A COVARIANCE KERNEL FOR SVM LANGUAGE RECOGNITION*

W. M. Campbell

MIT Lincoln Laboratory
Lexington, MA 02420

ABSTRACT

Discriminative training for language recognition has been a key tool for improving system performance. In addition, recognition directly from shifted-delta cepstral features has proven effective. A recent successful example of this paradigm is SVM-based discrimination of languages based on GMM mean supervectors (GSVs). GSVs are created through MAP adaptation of a universal background model (UBM) GMM. This work proposes a novel extension to this idea by extending the supervector framework to the covariances of the UBM. We demonstrate a new SVM kernel including this covariance structure. In addition, we propose a method for pushing SVM model parameters back to GMM models. These GMM models can be used as an alternate form of scoring. The new approach is demonstrated on a fourteen language task with substantial performance improvements over prior techniques.

Index Terms— language recognition, support vector machines

1. INTRODUCTION

Automatic language recognition has been typically performed with two techniques. A first method is based upon phone tokenization followed by language modeling, the PPRLM approach [1]. A second approach that has recently received significant attention is methods based upon shifted-delta cepstral coefficients (SDCCs), see e.g., [2]. The PPRLM approaches have traditionally been the most accurate. The SDCC methods have the advantage that they require no specialized language knowledge (i.e., phone labeling) and are very computationally simple.

The introduction of discriminative classifier methods into SDCC-based language recognition resulted in significant improvements in performance. Some examples of discriminative methods include the following. First, SVM techniques using polynomial kernels were introduced in [3]. Second, Gaussian mixture models (GMMs) were trained with maximum mutual information (MMI) training with excellent success in [4]. Finally, methods with GMM supervectors and SVM training have been shown to work well [5].

We focus on SVM based methods for language recognition. Our starting point is recognition using GMM supervectors. The basic concept for this approach is to adapt a universal background model (UBM) GMM on a per utterance basis and then use the resulting shift in means to predict the class. The stacked adapted means form a supervector. This approach was initially used in speaker recognition [6], but was subsequently adapted to language recognition in [5].

A drawback of using only adapted means in the GMM supervector system is that significant language information is also found in the covariance structure. Earlier MAP and MMI training methods for GMMs adapt both means and covariances [2]. Incorporating this adaptation into the SVM framework requires that we have a kernel that computes distances in mean and covariance space. We propose a method based on approximations introduced in earlier work [6].

The outline of the paper is as follows. In Sections 2 and 3, we review language recognition based on SVMs and GMM mean supervectors. Section 4 outlines our new kernel incorporating both mean and covariance GMM supervectors. Section 5 discusses an alternate scoring technique for the SVM model. Finally, Section 6 details experiments on the proposed technique on a 14 language recognition task.

2. LANGUAGE RECOGNITION WITH SVMS

An SVM [7] is a two-class classifier constructed from sums of a kernel function $K(\cdot,\cdot)$,

$$f(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i K(\mathbf{x}, \mathbf{x}_i) + d, \tag{1}$$

where $\sum_{i=1}^{N} \alpha_i = 0$ and $\alpha_i \neq 0$. The vectors $\mathbf{x}_i$ are support vectors and obtained from the training set by an optimization process [8].

For language recognition, the goal is to determine the language of an utterance from a set of known languages. Since the SVM is a two-class classifier, we handle language recognition as a verification problem. That is, we use a *one vs. rest* strategy. For language recognition, we train a target model for the language. The set of known non-targets are used as the remaining class.

*This work was sponsored by the Department of Defense under Air Force Contract FA8721-05-C-0002. Opinions, interpretations, conclusions, and recommendations are those of the authors and are not necessarily endorsed by the United States Government.

3. SVM GSV LANGUAGE RECOGNITION

A straightforward method of performing language recognition with SVMs is to use kernels that compare sequences of feature vectors [3]. One technique for comparing sequences is to adapt a language independent GMM per utterance and then calculate a distance between the distributions [6].

Assuming this strategy, suppose we have a Gaussian mixture model UBM,

$$g(\mathbf{x}) = \sum_{i=1}^{N} \lambda_i \mathcal{N}(\mathbf{x}; \mathbf{m}_i, \mathbf{\Sigma}_i) \tag{2}$$

where λ_i are the mixture weights, $\mathcal{N}()$ is a Gaussian distribution, and $\mathbf{m}_i$ and $\mathbf{\Sigma}_i$ are the mean and covariance of the Gaussians distributions, respectively.

Also, assume we have two utterances, utt_a and utt_b. We train GMMs, g_a and g_b as in (2), on the two utterances, respectively, using MAP adaptation. A natural distance between the two utterances is the KL divergence,

$$D(g_a \| g_b) = \int_{R^n} g_a(\mathbf{x}) \log \left(\frac{g_a(\mathbf{x})}{g_b(\mathbf{x})} \right) d\mathbf{x} \tag{3}$$

Unfortunately, the KL divergence does not satisfy the Mercer condition, so using it in an SVM is difficult and computationally expensive.

Instead of using the divergence directly, an approximation is typically used [6]. The idea is to bound the divergence using the log-sum inequality [9],

$$D(g_a \| g_b) \leq \sum_{i=1}^{N} \lambda_i D\left(\mathcal{N}(\cdot; \mathbf{m}_{a,i}, \mathbf{\Sigma}_{a,i}) \| \mathcal{N}(\cdot; \mathbf{m}_{b,i}, \mathbf{\Sigma}_{b,i})\right) \tag{4}$$

where we have represented the ith mixture component means of the adapted supervectors by $\mathbf{m}_{a,i}$ and $\mathbf{m}_{b,i}$ and the adapted covariances are similarly denoted. A closed-form formula for the divergence between Gaussian distributions is given by

$$\begin{aligned} &D(\mathcal{N}(\cdot; \mathbf{m}_{a,i}, \mathbf{\Sigma}_i) \| \mathcal{N}(\cdot; \mathbf{m}_{b,i}, \mathbf{\Sigma}_i)) = \\ &\quad 0.5 \log \frac{|\mathbf{\Sigma}_{b,i}|}{|\mathbf{\Sigma}_{a,i}|} + 0.5 \operatorname{tr}(\mathbf{\Sigma}_{b,i}^{-1} \mathbf{\Sigma}_{a,i}) \\ &\quad + 0.5 (\mathbf{m}_{a,i} - \mathbf{m}_{b,i})^t \mathbf{\Sigma}_{b,i}^{-1} (\mathbf{m}_{a,i} - \mathbf{m}_{b,i}) - \frac{n}{2} \end{aligned} \tag{5}$$

where $\operatorname{tr}(\cdot)$ is the trace, and n is the dimension of the space. In our original work on mean supervectors, we assumed $\mathbf{\Sigma}_{a,i} = \mathbf{\Sigma}_{b,i} = \mathbf{\Sigma}_i$. In other words, only the means are adapted. Then the right hand side of (4) can be calculated in closed form (ignoring constants) as

$$d(\mathbf{m}_a, \mathbf{m}_b) = \sum_{i=1}^{N} \lambda_i (\mathbf{m}_{a,i} - \mathbf{m}_{b,i}) \mathbf{\Sigma}_i^{-1} (\mathbf{m}_{a,i} - \mathbf{m}_{b,i}). \tag{6}$$

From the distance in (6), we can find the corresponding inner product via polarization [10] which is the kernel function,

$$\begin{aligned} K(g_a, g_b) &= \sum_{i=1}^{N} \lambda_i \mathbf{m}_{a,i}^t \mathbf{\Sigma}_i^{-1} \mathbf{m}_{b,i} \\ &= \sum_{i=1}^{N} \left(\sqrt{\lambda_i} \mathbf{\Sigma}_i^{-\frac{1}{2}} \mathbf{m}_{a,i} \right)^t \left(\sqrt{\lambda_i} \mathbf{\Sigma}_i^{-\frac{1}{2}} \mathbf{m}_{b,i} \right). \end{aligned} \tag{7}$$

4. A COVARIANCE KERNEL

A covariance kernel can be derived using (4) and (5) as the starting point. First, we use the symmetrized version of the KL divergence (without dividing by 2) between two Gaussian distributions which results in (5) being transformed to

$$\begin{aligned} &D_s(\mathcal{N}(\cdot; \mathbf{m}_{a,i}, \mathbf{\Sigma}_i) \| \mathcal{N}(\cdot; \mathbf{m}_{b,i}, \mathbf{\Sigma}_i)) = \\ &\quad 0.5 \operatorname{tr}(\mathbf{\Sigma}_{b,i}^{-1} \mathbf{\Sigma}_{a,i}) + 0.5 \operatorname{tr}(\mathbf{\Sigma}_{b,i}^{-1} \mathbf{\Sigma}_{a,i}) - n \\ &\quad + 0.5 (\mathbf{m}_{a,i} - \mathbf{m}_{b,i})^t \left(\mathbf{\Sigma}_{a,i}^{-1} + \mathbf{\Sigma}_{b,i}^{-1} \right) (\mathbf{m}_{a,i} - \mathbf{m}_{b,i}); \end{aligned} \tag{8}$$

i.e., we lose the log term and have symmetric roles for $\mathbf{\Sigma}_{a,i}$ and $\mathbf{\Sigma}_{b,i}$. The resulting symmetric distance (8) is not a kernel induced distance.

We assume that $\mathbf{\Sigma}_{a,i}$ and $\mathbf{\Sigma}_{b,i}$ are diagonal. Let σ_a^2, σ_b^2, and σ^2 be corresponding diagonal terms in the matrices, $\mathbf{\Sigma}_{a,i}$, $\mathbf{\Sigma}_{b,i}$, and $\mathbf{\Sigma}_i$, respectively. Then, expanding out the trace leads to terms of the form

$$0.5 \frac{\sigma_a^2}{\sigma_b^2} + 0.5 \frac{\sigma_b^2}{\sigma_a^2} = 0.5 \frac{\sigma^2 + \delta\sigma_a^2}{\sigma^2 + \delta\sigma_b^2} + 0.5 \frac{\sigma^2 + \delta\sigma_b^2}{\sigma^2 + \delta\sigma_a^2} \tag{9}$$

where $\delta\sigma_a^2 = \sigma_a^2 - \sigma^2$ and $\delta\sigma_b^2 = \sigma_b^2 - \sigma^2$. We can do a Taylor expansion around $(\delta\sigma_a^2, \delta\sigma_b^2) = (0, 0)$ of (9) using two derivatives. This results in the approximation

$$0.5 \frac{\sigma_a^2}{\sigma_b^2} + 0.5 \frac{\sigma_b^2}{\sigma_a^2} \approx 1 + 0.5 \left(\frac{\delta\sigma_a^2 - \delta\sigma_b^2}{\sigma^2} \right)^2. \tag{10}$$

We can rewrite (8) using the approximation (10),

$$\begin{aligned} &D_s(\mathcal{N}(\cdot; \mathbf{m}_{a,i}, \mathbf{\Sigma}_i) \| \mathcal{N}(\cdot; \mathbf{m}_{b,i}, \mathbf{\Sigma}_i)) \approx \\ &\quad 0.5 \operatorname{tr}((\mathbf{\Sigma}_{a,i} - \mathbf{\Sigma}_{b,i}) \mathbf{\Sigma}_i^{-2} (\mathbf{\Sigma}_{a,i} - \mathbf{\Sigma}_{b,i})) \\ &\quad + (\mathbf{m}_{a,i} - \mathbf{m}_{b,i})^t \left(0.5 \mathbf{\Sigma}_{a,i}^{-1} + 0.5 \mathbf{\Sigma}_{b,i}^{-1} \right) (\mathbf{m}_{a,i} - \mathbf{m}_{b,i}) \end{aligned} \tag{11}$$

The resulting approximation (11) is almost a kernel. The main difficulty at this point is that the mean vectors are interacting with the covariances in a coupled manner. One way around this difficulty is to replace the average of the two covariances in the second term on the right hand side of (11) with $\mathbf{\Sigma}_i$, see (2), from the UBM. Performing this operation results in

our proposed kernel,

$$K(g_a, g_b) = \sum_{i=1}^{N} \lambda_i \mathbf{m}_{a,i}^t \boldsymbol{\Sigma}_i^{-1} \mathbf{m}_{b,i} + \sum_{i=1}^{N} \frac{\lambda_i}{2} \operatorname{tr}\left(\boldsymbol{\Sigma}_{a,i} \boldsymbol{\Sigma}_i^{-2} \boldsymbol{\Sigma}_{b,i}\right) \tag{12}$$

We note that our goal in performing the above approximations is to satisfy the Mercer condition. An alternate approach would be to use a more accurate approximation, e.g. (11), with an SVM training package that handles "approximate" kernels. Our approach simplifies the model pushing process described in Section 5.

5. MODEL PUSHING

The kernel (12) can be used in an SVM training paradigm to obtain a set of models per target language. Standard SVM scoring would use the kernel at test time with the scoring in (1). An alternate method recently proposed for scoring is to transfer the SVM supervector back to a GMM model and then score using the GMM model. In certain situations, especially on short duration utterances, this interesting approach has improved accuracy, see [5].

We propose an alternate method for *pushing* an SVM model to a GMM model via some heuristic ideas. Our basic approach is to use a GMM to model boundaries rather than class distributions. One difficulty that should be noted is that covariance parameters must be positive when they are pushed back to the GMM model.

Our starting point is to observe that the standard kernel scoring in (1) can be split as follows,

$$f(\mathbf{x}) = \sum_{\{i|\alpha_i>0\}} \alpha_i K(\mathbf{x}, \mathbf{x}_i) - \sum_{\{i|\alpha_i<0\}} \alpha_i K(\mathbf{x}, \mathbf{x}_i) + d. \tag{13}$$

The two terms on the right hand side resemble a log-likehood ratio between an in-class and out-of-class model [11]. Since we are working with a linear kernel, we can write (13) as

$$f(\mathbf{x}) = \mathbf{w}_p^t \mathbf{D}\mathbf{x} - \mathbf{w}_m^t \mathbf{D}\mathbf{x} + d. \tag{14}$$

where $\mathbf{D}$ is a diagonal matrix and

$$\mathbf{w}_p = \sum_{\{i|\alpha_i>0\}} \alpha_i \mathbf{x}_i, \qquad \mathbf{w}_m = \sum_{\{i|\alpha_i<0\}} \alpha_i \mathbf{x}_i. \tag{15}$$

For the kernel (12), the vectors $\mathbf{x}_i$ and $\mathbf{x}$ will be the stacked means and covariances of the GMM models.

A natural observation is to view the vectors, $\mathbf{w}_p$ and $\mathbf{w}_m$, as representations of in-class and out-of-class data and incorporate them into GMM models. One strategy is to normalize the vectors to produce means and covariances. A simple method is to use the support vector weights; e.g., for the positive class, we obtain

$$\mathbf{x}_p = \frac{1}{\sum_{\{i|\alpha_i>0\}} \alpha_i} \sum_{\{i|\alpha_i>0\}} \alpha_i \mathbf{x}_i. \tag{16}$$

The vector $\mathbf{x}_p$ can then be transferred to a GMM model $g_p(\mathbf{x})$. A similar process yields a negative model, $g_m(\mathbf{x})$. It is easy to verify that the covariances for this process are positive.

Intuitively, the combination (16) is on the hyperplane boundary that supports the in-class data. A similar statement can be made for $\mathbf{x}_m$. Thus, the resulting pushed GMMs are modeling the location of the positive and negative boundaries of the classes. Another interesting observation is that this method weights data unequally—this contrasts to a standard EM technique that would weight these statistics equally.

Scoring with these GMM models is straightforward. For an input set of vectors, $\mathbf{y}_i$, we produce a log likelihood ratio,

$$\text{score} = \sum_i \log\left(g_p(\mathbf{y}_i)\right) - \sum_i \log\left(g_m(\mathbf{y}_i)\right). \tag{17}$$

6. EXPERIMENTS

Experiments were performed using a 14 language task in anticipation of the NIST 2007 language recognition evaluation (LRE). Target languages include Arabic, Bengali, Chinese, English, Farsi, German, Hindustani, Japanese, Korean, Russian, Spanish, Tamil, Thai, and Vietnamese.

Training data was primarily from Callfriend and Callhome; although, for languages such as Arabic, data was also used from Fisher and Mixer. Our test set, LRE07 DEV, included approximately 6000 utterances per duration for durations of 3, 10, and 30 seconds. The test set included all trials from NIST LRE 2005. Additional data was supplied by LDC for Arabic, Bengali, Thai, and Chinese dialects. For reported NIST 2005 LRE results, the performance is calculated on the primary data set; NIST defined this as the OHSU subset of the NIST 2005 LRE corpus.

The criterion for evaluation used is pooled EER. In the EER calculation, we used the priors of languages in the test set so that each language had an equal contribution to the EER. This strategy corresponds to the NIST scoring criterion which balances priors for the minDCF score.

For feature extraction, SDCC features were used with a 7-1-3-7 parameterization [2]. This corresponds to 7 delta-cepstral coefficients stacked from 7 different time locations. We also included cepstral coefficients for a total of 56 features per frame at 100 frames per second. Additional processing included RASTA, 0/1 feature normalization, and VTLN. Finally, a feature space version of NAP [12] (fNAP) was used analogous to the feature domain version of factor analysis in [13]. The corank for the fNAP projection was 128, and the mixture order was 256. The projection was designed using the training set with variation from the target language as the nuisance variable.

For a language and gender independent GMM UBM, we trained a 1024 mixture GMM using all of the training data with 5 iterations of EM adapting all parameters—means, mixture weights, and diagonal covariances.

Table 1. Comparison of EER for different models and training methods with a 30s test. In the table, M denotes mean, V denotes covariance, and P denotes scoring with a pushed model. Results for LRE05 are on the primary condition.

System	LRE07 DEV EER(%)	LRE05 EER (%)
GMM MAP	13.53	11.52
GMM MAP BE	6.56	6.65
SVM M	6.16	5.34
SVM MV	5.70	4.75
SVM MP	4.28	4.01
SVM MVP	3.47	3.74

Table 2. Comparison of EERs for different models and training methods at different durations for the LRE07 DEV set.

	30s	10s	3s
GMM MAP BE	6.56	10.83	22.02
SVM MP	4.28	10.30	23.66
SVM MVP	3.47	8.74	22.18

We implemented SVMs with both mean-only (SVM M) and mean plus covariance kernels (SVM MV) using the kernels given by (7) and (12) respectively. We also pushed these models back to GMM models for scoring as described in Section 5. As a baseline for the SVM systems, we trained a GMM model using mean and diagonal convariance adaptation with MAP adaptation using a maximum of 5 iterations.

For all systems, using the raw scores for classification was suboptimal. For all systems, final scores were processed using a "max" log-likelihood ratio,

$$s'_j = s_j - \max_{i \neq j} s_j. \tag{18}$$

In addition, for the GMM MAP trained model, a backend (BE) was used [14] for calibration. BE transforms scores using linear discriminant analysis and models the resulting vector using a tied covariance Gaussian per language. Note that the SVM systems scored in the standard manner required Z-norm based on a development set for proper calibration.

Results for the various systems are shown in Table 1. First, note that all SVM systems outperform the baseline MAP system. Second, we see that with both standard scoring and GMM scoring, the MV systems outperform the M only systems. Third, we see that using GMM scoring improves accuracy quite substantially; another benefit is that scores did not need to be normalized. Overall, the best result on LRE05, 3.74%, compares favorably with past systems on this task [5, 15].

Table 2 shows a breakout of the performance of the methods for different durations. Again, we see that the new MV kernel outperforms the M only kernel. Performance at 10s is substantially better for the new kernel including covariances. We note that 3s is a difficult task and none of the methods do particularly well at this duration.

7. CONCLUSIONS

A new kernel for SVM GMM language recognition was demonstrated that used both mean and covariance parameters. This kernel was combined with a GMM scoring technique to produce a system that performed well on both a standard NIST LRE task and a new development task.

8. REFERENCES

[1] M. Zissman, "Comparison of four approaches to automatic language identification of telephone speech," *IEEE Trans. Speech and Audio Processing*, vol. 4, no. 1, pp. 31–44, 1996.

[2] P. A. Torres-Carrasquillo, E. Singer, M. A. Kohler, R. J. Greene, D. A. Reynolds, and J. R. Deller, Jr., "Approaches to language identification using Gaussian mixture models and shifted delta cepstral features," in *International Conference on Spoken Language Processing*, 2002, pp. 89–92.

[3] W. M. Campbell, P. A. Torres-Carrasquillo, and D. A. Reynolds, "Language recognition with support vector machines," in *Proc. of Odyssey 04*, 2004, pp. 285–288.

[4] Lukas Burget, Pavel Matejka, and Jan Cernocky, "Discriminative training techniques for acoustic language identification," in *Proceedings of ICASSP*, 2006, pp. 209–212.

[5] Fabio Castaldo, Daniele Colibro, Emanuele Dalmasso, Pietro Laface, and Claudio Vair, "Acoustic language identification using fast discriminative training," in *Proc. Interspeech*, 2007.

[6] W. M. Campbell, D. E. Sturim, D. A. Reynolds, and A. Solomonoff, "SVM based speaker verification using a GMM supervector kernel and NAP variability compensation," in *Proceedings of ICASSP*, 2006, pp. I–97–I–100.

[7] Nello Cristianini and John Shawe-Taylor, *Support Vector Machines*, Cambridge University Press, Cambridge, 2000.

[8] Ronan Collobert and Samy Bengio, "SVMTorch: Support vector machines for large-scale regression problems," *Journal of Machine Learning Research*, vol. 1, pp. 143–160, 2001.

[9] Minh N. Do, "Fast approximation of Kullback-Leibler distance for dependence trees and hidden Markov models," *IEEE Signal Processing Letters*, vol. 10, no. 4, pp. 115–118, 2003.

[10] John B. Conway, *Functional Analysis*, Springer-Verlag, 1990.

[11] W.M. Campbell, J.P. Campbell, D.A. Reynolds, E. Singer, and P.A. Torres-Carrasquillo, "Support vector machines for speaker and language recognition," *Computer Speech & Language*, vol. 20, no. 2-3, pp. 210–229, 2006.

[12] Alex Solomonoff, Carl Quillen, and William M. Campbell, "Channel compensation for SVM speaker recognition," in *Proceedings of Odyssey-04, The Speaker and Language Recognition Workshop*, 2004, pp. 57–62.

[13] C. Vair, D. Colibro, F. Castaldo, E. Dalmasso, and P. Laface, "Channel factors compensation in model and feature domain for speaker recognition," in *Proc. IEEE Odyssey: The Speaker and Language Recognition Workshop*, 2006.

[14] W. M. Campbell, T. Gleason, J. Navratil, D. Reynolds, W. Shen, E. Singer, and P. Torres-Carrasquillo, "Advanced language recognition using cepstra and phonotactics: MITLL system performance on the NIST 2005 language recognition evaluation," in *Proc. IEEE Odyssey*, 2006.

[15] P. Matejka, L. Burget, P. Schwarz, and J. Cernocky, "Brno University of Technology system for NIST 2005 language recognition evaluation," in *Proc. IEEE Odyssey: The Speaker and Language Recognition Workshop*, 2006.