

ACOUSTIC MODELLING USING CONTINUOUS RATIONAL KERNELS

M.I. Layton and M.J.F. Gales

Engineering Department, Cambridge University, Trumpington St., Cambridge, CB2 1PZ U.K.
Email: {ml362,mjfg}@eng.cam.ac.uk

ABSTRACT

There has been significant interest in developing alternatives to hidden Markov models (HMMs) for speech recognition. In particular, interest has been focused upon models that allow additional dependencies to be incorporated. One such model is the Augmented Statistical Model. Here a local exponential approximation, based upon derivatives of a base distribution, is made about some distribution of the base model. Augmented statistical models can be trained using a maximum margin criterion, which may be implemented using an SVM with a generative kernel. Calculating derivatives of the base distribution, in particular higher-order derivatives, to form the generative kernel requires complex dynamic programming algorithms. In this paper a new form of rational kernel, a continuous rational kernel is proposed. This allows elements of the generative kernel, including those based on higher-order derivatives, to be computed using standard forms of transducer within a rational kernel framework. In addition, the derivatives are shown to be a principled method of defining marginalised kernels. Continuous rational kernels are evaluated using a large vocabulary continuous speech recognition (LVCSR) task.

1. INTRODUCTION

Many classification algorithms operate on only fixed-length vectors. However, applications such as text processing, computational biology and speech processing require classification of variable-length sequences of vectors. For such data, it is usual to estimate class-conditional generative models, such as Gaussian mixture models (GMMs) and hidden Markov models (HMMs), and then to use Bayes' rule to calculate the posterior probability of the class labels. For example, in speech recognition, HMMs are used to map sequences of observations, $\mathbf{O} = \{\mathbf{o}_1, \dots, \mathbf{o}_T\}$, into a one-dimensional likelihood space,

$$\hat{p}(\mathbf{O}; \boldsymbol{\lambda}) = \sum_{\theta \in \Theta} \prod_{t=1}^T a_{\theta_t \theta_{t-1}} p(\mathbf{o}_t | \theta_t; \boldsymbol{\lambda}) \quad (1)$$

where a_{ij} and $p(\mathbf{o}_t | \theta_t; \boldsymbol{\lambda})$ are the state transition probabilities and state-output distributions respectively; θ is a state sequence from the set of all possible sequences, Θ . Unfortunately, the independence and conditional-independence assumptions associated with HMMs are not correct for the speech signal, and so may degrade recognition performance.

In [1, 2], Augmented Statistical Models are proposed as a principled method of adding higher-order dependencies to a base statistical model. First, a base statistical model, $\hat{p}(\mathbf{O}; \boldsymbol{\lambda})$, is defined;

The authors would like to thank Bennett Rogers for initial work on the LVCSR task. Martin Layton would like to thank the Schiff Foundation for funding. Extensive use was made of equipment supplied to the Speech Group at Cambridge University by IBM under an SUR award.

this is typically a GMM or HMM. The parameters of this base model, $\boldsymbol{\lambda}$, are normally estimated using maximum likelihood estimation (MLE). Next, an augmented statistical model is generated using a local exponential approximation to the base distribution,

$$p(\mathbf{O}; \boldsymbol{\lambda}, \boldsymbol{\alpha}) = \frac{1}{\tau(\boldsymbol{\lambda}, \boldsymbol{\alpha})} \hat{p}(\mathbf{O}; \boldsymbol{\lambda}) \exp \left[\boldsymbol{\alpha}^T \nabla_{\boldsymbol{\lambda}}^{(\rho)} \ln \hat{p}(\mathbf{O}; \boldsymbol{\lambda}) \right] \quad (2)$$

where $\nabla_{\boldsymbol{\lambda}}^{(\rho)} \ln \hat{p}(\mathbf{O}; \boldsymbol{\lambda})$ represents all derivatives up to an order ρ . When latent-variable base models are used, the augmented model breaks the conditional independence assumptions of the base model [2]; independence assumptions remain unchanged. Unfortunately the augmented parameters, $\boldsymbol{\alpha}$, cannot be estimated easily using MLE since the normalisation term, $\tau(\boldsymbol{\lambda}, \boldsymbol{\alpha})$, cannot usually be calculated. However, by only considering binary classification tasks, estimation of $\boldsymbol{\alpha}$ can be reduced to finding a linear decision boundary in a generative score-space [3, 1] (an extension of the Fisher score-space [4]), formed from the log-likelihood ratio of the two base distributions and $\nabla_{\boldsymbol{\lambda}}^{(\rho)} \ln \hat{p}(\mathbf{O}; \boldsymbol{\lambda})$ for each class. Support vector machines (SVMs) – an implementation of maximum margin estimation – are commonly used to find this decision boundary since they offer good generalisation performance in high-dimensional score-spaces. However, calculating higher-order derivatives of models such as HMMs normally requires complex dynamic programming algorithms, tuned to specific derivatives.

Recently, Cortes *et al.* [5, 6] introduced a family of kernels, based upon weighted finite-state transducers, known as *Rational Kernels*. These operate by mapping variable-length sequences of discrete symbols into a fixed-dimensional feature-space in which 'distances' may be computed. String kernels have been shown to be a special case of rational kernels [5]. However, unlike string kernels (which require dynamic programming algorithms), rational kernel calculations use efficient weighted transducer composition. This enables them to operate on weighted lattices of observations (in addition to linear sequences) allowing effective classification when observation labels are uncertain. Rational kernels have been used in the construction of speech recognition systems [5, 7, 8]. In these systems, HMMs map continuous observations into a one-dimensional log-likelihood-space – the output of each HMM – and a label identifying the HMM. These labels and scores are passed to the recogniser. Rational kernels, trained upon the recogniser output, are then used to disambiguate sequences of words [5]. Unfortunately, unlike generative kernels, this technique discards all acoustic and HMM state-space information after the initial stage of recognition.

Previous work has examined the relationships between Fisher kernels, finite-state automata and string kernels. In Saunders *et al.* [9] the links between string kernels and Fisher kernels were discussed. Saunders showed that if generative models, parametrised by the probabilities of contiguous word sequences, are trained on

text documents, string kernels on those documents are equivalent to the Fisher derivatives of the generative model of the document.

In this paper, a form of continuous rational kernel is proposed. Here, rational kernels are used as a framework for calculating generative kernels with various-order derivatives. This allows continuous data to be directly used within the rational kernel framework. In contrast to the standard approach of calculating generative score-spaces explicitly (using dynamic programming algorithms that vary with each derivative), continuous rational kernels use standard transducer operations – different derivatives are calculated using the same algorithms with different finite-state transducers.

2. FINITE-STATE TRANSDUCERS AND RATIONAL KERNELS

In this section weighted finite-state transducers and acceptors are introduced. These consist of a set of states labelled from 1 to N . All paths start from state one¹ and terminate in an end-state (denoted by a double circle). Transitions between states are represented by directed arcs labelled in the form $\delta:\gamma$, where $\delta \in \Sigma$ and $\gamma \in \Delta$ are the input and output symbols. Σ and Δ are input and output alphabets. Either δ and γ can be the null symbol, ϵ . These represent transitions that either do not absorb symbols (null input, $\delta = \epsilon$) or do not emit symbols (null output, $\gamma = \epsilon$). In this paper, input and output alphabets are assumed to be identical, $\Sigma = \Delta$. Arcs labelled with $\Delta:\Delta$ or $\Delta:\epsilon$ denote the set of transitions $\delta:\delta$ and $\delta:\epsilon$, for all $\delta \in \Delta$.

In addition to input and output symbols, transitions between states are assigned a weight, w . Valid values for this weight and the operations used to propagate it through lattices are determined by the *semiring*, $(\mathbb{K}, \oplus, \otimes, \bar{0}, \bar{1})$ [10]. The set of weights, $w \in \mathbb{K}$, is closed under the operations of addition, \oplus , and multiplication, \otimes . The zero and identity elements are denoted by $\bar{0}$ and $\bar{1}$ respectively. Some standard semirings are the Real, Log and Tropical semirings. For transducers, weights in the real semiring represent probabilities: they are multiplied along paths and summed when paths merge. The log semiring is an isomorphism of the real semiring and is used to avoid numerical accuracy problems when long sequences are considered. When large transducers are considered, the Viterbi semiring is often used. This is similar to the log semiring except that when paths merge, only the most likely path is propagated. Unless explicitly stated (using $/w$ notation), weights are assumed to be identically $\bar{1}$. For clarity, all transducers in this paper are defined in the real semiring.

The advantage of representing sequences and operations as transducers is that they can be transformed using a number of standard algorithms. Given two transducers, U_1 and U_2 , efficient algorithms exist for the operations: inversion, U_1^{-1} ; composition, $U_1 \circ U_2$ [7]; and shortest-distance/transducer weight, $[[U_1]]$ [10]. These operations allow efficient *rational kernels* to be defined.

Rational kernels [5] are an attractive method of mapping discrete-symbol sequences and lattices to a high-dimensional feature-space since all calculations are performed using standardised transducer operations. Consider two sequences, O_a and O_b , with observations chosen from a discrete alphabet Δ . One simple feature-space is defined as the space where each dimension represents the frequency of a particular symbol (similar to a Bag-of-Words ker-

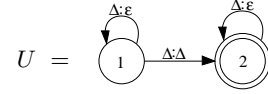
nel [11]). This is known as the unigram feature-space. In vector notation this is written as,

$$\phi(O) = \begin{bmatrix} f(a|O) \\ f(b|O) \\ \vdots \end{bmatrix} \quad (3)$$

where $f(\delta|O)$ represents the number of occurrences of δ in O . A similarity measure between examples in this feature-space is normally given by the dot-product. This is known as a unigram kernel,

$$K(O_a, O_b) = \phi(O_a)^T \phi(O_b) = \sum_{\delta \in \Delta} f(\delta|O_a) f(\delta|O_b) \quad (4)$$

Both the unigram feature-space and kernel can be efficiently calculated within the finite-state transducer framework. First, acceptors, A_i , are constructed to represent the observation sequences O_i . Composing these acceptors with the unigram transducer, U ,



transforms the linear sequences of observations into lattices, $A_i \circ U$. Each lattice contains T_i distinct paths of length T_i (where T_i is the length of an observation sequence O_i). All output symbols in the t -th, $t \in [1, T_i]$, path are null except for the transition associated with o_t which has an output label $o_t \in \Delta$. The number of paths with a particular output label, $\delta \in \Delta$, is equal to the number of occurrences of δ in the sequence. Each path has a unit weight, and so the total weight of these paths is $f(\delta|x)$ – the unigram count. The *unigram rational kernel* is given by the dot-product in this feature-space, and is written as [5],

$$\begin{aligned} K(O_a, O_b) &= [[(A_a \circ U) \circ (A_b \circ U)^{-1}]] \\ &= [[A_a \circ U \circ U^{-1} \circ A_b]] \end{aligned} \quad (5)$$

Inversion of the second operand transposes the input and output symbols. The outputs of the first operand therefore join with the inputs of the second, matching feature-space dimensions.

Unfortunately the unigram kernel is not invariant to sequence lengths: longer sequences have more observations and therefore higher counts associated with each dimension. Sequence-length normalisation is therefore preferable; the length-normalised unigram kernel is given by,

$$K(O_a, O_b) = \frac{1}{T_a T_b} [[A_a \circ U \circ U^{-1} \circ A_b]] \quad (6)$$

In addition to basic sequences of observations, rational kernels can operate on lattices (represented by weighted acceptors). When labelling of observations is uncertain, the use of lattices allows all² possible sequences of labels to be represented. Each path in the lattice represents a single hypothesised label sequence and is assigned a weight according to its likelihood. When the kernel is calculated, all information in the lattice is utilised, potentially improving classification performance. The rational kernel for lattices is identical to that for linear sequences and so can be calculated using only standard transducer operations.

¹Since state numbering is arbitrary, the states of a transducer with starting state $s \neq 1$ can be simply renumbered so that the $s = 1$.

²To reduce memory and CPU requirements, lattices are often pruned to remove highly unlikely paths.

3. CONTINUOUS RATIONAL KERNELS

As discussed in the previous section, finite-state automata offer a desirable framework for defining kernels on variable-length sequences of *discrete* data. Unfortunately, many tasks require classification of sequences of *continuous* observations. A method of mapping variable-length sequences to a fixed-dimensional feature-space is required. For example, in speech recognition generative models are used to map observation sequences, $\mathbf{O} = \{o_1, \dots, o_T\}$, into a one-dimensional likelihood score-space; HMMs are commonly used. Unfortunately, this one-dimensional representation of the observations severely limits the information available to the classifier. In [4], Jaakkola and Haussler introduced a score-space that captures differences in the generative process between examples instead of differences in likelihood. This score-space is based upon the derivative-space of the generative model and is known as the Fisher score-space.

The Fisher score-space was subsequently generalised to class-conditional generative models [3]. Here examples belonging to classes ω_1 and ω_2 , are mapped into a *score-space* consisting of the log-likelihood ratio and the first- and higher-order derivatives of the generative models with respect to their parameters, $\lambda^{(1)}$ and $\lambda^{(2)}$. These generative score-spaces can be used to train parameters of augmented models (equation 2) for binary classification tasks [2]. Bayes' rule for this binary case may be expressed as,

$$\left[1, \alpha^{(1)T}, \alpha^{(2)T}\right]^T \phi^{\text{gen}}(\mathbf{O}; \lambda) + b \underset{\omega_2}{\overset{\omega_1}{>}} 0 \quad (7)$$

where the ρ -order *generative score-space* is given by, $\phi^{\text{gen}}(\mathbf{O}; \lambda) =$

$$\begin{bmatrix} \phi^{11r}(\mathbf{O}; \lambda) \\ \phi^d(\mathbf{O}; \lambda^{(1)}) \\ -\phi^d(\mathbf{O}; \lambda^{(2)}) \end{bmatrix} = \begin{bmatrix} \ln \hat{p}(\mathbf{O}; \lambda^{(1)}) - \ln \hat{p}(\mathbf{O}; \lambda^{(2)}) \\ \nabla_{\lambda^{(1)}}^{(\rho)} \ln \hat{p}(\mathbf{O}; \lambda^{(1)}) \\ -\nabla_{\lambda^{(2)}}^{(\rho)} \ln \hat{p}(\mathbf{O}; \lambda^{(2)}) \end{bmatrix}$$

and $\lambda = [\lambda^{(1)T} \lambda^{(2)T}]^T$. The bias, b , subsumes the class priors and normalisation terms. A linear decision boundary in the generative score-space yields the augmented model parameters α . Since a distance-based learner (an SVM) will be used, a kernel must be defined. The generative kernel is defined as the inner-product in the generative score-space³,

$$K^{\text{gen}}(\mathbf{O}_a, \mathbf{O}_b; \lambda) = \phi^{\text{gen}}(\mathbf{O}_a; \lambda)^T \mathbf{G}^{-1}(\lambda) \phi^{\text{gen}}(\mathbf{O}_b; \lambda) \quad (8)$$

where $\mathbf{G}(\lambda) = \mathcal{E}[\phi^{\text{gen}}(\mathbf{O}; \lambda) \phi^{\text{gen}}(\mathbf{O}; \lambda)^T]$. The normalisation term, \mathbf{G} , ensures that the kernel is independent of the parametrisation of the generative models. However, for computational simplicity, it is traditional to approximate it by the identity matrix. The generative kernel can thus be written as,

$$K^{\text{gen}}(\mathbf{O}_a, \mathbf{O}_b; \lambda) = K^{11r}(\mathbf{O}_a, \mathbf{O}_b; \lambda) + K^d(\mathbf{O}_a, \mathbf{O}_b; \lambda^{(1)}) + K^d(\mathbf{O}_a, \mathbf{O}_b; \lambda^{(2)}) \quad (9)$$

where $K^{11r}(\cdot)$ and $K^d(\cdot)$ are dot-products in the feature-spaces ϕ^{11r} and ϕ^d respectively. Unfortunately, although derivatives of the acoustic model (in ϕ^d) can be calculated using dynamic programming algorithms, the complexity of these algorithms increases significantly for higher-order derivatives.

³The generative kernel satisfies Mercer's condition since inner-product kernels are always positive semi-definite.

In this section, continuous rational kernels are proposed as a powerful method of calculating both the first- and higher-order derivatives of acoustic models. First, class-conditional generative models are trained using maximum likelihood estimation (MLE) on the data. These model spatial and (for HMMs) temporal dependencies within observation sequences. Observation sequences are passed through these acoustic models and the sequences of latent-states and likelihoods recorded; these are compactly represented by a weighted acceptor, L . Each transition in the acceptor is labelled with a latent-state (components for GMMs; state/component pairs for HMMs) and a weight. The total weight of all paths through the acceptor, $[[L]]$, is one. Since the weighted acceptors have discrete labels, rational kernels can be trained. We call them *continuous rational kernels*. The feature-space is defined by a transducer operating on the latent-state acceptor, L . In this paper, n -gram and gappy- n -gram transducers will be described. These allow many different state-posterior probabilities to be calculated and, with appropriate weighting, allow first and higher-order derivatives of the acoustic models with respect to their parameters to be calculated within the continuous rational kernel framework. Unlike direct calculation of generative kernels, custom dynamic programming algorithms are not required.

To simplify the process, subsets of the acoustic model derivatives are considered. As shown in equation (9), these subsets can be combined to calculate the final generative kernel.

3.1. Component and transition probability kernels

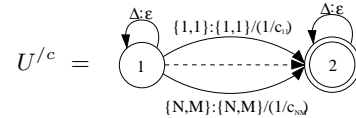
First consider the derivatives of the log-likelihood of an M -component GMM base acoustic model (with parameters λ) with respect to its component priors, c_m ,

$$\phi^c(\mathbf{O}; \lambda) = \nabla_{c_m} \ln \hat{p}(\mathbf{O}; \lambda) = \sum_{t=1}^T \left[\frac{P(m|o_t; \lambda)}{c_m} - 1 \right] \quad (10)$$

It is interesting to compare this component-prior score-space with the unigram feature-space described in section 2. This feature-space is obtained by composing the lattice of latent-states, L , with the unigram transducer, U . Each feature in the resulting space is given by the posterior probability of obtaining a particular state. The unigram feature-space is thus given by,

$$\phi^{\text{uni}}(\mathbf{O}; \lambda) = L \circ U = \sum_{t=1}^T \begin{bmatrix} P(m_t = 1|\mathbf{O}; \lambda) \\ P(m_t = 2|\mathbf{O}; \lambda) \\ \dots \\ P(m_t = M|\mathbf{O}; \lambda) \end{bmatrix} \quad (11)$$

From (10) and (11), it is clear that the derivative score-space is simply a scaled version of the unigram feature-space (the additional constant in equation (10) can be ignored since it does not affect classification). A scaled unigram transducer, U^c , that generates this component-prior derivative-space for an N -state, M -component HMM is,



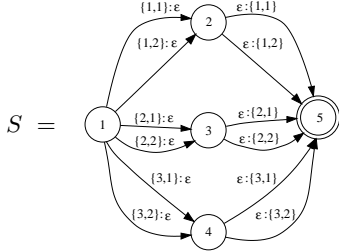
The corresponding component-prior continuous rational kernel (after length-normalisation) is given by,

$$K^c(\mathbf{O}_a, \mathbf{O}_b; \lambda) = \frac{1}{T_a T_b} [[L_{a \circ} U^c \circ U^{c^{-1}} \circ L_b]] \quad (12)$$

In equation (12), the component-prior kernel is expressed entirely in terms of transducer operations. Since standard algorithms are available for these, the problem is reduced to the task of selecting appropriate transducers. In addition, using this framework, more complex derivatives can be calculated, for example, the derivatives of an HMM with respect to its component priors,

$$\begin{aligned} \phi^\circ(\mathbf{O}; \lambda) &= \nabla_{c_{jm}} \ln \hat{p}(\mathbf{O}; \lambda) \\ &= \frac{1}{c_{jm}} \sum_{t=1}^T P(\theta_t = \{j, m\} | \mathbf{O}; \lambda) - \sum_{t=1}^T P(\theta_t = s_j | \mathbf{O}; \lambda) \end{aligned} \quad (13)$$

The first term of this expression is identical to that of the GMM derivative. Therefore the scaled unigram transducer, U'^c , defined above (using an alphabet of state/component pairs) can be reused. Expanding the second term, $P(\theta_t = s_j | \mathbf{O}; \lambda) = \sum_{m=1}^M P(\theta_t = \{j, m\} | \mathbf{O}; \lambda)$, yields a summation of the component posteriors over the components in each state. Given the unigram feature-space, a transducer, S , can be defined to perform this summation. An example of S for a three-state HMM with two mixture-components is,



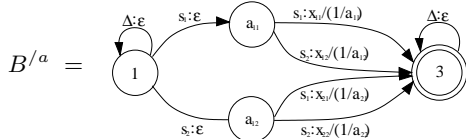
The state posterior, $P(\theta_t = s_j | \mathbf{O}; \lambda)$, is therefore the composition of the two transducers: $U \circ S$. The resulting component-prior kernel for an HMM is given by,

$$\begin{aligned} K^c(\mathbf{O}_a, \mathbf{O}_b; \lambda) &= \frac{1}{T_a T_b} \left([[L_a \circ U'^c \circ U'^{c-1} \circ L_b]] \right. \\ &\quad \left. - 2[[L_a \circ U'^c \circ S^{-1} \circ U^{-1} \circ L_b]] \right. \\ &\quad \left. + [[L_a \circ U \circ S \circ S^{-1} \circ U^{-1} \circ L_b]] \right) \end{aligned} \quad (14)$$

This kernel uses exactly the same framework and calculations as the component-prior kernel for GMMs. The only change required is the addition of a new transducer, S . In addition to the component-prior derivatives, derivatives with respect to the HMM transition probabilities can also be calculated. These are given by,

$$\begin{aligned} \nabla_{a_{ij}} \ln \hat{p}(\mathbf{O}; \lambda) &= \frac{1}{a_{ij}} \sum_{t=1}^T P(\theta_t = s_i, \theta_{t+1} = s_j | \mathbf{O}; \lambda) \\ &\quad - \sum_{t=1}^T P(\theta_t = s_j | \mathbf{O}; \lambda) \end{aligned} \quad (15)$$

The first term can be calculated using a modified form of the bigram feature-space, calculated using the transducer, B'^a . An example of B'^a for a two-state HMM is,



where input labels s_i represent the set of all transitions with a state

i . Dimensions in the feature-space are indexed by pairs of consecutive states, $\{i, j\}$, and labelled x_{ij} . The continuous rational kernel for the transition probability score-space can therefore be calculated using an expression similar to (14).

In this section, continuous rational kernels have been proposed as an attractive method for calculating derivatives of latent-variable generative models for use in generative kernels. However, these derivatives also provide a principled method of defining marginalised kernels [12]. In particular, GMM derivatives with respect to mixture-component priors correspond to the first-order marginalised count kernel; derivatives with respect to HMM transition probabilities correspond to second-order marginalised count kernels.

3.2. Fisher and generative score-spaces

In the previous section, the use of modified unigram and bigram transducers for the calculation of derivatives of the base acoustic model with respect to the component priors and transition probabilities was discussed. However, score-spaces can also be defined using the derivatives with respect to the component means, μ_{jm} , and covariances⁴, Σ_{jm} . Consider the score-space of derivatives with respect to the means [3],

$$\nabla_{\mu_{jm}} \ln \hat{p}(\mathbf{O}; \lambda) = \sum_{t=1}^T P(\theta_t = \{j, m\} | \mathbf{O}; \lambda) \Sigma_{jm}^{-1}(\mathbf{o}_t - \mu_{jm}) \quad (16)$$

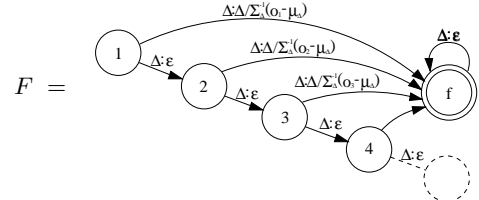
Although, given an acceptor of state-sequences, unigram transducers generate state/component posteriors, they cannot weight them by the observations. This is because observations are vector quantities whereas transducer weights are scalar. To overcome this a new semiring, the *vector semiring*, is introduced that allows weight *vectors* to be associated with transducer transitions. It is defined as $(\mathbb{R}^{d^+}, +, \otimes_{\text{vec}}, [0]^d, [1]^d)$ where $[x]^d$ represents a vector of length d with elements equal to x . Both addition and multiplication are

Semi-ring	\mathbb{K}	\oplus	\otimes	$\bar{0}$	$\bar{1}$
Vector	\mathbb{R}^{d^+}	$+$	\otimes_{vec}	$\{0\}^d$	$\{1\}^d$

Table 1. The Vector semiring

performed on a per-dimension basis; multiplication is defined as, $x \otimes_{\text{vec}} y = \{x_1 y_1, x_2 y_2, \dots, x_d y_d\}$.

The second problem with weighting posteriors by observations is that observations are time-dependent, whereas unigram transducer weights are constant. The solution is to expand the self-transitions in the unigram transducer to yield time-dependent transducer paths. These can then be weighted by a function of the t -th observation, $\Sigma_{jm}^{-1}(\mathbf{o}_t - \mu_{jm})$. The expanded unigram transducer with vector weights is given by,



⁴Covariance derivatives have a similar functional form to derivatives with respect to the mean and so, for brevity, are not presented.

This is known as the *Fisher transducer*. The Fisher kernel with acoustic model derivatives with respect to the means is given by,

$$K(\mathbf{O}_a, \mathbf{O}_b; \lambda) = \frac{1}{T_a T_b} [[L_a \circ F \circ F^{-1} \circ L_b]] \quad (17)$$

Covariance derivatives are obtained simply by setting the path weights of the Fisher transducer to $-\frac{1}{2}[\Sigma_{jm}^{-1} + \Sigma_{jm}^{-1}(\mathbf{o}_t - \boldsymbol{\mu}_{jm})(\mathbf{o}_t - \boldsymbol{\mu}_{jm})^T \Sigma_{jm}^{-1}]$.

3.3. Second and higher-order derivatives

Previous sections have concentrated on the use of continuous rational kernels for the calculation of the first-derivatives of GMM and HMM acoustic models with respect to their parameters. However additional information, that may be useful for classification, is contained within the higher-order derivatives of the acoustic models. When these higher-order derivatives are considered, rational kernels offer significant benefits over their dynamic programming counterparts. Consider, for example, the second derivatives of an HMM with respect to its component priors,

$$\begin{aligned} \nabla_{c_{kn}} \nabla_{c_{jm}}^T \ln \hat{p}(\mathbf{O}; \lambda) = & \frac{1}{c_{jm} c_{kn}} \sum_{t=1}^T \left\{ -2P(\theta_t^{jm} | \mathbf{O}; \lambda) \delta_{jk} \delta_{mn} \right. \\ & + \sum_{\tau=1}^T \left(D(\theta_t^{jm}, \theta_\tau^{kn}) - c_{jm} D(\theta_t^j, \theta_\tau^{kn}) \right. \\ & \left. \left. - c_{kn} D(\theta_t^{jm}, \theta_\tau^k) + c_{jm} c_{kn} D(\theta_t^j, \theta_\tau^k) \right) \right\} \quad (18) \end{aligned}$$

where

$$D(\theta_t^{jm}, \theta_\tau^{kn}) = P(\theta_t^{jm}, \theta_\tau^{kn} | \mathbf{O}; \lambda) - P(\theta_t^{jm} | \mathbf{O}; \lambda) P(\theta_\tau^{kn} | \mathbf{O}; \lambda) \quad (19)$$

and θ_t^{jm} and θ_t^j denote $\theta_t = \{j, m\}$ and $s_t = j$ respectively. Dynamic programming algorithms for calculating the joint posterior of being in state $\{j, m\}$ at time t and state $\{k, n\}$ at time τ are highly complex. This makes direct calculation of second derivatives difficult. However, when calculated within the continuous rational kernel framework, standard algorithms are reused with new transducers. In particular, the transducer for calculating the joint probability, $P(\theta_t^{jm}, \theta_\tau^{kn} | \mathbf{O}; \lambda)$, is given by the gappy bigram transducer, G . This generates the feature-space, $\phi_{jm, kn}^{gbi}(\mathbf{O}; \lambda)$,

$$G = \begin{array}{c} \begin{array}{c} \Delta \varepsilon \\ \curvearrowright \\ \textcircled{1} \end{array} \xrightarrow{\Delta \Delta} \begin{array}{c} \Delta \varepsilon \\ \curvearrowright \\ \textcircled{2} \end{array} \xrightarrow{\Delta \Delta} \begin{array}{c} \Delta \varepsilon \\ \curvearrowright \\ \textcircled{3} \end{array} \\ \phi_{jm, kn}^{gbi}(\mathbf{O}; \lambda) = \sum_{t=1}^T \sum_{\tau=1}^T P(\theta_t = \{j, m\}, \theta_\tau = \{k, n\} | \mathbf{O}; \lambda) \quad (20) \end{array}$$

Scaling this transducer by $1/c_{jm} c_{kn}$ (similarly to the unigram and bigram transducers) yields a transducer that calculates the first term in equation (19). All other terms can be generated using combinations of the unigram (U), gappy-bigram (G), and summation (S) transducers. For example, the term, $P(\theta_t^{jm} | \mathbf{O}; \lambda) \times P(\theta_\tau^{kn} | \mathbf{O}; \lambda)$, is given by the product of two unigram feature-spaces, $(L_a \circ U) \otimes (L_b \circ U)$, where \otimes denotes transducer concatenation. The second-order component-prior kernel can thus be written as a sum of the kernels of the parts; for brevity it is omitted. In addition to second derivatives with respect to the component priors, second derivatives with respect to other parameters

can be calculated. These derivatives take a similar form to equation (18) and may be calculated using the transducers introduced in this section. Higher-order derivatives are calculated using other n -gram transducers, for example, the gappy-trigram transducer for third-derivatives.

4. LVCSR RESULTS

Generative kernels (calculated using either the continuous rational kernel framework or dynamic programming) are a powerful form of acoustic model. Unfortunately, the distance-learning algorithms, in particular SVMs, used to train these models are binary classifiers whereas large vocabulary continuous speech recognition (LVCSR) has a vast number of possible classes. In order to apply continuous rational kernels to LVCSR, it is necessary to map this highly complex problem into a small set of binary classification problems; an approach related to that described in [13] is used. Given an utterance, the standard LVCSR Viterbi decoder is used to generate a *word lattice*. This represents the most likely word sequences for that utterance. The arcs are labelled with words and the language and acoustic model likelihood; nodes are labelled with time-stamps. Next, word lattices are converted to *confusion networks* [14]. These consist of a series of nodes with a linear graph. Each arc is labelled with a word, start and end times and a log-posterior, $\mathcal{F}(\omega_i)$. Finally the confusion networks are pruned so that at each time instance, only two words remain. Acoustic models and continuous rational kernels are then trained on these word pairs.

The database used for the LVCSR experiments was a 400 hour subset of the Fisher LDC data. This is the `fish2004sub` data set used for initial system development of the system described in [15]. The model set used in experiments was based upon the standard models and front-end described in [15]. Confusion networks were generated, using a bigram language model (LM), on the same 400 hours of training data.

Word Pair (examples)	Training	CN post.	# Components		
			1	2	4
CAN/CAN'T (7,522)	ML	78.5	81.7	86.0	88.0
	SVM ϕ^w		–	86.4	88.9
	SVM ϕ^{mc}		83.9	87.6	89.4
	SVM ϕ^{wmc}		–	87.7	89.6
	SVM ϕ^{pmc}		88.6	90.1	91.3
KNOW/NO (8,950)	ML	83.1	68.4	69.4	70.8
	SVM ϕ^w		–	69.3	71.9
	SVM ϕ^{mc}		69.3	71.8	73.6
	SVM ϕ^{wmc}		–	71.7	73.9
	SVM ϕ^{pmc}		84.9	85.3	86.3
SVM ϕ^{pwmc}	–	85.3	86.3		

Table 2. 8-Fold cross validation results (% correct) using a variable number of mixture components

Classifier performance was evaluated using 8-fold cross-validation on the training data. All experiments used diagonal covariance matrix GMMs, trained on the acoustic data using the longest time-stamps from the confusion networks⁵ for the two confusable

⁵The earliest time of the two words and the latest time of the two words

words. The number of positive and negative examples within each word pair were equalised by sampling – random selection will yield an accuracy of 50%. A number of pairwise classifiers were trained; two examples are shown in table 2. The baseline performance of the LVCSR system for each confusable pair is given by the confusion network (CN) posteriors – the baseline accuracy is approximately 80%.

Standard MLE GMMs with one, two and four Gaussian mixture-components were used as baseline pairwise classifiers. Next, a number of SVMs were trained using generative score-spaces of the log-likelihood ratio plus derivatives. Different combinations of derivatives with respect to the component priors (w), means (m) and covariances (c) were considered. In most cases, SVMs gave performance gains over the MLE acoustic models. As the number of derivatives increased, performance increased. Best performance was achieved when all derivatives (ϕ^{wmc}) were included. Similar trends were observed by Smith *et al.* [3, 1] for the Deterding and Isolet datasets. For the “CAN/CAN’T” pair, the GMM and SVM systems obtained better results than the baseline confusion network score. However in general, this was not the case. Classifier accuracy was improved when the posterior ratio, $\mathcal{F}(\omega_1) - \mathcal{F}(\omega_2)$, was included in the score-space; this added context information from the LVCSR language model. Performance gains were seen in all cases – significant gains were observed for word pairs where the GMM and SVM performance was poor.

No experiments were performed using second- and higher-order score-spaces. This is because in this speech recognition task, observations are sampled from a high-dimensional (39-dimension) observation-space. State/component posteriors are therefore polarised to be either one or zero. However since, as shown in equations (18) and (19), second derivatives are expressed in terms of the joint posterior of two states minus the product of the posteriors of the two states, they are always zero⁶. In general, this is not the case. For low-dimensional spaces, state/component posteriors assume a range of values and so have non-zero second derivatives.

5. CONCLUSION

In this paper continuous rational kernels, an extension to rational kernels, are proposed. These can be used to efficiently calculate elements of generative kernels which may in turn be used to estimate parameters of augmented models. These augmented models are a powerful extension to many latent-variable acoustic models such as HMMs. They make use of derivatives of the log-likelihood of the base acoustic model to allow additional dependencies to be modelled. Within the continuous rational kernel framework, it is shown that n -gram and gappy- n -gram transducers map observation sequences into feature-spaces of state posteriors. Weighting and combining these posterior-spaces allows derivatives of the base acoustic model with respect to the component priors and transition probabilities to be calculated using only standard transducer operations. The vector semiring allows derivatives with respect to the means and covariances to be calculated. Continuous rational kernels on these derivative-spaces can be combined to yield powerful generative kernels and thus can be used to train augmented

are used for the start and end time respectively. This ensures that the full acoustic data for both words is included.

⁶If either of the posteriors, $P(\theta_t = \{j, m\} | \mathcal{O}; \lambda)$ or $P(\theta_\tau = \{k, n\} | \mathcal{O}; \lambda)$ are zero, then the joint posterior, $P(\theta_t = \{j, m\}, \theta_\tau = \{k, n\} | \mathcal{O}; \lambda)$ will also be zero. However, if both posteriors are one, the joint posterior will also be one and so the difference will be zero.

statistical models. The continuous rational kernel framework has a significant advantage over direct calculation of score-space elements since it generalises well to higher dimensions – the same algorithms are used with different transducers. Initial experiments using SVM training on a large vocabulary speech recognition task indicate that the score-space and kernels detailed in this paper may be useful for speech recognition.

6. REFERENCES

- [1] N.D. Smith, *Using Augmented Statistical Models and Score Spaces for Classification*, Ph.D. thesis, University of Cambridge, September 2003.
- [2] M.J.F. Gales and M.I. Layton, “SVMs, score-spaces and maximum margin statistical models,” in *Beyond HMM workshop, ATR*, 2004.
- [3] N. Smith and M. Gales, “Speech recognition using SVMs,” in *Advances in Neural Information Processing Systems 14*, T. G. Dietterich, S. Becker, and Z. Ghahramani, Eds. 2002, pp. 1197–1204, MIT Press.
- [4] T. Jaakkola and D. Hausser, “Exploiting generative models in discriminative classifiers,” in *Advances in Neural Information Processing Systems 11*, S.A. Solla and D.A. Cohn, Eds. 1999, pp. 487–493, MIT Press.
- [5] C. Cortes, P. Haffner, and M. Mohri, “Rational kernels: Theory and algorithms,” *Journal of Machine Learning Research*, vol. 5, pp. 1035–1062, 2004.
- [6] C. Cortes, P. Haffner, and M. Mohri, “Positive definite rational kernels,” in *16th Annual Conference on Computational Learning Theory (COLT 2003)*, 2003, pp. 656–670.
- [7] M. Mohri, F. Pereira, and M. Riley, “Weighted finite-state transducers in speech recognition,” *Computer Speech & Language*, vol. 16, pp. 69–88, January 2002.
- [8] F.C.N. Pereira and M.D. Riley, “Speech recognition by composition of weighted finite automata,” in *Finite-State Devices for Natural Language Processing*, E. Roche and Y. Schabes, Eds. MIT Press, 1997.
- [9] C. Saunders, J. Shawe-Taylor, and A. Vinokourov, “String kernels, fisher kernels and finite state automata,” in *Advances in Neural Information Processing Systems 15*, S. Becker, S. Thrun, and K. Obermayer, Eds., 2003, pp. 633–640.
- [10] M. Mohri, “Semiring frameworks and algorithms for shortest-distance problems,” *Journal of Automata, Languages and Combinatorics*, vol. 7, pp. 321–350, 2002.
- [11] J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*, Cambridge University Press, 2004.
- [12] K. Tsuda, T. Kin, and K. Asai, “Marginalized kernels for biological sequences,” *Bioinformatics*, vol. 18, pp. S268–S275, 2002.
- [13] V. Venkataramani, S. Chakrabarty, and W. Byrne, “Support vector machines for segmental minimum Bayes risk decoding of continuous speech,” in *ASRU 2003*, 2003.
- [14] L. Mangu, E. Brill, and A. Stolcke, “Finding consensus among words: Lattice-based word error minimization,” in *Proc. Eurospeech*, 1999.
- [15] G. Evermann, H.Y. Chan, M.J.F. Gales, B. Jia, D. Mrva, P.C. Woodland, and K. Yu, “Training LVCSR systems on thousands of hours of data,” in *Proc. ICASSP*, 2005.