DISCRIMINATIVE FEATURES FOR LANGUAGE IDENTIFICATION

Chris Alberti, Michiel Bacchiani

Google Inc., New York

ABSTRACT

In this paper we investigate the use of discriminatively trained feature transforms to improve the accuracy of a MAP-SVM language recognition system. We train the feature transforms by alternatively solving an SVM optimization on MAP supervectors estimated from transformed features, and performing a small step on the transforms in the direction of the antigradient of the SVM objective function. We applied this method on the LRE2003 dataset, and obtained an 5.9% relative reduction of pooled equal error rate.

Index Terms— Language recognition, support vector machines, discriminative feature transforms.

1. INTRODUCTION

Language recognition is an important component of any spoken language application where lightweight decisions have to be made to route utterances in several languages to the correct language-specific processing, for example to the correct automatic speech recognition system. In recent years, a popular approach to language recognition has been the MAP-SVM method [1] [2] [3], also used in speaker verification [4].

In this paper we investigate the effect of a discriminatively trained feature transform on the accuracy of a MAP-SVM language-id system. Inspired by the fMPE method [5], we propose transforms that shift the observations in dependence of their posteriors. The method fits naturally in the MAP-SVM approach, since the components of the universal background model can function as the classes for which posterior probabilities are computed. In contrast to fMPE where the feature transform is estimated by minimizing the phone error rate, we instead estimate the feature transform by optimizing the objective function of the SVM.

Section 2 gives an overview of a typical MAP-SVM language-id system. In Section 3, we describe our proposed discriminative feature transform. Experiments on the LRE2003 dataset are reported in Section 4, and Section 5 concludes the paper.

2. LANGUAGE RECOGNITION USING SVMS

Assume a dataset of utterances is given. For utterance u, let $l_u \in \{1, 2, .., L\}$ be the known language label, and let $o_u(t)$, for $t \in \{1, 2, ..., T_u\}$ be a sequence of feature vectors (or observations) extracted from the utterance audio waveform. Let the dataset be partitioned in four parts with no overlap in speaker, and let the partitions be labeled "UbmTrain", "SvmTrain", "Dev", and "Eval".

An N component Gaussian Mixture Model (GMM) with diagonal covariances, referred to as the "Universal Background Model" (UBM), is trained on "UbmTrain". Let λ_n, μ_n, Σ_n, for $n \in \{1, 2, ..., N\}$, be the Gaussian mixture component weights, means and covariance matrices of the UBM. The UBM is used to estimate an utterance specific adapted model on each of the utterances in the remaining partitions. The estimation is performed using Maximum A Posteriori (MAP) adaptation [6] of the means. For a given utterance u, the MAP adapted model is an N component GMM with the same weights and covariance matrices as the UBM, and with means

$$\mu_{u,n} = \frac{\tau\mu_n + \sum_{t=1}^{T_u} \gamma_{u,n}(t) o_u(t)}{\tau + \sum_{t=1}^{T_u} \gamma_{u,n}(t)}, \quad (1)$$

where τ is the parameter of the conjugate prior distribution used in MAP adaptation [6], and $\gamma_{u,n}(t)$ is the posterior probability that $o_u(t)$ was emitted by the n-th UBM component

$$\gamma_{u,n}(t) = \frac{\lambda_n \mathcal{N}(o_u(t), \mu_n, \Sigma_n)}{\sum_{k=1}^{N} \lambda_k \mathcal{N}(o_u(t), \mu_k, \Sigma_k)}. \quad (2)$$

For every utterance u, the means of the corresponding adapted model are rescaled

$$\hat{\mu}_{u,n} = \lambda_n^{\frac{1}{2}} \Sigma_n^{-\frac{1}{2}} \mu_{u,n}, \quad (3)$$

and stacked in a single column vector $\phi_u = [\hat{\mu}_{u,1} \dots \hat{\mu}_{u,N}]^{\mathsf{T}}$ dubbed "supervector". The rescaling in equation 3 is motivated by the fact that it makes the distance between supervectors an upper bound of the KL divergence between models adapted to the corresponding utterances [1].

L support vector machines (SVM) with linear kernels [7], one for each language, are trained on the supervectors computed from the utterances in "SvmTrain". For each language l, the SVM training is done by solving the problem

$$\alpha_l^* = \text{argmax}_\alpha \sum_u \alpha_u - \frac{1}{2} \sum_{u,v} \alpha_u \alpha_v y_u y_v \phi_u^{\mathsf{T}} \phi_v$$
$$\text{s.t. } 0 \le \alpha_u \le C \ \wedge \ \sum_u \alpha_u y_u = 0, \quad (4)$$

where u and v enumerate the utterances in "SvmTrain", α is a set of optimization variables α_u, C is the hinge loss parameter [7], and $y_u = +1$, if $l = l_u$, -1 otherwise. This is known as the "dual" formulation of the SVM optimization problem [7].

Once the optimization problem is solved the solution will be in the form of a sequence of $\alpha^*_{u,l}$. The solution can be turned into a linear classifier computing score $d_l(u)$ for utterance u in language l as

$$d_l(u) = \phi_u^\mathsf{T} w_l^* + b_l^*, \tag{5}$$

where

$$w_l^* = \sum_v \alpha^*_{v,l} y_v \phi_v, \;\; b_l^* = y_{v'} - \sum_v \alpha^*_{v,l} y_v \phi_v^\mathsf{T} \phi_{v'}, \tag{6}$$

for any utterance v'. The margin of the l-th SVM is defined as $\rho_l = 1/||w_l^*||$ and it is related to the generalization power of the that SVM classifier [7].

The L scores $d_1(u), \ldots, d_L(u)$ produced by the SVMs are merged by a fusion module, in our case a maximum entropy classifier, trained on "Dev". The accuracy of the system is finally tested on "Eval".

3. TRAINING A DISCRIMINATIVE FEATURE TRANSFORM

In this work, we propose changing the system described in section 2 by introducing a language specific feature transform. In analogy to [5], we propose using the transform

$$x_u(t) = o_u(t) + M h_u(t), \tag{7}$$

where $h_u(t)$ is a high dimensional sparse feature vector calculated at every frame t, and M is a matrix that projects $h_u(t)$ down to the dimensionality of $o_u(t)$. We then use the transformed features $x_u(t)$, rather than $o_u(t)$, in equations 1 and 2 to compute the supervectors that are input to the SVM. This transform choice is advantageous in that it provides a flexible way of incorporating frame level additional information in the training procedure. It is also practical since it admits a convenient initialization by setting all elements of M to zero, and is straightforward to differentiate, making it feasible to implement gradient methods to estimate M. A separate transform is trained for every language, however in the following we omit for simplicity of notation the l index.

In this work we fix $h_u(t)$ to be the N dimensional vector of posteriors of $o_u(t)$ with respect to the UBM components. For this choice of $h_u(t)$, the dimensionality of M is $D \times N$, where D is the dimensionality of $o_u(t)$.

We estimate M by optimizing an objective function related to the discriminative capacity of our classifier. We choose to minimize the SVM objective function from equation 4,

$$J(\alpha, M) = \sum_u \alpha_u - \frac{1}{2} \sum_{u,v} \alpha_u \alpha_v y_u y_v \phi_u(M)^\mathsf{T} \phi_v(M), \tag{8}$$

where we write ϕ_u and ϕ_v as a function of M to emphasize that it is through the supervectors that the dependency on the feature transform comes in. Minimizing $J(\alpha, M)$ corresponds to maximizing the margin of the SVM classifier since, for fixed M, $J(\alpha^*, M) = 1/(2\rho^2)$. Therefore the problem we wish to solve is

$$\begin{aligned} &(\alpha_l^*, M^*) = \mathrm{argmax}_\alpha \mathrm{argmin}_M J(\alpha, M) \\ &\text{s.t. } 0 \le \alpha_u \le C \;\; \wedge \;\; \sum_u \alpha_u y_u = 0. \end{aligned} \tag{9}$$

As a computationally feasible solution, we propose to iteratively (step 1) solve the SVM optimization to find α^* for a fixed feature transform M, and then (step 2) perform a small step on M in the direction of the antigradient of $J(\alpha^*, M)$. Instead of explicitly imposing a restriction on the norm of M, we allow only a small number of gradient descent steps, using as stopping condition the fact that we reach a minimum of the error rate on a development set. The first step amounts to re-estimating the MAP supervectors for the new feature transform, and rerunning the SVM solver. The second step applies the update

$$M_{i,j} := M_{i,j} - \nu_{i,j} \frac{dJ}{dM_{i,j}}, \tag{10}$$

where $M_{i,j}$ is the element at row i and column j of the transform matrix M, and $\nu_{i,j}$ is the learning rate for that particular element. A way of efficiently computing $\frac{dJ}{dM_{i,j}}$ is discussed in section 3.1. The choice of $\nu_{i,j}$ is described in section 3.2.

3.1. Gradient Computation

The key quantity used in gradient descent is $\frac{dJ}{dM_{i,j}}$. In the following we will show how to compute this derivative efficiently by performing two passes over the training utterances.

Using total derivative and equation 7 we have

$$\frac{dJ}{dM_{i,j}} = \sum_{u,t} \frac{dJ}{dx_{u,i}(t)} \frac{dx_{u,i}(t)}{dM_{i,j}} = \sum_{u,t} \frac{dJ}{dx_{u,i}(t)} h_{u,j}(t), \tag{11}$$

where $x_{u,i}(t)$ is the i-th component of the transformed observation $x_u(t)$, $h_{u,j}(t)$ is the j-th component of the feature vector $h_u(t)$. In vector notation we can rewrite the above as

$$\frac{dJ}{dM} = \sum_{u,t} (\nabla_{x_u(t)} J) h_u^\mathsf{T}(t), \tag{12}$$

where $\frac{dJ}{dM}$ is a matrix holding all the derivatives of J with respect to the components of M. The dimensionality of $\frac{dJ}{dM}$ is $D \times N$, that of $\nabla_{x_u(t)} J$ is $D \times 1$, and that of $h_u(t)$ is $N \times 1$.

Inserting equation 8 in equation 12, and observing now

that $\nabla_{x_u(t)}\phi_v = 0$ for $u \neq v$, we have

$$\begin{aligned}\nabla_{x_u(t)}J &= -\nabla_{x_u(t)}\sum_{v',v}\alpha^*_{v'}\alpha^*_v y_{v'}y_v\phi^\mathsf{T}_{v'}\phi_v \\ &= -\sum_v \alpha^*_u\alpha^*_v y_u y_v(\nabla_{x_u(t)}\phi^\mathsf{T}_u)\phi_v \\ &= -\alpha^*_u y_u(\nabla_{x_u(t)}\phi^\mathsf{T}_u)\sum_v \alpha^*_v y_v\phi_v. \quad (13)\end{aligned}$$

The matrix $\nabla_{x_u(t)}\phi^\mathsf{T}_u$ has dimension $D \times DN$ and is the horizontal stacking of matrices $\nabla_{x_u(t)}\hat{\mu}^\mathsf{T}_{u,n}$,

$$\nabla_{x_u(t)}\phi^\mathsf{T}_u = \left[\begin{array}{ccc}\nabla_{x_u(t)}\hat{\mu}^\mathsf{T}_{u,1} & \ldots & \nabla_{x_u(t)}\hat{\mu}^\mathsf{T}_{u,N}\end{array}\right]. \quad (14)$$

We can express the derivative of the rescaled mean (equation 3) as

$$\nabla_{x_u(t)}\hat{\mu}^\mathsf{T}_{u,n} = \lambda_n^{\frac{1}{2}}\Sigma^{-\frac{1}{2}}\nabla_{x_u(t)}\mu^\mathsf{T}_{u,n}, \quad (15)$$

and the derivative of the adapted mean (equation 1) as

$$\nabla_{x_u(t)}\mu^\mathsf{T}_{u,n} = \frac{I\gamma_{u,n}(t) + (\nabla_{x_u(t)}\gamma_{u,n}(t))(x^\mathsf{T}_u(t) - \mu^\mathsf{T}_{u,n})}{\tau + \sum_{t=1}^T \gamma_{u,n}(t)}, \quad (16)$$

where I is a $D \times D$ identity matrix. Finally the gradient of GMM posteriors with respect to the observation is

$$\nabla_{x_u(t)}\gamma_{u,n}(t) = \gamma_{u,n}(t)\sum_{n'=1}^N \Sigma^{-1}_{n'}(\mu_{n'} - x_u(t))(\delta_{n,n'} - \gamma_{u,n'}(t)), \quad (17)$$

where $\delta_{n,n'}$ is Kronecker's delta.

In summary, to compute the gradient of $J(\alpha^*, M)$ with respect to M, we iterate over the observations of the utterances with $\alpha^*_u > 0$ twice:

1. in the first pass we compute the supervectors ϕ_u with the current feature transform,
2. in the second pass we compute $\nabla_{x_u(t)}J$ using equations 13-17 for every frame t, and cumulate the resulting gradient updates as in equation 12.

3.2. Learning Rate

To choose the learning rate we adopt the same procedure as for fMPE [5]. We refer the reader there for justification and discussion. The learning rate is computed as

$$\nu_{i,j} = \frac{\sigma_i}{E(p_{i,j} + n_{i,j})}, \quad (18)$$

where σ_i is the average standard deviation of dimension i for the Gaussians in the UBM, E is a constant controlling the general learning rate, and

$$\begin{aligned}p_{i,j} &= \sum_{u,t}\max\left(\frac{dJ}{dx_{u,i}(t)}h_{u,j}(t), 0\right), \\ n_{i,j} &= \sum_{u,t}\min\left(\frac{dJ}{dx_{u,i}(t)}h_{u,j}(t), 0\right), \quad (19)\end{aligned}$$

are the accumulation of the positive and the negative part of the transform update for every training data frame.

	30s	10s	3s	pooled
Baseline	4.12%	8.78%	18.24%	12.87%

Table 1. Baseline equal error rates on the "Eval" partition of LRE2003, where the SVM classifiers are trained on all of the "Train" and "Dev" partitions of LRE2003.

	30s	10s	3s	pooled
Baseline	7.2%	13.4%	23.4%	18.6%
Feat-trans	8.4%	14.1%	23.9%	17.5%

Table 2. Equal error rates on the "Eval" partition of LRE2003, where the SVM classifiers are trained on the "Train" partition of LRE2003, while the "Dev" partition is used for determining the stopping condition of gradient descent. "Baseline" is a trained using the original features, "Feat-trans" using the transformed observations.

4. EXPERIMENTS

We evaluated the proposed method using the CallFriend1996 and the LRE2003 datasets. CallFriend1996 contains 900 data samples, in 12 languages, each nominally 30 minutes long. LRE2003 contains 11830 data samples, in 12 languages, of nominal length 3s, 10s and 30s.

The feature type used in this work is "Static plus Shifted Delta Cepstral Coefficients" (SSDCC) with parametrization 7-1-3-7 [8][9]. A speech/non-speech segmenter trained on a separate dataset is used to discard non-speech feature vectors from the utterances. The speech feature vectors are then normalized on a per utterance basis so that the mean and variance are 0 and 1 respectively. We trained a 1024 mixture component UBM on the "Train" partition of CallFriend1996. We used $\tau = 16$ for MAP adaptation, and a hinge loss $C = 1$ for SVM training. For gradient computation we thresholded posteriors at 10^{-2}. For the learning rate we used an E such that the margin increases by a few percents at the first iteration. To fuse the scores produced by the per-language SVMs, we trained a maximum entropy classifier on the "Dev" partition of LRE2003. Finally, we evaluated the performance of the system on the "Eval" partition of LRE2003.

Running SVM training on the partitions "Train" and "Dev" of LRE2003 (7990 instances) gives the equal error rates shown in table 1, which are directly comparable to the ones reported in [1].

For the method proposed in this paper we needed however to have a development set to determine the stopping conditions of gradient descent, so we ran SVM training only on the "Train" partition of LRE2003 (3493 instances), and used the "Dev" partition to determine the stopping condition. Figure 1 shows the evolution of the average margin as a function of the number of gradient descent iterations. For a suf-

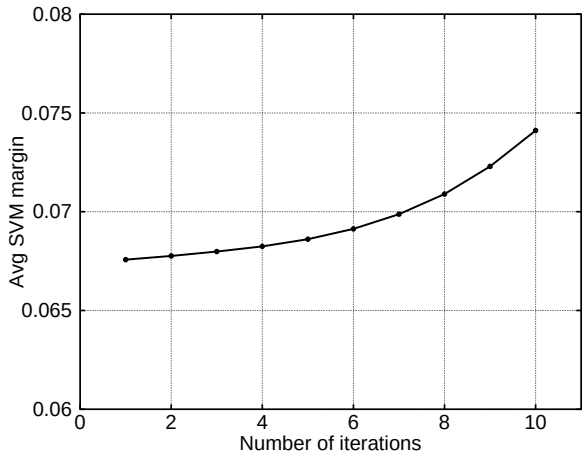

Fig. 1. Average margin across the 12 SVM language classifiers as a function of the number of iterations of gradient descent.

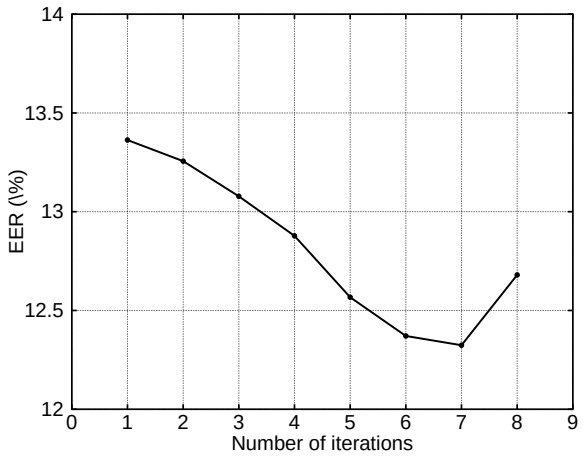

Fig. 2. Pooled equal error rate on the "Dev" partition of LRE2003, as a function of the number of iterations of SVM margin gradient descent.

ficiently small step size, the algorithm guarantees the margin to be non-decreasing and the plot shows that the algorithm succeeds in increasing the margin by training the transforms. Finally figure 2 shows the evolution of equal error rate on the "Dev" partition of LRE2003: the equal error rate decreases from 13.4% to 12.3% over the course of 7 training steps.

Table 2 shows the evaluation results on the "Eval" partition of LRE2003. Pooled equal error rate decreases from 18.6% to 17.5%, which is a 5.9% relative error rate reduction. Unfortunately the improvement in pooled equal error rate does not hold when EERs are computed separately on test instances of different durations. This is not surprising as the SVM margin we are maximizing only guarantees improvement for the overall accuracy, and not for the accuracy of specific subsets of the test data.

5. CONCLUSION

In this paper we experimented incorporating UBM posteriors into a feature transform to improve the accuracy of a MAP-SVM system for language recognition. We proposed a new algorithm to train the feature transform discriminatively by running gradient descent on the SVM dual objective, which guarantees a non-decreasing classification margin. We found that the proposed algorithm correctly drives the SVM margin, and that the accuracy of the resulting classifier improves after a small number of iterations. After many iterations the accuracy starts dropping, possibly because we are increasing the margin excessively at the expense of the error on the training data.

The algorithm we proposed has proven to be effective at incrementing the classification margin. We showed that maximizing the margin is however insufficient in this setting, as it is possible to continue increasing it without achieving an improvement in classification accuracy. A solution to this problem could be adding a regularization term penalizing either the norm of M or directly the norm of the supervectors in the training set. In future work we plan to investigate this issue further, as well as to experiment with larger feature vectors $h_u(t)$ obtained e.g. by stacking posteriors over consecutive observations.

6. REFERENCES

[1] WM Campbell, E. Singer, PA Torres-Carrasquillo, and DA Reynolds, "Language recognition with support vector machines," in *Odyssey: The Speaker and Language Recognition Workshop*, vol. 4, p. 3.

[2] F. Castaldo, D. Colibro, E. Dalmasso, P. Laface, and C. Vair, "Acoustic language identification using fast discriminative training," in *Proc. Interspeech*, 2007, pp. 346–349.

[3] WM Campbell, DE Sturim, DA Reynolds, and A. Solomonoff, "SVM based speaker verification using a GMM supervector kernel and NAP variability compensation," in *Proc. ICASSP 2006*, vol. 1.

[4] WM Campbell, DE Sturim, and DA Reynolds, "Support vector machines using GMM supervectors for speaker verification," *IEEE Signal Processing Letters*, vol. 13, no. 5, pp. 308–311, 2006.

[5] D. Povey, B. Kingsbury, L. Mangu, G. Saon, H. Soltau, and G. Zweig, "fMPE: Discriminatively trained features for speech recognition," in *Proc. ICASSP*, 2005, vol. 1, pp. 961–964.

[6] J.L. Gauvain and C.H. Lee, "Maximum a posteriori estimation for multivariate Gaussian mixture observations of Markov chains," *IEEE Transactions on Speech and Audio Processing*, vol. 2, no. 2, pp. 291–298, 1994.

[7] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[8] P.A. Torres-Carrasquillo, E. Singer, M.A. Kohler, R.J. Greene, D.A. Reynolds, and JR Deller Jr, "Approaches to language identification using Gaussian mixture models and shifted delta cepstral features," in *Proc. ICSLP*, 2002.

[9] WM Campbell, "A covariance kernel for SVM language recognition," in *Proc. ICASSP 2008*, pp. 4141–4144.