REGULARIZED ADAPTATION OF DISCRIMINATIVE CLASSIFIERS

Xiao Li and Jeff Bilmes

Department of Electrical Engineering
University of Washington
Seattle, WA 98195-2500
{lixiao,bilmes}@ee.washington.edu

ABSTRACT

We introduce a novel method for adapting discriminative classifiers (multi-layer perceptrons (MLPs) and support vector machines (SVMs)). Our method is based on the idea of regularization, whereby an optimization cost criterion to be minimized includes a penalty in accordance to how "complex" the system is. Specifically, our regularization term penalizes depending on how different an adapted system is from an unadapted system, thus avoiding the problem of overtraining when only a small amount of adaptation data is available. We justify this approach using a max-margin argument. We apply this technique to MLPs and produce a working real-time system for rapid adaptation of vowel classifiers in the context of the Vocal Joystick project. Overall, we find that our method outperforms all other MLP-based adaptation methods we are aware of. Our technique, however, is quite general and can be used whenever rapid adaptation of MLP or SVM classifiers are needed (e.g., from a speaker-independent to a speaker-dependent classifier in a hybrid MLP/HMM or SVM/HMM speech-recognition system).

1. INTRODUCTION

The idea of adaptation in automatic speech recognition (ASR) has become one of the crucial techniques that any state-of-the-art ASR system cannot do without. Why adaptation works rests on the assumption that test data $\mathcal{D}_{TE} = \{(\mathbf{x}_t, y_t)|(\mathbf{x}_t, y_t) \sim P_{TE}\}_{t=1}^{|\mathcal{D}_{TE}|}$ is drawn from an inherently different probability distribution than that of (most of) the training data $\mathcal{D}_{TR} = \{(\mathbf{x}_t, y_t)|(\mathbf{x}_t, y_t) \sim P_{TR}\}_{t=1}^{|\mathcal{D}_{TR}|}$ i.e., a *training-data distribution* P_{TR} and a *test-data distribution* P_{TE} are different. Only a small or a fixed-size amount of data from the test distribution, called the *adaptation data*, $\mathcal{D}_{AD} = \{(\mathbf{x}_t, y_t)|(\mathbf{x}_t, y_t) \sim P_{TE}\}_{t=1}^{|\mathcal{D}_{AD}|}$, is available (for unsupervised adaptation, we do not know the y_t values in $\mathcal{D}_{AD}$). Therefore, no matter how much data is available from the training-data distribution, $|\mathcal{D}_{TR}| \to \infty$, it will not be possible to obtain an asymptotically consistent estimate of P_{TE}. On the other hand, training a model using only the adaptation data would [1] either lead to: 1) overfitting, due to a high-variance parameter estimate of a complex model, or 2) high-bias due to estimating the parameters of an excessively simple model.

Many techniques for adaptation (often applied to hidden Markov models) have been proposed in the past. This includes MLLR and MAP methods (summarized in [2]) and Eigenvoice methods [3]. Many methods for the adaptation of discriminatively trained multi-layer perceptrons (MLPs) have also been proposed [4, 5, 6, 7, 8].

In this paper, we make the observation that the notion of "regularization" can be applied to the adaptation framework. Regularization is a general idea applicable to regression, classification, machine learning, and statistical model selection, and is an example of Occam's razor, Kolmogorov complexity, Bayesian parameter estimation, and the principles of both maximum entropy and minimum description length. The essential component is that one desires a model, on the one hand, to achieve a low empirical error on training data, but on the other hand, one wishes that model to be as simple as possible. Typically, one has control over the tradeoff between these two (often conflicting) goals using a tradeoff coefficient which carves out what is now called the regularization path [9] or more recently the "accuracy-regularization frontier." Indeed, this is the essence behind the success of the support-vector and Kernel methods [10], where the function to minimize is a sum of a (hinge) loss function and a regularizer (typically the ℓ_p-norm, where $p = 2$).

We extend this idea to that of "adaptation", where we produce an *adapted model* (e.g., a speaker-dependent model) by training using a regularizer that penalizes distance from an *unadapted model* (e.g., a speaker-independent (SI) model) rather than from the simplest model possible. The unadapted model is one that presumably will have been robustly trained using a large amount of training data $\mathcal{D}_{TR}$ — thus, the unadapted model becomes our exemplar of simplicity. Our tradeoff coefficient, moreover, controls the degree of adaptation. We call this general idea *regularized adaptation*.

Note that there are a number of ways this can be applied. For example, in the context of a maximum entropy estimation, we maximize entropy of some distribution (regularizer) subject to some constraints (training data). This is equivalent to minimizing over $\mathcal{P}$ in the Kullback-Leibler (KL) divergence $D(\mathcal{P}||\mathcal{P}_0)$ with respect to a uniform distribution $\mathcal{P}_0$ (regularizer) subject to the same constraints. We alternatively could minimize the KL-divergence relative to some unadapted distribution $\mathcal{P}_1$ subject to constraints corresponding to adaptation data. MLLR [2] itself can be seen as regularized adaptation, where the number of regression classes is viewed as the tradeoff coefficient. In previous work, moreover, we [8] demonstrated how regularized adaptation can be applied in the support vector machine framework, where some of the support vectors produced from the training set are included within the adaptation data, to produce a resulting support vector machine (SVM) that is both trained on the adaptation data and regularized towards the training set.

We have empirically found, however, that SVM training is computationally expensive, especially given the large training set sizes that we have (many hundreds of thousands of many hundred dimensional feature vectors). MLPs [11], however, can be rapidly and discriminatively trained even when the training sets are extremely large [12] and even when using only simple first-order (e.g., gradient descent) training methods [11]. Moreover, ℓ_2-regularization of the MLP parameters is easily applicable to MLP training [13, 11].

This material is based on work supported by the National Science Foundation under grant IIS-0326382.

Therefore, regularized adaptation also works under the MLP framework.

This paper applies regularized adaptation to MLPs — our approach produces an adaptation strategy that is both rapid (requires only a small amount of adaptation data, and the adaptation algorithm is fast) and that works well. Indeed, our method outperforms all other MLP-based adaptation methods we are aware of.

2. ACCURACY, REGULARIZATION, MLPS, & SVMS

The accuracy-regularization view of classification establishes strong relationships between MLPs and SVMs [14]. For a binary classification problem, the discriminant functions of both SVMs and MLPs take on the same general form:

$$d(\mathbf{x_t}) = \langle \mathbf{w}, \Phi_\theta(\mathbf{x}_t) \rangle + b \tag{1}$$

where $\Phi_\theta(\cdot)$ is a nonlinear transformation, parameterized by θ, from the input space $\mathbf{x_t}$ to a feature space where linear classification takes place. Given this mapping, and considering only the optimization of the last layer, the training objectives for regularized MLPs [11] and SVMs can be written as a regularized error function as follows:

$$\min_{\mathbf{w},b} \quad \tfrac{\mu}{2}\|\mathbf{w}\|^2 + \tfrac{1}{T}\textstyle\sum_{t=1}^{T} Q(y_t, d(\mathbf{x_t})) \tag{2}$$

where $\|\mathbf{w}\|^2$ is the squared ℓ_2 norm, $Q(\cdot)$ is a loss function, and μ is an accuracy-regularization tradeoff coefficient. Note $d(\cdot)$ is a function of $(\mathbf{w}, b)$ defined in Equation (1).

For SVMs, $\Phi_\theta(\cdot)$ is a particular input-to-feature space mapping that implicitly defines a reproducing kernel [15] $k(\mathbf{x}, \mathbf{x}') = \langle \Phi_\theta(\mathbf{x}), \Phi_\theta(\mathbf{x}') \rangle$. For binary SVMs, moreover, we typically assume a hinge loss function, where:

$$\xi_t \triangleq Q(y_t, f(\mathbf{x_t})) = \max(0, 1 - y_t d(\mathbf{x_t})). \tag{3}$$

Since this is not a continuous function, it cannot be solved using stochastic gradient descent (as can an MLP) but it can readily be solved using quadratic programming with linear constraints to express the error:

$$\xi_t \geq 0 \quad \text{and} \quad \xi_t \geq 1 - y_t d(\mathbf{x_t}). \tag{4}$$

This can be formulated as a constrained convex optimization problem solved in the dual space with a unique optimal solution, and the "kernel trick" can be used so that $\Phi_\theta(\cdot)$ is never explicitly evaluated.

For MLPs, on the other hand, Equation (2) is equivalent to training the last layer of an MLP with "weight decay" [11] which serves as the regularization term. Here, $\Phi_\theta(\cdot)$ corresponds to the mapping produced by the input-to-hidden layer of the network:

$$\Phi_\theta(\mathbf{x_t}) = \mathbf{h_t} = [\phi_1(\mathbf{x}_t), \ldots, \phi_K(\mathbf{x}_t)]^T$$

where $\phi_k(\mathbf{x}_t) = g(\langle \mathbf{v_k}, \mathbf{x}_t \rangle + d_k))$ is a non-linearity (typically as sigmoid function) applied to $\langle \mathbf{v}_k, \mathbf{x_t} \rangle$, and $\theta = [\mathbf{v_1} \ldots \mathbf{v_K}]^T$ is the MLP's input-to-hidden weight matrix, and where $\mathbf{v_k}^T$ is the k^{th} row of this matrix. Cross entropy is most popularly used as the error function $Q(\cdot)$ due to the desirable property that in such a case the trained MLP outputs can be interpreted as posterior probabilities [11]. In fact, in previous work [8], an MLP-trained $\Phi_\theta(\cdot)$ function was used as an input-to-feature space mapping for subsequent max-margin training using an SVM, producing what we now call the "MLP-kernel."

Indeed, as the community learns more about both MLPs and SVMs, their interpretation appears to be converging towards one idea. A key difference is training, e.g., with MLPs, both layers are trained; and from the SVM perspective, the MLP simultaneously learns both the linear separator and the parameterized kernel. With an SVM, however, the kernel is fixed at training time. Moreover, [14] proved that an MLP trained with either weight decay or early stopping can be viewed as max-margin training, and [16] showed that the use of cross-entropy as an error function also has a max-margin interpretation.

3. REGULARIZED ADAPTATION

We next introduce justification for regularized adaptation from a SVM/max-margin viewpoint and the ideas of Section 2 show that our results apply just as easily to MLPs.

Let $(\mathbf{w}^*, b^*)$ be the optimum parameters found (either using SVM or MLP training) for the unadapted model and let $d^*(\mathbf{x_t}) \triangleq \langle \mathbf{w}^*, \Phi_\theta(\mathbf{x}_t) \rangle + b^*$ be the discriminant function for this unadapted model. These parameters are presumed to be constants during any adaptation data training.

We generalize the constraints in an SVM as follows:

$$\begin{array}{rl} \displaystyle\min_{\mathbf{w},b} & \tfrac{\mu}{2}\|\mathbf{w}\|^2 + \tfrac{1}{T}\sum_{t=1}^{T} \xi_t \\ \text{subject to} & \xi_t \geq 0; \\ & \xi_t \geq \psi(\mathbf{x_t}); \end{array} \tag{5}$$

In this equation, we depict the minimization of $\mathbf{w}$ subject to various constraints over *only* the adaptation data $\mathcal{D}_{AD}$. The forms of $\psi(\cdot)$ yield various forms of adaptation strategies as we next show.

First, if we let $\psi(\mathbf{x_t}) = 1 - y_t d(\mathbf{x_t})$, then Equation (5) is equivalent to Equation (2) under the hinge loss, but where the adaptation process only uses the adaptation data and entirely ignores information from the training data. This means that if the adaptation sample has a large margin ($y_t d(\mathbf{x}_t)$ is large and positive), then there is no additional charge to the error penalty.

Next, if we let $\psi(\mathbf{x_t}) = \delta(y_t d^*(\mathbf{x_t}) < 0)(1 - y_t d(\mathbf{x_t}))$, an adaptation sample is used only when it is misclassified by the unadapted model, creating a form of "hard boosting."

Finally, our new approach combines the margins of the unadapted and adapted model. Specifically, we let $\psi(\mathbf{x_t}) = 1 - y_t d(\mathbf{x_t}) - \alpha y_t d^*(\mathbf{x_t})$ where $\alpha \geq 0$. Intuitively, if an adaptation sample has a large margin with respect to the decision boundary of the unadapted model, we decrease the importance of any margin error made by the adapted model in its total contribution to the loss. In other words, if $\alpha y_t d^*(\mathbf{x_t})$ is large and positive, then we do not incur a penalty even if $y_t d(\mathbf{x_t})$ is less than one. We next see that this yields a generalized objective that can avoid overfitting on adaptation data.

We rewrite the above constraint as:

$$\begin{array}{rcl} \psi(\mathbf{x_t}) & = & 1 - y_t(\langle(\mathbf{w} + \alpha\mathbf{w}^*), \phi(\mathbf{x_t})\rangle + b + \alpha b^*) \\ & = & 1 - y_t(\langle \mathbf{w}', \phi(\mathbf{x_t})\rangle + b') \end{array} \tag{6}$$

where $\mathbf{w}' = \mathbf{w} + \alpha\mathbf{w}^*$ and $b' = b + \alpha b^*$ are the new adapted parameters. Therefore, minimizing over $(\mathbf{w}, b)$ with $(\alpha\mathbf{w}^*, b^*)$ fixed (Equation (5)) is equivalent to minimizing over $(\mathbf{w}', b')$ with a regularizer of $\|\mathbf{w}' - \alpha\mathbf{w}^*\|^2$, yielding the objective criterion:

$$\begin{array}{rl} \displaystyle\min_{\mathbf{w}',b'} & \tfrac{\mu}{2}\|\mathbf{w}' - \alpha\mathbf{w}^*\|^2 + \tfrac{1}{T}\sum_{t=1}^{T} \xi_t \\ \text{subject to} & \xi_t \geq 0; \\ & \xi_t \geq 1 - y_t(\langle \mathbf{w}', \phi(\mathbf{x_t})\rangle + b'); \end{array} \tag{7}$$

In other words, we no longer minimize the ℓ_2 norm of the weight vector alone, but rather the complexity penalty comes from the normed difference from the scaled unadapted model $\alpha\mathbf{w}^*$. Therefore, we have proven that regularized adaptation corresponds to max-margin training with modified constraints. Note that we have recently discovered a similar formulation (where $\alpha = 1$) referred to as "biased regularization" [17] but which was not used for adaptation.

From Equation (2), we easily see how our objective function extends to MLPs (we continue this discussion with $\alpha = 1$ for simplicity), where we get:

$$\min_{\mathbf{w},b} \quad \frac{\mu}{2}\|\mathbf{w} - \mathbf{w}^*\|^2 + \frac{1}{T}\sum_{t=1}^{T} Q(y_t, d(\mathbf{x_t})) \tag{8}$$

As with the above, instead of regularizing towards zero, we do so towards an unadapted model, and we incur a penalty in accordance to large deviations from this model. Moreover, we can extend this to a multi-class two-layer MLP where we regularize each of the input-to-hidden $\mathbf{W}_{i2h}$ and the hidden-to-output $\mathbf{W}_{h2o}$ weight matrices with separate tradeoff coefficients ν and μ:

$$\min_{\substack{\mathbf{W}_{h2o},\mathbf{W}_{i2h}\\ \mathbf{b}_{h2o},\mathbf{b}_{i2h}}} \left(\frac{\mu}{2}\|\mathbf{W}_{h2o} - \mathbf{W}^*_{h2o}\|^2 + \frac{\nu}{2}\|\mathbf{W}_{i2h} - \mathbf{W}^*_{i2h}\|^2 + \frac{1}{T}\sum_{t=1}^{T} Q(\mathbf{y_t}, \mathbf{d}(\mathbf{x_t}))\right)$$

where $\|A\|^2 = tr(AA^T)$. Due to the mathematical tractability of the ℓ_2-norm, all of the above MLP objectives can be easily optimized using stochastic gradient descent [11], which is typically much faster than the quadratic-programming-like procedures needed for SVM optimization.

There have in fact been a number of adaptation methods for MLPs introduced in the past. First, there is RLL *(Retrained only Last-Layer)*, discussed in [6, 7, 8]. This approach starts from the SI model and retrains only the last layer of the MLP. In [7], a portion of the last layer is adapted to minimize cross-entropy, while in [6] and [8], the last layer is re-estimated, under a maximal margin objective, by combining adaptation data with support vectors of the user-independent system. Note that this approach is akin to our approach where $\mu = 0$ and $\nu \to \infty$ thereby disallowing any change in the first layer. Next, there is RSI *(Retrained Speaker-Independent)*, discussed in [4]. This approach starts from the SI model and retrains the entire network, usually with early stopping applied to avoid overfitting. This is akin to $\mu = \nu = 0$. Last, there is LIN *(Linear Input Network)*, discussed in [4, 5] where an MLP is augmented with an additional linear transformation input layer, keeping the rest of the network fixed. The number of free parameters is often further reduced using parameter tying. Since a cascade of linear transforms is a linear transform, this method is also akin to $\mu \to \infty$ and $\nu = 0$. We also define in this paper a new method, entitled RFL *(Retrained First Layer)* which also corresponds to $\mu \to \infty$ and $\nu = 0$. Of course, all of these methods are generalized by varying the μ and ν tradeoff coefficients.

4. APPLICATION: THE VOCAL JOYSTICK

The above procedure was developed in the context of the Vocal Joystick (VJ) project [18], which is a human-computer interface system that enables individuals with motor impairments to control computing devices (mice pointers, etc.) with continuous aspects of their voice (pitch, vowel quality). Within this project, we needed a frame-level vowel-classifier: 1) that had real-time performance, 2) that was extremely accurate and robust, and where 3) speaker adaptation was fast and consumed few computational resources. Our regularized adaptation approach meets all of these requirements and is now in use in our most up-to-date VJ engine.

5. EXPERIMENTS

We compared our approach (in all cases, $\alpha = 1$) to a number of standard MLP adaptation techniques as outlined earlier. The Vocal Joystick project contains training/test sets for two vowel classification tasks, a 4-class case (vowels [ae], [uw], [aa], and [iy]) and an 8-class case (the addition of vowels [a], [ow], [ey], and [ix]) which we use for all evaluations. The training/test data, 15 speakers total, was collected as part of the project since there was no clean collection of vowel data that did not exhibit significant amounts of coarticulation. Unlike in ASR, heavily coarticulated vowels in training are less desirable since vowels uttered during VJ control are typically not heavily coarticulated. In addition to the above 15 speakers above, we have very recently collected an additional 10 speakers of data that we here used only as a development set (used to produce values for tradeoff coefficients, learning rates, and number of hidden units). Note that the data we use in this work is different than in [8].

In all experiments, we produced standard MFCC speech feature vectors with a window size of 25ms and 10ms frame step size, and computed the MFCC deltas producing a 26-dimensional feature vector. A simple form of online (i.e., a causal filter) mean and variance normalization was applied to the features. Each classifier has as input features a window of 7 such frames (empirically determined to be optimal) leading to an $\mathbf{x_t}$ of size $7 \times 26 = 182$. Also, each MLP used 50 hidden units (also empirically optimal) sigmoidal non-linearity, and used a softmax non-linearity at the last layer.

Ten of the speakers were allocated to the training set, and 5 speakers were used for testing (adaptation and evaluation). There are 180 utterances in the training set for each vowel, giving a total of 4*180 (85k frames) and 8*180 (174k frames) training utterances for the 4- and 8-class systems respectively, excluding silence. Each vowel class for each speaker has 18 utterances spoken in various ways (rising/falling pitch, etc.). For a particular test speaker, the 18 utterances for each vowel were further divided into 6 subsets with 3 utterances each. Each subset (3 utterances) was used for adaptation and the remaining subsets (15 utterances in total) for evaluation. We calculated the mean of the error rates over these 6 adaptation/evaluation subsets, and repeated this for each speaker, so the classification error rates reported below are an average over the 5 test speakers, and hence an average of 30 subsets. For each speaker, this yields an evaluation subset of 8K (resp. 16K) frames for the 4-class (resp. 8-class) system for each test speaker, and we multiply these numbers by 5 to get the overall effective evaluation set size.

There are several ways to measure the amount and type of adaptation data. For example, one way measures the amount (in seconds) of adaptation data available balanced across all classes. An alternative approach is to measure the data as a fraction of the number of different class categories available to adapt on (e.g., for an 8 class classifier, it may be desirable to adapt using data from only the unbalanced instances of 3 of those 8 classes). This latter approach is more like adaptation in ASR, where limited adaptation data almost assuredly means that certain categories (phones or words) are not available. We evaluate both approaches below.

Classification error rate baseline (unadapted model) results on the test set are show in Table 1. The results show that involving an SVM always hurts performance relative to the MLP. Since the MLPs train so much faster, we were able to perform a more thorough search of the tuning parameters (learning rates, tradeoff coefficients, etc) than we could with the SVMs. Note also, the MLPs were trained using cross-entropy on the soft-max output layer, and thus directly optimized a multi-class error objective function. Each of the SVMs were trained using n one-vs-all schemes (where n is the number of classes) so this may also have given the MLP an advantage (even under its harder task of non-convex optimization).

	4 classes		8 classes	
SVM(RBF kernel)	9.0	1hr	41.2	2hr
SVM(2nd-order poly kernel)	9.4	1hr	40.3	2hr
MLP	**8.9**	2min	40.0	5min
MLP-SVM(linear)[8]	**8.8**	1hr	39.8	2hr
REG (MLP with weight decay)	**8.5**	2min	**39.0**	5min

Table 1. Error rate results for various unadapted models, and their approximate training times. In this and other tables, we highlight the best error rate, and also the cases that are not significantly different at the $p < 0.002$ level relative to this best.

We next compared a number of adaptation strategies including: LIN, RLL, RFL, RSI (retrain both MLP layers using only adaptation data starting from the SI model), RRI (differs from RSI by starting from *r*andom *i*nitial parameters), and REG (regularized adaptation). We vary the amount of adaptation data that was available on all 4 or 8 classes (so all classes in each case had an equal amount of adaptation data). The results are shown in Table 2. Results show that RSI, RFL, and REG are all the best performing methods when the adaptation data is balanced among the classes.

	4 classes			8 classes		
amount of data/class	1sec	2sec	3sec	1sec	2sec	3sec
RRI	10.62	8.62	5.4	33.2	29.0	24.9
RLL	8.34	7.99	7.1	31.8	28.4	26.5
MLP-SVM [8]	8.74	6.80	5.7	32.1	28.0	26.0
LIN	**7.55**	**5.69**	5.1	32.6	29.9	28.1
RSI	**7.09**	**5.50**	**4.6**	**28.4**	**24.8**	**22.9**
RFL	**7.16**	**5.46**	**4.5**	**28.2**	**24.7**	**22.5**
REG	**7.05**	**5.42**	**4.5**	**28.1**	**24.4**	**22.4**

Table 2. 4- and 8-class test results, in % error rate, with varying amounts of class-balanced adaptation data.

# classes	1	2	3	4	5	6	7	8
RLL	48.8	48.5	49.1	42.2	46.2	41.7	34.0	26.5
LIN	46.1	49.8	49.1	45.5	42.1	37.7	35.7	28.1
RSI	46.1	45.3	47.4	41.0	43.5	38.3	30.6	**22.9**
RFL	41.4	39.3	37.9	33.1	33.5	31.7	27.6	**22.5**
REG	**39.6**	**38.3**	**36.0**	**32.3**	**30.8**	**28.8**	**26.3**	**22.4**

Table 3. 8-class test results, in % error rate, with different number of classes of adaptation data available (unbalanced classes). Three seconds of data is used for each class.

Table 3 shows results when the amount of adaptation data is unbalanced across the classes. Column i means a system is adapted only with i out of the 8 possible number of classes. These results are therefore more akin to the situation in ASR where a limited amount of adaptation data corresponds to a very small number of the possible classes that can exist. We experimented with different orders that the vowel classes are available for adaptation and observed similar patterns in results, and here we only report the results for one such order. As the table shows, the REG approach performs significantly (at the $p < 0.002$ level) better than any of the other approaches for all classes up to all 8 classes at which time it performs identically (statistically) with the RFL and RSI approaches.

6. CONCLUSION

We have introduced regularized adaptation as a framework for moving from an unadapted (speaker-independent) to an adapted (speaker-dependent) discriminative classifier and have shown that it achieves results that improve on all other MLP adaptation schemes we are aware of. In addition, the adaptation scheme works particularly well when the data is unbalanced, and it yields a relatively simple and computationally cheap adaptation algorithm that does not require a large amount of adaptation data. Our approach, moreover, could easily be applied to adapt the front ends of a hybrid HMM/ANN [12] or HMM/SVM ASR system. Lastly, we wish to thank Jon Malkin for both Vocal Joystick development and idea contribution, and thank Kelley Kilanski for data collection.

7. REFERENCES

[1] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, ser. Springer Series in Statistics. Springer, 2001.

[2] P. Woodland, "Speaker adaptation: Techniques and challenges," in *Proc. IEEE ASRU*, 1999.

[3] R. Kuhn, J.-C. Junqua, P. Nguyen, and N. Niedzielski, "Rapid speaker adaptation in eigenvoice space," *IEEE Transactions on Speech and Audio Processing*, vol. 8, pp. 695–707, Nov. 2000.

[4] J.Neto, L. Almeida, M. Hochberg, C. Martins, L. Nunes, S. Renals, and T. Robinson, "Speaker-adaptation for hybrid HMM-ANN continuous speech recognition system," in *Proc. Eurospeech*, 1995.

[5] V. Abrash, H. Franco, A. Sankar, and M. Cohen, "Connectionist speaker normalization and adaptation," in *Eurospeech*, September 1995, pp. 2183–2186.

[6] N. Matic, I. Guyon, J. Denker, and V. Vapnik, "Writer adaptation for on-line handwritten character recognition," in *Proc. Intl. Conf. on Pattern Recognition and Document Analysis*, 1993.

[7] J.Stadermann and G.Rigoll, "Two-stage speaker adaptation of hybrid tied-posterior acoustic models," in *ICASSP*, 2005.

[8] X.Li, J.Bilmes, and J.Malkin, "Maximum margin learning and adaptation of MLP classifi ers," in *Eurospeech*, 2005.

[9] T. Hastie, S. Rosset, R. Tibshirani, and J. Zhu, "The entire regularization path for the support vector machine," *Journal of Machine Learning Research*, vol. 5, pp. 1391–1415, Oct 2004.

[10] V. Vapnik, *Statistical Learning Theory*. Wiley, 1998.

[11] C. Bishop, *Neural Networks for Pattern Recognition*. Clarendon Press, Oxford, 1995.

[12] H. Bourlard and N. Morgan, *Connectionist Speech Recognition: A Hybrid Approach*. Kluwer Academic Publishers, 1994.

[13] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A nerual probabilistic language model," *Journal of Machine Learning Research*, vol. 3, pp. 1137–1155, 2003.

[14] R.Collobert and S.Bengio, "Links between perceptrons, MLPs and SVMs," in *Intl. Conf. on Machine Learning*, 2004.

[15] B.Scholkopf and A.J.Smola, *Learning with kernels*. The MIT Press, 2001.

[16] S. Rosset, J. Zhu, and T. Hastie, "Margin maximizing loss functions," in *Neural Information Processing Systems (NIPS)*, ser. 16. Cambridge, MA: MIT Press., 2004.

[17] B. Schölkopf, R. Herbrich, and A. J. Smola, "A generalized representer theorem," in *COLT '01/EuroCOLT '01: Proc. 14th Conf. Comp. Learning Theory/5th European Conf. Comp. Learning Theory*. London, UK: Springer-Verlag, 2001, pp. 416–426.

[18] J. Bilmes et al., "The Vocal Joystick: A voice-based human-computer interface for individuals with motor impairments," in *Human Language Technology Conf. and Conf. on Empirical Methods in Natural Language Processing*, Vancouver, October 2005.