boldsymbol{o}_2 | \boldsymbol{o}_1; \boldsymbol{\lambda}) \ldots p(\boldsymbol{o}_T | \boldsymbol{o}_1, \ldots, \boldsymbol{o}_{T-1}; \boldsymbol{\lambda})$$

  - impractical to directly model in this form
- Two possible forms of conditional independence used:
  - observed variables
  - latent (unobserved) variables
- Even given dependencies (form of Bayesian Network):
  - need to determine how dependencies interact

Hidden Markov Model - A Dynamic Bayesian Network



(a) Standard HMM phone topology



(b) HMM Dynamic Bayesian Network

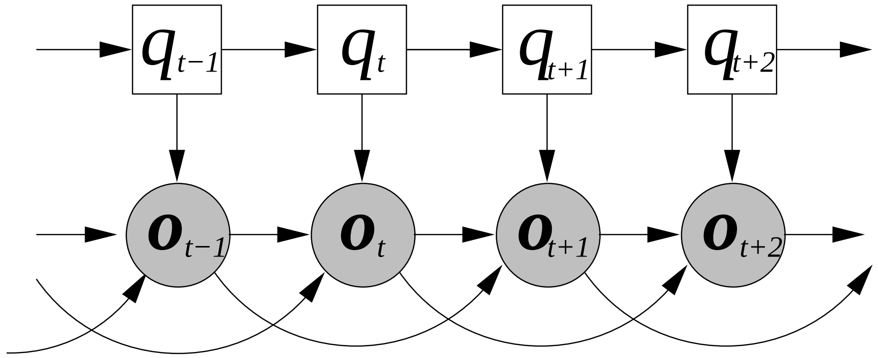
- Notation for DBNs [16]:

  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 Observed Variables



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

$$p(\mathbf{O};\boldsymbol{\alpha}) = \frac{1}{Z}h(\mathbf{O})\exp\left(\boldsymbol{\alpha}^{\mathsf{T}}\mathbf{T}(\mathbf{O})\right)$$

  - $h(\mathbf{O})$ is the reference distribution; Z is the normalisation term
  - $\boldsymbol{\alpha}$ are the natural parameters
  - the function $\mathbf{T}(\mathbf{O})$ is a sufficient statistic.

- What is the appropriate form of statistics $(\mathbf{T}(\mathbf{O}))$ - needs DBN to be known
  - for example in diagram one feature, $T(\mathbf{O}) = \sum_{t=1}^{T-2} o_t o_{t+1} o_{t+2}$

Score-Space Sufficient Statistics

- Need a systematic approach to extracting sufficient statistics
  - what about using the sequence-kernel score-spaces?

$$\mathbf{T}(\mathbf{O}) = \phi(\mathbf{O};\boldsymbol{\lambda})$$

  - does this help with the dependencies?

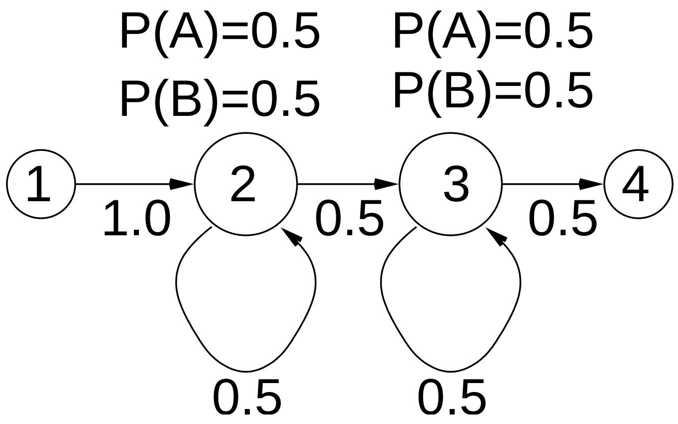
- For an HMM the mean derivative elements become

$$\boldsymbol{\nabla}_{\boldsymbol{\mu}^{(jm)}} \log(p(\mathbf{O};\boldsymbol{\lambda})) = \sum_{t=1}^{T} P(\mathbf{q}_t = \{\theta_j, m\}|\mathbf{O};\boldsymbol{\lambda})\boldsymbol{\Sigma}^{(jm)\text{-}1}(\boldsymbol{o}_t - \boldsymbol{\mu}^{(jm)})$$

  - state/component posterior a function of complete sequence $\mathbf{O}$
  - introduces longer term dependencies
  - different conditional-independence assumptions than generative model

Score-Space Dependencies

- Consider a simple 2-class, 2-symbol $\{\mathtt{A}, \mathtt{B}\}$ problem:
  - Class ω_1: AAAA, BBBB
  - Class ω_2: AABB, BBAA



Feature	Class ω_1		Class ω_2	
	AAAA	BBBB	AABB	BBAA
Log-Lik	-1.11	-1.11	-1.11	-1.11
∇_{2A}	0.50	-0.50	0.33	-0.33
$\nabla_{2A}\nabla_{2A}^{\mathsf{T}}$	-3.83	0.17	-3.28	-0.61
$\nabla_{2A}\nabla_{3A}^{\mathsf{T}}$	-0.17	-0.17	-0.06	-0.06

- ML-trained HMMs are the same for both classes
- First derivative classes separable, but not linearly separable
  - also true of second derivative within a state
- Second derivative across state linearly separable

Parametric Models with Score-Spaces

- Use the score-spaces as the sufficient statistics
  - discriminative form is the conditional augmented model [17]

$$P(\omega_i|\mathbf{O};\boldsymbol{\lambda},\boldsymbol{\alpha}) = \frac{1}{Z}\exp\left(\boldsymbol{\alpha}^{(i)\mathsf{T}}\boldsymbol{\phi}(\mathbf{O};\boldsymbol{\lambda}^{(i)})\right)$$

- Simple to apply to isolated/whole-segment models
- More difficult to extend to continuous tasks
  - one option is to consider all possible word alignments as latent variables

$$P(\omega_1,\ldots,\omega_N|\mathbf{O};\boldsymbol{\lambda},\boldsymbol{\alpha}) = \frac{1}{Z}\sum_{\mathbf{q}} P(\mathbf{q}|\mathbf{O};\boldsymbol{\lambda})\prod_{i=1}^{N}\exp\left(\boldsymbol{\alpha}^{(i)\mathsf{T}}\boldsymbol{\phi}(\mathbf{O}^{(q_i)};\boldsymbol{\lambda}^{(i)})\right)$$

Initial results interesting, but needs more work

SVMs for Noise Robust ASR

- Alternative: use non-parametric classifier such as the SVM
  - combine parametric (HMM) and non-parametric technique (SVM)
  - combine generative model (HMM) and discriminative function (SVM)
- Parametric form allows speaker/noise compensation (remove outliers)
- Non-parametric form allows longer term dependencies
  - nature of dependencies related to kernel (and order of kernel)
- Derivative generative kernels with maximally non-committal metric used here
  - LLR ratio most discriminatory - weight by ϵ (set empirically):

$$\mathcal{S}(\mathbf{O};\boldsymbol{\lambda}) + \epsilon\left(\log\left(\frac{p(\mathbf{O};\boldsymbol{\lambda}^{(i)})}{p(\mathbf{Y};\boldsymbol{\lambda}^{(j)})}\right)\right)$$

  - $\mathcal{S}(\mathbf{O};\boldsymbol{\lambda})$ is the score from the SVM for classes ω_i and ω_j

Adapting SVMs to Speaker/Noise Conditions

- Decision boundary for SVM is ($z_i \in \{-1, 1\}$ label of training example)

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i^{\mathtt{svm}} z_i \mathbf{G}^{-1} \boldsymbol{\phi}(\mathbf{O}_i; \boldsymbol{\lambda})$$

  - $\boldsymbol{\alpha}^{\mathtt{svm}} = \{\alpha_1^{\mathtt{svm}}, \ldots, \alpha_n^{\mathtt{svm}}\}$ set of SVM Lagrange multipliers

- Choice in adapting SVM to condition, modify:
  - $\boldsymbol{\alpha}^{\mathtt{svm}}$ - non-trivial though schemes have recently been proposed
  - $\boldsymbol{\lambda}$ - simple, model compensation [18]

- Approach adopted in this work is to modify generative model parameters, $\boldsymbol{\lambda}$
  - noise/speaker-independent SVM Lagrange multipliers
  - noise/speaker-dependent generative kernels

Model-Based Compensation Techniques

- A standard problem with kernel-based approaches is adaptation/robustness
  - not a problem with generative kernels
  - adapt generative models using model-based adaptation
- Standard approaches for speaker/environment adaptation
  - (Constrained) Maximum Likelihood Linear Regression [19]

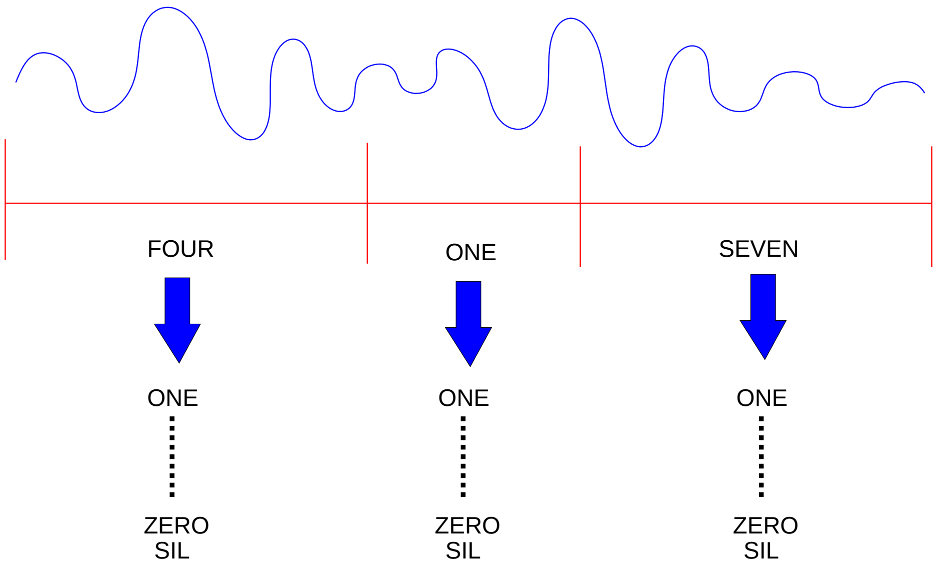
$$\boldsymbol{x}_t = \mathbf{A}\boldsymbol{o}_t + \mathbf{b}; \quad \boldsymbol{\mu}^{(m)} = \mathbf{A}\boldsymbol{\mu}_{\mathrm{x}}^{(m)} + \mathbf{b}$$

  - Vector Taylor Series Compensation [20] (used in this work)

$$\boldsymbol{\mu}^{(m)} = \mathbf{C}\log\left(\exp(\mathbf{C}^{-1}(\boldsymbol{\mu}_{\mathrm{x}}^{(m)} + \boldsymbol{\mu}_{\mathrm{h}}^{(m)})) + \exp(\mathbf{C}^{-1}\boldsymbol{\mu}_{\mathrm{n}}^{(m)})\right)$$

- Adapting the generative model will alter score-space

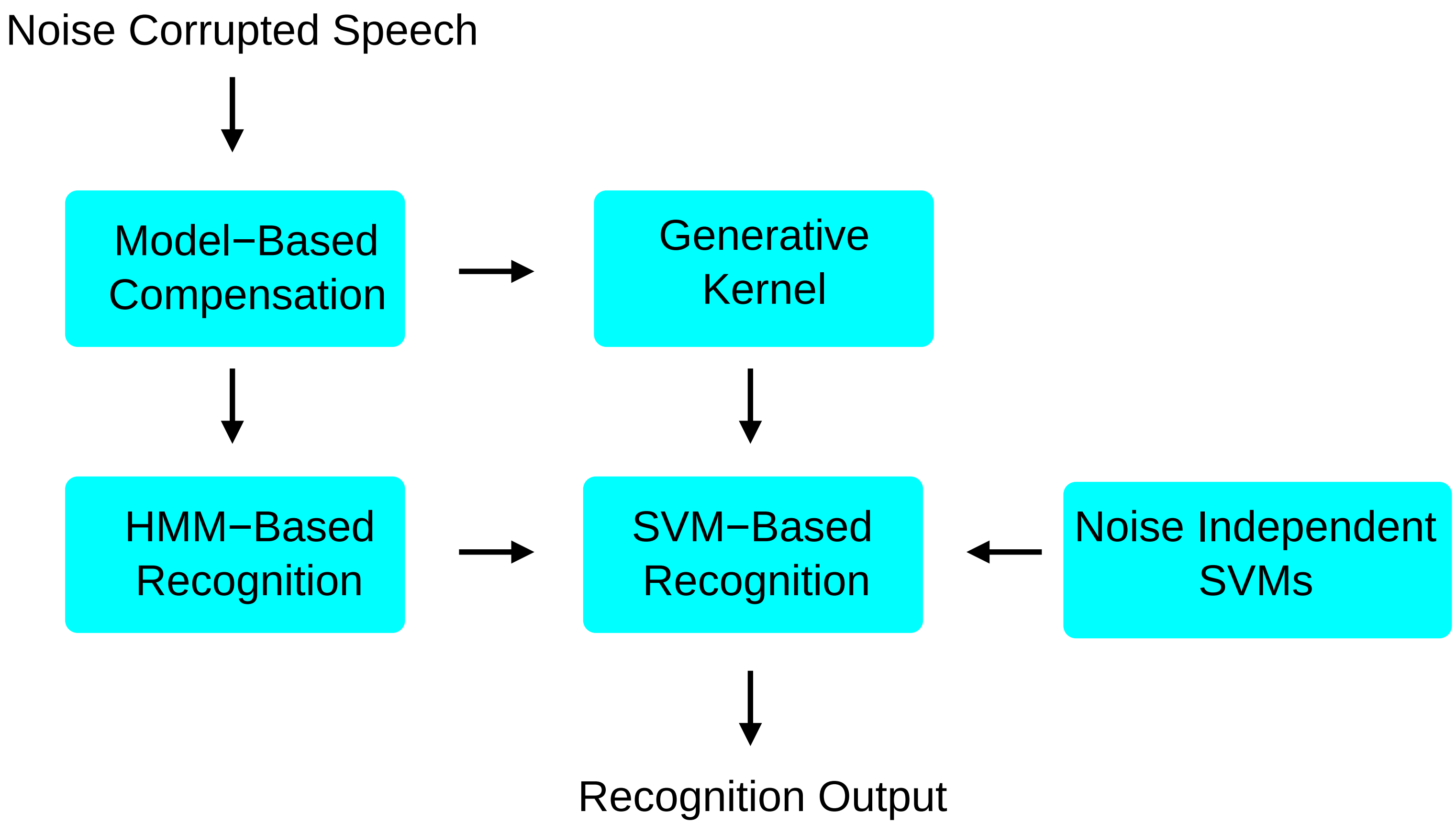
Handling Continuous Digit Strings



- Using HMM-based hypothesis
  - "force-align" - word start/end
- Foreach word start/end times
  - find "best" digit + silence
- Can use multi-class SVMs

- Simple approach to combining generative and discriminative models
  - related to acoustic code-breaking [21]
- Initial implementation uses a 1-v-1 voting SVM combination scheme
  - ties between pairs resolved using appropriate SVM output
  - > 2 ties back-off to standard HMM output

SVMs Rescoring Scheme



- Model compensation needs to "normalise" the score-spaces
  - derivative generative-kernels suited for this
  - when data "matches" models a score of zero results

Evaluation Tasks

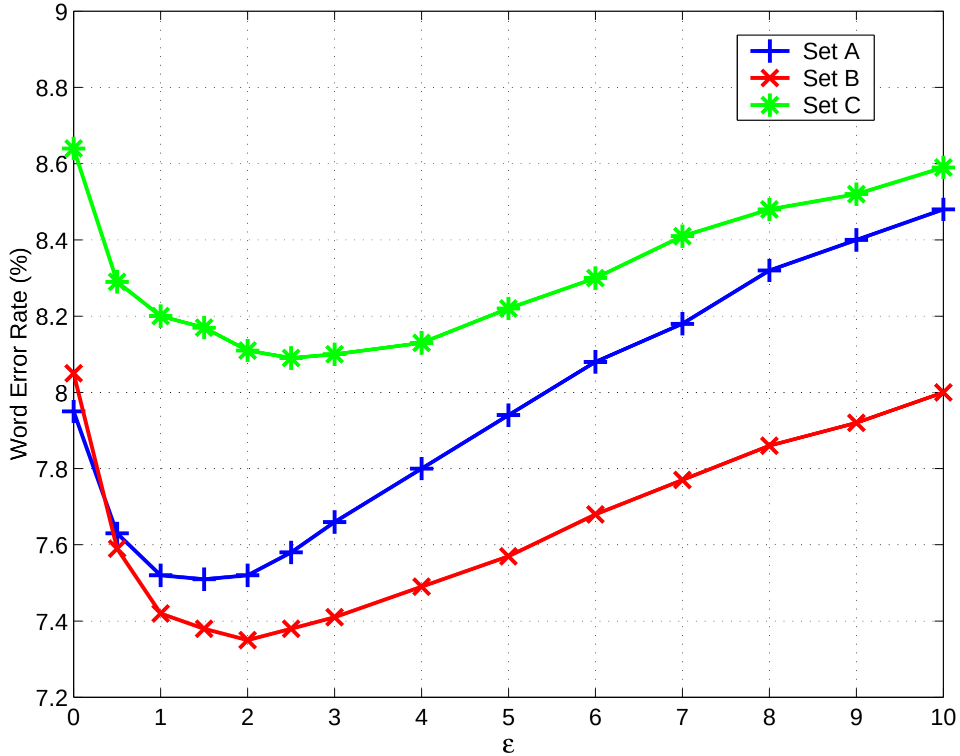
- AURORA 2 small vocabulary digit string recognition task
  - whole-word models, 16 emitting-states with 3 components per state
  - clean training data for HMM training - HTK parameterisation
  - SVMs trained on subset of multi-style data - Set A N2-N4, 10-20dB SNR
  - Set A N1 and Set B and Set C unseen noise conditions
  - Noise estimated in a ML-fashion for each utterance

- Toshiba In-Car Task
  - training data from WSJ SI284 to train clean acoustic models
  - state-clustered states, cross-word triphones (650 states ≈7k components) word-internal triphones for SVM rescoring models
  - test data collected in car (idle, city, highway), unknown length digits other test sets available, e.g. command and control
  - 35, 25, 18 SNR averages for the idle, city, highway condition, respectively
  - Noise estimated in a ML-fashion for each utterance

SVM Rescoring on AURORA 2.0

System	Test Set		
	A	B	C
VTS	9.8	9.1	9.5
+SVM	7.5	7.4	8.1

WER (%) averaged over 0-20dB



- 1-v-1 majority voting
- SVM rescoring used $\epsilon = 2$
- Large gains using SVM

- Noise-independent SVM performs well on unseen noise conditions
- Graph shows variation of performance with ϵ - $\epsilon = 0$ better than VTS

SVM Rescoring on the Toshiba Data

System	VTS iter	WER (%)		
		ENON	CITY	HWY
VTS	1	1.2	3.1	3.8
+SVM		1.3	2.6	3.2
VTS	2	1.4	2.7	3.2
+SVM		1.3	2.1	2.5

Performance on phone-number task with SVM rescoring

- More complicated acoustic models - 12 components per state
  - 1-v-1 majority voting used
- SVM rescoring shows consistent over VTS compensation
  - larger gains for lower SNR conditions (CITY and HWY)

Conclusions

- Sequence kernels are an interesting extension to standard "static" kernels
  - currently successfully applied to binary tasks such as speaker verification
- Score-spaces associates with generative kernels interesting
  - systematic way of extracting statistics from continuous data
  - different conditional independence assumptions to generative model
  - score-space/kernels can be adapted using model-based approaches
- Application of score-spaces and kernels to speech recognition
  - parametric classifiers: augmented statistical models
  - non-parametric classifiers: support vector machines

Interesting classifier options - without throwing away HMMs

References

[1] V.N. Vapnik, *Statistical learning theory*, John Wiley & Sons, 1998.

[2] W.M. Campbell, J.P. Campbell, D.A. Reynolds, E. Singer, and P.A. Torres-Carraquillo, "Support vector machines for speaker and language recognition," *Computer Speech Language*, vol. 20, pp. 210–229, 2005.

[3] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins, "Text classification using string kernels," *Journal of Machine Learning Research*, vol. 2, pp. 419–444, 2002.

[4] C. Cortes, P. Haffner, and M. Mohri, "Weighted automata kernels - general framework and algorithms," in *Proc. Eurospeech*, 2003.

[5] Layton MI and MJF Gales, "Acoustic modelling using continuous rational kernels," *Journal of VLSI Signal Processing Systems for Signal, Image, and Video Technology*, August 2007.

[6] W.M. Campbell, D.Sturim, D.A. Reynolds, and A. Solomonoff, "SVM based speaker verification using a GMM supervector kernel and NAP variability compensation," in *Proc. ICASSP*, 2006.

[7] N.D. Smith and M.J.F. Gales, "Speech recognition using SVMs," in *Advances in Neural Information Processing Systems*, 2001.

[8] Stolcke et al, "MLLR transforms as features in speaker recognition," in *Proc. ICASSP*, 2005.

[9] H Yang, Y Dong, X Zhao, L Lu, and H Wang, "Cluster adaptive training weights as features in SVM-based speaker verification," in *Proceedings InterSpeech*, 2007.

[10] T. Jaakkola and D. Haussler, "Exploiting generative models in disciminative classifiers," in *Advances in Neural Information Processing Systems 11*, S.A. Solla and D.A. Cohn, Eds. 1999, pp. 487–493, MIT Press.

[11] C. Longworth and M.J.F. Gales, "Derivative and parametric kernels for speaker verification," in *Proceedings InterSpeech*, September 2007.

[12] V. Wan and S. Renals, "Speaker verification using sequence discriminant support vector machines," *IEEE Transactions Speech and Audio Processing*, 2004.

[13] K. Tsuda, T. Kin, and K. Asai, "Marginalized kernels for biological sequences," *Bioinformatics*, vol. 18, pp. S268–S275, 2002.

[14] A. Solomonoff, W.M. Campbell, and I. Boradman, "Advances in channel compensation foe SVM speaker recognition," in *Proc. ICASSP*, 2005.

[15] AO Hatch, S Kajarekar, and A Stolcke, "Within-class covariance normalisation for SVM-based speaker verification," in *Proceedings InterSpeech*, 2006.

[16] J.A. Bilmes, "Graphical models and automatic speech recognition," in *Mathmatical Foundations of Speech and Language Processing*, 2003.

[17] M.I. Layton and M.J.F. Gales, "Augmented statistical models for speech recognition," in *ICASSP*, 2006.

[18] MJF Gales and F Flego, "Discriminative classiers with generative kernels for noise robust speech recognition," in *Proc. ICASSP*, 2009.

[19] M J F Gales, "Maximum likelihood linear transformations for HMM-based speech recognition," *Computer Speech and Language*, vol. 12, pp. 75–98, 1998.

[20] A Acero, L Deng, T Kristjansson, and J Zhang, "HMM Adaptation using Vector Taylor Series for Noisy Speech Recognition," in *Proc. ICSLP*, Beijing, China, 2000.

[21] V. Venkataramani, S. Chakrabartty, and W. Byrne, "Support vector machines for segmental minimum Bayes risk decoding of continuous speech," in *ASRU 2003*, 2003.