KERNELIZED LOG LINEAR MODELS FOR CONTINUOUS SPEECH RECOGNITION

Shi-Xiong Zhang and M.J.F. Gales

Department of Engineering, University of Cambridge, Cambridge, UK
{sxz20, mjfg}@eng.cam.ac.uk

ABSTRACT

Large margin criteria and discriminative models are two effective improvements for HMM-based speech recognition. This paper proposed a large margin trained log linear model with kernels for CSR. To avoid explicitly computing in the high dimensional feature space and to achieve the nonlinear decision boundaries, a kernel based training and decoding framework is proposed in this work. To make the system robust to noise a kernel adaptation scheme is also presented. Previous work in this area is extended in two directions. First, most kernels for CSR focus on measuring the similarity between two observation sequences. The proposed *joint* kernels defined a similarity between two observation-label sequence pairs on the sentence level. Second, this paper addresses how to efficiently employ kernels in large margin training and decoding with lattices. To the best of our knowledge, this is the first attempt at using large margin kernel-based log linear models for CSR. The model is evaluated on a noise corrupted continuous digit task: AURORA 2.0.

Index Terms— log linear model, large margin, kernel

1. INTRODUCTION

Most continuous speech recognition (CSR) is based on generative models, in the form of Hidden Markov Models (HMMs). Although discriminative training of HMMs has been shown to yield performance gains [1,2], the underlying models are still generative, with the sentence posterior being computed via the Bayes' rule. This has led to interest in discriminative models for CSR, e.g., Conditional Random Fields (CRF) [3], logistic regression machines [4], Conditional Augmented models (C-Aug) [5] and Structured Support Vector Machines (S-SVM) [6], where the posterior of the sentence given the observation can be directly modelled. For these discriminative models two important decisions need to be made: the form of the features to use and the appropriate training criterion.

A number of features have been investigated at the frame, model and word level [3, 7–9]. Although, it has been shown that the use of high-dimensional features, such as polynomial [10] and derivative features [8], can improve the performance, they are often trained using Conditional Maximum Likelihood (CML) [5, 11]. For high-dimensional features, there may be issues with generalisation. To address this there has been interest in large margin [9, 12] approaches. However, in these approaches, the features are explicitly defined and computed. Thus, the computational cost and memory requirement are at least propositional to the number of features.

To handle extremely high (or infinite) dimensional features, several methods based on the kernel trick have been developed [13]. These methods handle the high-dimensional feature $\boldsymbol{\phi}(\mathbf{O})$ of observation sequence $\mathbf{O}$ simply by focusing on the kernel function, $K(\mathbf{O}_i, \mathbf{O}_j) = \boldsymbol{\phi}(\mathbf{O}_i)^{\mathsf{T}}\boldsymbol{\phi}(\mathbf{O}_j)$. Kernels can be computed based on $\mathbf{O}$ to avoid computing the high-dimensional $\boldsymbol{\phi}(\cdot)$. Although kernel methods has been partially evaluated for frame-level phoneme classification tasks [14, 15], not much work has been reported on large margin kernel methods for continuous speech recognition.

To kernelize the sentence-level log linear models for CSR, this paper proposes a *joint* kernel $K((\mathbf{O}_i, \mathbf{w}_i), (\mathbf{O}_j, \mathbf{w}_j)) = \boldsymbol{\phi}(\mathbf{O}_i, \mathbf{w}_i)^{\mathsf{T}}\boldsymbol{\phi}(\mathbf{O}_j, \mathbf{w}_j)$, which defines a similarity between observation-word sequence pairs, $(\mathbf{O}, \mathbf{w})$. The proposed joint kernel can be decomposed at a segmental level, which allows efficient large margin training and decoding with lattices. One elegant property of this framework is the interface between the speech data and the learning algorithm is made uniquely through the joint kernel function, where all the domain knowledge can be incorporated. The system is evaluated on a noise corrupted continuous digit task: AURORA 2.0.

2. LOG LINEAR MODELS

Given an observation sequence, $\mathbf{O} = \{\boldsymbol{o}_1, \ldots, \boldsymbol{o}_T\}$, the posterior of the hypothesised labels $\mathbf{w} = \{w_1, \ldots, w_{|\mathbf{w}|}\}$ for many generative and discriminative models, e.g., CRFs and C-Augs can be expressed as a log linear model (LLM)

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

where Z is the normalisation term and $\boldsymbol{\alpha}$ are the log linear model parameters. $\boldsymbol{\phi}(\mathbf{O}, \mathbf{w})$ is a joint feature vector characterizing the statistical dependencies of $(\mathbf{O}, \mathbf{w})$. Recognition with this form of model only depends on the inner product of parameters and features

$$\hat{\mathbf{w}} = \arg\max_{\mathbf{w}} P(\mathbf{w}|\mathbf{O}; \boldsymbol{\alpha}) = \arg\max_{\mathbf{w}} \boldsymbol{\alpha}^{\mathsf{T}}\boldsymbol{\phi}(\mathbf{O}, \mathbf{w}). \tag{2}$$

The authors would like to thank Toshiba Research Europe Ltd, Cambridge Research Lab, for partly funding this work.

In CSR the training data is always limited, to make the model generalize well on a high-dimensional space, a large margin criterion can be applied [9, 12]. Given that the training data consists of pairs $\left\{\mathbf{O}^{(r)}, \mathbf{w}_{\texttt{ref}}^{(r)}\right\}_{r=1}^{R}$, where $\mathbf{O}^{(r)}$ is the r^{th} observation sequence and $\mathbf{w}_{\texttt{ref}}^{(r)}$ is its reference label sequence, the LLM can be large margin trained by *minimising*

$$\sum_{r=1}^{R}\left[\max_{\mathbf{w}\neq\mathbf{w}_{\texttt{ref}}^{(r)}}\left\{\mathcal{L}(\mathbf{w}_{\texttt{ref}}^{(r)},\mathbf{w}) - \log\left(\frac{P(\mathbf{w}_{\texttt{ref}}^{(r)}|\mathbf{O}^{(r)};\boldsymbol{\alpha})}{P(\mathbf{w}|\mathbf{O}^{(r)};\boldsymbol{\alpha})}\right)\right\}\right]_{+} \quad (3)$$

where $\mathcal{L}(\mathbf{w}_{\texttt{ref}}^{(r)},\mathbf{w})$ is a loss function introduced to control the size of the margin and $[\;]_{+}$ is the hinge-loss function. Here the margin is defined as the distance between the reference $\mathbf{w}_{\texttt{ref}}^{(r)}$ and "closest" competing word sequence $\mathbf{w}$ in the log posterior domain [9]. A Gaussian prior, $\log p(\boldsymbol{\alpha}) \propto -\frac{1}{2C}||\boldsymbol{\alpha}||^2$, is usually introduced into the training criterion, where $C\mathbf{I}$ is the covariance matrix of Gaussian prior [9]. Substituting equation (1) into (3) and canceling out the normalization term yields the following convex optimization

$$\min_{\boldsymbol{\alpha},\xi}\ \frac{1}{2}||\boldsymbol{\alpha}||_2^2 + \frac{C}{R}\xi \quad (4)$$

$$\text{s.t. } \forall \text{ competing hypothesis } \left\{\mathbf{w}^{(1)},\ldots,\mathbf{w}^{(R)}\right\} \in \mathcal{W}^R :$$

$$\boldsymbol{\alpha}^{\mathsf{T}}\sum_{r=1}^{R}\left[\boldsymbol{\phi}(\mathbf{O}^{(r)},\mathbf{w}_{\texttt{ref}}^{(r)}) - \boldsymbol{\phi}(\mathbf{O}^{(r)},\mathbf{w}^{(r)})\right] \geq \sum_{r=1}^{R}\mathcal{L}(\mathbf{w}_{\texttt{ref}}^{(r)},\mathbf{w}^{(r)}) - \xi$$

where $\xi \geq 0$ is a slack variable introduced to replace $[\;]_{+}$. (4) can be solved using the cutting plane algorithm [16].

3. KERNELIZATION

The discussion of LLM parameters and features has so far assumed that there is an explicit representation of each of these. It is also possible to consider a more general form that can be highly efficient in dealing with large feature spaces. Since the model uses an inner product between model parameters and features, it is possible to kernelize this operation in the same way as in non-linear SVMs [17]. This allows the "kernel trick" to be used to avoid explicitly computing and saving the large feature space.

Similar to SVMs [17], to kernelize the log linear model, the large margin training criterion (4) must be rewritten in the dual form. Note that the LLM parameters $\boldsymbol{\alpha}$ are not trained directly in the dual case. Instead the dual variables $\boldsymbol{\alpha}^{\texttt{dual}} = [\alpha_1^{\texttt{dual}},\ldots,\alpha_\tau^{\texttt{dual}},\ldots,\alpha_n^{\texttt{dual}}]$, where n is the number of training iterations, are learned by solving the optimization

$$\max_{\alpha_\tau^{\texttt{dual}}\geq 0}\sum_{\tau=1}^{n}\alpha_\tau^{\texttt{dual}}\mathcal{L}_\tau - \frac{1}{2}\sum_{\tau=1}^{n}\sum_{t=1}^{n}\alpha_\tau^{\texttt{dual}}\alpha_t^{\texttt{dual}}g_{t,\tau} \quad (5)$$

$$\text{s.t.}\sum_{\tau=1}^{n}\alpha_\tau^{\texttt{dual}} = C$$

where $\mathcal{L}_\tau = \frac{1}{R}\sum_{r=1}^{R}\mathcal{L}(\mathbf{w}_{\texttt{ref}}^{(r)},\mathbf{w}_\tau^{(r)})$ is the average loss and $\mathbf{w}_\tau^{(r)}$ is the competing word sequence for the r^{th} utterance on the τ^{th} iteration. $g_{t,\tau}$ is defined as the following inner product

$$g_{t,\tau} = \frac{1}{R^2}\left[\sum_{i=1}^{R}\left(\boldsymbol{\phi}(\mathbf{O}^{(i)},\mathbf{w}_{\texttt{ref}}^{(i)}) - \boldsymbol{\phi}(\mathbf{O}^{(i)},\mathbf{w}_\tau^{(i)})\right)\right]^{\mathsf{T}} \left[\sum_{j=1}^{R}\left(\boldsymbol{\phi}(\mathbf{O}^{(j)},\mathbf{w}_{\texttt{ref}}^{(j)}) - \boldsymbol{\phi}(\mathbf{O}^{(j)},\mathbf{w}_t^{(j)})\right)\right] \quad (6)$$

Algorithm 1: Kernelized large-margin log linear model

```
Input: $\{(\mathbf{O}^{(r)}, \mathbf{w}_{\texttt{ref}}^{(r)}\}_{r=1}^{R}$ and joint kernel function $K$;
repeat
    /* Step-1: Solve current dual program */
        $\boldsymbol{\alpha}^{\texttt{dual}} \leftarrow$ maxsimising equation (5)
    /* Step-2: Find "closest" competing hypothesis */
    for $r = 1..R$ do
        $\mathbf{w}_{n+1}^{(r)} \leftarrow \arg\max_{\mathbf{w}}\left\{\mathcal{L}(\mathbf{w}_{\texttt{ref}}^{(r)},\mathbf{w}) + \boldsymbol{\alpha}^{\mathsf{T}}\boldsymbol{\phi}(\mathbf{O}^{(r)},\mathbf{w})\right\}$ (8)
        where $\boldsymbol{\alpha}^{\mathsf{T}}\boldsymbol{\phi}(\mathbf{O}^{(r)},\mathbf{w})$ is implicitly computed using (13)
    /* Step-3: Update Gram matrix $\mathbf{H}$ */
    Compute $[g_{\tau,n+1}]_{\tau=1}^{n+1}$, Update $\mathbf{G}_{n\times n} \rightarrow \mathbf{G}_{(n+1)\times(n+1)}$
    $n = n + 1$;
until /* no new "closest" competing hypothesis can be found */;
return $\boldsymbol{\alpha}^{\texttt{dual}}$
```

Note that the dual optimization (5) only depends on the Gram matrix $\mathbf{G} = [g_{t,\tau}]_{n\times n}$, where $g_{t,\tau}$ depends on the inner product of the joint feature vectors $\boldsymbol{\phi}(\cdot)$, and thus can be replaced by a *joint* kernel function,

$$K\left((\mathbf{O}^{(i)},\mathbf{w}^{(i)}),(\mathbf{O}^{(j)},\mathbf{w}^{(j)})\right) = \boldsymbol{\phi}(\mathbf{O}^{(i)},\mathbf{w}^{(i)})^{\mathsf{T}}\boldsymbol{\phi}(\mathbf{O}^{(j)},\mathbf{w}^{(j)}) \quad (7)$$

These joint kernels are easier to describe analytically, since they express the correlation between two $(\mathbf{O},\mathbf{w})$ pairs [18]. More details will be discussed in Section 3.1. Thus the interface between the speech data and the learning algorithm is made uniquely through this joint kernel function.

The kernelized training algorithm can be simply describes in three steps. First, solve the dual quadratic program (5) based on the current Gram matrix $\mathbf{G}$. At iteration n this will return an n-dimensional $\boldsymbol{\alpha}^{\texttt{dual}}$. Second, use the current $\boldsymbol{\alpha}^{\texttt{dual}}$ to find the "closest" competing hypothesis $\mathbf{w}_{n+1}^{(r)}$ for each utterance r in parallel. These $\mathbf{w}_{n+1}^{(r)}|_{r=1,\ldots,R}$ will be used to compute the losses and the kernels. Third, compute the kernel values and update the Gram matrix. The process is summarized in Alg. 1. The algorithm is guaranteed to converge as long as the Gram matrix $\mathbf{G}$ is positive definite. Note that in kernelized SVMs the size of Gram matrix $\mathbf{G}_{R\times R}$ is fixed [17]; however for kernelized LLMs the size of $\mathbf{G}_{n\times n}$ is dynamic and depends on the number of "closest" competing hypotheses. In practice, the returned $\boldsymbol{\alpha}^{\texttt{dual}}$ is usually sparse. To reduce the memory cost, $\mathbf{w}_\tau^{(1)},\ldots,\mathbf{w}_\tau^{(R)}$ and the $\tau^{\texttt{th}}$ row of Gram matrix can be pruned when the corresponding $\alpha_\tau^{\texttt{dual}}$ remains 0 for more than 100 iterations.

Note that similar to SVMs, the LLM parameter $\boldsymbol{\alpha}$ can be retrieved from $\boldsymbol{\alpha}^{\texttt{dual}}$ by linearly combining the joint features of the reference and competing hypothesises,

$$\boldsymbol{\alpha} = \frac{1}{R}\sum_{\forall\tau,r} \alpha_{\tau}^{\texttt{dual}} \left[\boldsymbol{\phi}(\mathbf{O}^{(r)}, \mathbf{w}_{\texttt{ref}}^{(r)}) - \boldsymbol{\phi}(\mathbf{O}^{(r)}, \mathbf{w}_{\tau}^{(r)})\right] \quad (9)$$

3.1. Joint Kernels

To avoid working in the high-dimensional features space, in the previous section the joint kernel is introduced to replace the inner product of joint features. The point of joint kernel $K\left((\mathbf{O}^{(i)}, \mathbf{w}^{(i)}), (\mathbf{O}^{(j)}, \mathbf{w}^{(j)})\right)$ is to describe a non-linear similarity between two observation-label pairs by mapping the pairs into a joint feature space. Joint kernels have already begun to be studied in [16, 18]. In theory any function in the form of (7) can be treated as a joint kernel. To enable efficient decoding (see more in Section 3.2), denoting $\mathbf{O}^{(i)} = \{\mathbf{O}_1^{(i)}, \ldots, \mathbf{O}_k^{(i)}, \ldots\}$ and its corresponding word labels $\mathbf{w}^{(i)} = \{w_1^{(i)}, \ldots, w_k^{(i)}, \ldots\}$, this paper proposes the following joint kernel function

$$K\left((\mathbf{O}^{(i)}, \mathbf{w}^{(i)}), (\mathbf{O}^{(j)}, \mathbf{w}^{(j)})\right) = \sum_{k=1}^{|\mathbf{w}^{(i)}|}\sum_{m=1}^{|\mathbf{w}^{(j)}|} \delta(w_k^{(i)}, w_m^{(j)})\, k(\mathbf{O}_k^{(i)}, \mathbf{O}_m^{(j)}) \quad (10)$$

where $k(\cdot,\cdot)$ is a kernel commonly used in SVMs [13]. $\mathbf{O}_k^{(i)}$ is the observations for the k^{th} segment in utterance i, and $w_k^{(i)}$ is its label. One interesting property of this joint kernel is that it can be decomposed into a set of word-level kernels k. If $w_k^{(i)} \neq w_m^{(j)}$, the term δ will be zero and there is no need to compute the kernel k. This makes efficient kernel-based decoding become possible (see more in Section 3.2).

As the length of observation sequences varies, $k(\cdot,\cdot)$ should be a sequence kernel, e.g., generative kernels [9] are a good option as they allow standard model-based compensation schemes to be used to make the kernel robust to noise. In this paper, a generative kernel is combined with three commonly used static kernels,

$$\begin{aligned}
k_{\texttt{lin}}(\mathbf{O}_k, \mathbf{O}_m) &= \boldsymbol{\varphi}^{\lambda}(\mathbf{O}_k)^{\mathsf{T}}\boldsymbol{\varphi}^{\lambda}(\mathbf{O}_m)\\
k_{\texttt{poly}}(\mathbf{O}_k, \mathbf{O}_m) &= \left(\boldsymbol{\varphi}^{\lambda}(\mathbf{O}_k)^{\mathsf{T}}\boldsymbol{\varphi}^{\lambda}(\mathbf{O}_m) + b\right)^d \quad (11)\\
k_{\texttt{rbf}}(\mathbf{O}_k, \mathbf{O}_m) &= \exp\left(-\frac{1}{2\sigma^2}||\boldsymbol{\varphi}^{\lambda}(\mathbf{O}_k) - \boldsymbol{\varphi}^{\lambda}(\mathbf{O}_m)||^2\right)
\end{aligned}$$

where the $\boldsymbol{\varphi}^{\lambda}$ is a generative kernel induced feature vector

$$\boldsymbol{\varphi}^{\lambda}(\mathbf{O}) = \begin{bmatrix} \log p_{\boldsymbol{\lambda}}(\mathbf{O}|v_1) \\ \vdots \\ \log p_{\boldsymbol{\lambda}}(\mathbf{O}|v_{\text{M}}) \end{bmatrix}, \quad (12)$$

where $\boldsymbol{\lambda}$ denotes the HMM parameters, $p_{\boldsymbol{\lambda}}(\mathbf{O}|v_k)$ is the likelihood for HMM v_k, and M is the total number of HMMs. This feature space concatenates the log-likelihoods from all models, including the correct model and competing ones, to yield additional information from the observations. Other features, such as derivative features $\boldsymbol{\varphi}^{\nabla}(\mathbf{O})$ [8], can also be applied here. The relationship between the kernel k in (11) and joint kernel K in (10) can be illustrated in Fig 1.

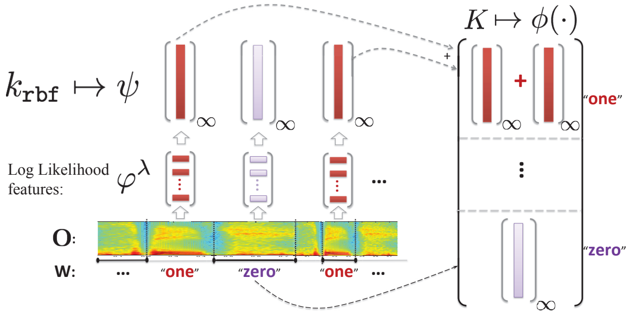

Fig. 1. An example of joint kernel K and its joint feature $\boldsymbol{\phi}$. On each segment kernel $k_{\texttt{rbf}}(\cdot,\cdot) = \boldsymbol{\psi}(\cdot)^{\mathsf{T}}\boldsymbol{\psi}(\cdot)$ is applied.

3.2. Kernel-based Decoding

Substituting equation (9) into (2), kernel-based decoding can be achieved by

$$\hat{\mathbf{w}} = \arg\max_{\mathbf{w}} \boldsymbol{\alpha}^{\mathsf{T}}\boldsymbol{\phi}(\mathbf{O}, \mathbf{w}) = \arg\max_{\mathbf{w}} \quad (13)$$
$$\sum_{\forall\tau,r}\alpha_{\tau}^{\texttt{dual}}\left[K\left((\mathbf{O}, \mathbf{w}), (\mathbf{O}^{(r)}, \mathbf{w}_{\texttt{ref}}^{(r)})\right) - K\left((\mathbf{O}, \mathbf{w}), (\mathbf{O}^{(r)}, \mathbf{w}_{\tau}^{(r)})\right)\right]$$

where $\mathbf{w}_{\tau}^{(r)}|_{\tau=1,\ldots,n}^{r=1,\ldots,R}$ are the competing word sequences found in the training phase.[1] Similar to the discriminative training in [1], lattices $\mathbb{L}$ are generated during training and decoding to restrict the search space of $\mathbf{w}$. Since the joint kernel K in (10) can be decomposed at the word level, the $\mathbf{w}$ that maximises (13) can be efficiently found via the following arc-level Viterbi search

$$\boldsymbol{\rho}_e = \max_{e'\in\mathbb{L}}\{\boldsymbol{\rho}_{e'} + s_{e'e}\} \quad (14)$$

where e is a node in the $\mathbb{L}$, e' is one of its previous nodes, and $\boldsymbol{\rho}_e$ is the best path score at node e as shown in Fig. 2. $s_{e'e}$ is the decomposed kernel scores for the arc $e'e$:

$$\sum_{\tau=1}^{n}\alpha_{\tau}^{\texttt{dual}}\sum_{\substack{\texttt{arc}=e'e\\ \texttt{arc}\in\mathcal{A}}} k(\mathbf{O}_{e'e}, \mathbf{O}_{\texttt{arc}}) - \sum_{\tau=1}^{n}\alpha_{\tau}^{\texttt{dual}}\sum_{\substack{\texttt{arc}=e'e\\ \texttt{arc}\in\mathcal{B}^{\tau}}} k(\mathbf{O}_{e'e}, \mathbf{O}_{\texttt{arc}})$$

where $\mathcal{A}$ denotes all the reference arcs in the numerator lattices and $\mathcal{B}^{\tau}$ denotes all the competing arcs in the "closest" competing hypotheses generated in (8) at iteration τ.

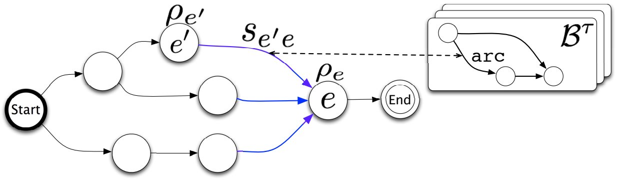

Fig. 2. Kernel-based decoding over a lattice.

[1] Similar to support vectors in the SVM classification [17], here $\mathbf{w}_{\tau}^{(r)}$ and $\mathbf{w}_{\texttt{ref}}^{(r)}$ can also be viewed as support vectors.

4. KERNEL ADAPTATION

As previously discussed, one advantage of using generative models to define the kernels (see equation (11) and (12)) is that it is possible to use state-of-the-art model-based noise robustness approaches. For standard generative models, model-based compensation schemes such as Vector Taylor Series (VTS) compensation [19] are a successful approach to handling this problem. Here, the HMM parameters $\boldsymbol{\lambda}$ associated with the kernel k are modified to represent the target acoustic environment [20]. The compensated mean and covariance for component m in $\boldsymbol{\lambda}$, are given by

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

$$\boldsymbol{\Sigma}^{(m)} = \mathbf{J}^{(m)}\boldsymbol{\Sigma}_{\mathtt{x}}^{(m)}\mathbf{J}^{(m)\mathsf{T}} + (\mathbf{I} - \mathbf{J}^{(m)})\boldsymbol{\Sigma}_{\mathtt{n}}(\mathbf{I} - \mathbf{J}^{(m)})^{\mathsf{T}}$$

where the additive noise mean $\boldsymbol{\mu}_{\mathtt{n}}$ and covariance $\boldsymbol{\Sigma}_{\mathtt{n}}$ are the parameters of the noise model estimated from the data [21]. Other terms include the DCT, matrix $\mathbf{C}$ and Jacobian matrix $\mathbf{J}^{(m)}$ are fully described in [19]. Thus in this work discriminative model parameters $\boldsymbol{\alpha}^{\mathtt{dual}}$ are noise-independent, whereas the generative model $\boldsymbol{\lambda}$ based kernel k is noise-dependent.

5. EXPERIMENTS AND CONCLUSION

The performance of the proposed kernelized LLM was evaluated on the AURORA 2 task. AURORA 2 is a standard noise-robust digit string recognition task. The vocabulary size M is 12 (one to nine, plus zero, oh and silence). The 8440 clean training utterances were used to train the acoustic generative models (HMMs). To compare with previously published results, this paper follows the same setup in [20], including the configurations of MFCCs and HMMs. Test set A was used as the development set for tuning parameters for all systems, such as the C in (4). To evaluate the performance of log linear models, large margin training and kernels, several configurations were compared. The baseline system was HMM with VTS compensation. These compensated HMMs were also used to derive: the noise robust log-likelihood features (12); the word-level segmentation for the multi-class SVMs; and the training and decoding lattices for the LLMs. The performances of VTS-compensated HMM, multi-class SVMs [9] and LLMs with different training criteria and kernels are shown in Table 1. The CML training includes L_2 regularization. Detailed results on different SNRs for Set A are also shown in Table 2.

Examining the results in Table 1 shows that the large margin LLM with 2nd order polynomial kernel achieved the best results among all the systems. For multi-class SVMs, the observation sequence is first segmented into words based on HMMs and individual words classified independently. The difference in performance between the LLM and multi-class SVM systems shows the impact of sentence-level modelling. The overall gain from using kernelized LLMs over the VTS-compensated HMM system was over 22%. The gain from using polynomial kernels over linear kernels was 3%. Note that without kernelization, it is impractical to apply large margin LLMs with a polynomial kernel, since it requires computing and keeping all the high dimensional joint features explicitly. However, in Alg. 1 only the Gram matrix is required.

Model	**Criterion**	**Kernel**	**Set A**	**Set B**	**Set C**	**Avg.**
HMM-VTS	ML	–	9.8	9.1	9.5	9.5
M-SVM	LM	linear	8.3	8.1	8.6	8.3
LLM -1	CML	linear	8.1	7.7	8.3	8.1
LLM -2	LM	linear	7.9	7.3	8.0	7.7
LLM -3	LM	2^{nd}-poly	7.6	7.1	7.9	7.5

Table 1. Results (WER %) of VTS based HMM, Multi-class SVMs [9] and LLMs trained using CML and large margin criteria, with linear and 2nd order polynomial kernels in (11).

SNR	**Test Set A**				
(dB)	HMM	M-SVM	LLM -1	LLM-2	LLM-3
20	1.7	1.5	1.4	1.3	1.1
15	2.4	2.0	1.9	1.8	1.7
10	4.4	3.6	3.5	3.3	3.2
05	11.2	9.2	8.9	8.8	8.4
00	29.6	25.1	24.9	24.1	23.8
Avg	9.8	8.3	8.1	7.9	7.6

Table 2. Results (WER %) of VTS based HMM, M-SVM and LLMs in different SNRs. LLM-1, LLM-2 and LLM-3 are the systems in Table 1.

Relation to prior work The work presented here is a kernelized version of structured LLMs perviously proposed in [7,9]. According to the large margin criterion (4), it can also be viewed as a kernelized structured SVM [16,22]. A kernelized LLM was also proposed in [15]. Note that the kernel in [15] was defined on the frame-level whereas the joint kernel in this work is defined on the sentence-level. The kernel algorithm in [15] is based on MMI, whereas the algorithm here is based on large margin training. The work in [15] is actually a low-rank approximation of kernel methods whereas in this paper the exact Gram matrix was used. To the best of our knowledge, this work is the first attempt at a sentence-level large-margin kernel method for CSR.

Conclusion This paper proposes a large margin trained log linear model with kernels for CSR. Kernelizing the LLM has two advantages. First, it avoids explicitly computing and saving the high dimensional features. Second, it introduces nonlinearity to the LLM. A kernel adaptation scheme is also described to make the system robust to noise. This work has two main contributions. First, most kernels for CSR focused on measuring the similarity between two observation sequences. The proposed *joint* kernels define a sentence-level similarity between two observation-label sequence pairs. Second, this paper addresses how to efficiently employ kernels in large margin training and decoding based on lattices. Results on AURORA 2 demonstrate that using large margin LLMs with nonlinear kernels yields significant improvements. Future work will examine RBF and derivative kernels in the proposed framework.

6. REFERENCES

[1] D. Povey, “Discriminative training for large vocabulary speech recognition,” Ph.D. dissertation, Cambridge University, 2004.

[2] F. Sha and L. K. Saul, “Large margin hidden Markov models for automatic speech recognition,” in *Neural Information Processing Systems*, 2007, pp. 1249–1256.

[3] G. Zweig and P. Nguyen, “A segmental CRF approach to large vocabulary continuous speech recognition,” in *ASRU*, 2009, pp. 152–157.

[4] O. Birkenes, T. Matsui, and K. Tanabe, “Isolated-word recognition with penalized logistic regression machines,” in *ICASSP*, vol. 1, 2006, pp. 405–408.

[5] M. I. Layton and M. J. F. Gales, “Augmented statistical models for speech recognition,” in *Proc. ICASSP*, 2006, pp. 129–132.

[6] S.-X. Zhang and M. J. F. Gales, “Structured support vector machines for noise robust continuous speech recognition,” in *Proc. Interspeech*, Florence, Italy, 2011, pp. 989–992.

[7] M. J. F. Gales, S. Watanabe, and E. Fosler-Lussier, “Structured discriminative models for speech recognition,” *Signal Processing Magazine, IEEE*, vol. 29, no. 6, pp. 70–81, 2012.

[8] A. Ragni and M. J. F. Gales, “Derivative kernels for noise robust ASR,” in *Proc. ASRU*, Hawaii, USA, 2011, pp. 119–124.

[9] S.-X. Zhang, A. Ragni, and M. J. F. Gales, “Structured log linear models for noise robust speech recognition,” *Signal Processing Letters, IEEE*, vol. 17, pp. 945–948, 2010.

[10] S. Wiesler, M. Nußbaum-Thom, G. Heigold, R. Schlüter, and H. Ney, “Investigations on features for log-linear acoustic models in continuous speech recognition,” in *ASRU*, 2009, pp. 52–57.

[11] G. Zweig *et al.*, “Speech recognitionwith segmental conditional random fields: A summary of the JHU CLSP 2010 summer workshop,” in *ICASSP*, 2011, pp. 5044–5047.

[12] B. Taskar, “Learning structured prediction models: a large margin approach,” Ph.D. dissertation, CA, USA, 2005.

[13] J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*. New York, NY, USA: Cambridge University Press, 2004.

[14] Y. Kubo, S. Watanabe, A. Nakamura, E. McDermott, and T. Kobayashi, “A sequential pattern classifier based on hidden Markov kernel machine and its application to phoneme classification.” *Journal of Selected Topics in Signal Processing*, vol. 4, no. 6, pp. 974–984, 2010.

[15] Y. Kubo, S. Wiesler, R. Schlter, H. Ney, S. Watanabe, A. Nakamura, and T. Kobayashi, “Subspace pursuit method for kernel-log-linear models.” in *ICASSP*, 2011, pp. 4500–4503.

[16] T. Joachims, T. Finley, and C.-N. J. Yu, “Cutting-plane training of structural SVMs,” *Machine Learning*, vol. 77, no. 1, pp. 27–59, 2009.

[17] V. N. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer-Verlag, 1995.

[18] J. Weston, B. Schölkopf, and O. Bousquet, “Joint kernel maps,” in *IWANN*, 2005, pp. 176–191.

[19] A. Acero, L. Deng, T. Kristjansson, and J. Zhang, “HMM Adaptation using Vector Taylor Series for Noisy Speech Recognition,” in *Proc. ICSLP*, vol. 3, 2000, pp. 869–872.

[20] M. J. F. Gales and F. Flego, “Discriminative classifiers with adaptive kernels for noise robust speech recognition,” *Comput. Speech Lang.*, vol. 24, no. 4, pp. 648–662, 2010.

[21] H. Liao and M. J. F. Gales, “Joint uncertainty decoding for robust large vocabulary speech recognition,” Cambridge University, Tech. Rep. CUED/F-INFENG/TR552, November 2006.

[22] S.-X. Zhang and M. J. F. Gales, “Structured SVMs for automatic speech recognition,” *IEEE Transactions Audio, Speech and Language Processing*, vol. 21, no. 3, pp. 544–555, 2013.