A FRAMEWORK FOR DISCRIMINATIVE SVM/GMM SYSTEMS FOR LANGUAGE RECOGNITION[*]

W. M. Campbell[†], *Z. Karam*[†‡]

[†]MIT Lincoln Laboratory, Lexington, MA
[‡]DSPG, Research Laboratory of Electronics at MIT, Cambridge MA

ABSTRACT

Language recognition with support vector machines and shifted-delta cepstral features has been an excellent performer in NIST-sponsored language evaluation for many years. A novel improvement of this method has been the introduction of hybrid SVM/GMM systems. These systems use GMM supervectors as an SVM expansion for classification. In prior work, methods for scoring SVM/GMM systems have been introduced based upon either standard SVM scoring or GMM scoring with a *pushed* model. Although prior work showed experimentally that GMM scoring yielded better results, no framework was available to explain the connection between SVM scoring and GMM scoring. In this paper, we show that there are interesting connections between SVM scoring and GMM scoring. We provide a framework both theoretically and experimentally that connects the two scoring techniques. This connection should provide the basis for further research in SVM discriminative training for GMM models.

***Index Terms*—** language recognition, support vector machines

1. INTRODUCTION

Automatic language techniques with shifted-delta cepstral coefficients (SDCCs) have become an attractive method for efficient language recognition [1]. These methods take input speech, convert to SDCCs, and then perform classification directly on the resulting sequence of feature vectors. Standard techniques used for classification are support vector machines (SVMs) [2] and Gaussian mixture models (GMMs) [1].

Several discriminative methods for GMMs and SVMs have been introduced in the literature. Discriminative techniques for SDCCs using SVMs with a polynomial kernel were introduced in the NIST 2003 language recognition evaluation (LRE) [2]. Maximum mutual information (MMI) training of GMMs was used in the NIST 2005 NIST LRE [3]. Finally, methods with GMM supervectors and SVM training were introduced in the NIST 2007 LRE [4, 5].

We focus on the method in [5] which is based upon three main components. The first component is the use of a feature domain subspace method, fNAP, which is an extension of the NAP technique [6, 7]. The second component is a mean and covariance kernel for comparing GMM supervectors. The use of both parameter types is critical for good performance in language recognition applications. The final component, which we explore in detail, is transferring the SVM model parameters back to a GMM model and then using that model for scoring. We call this process *model pushing*.

In a prior paper, a technique for model pushing was given based upon heuristic considerations [5]. Other model pushing techniques are possible, see [4]. In this paper, we show that the technique in [5] can be justified as an approximation to an exact technique. To this end, we give a theoretical correspondence between the SVM scoring and GMM scoring. Then, with Monte Carlo simulations and corresponding approximations, we demonstrate the validity of the methods.

The outline of the paper is as follows. In Section 2, we discuss prior work on language recognition based on SVMs and GMM supervectors. The covariance kernel discussed in prior work [5] is presented. Section 3 discusses the model pushing technique from [5]. Section 4 gives a theoretical justification for model pushing. Finally, Section 5 details experiments on the proposed framework using data from the NIST 2007 LRE.

2. LANGUAGE RECOGNITION WITH SVMS

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 each language; the set of known non-targets are used as the remaining class.

The standard form of an SVM, $f(\mathbf{x})$, is

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

where the $K(\cdot, \cdot)$ is a kernel function and $\sum_{i=1}^{N} \alpha_i = 0$ and $\alpha_i \neq 0$. The support vectors $\mathbf{x}_i$ and d are obtained by optimization.

A straightforward method of performing language recognition with SVMs is to use kernels that compare sequences of feature vectors [2]. One technique for comparing sequences is to adapt a language independent GMM per utterance and then calculate a distance between the distributions [5]. Assuming this strategy, suppose we have a Gaussian mixture model UBM,

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

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, and 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,

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

[*] 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.

Unfortunately, the KL divergence does not satisfy the Mercer condition usually assumed for SVM training.

In prior work, instead of using the divergence directly, several approximations were made to obtain a kernel-induced distance [5]. The first approximation involved the log-sum inequality,

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

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. Note that we have switched to the symmetrized form of the KL divergence, $D_s()$.

A closed-form formula for the symmetric divergence between Gaussian distributions is given by

$$\begin{aligned} &D_s(\mathcal{N}(\cdot; \mathbf{m}_{a,i}, \mathbf{\Sigma}_i) \| \mathcal{N}(\cdot; \mathbf{m}_{b,i}, \mathbf{\Sigma}_i)) = \\ &\quad 0.5\,\mathrm{tr}(\mathbf{\Sigma}_{b,i}^{-1}\mathbf{\Sigma}_{a,i}) + 0.5\,\mathrm{tr}(\mathbf{\Sigma}_{b,i}^{-1}\mathbf{\Sigma}_{a,i}) - n \\ &\quad + 0.5(\mathbf{m}_{a,i} - \mathbf{m}_{b,i})^t \left(\mathbf{\Sigma}_{a,i}^{-1} + \mathbf{\Sigma}_{b,i}^{-1}\right)(\mathbf{m}_{a,i} - \mathbf{m}_{b,i}); \end{aligned} \quad (5)$$

where $\mathrm{tr}(\cdot)$ is the trace, and n is the dimension of the feature space.

At this point, it is tempting to attempt to use (4) with (5) in an SVM framework. The difficulty is that the combination is not a kernel-induced distance. Therefore, in prior work, two additional approximations were performed. We assume that $\mathbf{\Sigma}_{a,i}$ and $\mathbf{\Sigma}_{b,i}$ are diagonal. The first additional approximation was to expand the trace terms in a Taylor expansion,

$$\begin{aligned} &D_s(\mathcal{N}(\cdot; \mathbf{m}_{a,i}, \mathbf{\Sigma}_i) \| \mathcal{N}(\cdot; \mathbf{m}_{b,i}, \mathbf{\Sigma}_i)) \approx \\ &\quad 0.5\,\mathrm{tr}((\mathbf{\Sigma}_{a,i} - \mathbf{\Sigma}_{b,i})\mathbf{\Sigma}_i^{-2}(\mathbf{\Sigma}_{a,i} - \mathbf{\Sigma}_{b,i})) \\ &\quad + (\mathbf{m}_{a,i} - \mathbf{m}_{b,i})^t \left(0.5\mathbf{\Sigma}_{a,i}^{-1} + 0.5\mathbf{\Sigma}_{b,i}^{-1}\right)(\mathbf{m}_{a,i} - \mathbf{m}_{b,i}). \end{aligned} \quad (6)$$

A second additional approximation was,

$$0.5\mathbf{\Sigma}_{a,i}^{-1} + 0.5\mathbf{\Sigma}_{b,i}^{-1} \approx \mathbf{\Sigma}_i^{-1} \quad (7)$$

where $\mathbf{\Sigma}_i$ is the covariance from the UBM, see (2). Performing this operation resulted in the distance,

$$\begin{aligned} D_s(g_a \| g_b) \approx\,& d^2(\mathbf{x}_a, \mathbf{x}_b) \\ &= \sum_{i=1}^{N} \lambda_i \big[(\mathbf{m}_{a,i} - \mathbf{m}_{b,i})^t \mathbf{\Sigma}_i^{-1}(\mathbf{m}_{a,i} - \mathbf{m}_{b,i}) \\ &\quad + 0.5\,\mathrm{tr}((\mathbf{\Sigma}_{a,i} - \mathbf{\Sigma}_{b,i})\mathbf{\Sigma}_i^{-2}(\mathbf{\Sigma}_{a,i} - \mathbf{\Sigma}_{b,i})) - n\big] \end{aligned} \quad (8)$$

and the corresponding covariance kernel in [5],

$$K(\mathbf{x}_a, \mathbf{x}_b) = \sum_{i=1}^{N} \lambda_i \mathbf{m}_{a,i}^t \mathbf{\Sigma}_i^{-1} \mathbf{m}_{b,i} + \sum_{i=1}^{N} \frac{\lambda_i}{2}\,\mathrm{tr}\left(\mathbf{\Sigma}_{a,i}\mathbf{\Sigma}_i^{-2}\mathbf{\Sigma}_{b,i}\right). \quad (9)$$

where $\mathbf{x}_a$ is the supervector of means and covariances for g_a.

To summarize, applying SVMs to language recognition involves several steps. To train language models, the mean and covariance parameters of a UBM are MAP adapted to each utterance in the training corpus. Then, an SVM is trained per language using the kernel (9). For standard SVM scoring, the input test utterance is used to adapt a UBM to obtain mean and covariance parameters and these are input to the scoring equation (1).

3. MODEL PUSHING

Instead of using standard SVM scoring (1), several researchers have explored the possibility of transferring the SVM parameters back to a GMM model and using standard log likelihood ratio scoring [5, 4]. The challenge in this paradigm is to understand the exact role of the SVM parameters in GMM scoring.

The approach presented in [5] was heuristically motivated. The starting point is to observe that the standard kernel scoring in (1) can be split as follows,

$$f(\mathbf{x}) = \sum_{\{i|\alpha_i>0\}} \alpha_i K(\mathbf{x}, \mathbf{x}_i) - \sum_{\{i|\alpha_i<0\}} \alpha_i K(\mathbf{x}, \mathbf{x}_i) + d. \quad (10)$$

The two terms on the right hand side resemble a log likelihood ratio between an in-class and out-of-class model. Since we are working with a symmetric linear kernel, we can write (10) as

$$f(\mathbf{x}) = cK(\mathbf{x}_p, \mathbf{x}) - cK(\mathbf{x}_m, \mathbf{x}) + d. \quad (11)$$

where

$$\mathbf{x}_p = \frac{1}{c}\sum_{\{i|\alpha_i>0\}} \alpha_i \mathbf{x}_i, \qquad \mathbf{x}_m = -\frac{1}{c}\sum_{\{i|\alpha_i<0\}} \alpha_i \mathbf{x}_i \quad (12)$$

and

$$c = \sum_{\{i|\alpha_i>0\}} \alpha_i. \quad (13)$$

For the kernel (9), the vectors $\mathbf{x}_p$, $\mathbf{x}_m$, etc., will be supervectors of stacked means and covariances.

A natural observation is to view the vectors, $\mathbf{x}_p$ and $\mathbf{x}_m$, as representations of in-class and out-of-class data and incorporate them into GMM models. Since the process in (12) is a convex combination of supervectors, it produces a valid GMM parameter instance; i.e., the covariances are positive. Thus, we can transfer $\mathbf{x}_p$ to a GMM model, $g_p(\mathbf{y})$. A similar process yields a negative model, $g_m(\mathbf{y})$.

Intuitively, the combination for $\mathbf{x}_p$ in (12) 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,

$$\text{score} = \sum_i \log\left(g_p(\mathbf{y}_i)\right) - \sum_i \log\left(g_m(\mathbf{y}_i)\right). \quad (14)$$

4. RELATING MODEL PUSHING TO GMM SCORING

We define the cross-entropy between two distributions as

$$H(p, q) = -\int p(x)\log q(x)dx. \quad (15)$$

We also denote the entropy of $p(x)$ by $H(p)$. Note that

$$D(p\|q) = H(p, q) - H(p). \quad (16)$$

Suppose we have a test utterance with feature vectors, $\mathbf{y}_1, \ldots, \mathbf{y}_m$ and we adapt a GMM UBM to this utterance to produce a GMM, $g_y(\mathbf{y})$. We can rewrite the standard SVM scoring equation (1) in terms of distances. That is, if $d(\mathbf{x}_y, \mathbf{x}_z)$ is the distance in (8), then

$$\begin{aligned} d^2(\mathbf{x}_p,\mathbf{x}_y) &- d^2(\mathbf{x}_m,\mathbf{x}_y) \\ &= K(\mathbf{x}_p,\mathbf{x}_p) - 2K(\mathbf{x}_p,\mathbf{x}_y) + K(\mathbf{x}_y,\mathbf{x}_y) \\ &\quad - K(\mathbf{x}_m,\mathbf{x}_m) + 2K(\mathbf{x}_m,\mathbf{x}_y) - K(\mathbf{x}_y,\mathbf{x}_y) \\ &= -2\left(K(\mathbf{x}_p,\mathbf{x}_y) - K(\mathbf{x}_m,\mathbf{x}_y)\right) + d_1. \end{aligned} \tag{17}$$

Now combining the SVM scoring equation (11) and (17) yields,

$$f(\mathbf{x}_y) = c_1(d^2(\mathbf{x}_p,\mathbf{x}_y) - d^2(\mathbf{x}_m,\mathbf{x}_y)) + d_2. \tag{18}$$

Note that the constant d_2 does not depend on g_y. Also, in the separable class case, $c_1 = -c/2$ and $d_2 = 0$ where c is given in (13).

Now, we can use the fact that

$$d^2(\mathbf{x}_y,\mathbf{x}_z) \approx D_s(g_y \| g_z) \tag{19}$$

and the SVM distance score (18) to give

$$f(\mathbf{x}_y) \approx c_1(D_s(g_y\|g_p) - D_s(g_y\|g_m)) + d_2 \tag{20}$$

The next connection that we make is that cross-entropy can be approximated using the law of large numbers. That is,

$$\text{if } \mathbf{y}_i \sim g_y(\mathbf{y}), \text{ then} \tag{21}$$

$$H(g_y,g_p) - H(g_y,g_m) = \lim_{M\to\infty} \frac{-1}{M}\sum_{i=1}^{M} \log\left(\frac{g_p(\mathbf{y}_i)}{g_m(\mathbf{y}_i)}\right) \tag{22}$$

where $\sim$ indicates "distributed as." That is, likelihood scoring is the same as cross-entropy.

Also, note that

$$D(g_y,g_p) - D(g_y,g_m) = H(g_y,g_p) - H(g_y,g_m) \tag{23}$$

so

$$D(g_y,g_p) - D(g_y,g_m) = \lim_{M\to\infty} \frac{-1}{M}\sum_{i=1}^{M} \log\left(\frac{g_p(\mathbf{y}_i)}{g_m(\mathbf{y}_i)}\right). \tag{24}$$

We additionally assume that

$$D_s(g_y,g_p) - D_s(g_y,g_m) \approx D(g_y,g_p) - D(g_y,g_m). \tag{25}$$

Putting (20), (24), and (25) together gives us our proposed relation between SVM scoring and GMM scoring,

$$f(\mathbf{x}_y) \approx \frac{-c_1}{M}\sum_{i=1}^{M} \log\left(\frac{g_p(\mathbf{y}_i)}{g_m(\mathbf{y}_i)}\right) + d_2 \tag{26}$$

for large M. The approximation (25) is difficult to demonstrate analytically, and we will show through experiments in the next section the validity of the method.

As an alternate approximation, we also derived another relation based on (20),

$$f(\mathbf{x}_y) \approx c_1(H_s(g_y\|g_p) - H_s(g_y\|g_m)) + d_3 \tag{27}$$

where d_3 does not depend upon g_y. This approximation (27) is compelling, but to use GMM scoring to approximate this quantity, we would have to perform standard scoring (26) as well as performing Monte-Carlo generation of points and GMM scoring using g_p and g_m. We leave exploration of this method to future work.

The distance interpretation of SVM scoring in (18) and (20) has a compelling geometric interpretation. For a given test distribution, g_y, we find the distance to the g_p distribution which is on the positive class boundary and the distance to the g_m distribution which is on the negative class boundary. The SVM score is based on which of these distance is smaller; i.e., which distribution, g_p or g_m, is closer to g_y .

5. EXPERIMENTS

Experiments were performed on development and evaluation sets from the NIST 2007 LRE 14 language closed-set evaluation task [6]. 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 development set, LRE07 DEV, included approximately 6000 utterances per duration for durations of 3, 10, and 30 seconds. The development set included all trials from NIST LRE 2005. Additional data was supplied by LDC for Arabic, Bengali, Thai, and Chinese dialects. The criterion for evaluation used is pooled EER; we used the priors of languages in the test set so that each language had an equal contribution to the EER.

For feature extraction, SDCC features were used with a 7-1-3-7 parameterization [1]. 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, fNAP features [6] were produced with a corank 128, a mixture order of 256, and variation from the target language as the nuisance variable.

For a language and gender independent GMM UBM, we trained a 512 mixture GMM using all of the training data with 5 iterations of EM adapting all parameters—means, mixture weights, and diagonal covariances. We implemented SVMs with the mean plus covariance kernel given in (9). We also pushed these models back to GMM models for scoring as described in Section 3.

The resulting system is used for exploring the approximations presented in Section 3. The basic performance of the systems is given in Table 1. In the table, scores were processed scores were processed using a "max" log likelihood ratio,

$$s_j' = s_j - \max_{i\neq j} s_j. \tag{28}$$

Also, a backend (BE) was used [8] for calibration. BE transforms scores using linear discriminant analysis and models the resulting vector using a tied covariance Gaussian per language. Note that GMM scoring performs better for both the max transformation and BE. This resulting system is very competitive with other NIST 2007 LRE systems.

Our first set of experiments was to explore the approximations (6), (7) to the KL divergence for a single Gaussian. We used random Gaussian mixture components from our LID data and calculated actual values versus approximated values. The mean of the

Table 1. Performance for the base system on the NIST 2007 LRE 30 second closed set task for 14 languages.

System	LRE07 EER (%)
SVM, Max	3.25
SVM, GMM score, Max	2.54
SVM, BE	2.53
SVM, GMM score, BE	1.70

Table 2. Performance of various KL approximations on NIST LRE data for a single Gaussian mixture component

Approximation	Mean of the Relative Error (%)
KL Approx 1 (6)	0.08
KL Approx 2 (7)	0.14

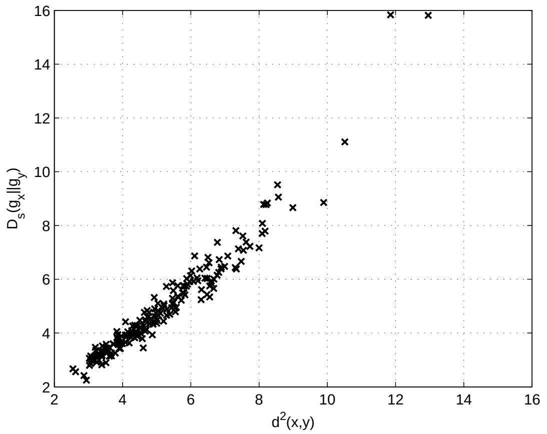

Fig. 1. MC D_s and approximation on NIST LRE data

relative error, $|\mathrm{actual} - \mathrm{approx}|/|\mathrm{actual}|$ between the actual and approximate values is shown in Table 2. The table shows that our KL approximations have about $1 - 2$ digits of accuracy. The table shows that the covariance by the UBM in (6) degrades accuracy.

In our next set of experiments, we performed a comparison of the full approximation to the KL divergence (8) to a Monte-Carlo (MC) approximation of the exact symmetric KL divergence $D_s(g_y\|g_z)$. For the distances, we used mixture models adapted from the UBM on our LID experimental data. For these experiments, approximately 100 samples per mixture component were used in the MC runs. This number of samples was found to produce reasonable accuracy. A scatter plot of the approximation and the corresponding MC D_s is shown in Figure 1. Note from the figure that the general trend is well established and reasonable between the two values.

In the next set of experiments, we explored the approximation between symmetric and non-symmetric KL divergence differences, see (25) and Figure 2. We used LID data again with the same MC setup. The figure justifies our approximation.

As a final set of experiments, we performed both GMM scoring and SVM scoring on LID data. A plot of the GMM score for various trials for a fixed model is shown in Figure 3. The figure shows the relation between the two methods and illustrates our final approximation (26).

6. CONCLUSIONS

We have demonstrated in this paper a framework for understanding the relation between KL-based SVM systems and log-likelihood ratio GMM scoring. Our paper justified prior model pushing techniques for transferring SVM parameters to a GMM. Through various MC simulations were able to demonstrate that the approximations were reasonable and justified the theoretical development. Further work is necessary to understand the source of degradations in the approximations. This work could potentially explain why GMM scoring and SVM scoring have different language recognition performance.

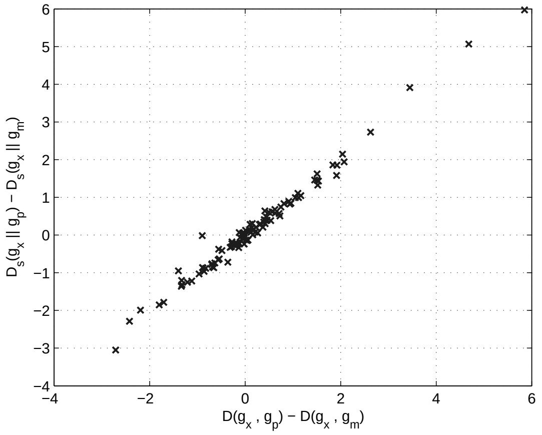

Fig. 2. Comparison of MC $D_s(g_y\|g_p) - D_s(g_y\|p)$ versus MC $D(g_y\|g_p) - D(g_y\|g_p)$, see (25)

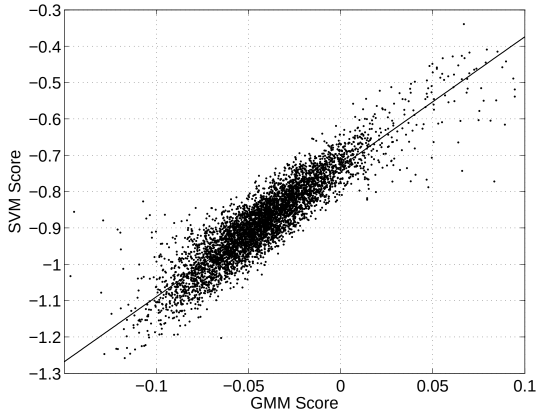

Fig. 3. Comparison of GMM score and SVM score for Arabic target model. A linear fit is included to show the trend.

7. REFERENCES

[1] 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 *Proc. ICSLP*, 2002, pp. 89–92.

[2] 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.

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

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

[5] W. M. Campbell, "A covariance kernel for SVM language recognition," in *Proceedings of ICASSP*, 2008, pp. 4141–4144.

[6] W. M. Campbell, D. E. Sturim, P. Torres-Carrasquillo, and D. A. Reynolds, "A comparison of subspace feature-domain methods for language recognition," in *Proc. Interspeech*, 2008.

[7] 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*, 2006.

[8] 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.