Expectation Maximization of Forward Decoding Kernel Machines

Shantanu Chakrabartty and Gert Cauwenberghs
Center for Language and Speech Processing.
The Johns Hopkins University
Baltimore, MD 21218
{*shantanu,gert*}*@bach.ece.jhu.edu*

Abstract

Forward Decoding Kernel Machines (FDKM) combine large-margin kernel classifiers with Hidden Markov Models (HMM) for Maximum a Posteriori (MAP) adaptive sequence estimation. This paper proposes a variant on FDKM training using Expectation-Maximization (EM). Parameterization of the expectation step controls the temporal extent of the context used in correcting noisy and missing labels in the training sequence. Experiments with EM-FDKM on TIMIT phone sequence data demonstrate up to 10% improvement in classification performance over FDKM trained with hard transitions between labels.

1 Introduction

Large Margin (LM) Classifiers like Support Vector Machines (SVM) have several attractive properties :

1. They generalize well even with relatively few data points in the training set, and bounds on the generalization error can be directly estimated from the training data.
2. The only parameter that needs to be chosen is a penalty term for misclassification which acts as a regularizer [4] and determines a trade-off between resolution and generalization performance, to control learning ability.
3. The algorithm finds, under general conditions, a unique separating decision surface that provides the best out-of-sample performance.
4. They provide a framework to model non-linear classification boundaries by projecting the input data point into higher dimensional space and then computing the distances with the aid of a kernel.
5. The learning algorithm performs model selection based on some optimization criterion, by which on the data points which are relevant to classification or the problem are used for computation.

Most of the theory and formulation of LM classifiers are based on a stronger independence assumption across input training data. Sequential structure across training data is not taken into account in the standard LM formulation. Graphical Models factor sequential dependencies between random variables into conditionally independent entities (cliques) to which existing LM principles can be readily applied. Forward Decoding Kernel Machines (FDKM) [6] are hybrid models combining graphical models with LM principles to perform MAP sequence estimation. State transitions in the sequence are conditioned on observed data using a kernel-based probability model, and forward decoding of the state transition probabilities with the sum-product algorithm directly produces the MAP sequence. The parameters in the probabilistic model are trained using a recursive scheme that maximizes a lower bound on the regularized cross-entropy. The recursion performs an expectation step on the outgoing state of the transition probability model, using the posterior probabilities produced by the previous maximization step. Similar to Expectation-Maximization (EM), the FDKM recursion deals effectively with noisy and partially labeled data.

The aim of this paper is to devise an FDKM training algorithm based on Expectation-Maximization to enhance the generalization ability for static and temporal recognition.

2 FDKM Decoder

The problem of FDKM recognition is formulated in the framework of MAP (maximum a posteriori) estimation, combining Markovian dynamics with kernel machines. A Markovian model is assumed with sym-

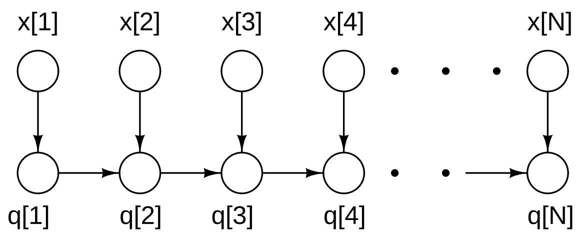

Figure 1: *Graphical model showing inter-dependencies between states q[n] conditioned on input vectors* $\mathbf{x}[n]$.

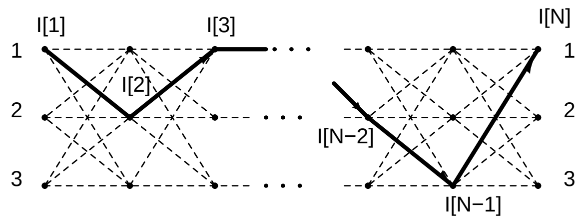

Figure 2: *Trellis diagram for a fully connected 3 state Markovian model. The training label sequence* $I[1], \ldots, I[N]$ *can be considered an instance of all possible paths through the trellis.*

bols belonging to S classes, as illustrated by a graphical model Figure 1. A corresponding trellis diagram for a fully connected Markovian model is shown for $S = 3$ classes. Transitions between the classes are modulated in probability by observation (data) vectors $\mathbf{x}$ over time.

The MAP forward decoder receives the sequence $\overline{\mathbf{X}}[n] = \{\mathbf{x}[n], \mathbf{x}[n-1], \ldots, \mathbf{x}[1]\}$ and produces an estimate of the probability of the state variable $q[n]$ over all classes i, $\alpha_i[n] = P(q[n] = i \mid \overline{\mathbf{X}}[n], \mathbf{w})$, where $\mathbf{w}$ denotes the set of parameters for the learning machine. Unlike *hidden* Markov models, the states directly encode the symbols, and the observations $\mathbf{x}$ modulate transition probabilities between states [7]. Estimates of the posterior probability $\alpha_i[n]$ are obtained from estimates of local transition probabilities using the *forward-decoding* procedure [7, 8]

$$\alpha_i[n] = \sum_{j=1}^{S} P_{ij}[n]\ \alpha_j[n-1] \tag{1}$$

where $P_{ij}[n] = P(q[n] = i \mid q[n-1] = j, \mathbf{x}[n], \mathbf{w}_j)$ denotes the probability of making a transition from class j at time $n-1$ to class i at time n, given the current observation vector $\mathbf{x}[n]$. The transition probabilities $P_{ij}[n]$ are parameterized by $\mathbf{w}_j = \{\mathbf{w}_{1j}, ..., \mathbf{w}_{Sj}\}$, where $\mathbf{w} = \bigcup_{ij}^{S} \mathbf{w}_{ij}$. The forward decoding (1) embeds sequential dependence of the data wherein the probability estimate at time instant n depends on all the previous data. An on-line estimate of the symbol $q[n]$ is thus obtained:

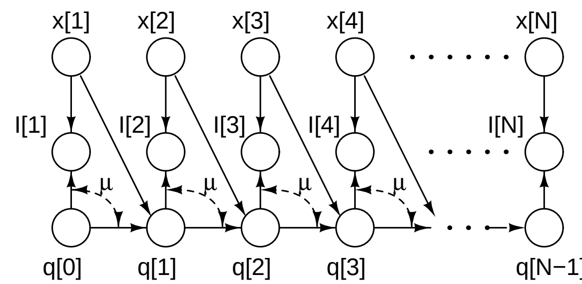

Figure 3: *Graphical model assumed for EM training showing inter-dependencies between states q[n], input vectors* $\mathbf{x}[n]$ *and labels* $I[n]$.

$$q^{\text{est}}[n] = \arg\max_i \alpha_i[n] \tag{2}$$

Accurate estimation of transition probabilities $P_{ij}[n]$ in (1) is crucial in decoding (2) to provide good performance. We use kernel logistic regression [6] with regularized maximum cross-entropy, to model conditional probabilities.

3 Training Formulation

3.1 Direct Partitioning

A brute-force method to obtain transition probabilities $P_{ij}[n]$ would be to partition the data into S subsets, based on the classification of their previous state/label and then performing S independent kernel logistic regressions. This method has the following drawbacks.

1. **Data Insufficiency:** For data sets with large number of classes/labels it may not be possible to obtain instances of all tuples of previous states and present states. This would produce inaccurate and biased estimates for missing empirical transitions.

2. **Noisy and Missing Labels:** Errors in previous labels propagate to the training of the subsequent labels. Missing labels cause information about subsequent labels to be discarded in training.

3. **Irregularity in Parameterization:** The regularization parameters for each of the kernel logistic regressions have to be tuned separately to prevent over-fitting or under-fitting to a part of the training data. In most of the scenarios, prior knowledge is not available to fine tune the parameters.

3.2 EM Sequential Training

The training algorithm described in this section mitigates the above problems. For the rest of the section a fully connected trellis is considered, without loss of generality for brevity of exposition. For training the MAP forward decoder, we assume access to a training sequence with labels (class memberships). For instance, the TIMIT speech database comes labeled with phonemes. Let $\overline{\mathbf{I}} = \{I[1], I[2], \ldots, I[N]\}$, be a sequence of class memberships, where $I[n] \in 1, 2, .., S$. Continuous (soft) labels could be assigned rather than binary indicator labels, to signify uncertainty in the training data over the classes. As probabilities $y_i[n] = P(I[n] = i)$, label assignments are normalized: $\sum_{i=0}^{S-1} y_i[n] = 1, y_i[n] \geq 0$. This label sequence $\overline{\mathbf{I}}$ can be considered to be one probabilistic realization of of all possible paths through the trellis 2.

The objective of the training is to maximize the MAP conditional likelihood of the training label sequence given training data.

$$L(\mathbf{w}) = \log P(\mathbf{w}) + \log P(\overline{\mathbf{I}} \,|\, \overline{\mathbf{X}}, \mathbf{w}) \tag{3}$$

where $P(\mathbf{w})$ denotes the prior distribution over the model parameters $\mathbf{w}$. The state variables $\overline{\mathbf{Q}} = q[0], .., q[N]$ serve as 'hidden' variables in the EM formulation (3). Ideally, the hidden variables and labels coincide, $q[n] = I[n]$, and the procedure reduces to the direct partitioning method of section 3.1. In the presence of missing and noisy labels $I[n]$, the EM procedure fills in previous states $q[n-1]$, with expected values conditioned over the training sequence.

The EM auxiliary function (3) becomes

$$\begin{aligned} Q(\mathbf{w}, \mathbf{w}^p) \;\; = \;\; & \log P(\mathbf{w}) \\ & + \frac{1}{Z} \sum_{\overline{\mathbf{Q}}} P(\overline{\mathbf{Q}}, \overline{\mathbf{I}} | \overline{\mathbf{X}}, \mathbf{w}^p) \log P(\overline{\mathbf{Q}}, \overline{\mathbf{I}} | \overline{\mathbf{X}}, \mathbf{w}) \end{aligned} \tag{4}$$

where $\mathbf{w}^p$ is the prior estimate of $\mathbf{w}$, and Z is a normalization constant to ensure $\sum_{\overline{\mathbf{Q}}} P(\overline{\mathbf{Q}} | \overline{\mathbf{I}}, \overline{\mathbf{X}}, \mathbf{w}^p) = 1$ [5]. Using (4) a sequence of parameters is obtained through the iteration $\mathbf{w}^{p+1} = \arg\max_{\mathbf{w}} Q(\mathbf{w}, \mathbf{w}^p)$, resulting in increase of likelihood function $L(\mathbf{w})$ after each iteration, till the algorithm converges to a local maxima [5].

The factorization of the joint probabilities $P(\overline{\mathbf{Q}}, \overline{\mathbf{I}} | \overline{\mathbf{X}}, \mathbf{w}^p)$ over graphs can be performed using standard forward-backward recursion [1, 14]. In this work a regularization parameter μ is introduced in the forward-backward recursion controlling the temporal extent of pooling context information to fill in missing or noisy labels $I[n]$ in the hidden sequence $q[n]$.

The forward-backward recursions are

$$\begin{aligned} \gamma_i[n] &= \sum_j \gamma_j[n-1] (\widehat{P}_{I[n]j}[n])^{\mu} \widehat{P}_{ij}[n] \\ \beta_i[n] &= \sum_j \beta_j[n+1] (\widehat{P}_{I[n]j}[n])^{\mu} \widehat{P}_{ji}[n] \end{aligned} \tag{5}$$

with

$$\begin{aligned} \gamma_i[n] &= \widehat{P}(I[1], \ldots, I[n], q[n] = i) \\ \beta_i[n] &= \widehat{P}(I[n+1], \ldots, I[N] | q[n-1] = i) \end{aligned} \tag{6}$$

where $\widehat{P}$ denotes the probability estimates obtained using parameters $\mathbf{w}^p$.

In the limit $\mu \to 0$ the forward recursion reduces to the standard form (1). By increasing the value of μ the estimates of $q[n]$ are biased more strongly towards the training labels $I[n]$. Details of this effect can be inferred through derivations in Appendix B.

A second regularization parameter C controls the complexity of the kernel classifier as in the standard SVM formulation. A Gaussian prior $P(\mathbf{w}) = \exp(-\frac{1}{2C} \sum_i^S \sum_j^S |\mathbf{w}_{ij}|^2)$ is chosen to favor 'smooth' solutions. $Q(\mathbf{w}, \mathbf{w}^p)$ can then be written as

$$Q(\mathbf{w}, \mathbf{w}^p) = \sum_j H_j \tag{7}$$

where

$$H_j = \sum_i^S -\frac{1}{2} |\mathbf{w}_{ij}|^2 + \sum_n^N C_j[n] \sum_i^S \widehat{y}_{ij}[n] \log P_{ij}[n] \tag{8}$$

with

$$\begin{aligned} \widehat{y}_{ij}[n] &= \sigma_{ij}[n] / \sum_i^S \sigma_{ij}[n] \\ C_j[n] &= C \sum_i^S \sigma_{ij}[n] / \sum_j \gamma_j[N] \end{aligned} \tag{9}$$

and

$$\sigma_{ij}[n] = \gamma_j[n-1] (\widehat{P}_{I[n]j}[n])^{\mu} \widehat{P}_{ij}[n] \beta_i[n+1] (1 + \mu \delta_{I[n],i}) \tag{10}$$

where $\delta_{x,y}$ is the delta Kronecker function. Mathematical details to arrive at (8) are given in appendix B. The formulation (7) is equivalent to S independent regressions of conditional probabilities $P_{ij}[n]$, for each outgoing state j, from data $\mathbf{x}[n]$ with effective labels $\widehat{y}_{ij}[n]$ and effective regularization factor $C_j[n]$.

3.3 Kernel Logistic Probability Regression

General estimation of conditional probabilities $\Pr(i|\mathbf{x})$ from training data $\mathbf{x}[n]$ and (soft) labels $y_i[n]$ can be obtained using a regularized form of kernel logistic regression [9]. For each outgoing state j, one such probabilistic model can be constructed for the incoming state i conditional on $\mathbf{x}[n]$:

$$P_{ij}[n] = \exp(f_{ij}(\mathbf{x}[n])) / \sum_{s}^{S} \exp(f_{sj}(\mathbf{x}[n])) \qquad (11)$$

As with SVMs, dot products in the expression for $f_{ij}(\mathbf{x})$ in (11) convert into kernel expansions over the training data $\mathbf{x}[m]$ by transforming the data to feature space [10]

$$\begin{aligned} f_{ij}(\mathbf{x}) &= \mathbf{w}_{ij}.\mathbf{x} + b_{ij} \\ &= \sum_{m} \lambda_{ij}^{m} \ \mathbf{x}[m].\mathbf{x} + b_{ij} \qquad (12) \\ &\xrightarrow{\Phi(\cdot)} \sum_{m} \lambda_{ij}^{m} \ K(\mathbf{x}[m], \mathbf{x}) + b_{ij} \end{aligned}$$

where $K(\cdot,\cdot)$ denotes any symmetric positive-definite kernel[1] that satisfies the Mercer condition, such as a Gaussian radial basis function or a polynomial spline [4, 11].

Optimization of (8) requires solving M disjoint but similar sub-optimization problems. The subscript j is omitted in the remainder of this section for clarity. The (primal) objective function of kernel logistic regression expresses regularized cross-entropy (8) of the logistic model (11) in the form [11, 12]

$$\begin{aligned} H &= -\sum_{i} \frac{1}{2} |\mathbf{w}_i|^2 \qquad (13) \\ &+ C[m] \sum_{m}^{N} [\sum_{i}^{M} y_i[m] f_k(\mathbf{x}[m]) - \log \sum_{p}^{M} e^{f_p(\mathbf{x}[m])}] \, . \end{aligned}$$

The parameters λ_{ij}^{m} in (12) are determined by minimizing a dual formulation of the objective function (13) obtained through the Legendre transformation, which for logistic regression takes the form of an entropy-based potential function in the parameters [9]

$$\begin{aligned} H_d &= \sum_{i}^{M} [\frac{1}{2} \sum_{l}^{N} \sum_{m}^{N} \lambda_i^l Q_{lm} \lambda_i^m \qquad (14) \\ &+ C[m] \sum_{m}^{N} (y_i[m] - \frac{\lambda_i^m}{C[m]}) \log(y_i[m] - \frac{\lambda_i^m}{C[m]})] \end{aligned}$$

subject to constraints

$$\sum_{m} \lambda_i^m = 0 \qquad (15)$$

$$\sum_{i} \lambda_i^m = 0 \qquad (16)$$

$$\lambda_i^m \leq C[m] y_i[m] \qquad (17)$$

[1] $K(\mathbf{x},\mathbf{y}) = \Phi(\mathbf{x}).\Phi(\mathbf{y})$. The map $\Phi(\cdot)$ need not be computed explicitly, as it only appears in inner-product form.

Derivations to arrive at the dual formulation are provided in the Appendix A. And alternative approach using Huber loss function gives rise to a sparse dual formulation where the Shannon entropy in equation (14) converts to the *Gini* entropy index [16].

3.4 Training Algorithm

The EM training algorithm is summarized as follows :

1. To obtain an initial estimate of parameters $\mathbf{w}^0$, regress probabilities using kernel logistic regression over the entire data set. This will give estimates of $\widehat{P}(q[n] = i|\mathbf{x}[n])$.

2. Bootstrap FDKM with estimates $\widehat{P}_{ij}[n] = \widehat{P}(q[n] = i|\mathbf{x}[n])$ and pick (derive) appropriate values for C and μ.

3. Compute $\widehat{y}_{ij}[n]$ and $C_j[n]$ using (5), (9) and (10).

4. Train S independent logistic kernel machines by maximizing equation (8) using its dual formulation (14), to obtain new set of parameters $\mathbf{w}^{p+1}$.

5. Compute new estimates of $\widehat{P}_{ij}[n]$ using $\mathbf{w}^{p+1}$ and go to step 3. Iterate till convergence.

Generally only 4-5 EM iterations are required to obtain a reasonable solution. Approximation and projection techniques used for FDKM in [16] can be used in the EM procedure to improve the computational efficiency of step 4.

4 Experiments and Results

To validate the training algorithm, experiments were first performed in a controlled setting with training data generated synthetically from a given distribution, possessing a known sequential structure. Performance was finally evaluated on real data, consisting of phone sequences from TIMIT corpus.

4.1 Controlled Experiments

Training data was produced by a two state Hidden Markov Model (HMM), with different single mixture Gaussian distribution attached to the states. The transition probabilities between the states determined the sequential nature of the data. The dimension of the input data was fixed to be 2 for the purpose of visualization. A sample output of 100 training points is shown in Figure 4. The purpose of this experiment was to determine if the machine is able to use the sequential information efficiently to yield better classification

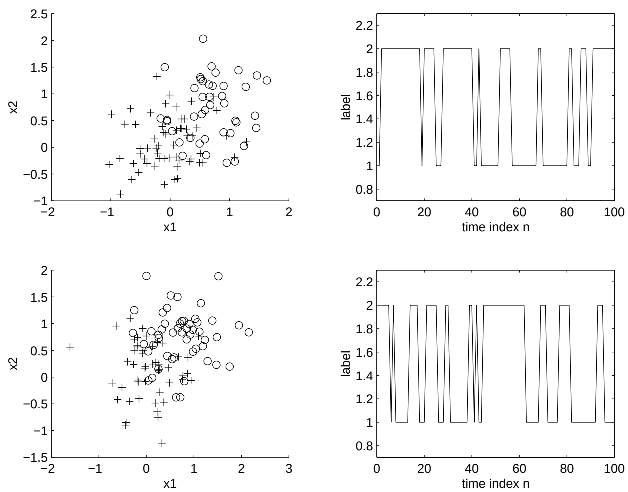

Figure 4: *Plot showing the spatial structure and temporal structure of the synthetic training (top) and test data (bottom). The temporal plot (right) shows that the data exhibits sequential structure.*

performance, especially when there is significant overlap in the two class distributions. For a Bayes classification error of 21%, FDKM achieved a classification error of 15.5% for a particular combination of μ and C, implying that it is able to leverage some of the sequential information present in the training data.

4.2 Phonetic Experiments

The performance of the training algorithm was evaluated on phonetic data, obtained from TIMIT database. The TIMIT speech dataset [13] consists of approximately 60 phone classes, which were first collapsed onto 6 broad categories according to TIMIT documentation. These categories comprised of *Vowels (V)*, *Stops(S)*, *Fricatives(F)*, *Nasals(N)*, *Semi-Vowels(SV)* and *Silence(Sil)*. An empirical phonetic language model for 6 broad categories is illustrated in Table 1. The table shows that there exists significant sequential structure across these classes, that can be used to improve the classification performance. The experiments in this section is based on 2000 phonetic instances from randomly chosen "sx" training sentences in the corpus.

The speech signal was first processed by a pre-emphasis filter with transfer function $1-0.97z^{-1}$. Subsequently, a 25 ms Hamming window was applied over 10 ms shifts to extract a sequence of phonetic segments. Cepstral coefficients were extracted from the sequence, combined with their first and second order time differences into a 39-dimensional vector. Cepstral mean subtraction and speaker normalization were subsequently applied. Each phone utterance was then subdivided into three segments with relative proportions 4:3:4 [15]. The features in the three segments were individually averaged and concatenated to obtain a 117-dimensional feature vector. Principal Component Analysis (PCA) was performed and feature vectors were projected onto 80-dimensional space, retaining 99% of the total signal energy.

Table 1: *Phonetic Language Model Computed using 2000 phones instances from TIMIT. The phones previous state and the first row is the destination state.*

	V	S	F	N	SV	Sil
V	4%	0.4%	8.5%	6.1%	6.3%	11.6%
S	5%	0%	0%	0.3%	1.8%	0.5%
F	8%	0.5%	0.5%	0.8%	1.0%	3.4%
N	4%	0.5%	1.4%	0.1%	0.4%	2.2%
SV	10%	0%	0.5%	0.2%	0.7%	1.3%
Sil	7%	6%	3.2%	0.7%	1.8%	1.5%

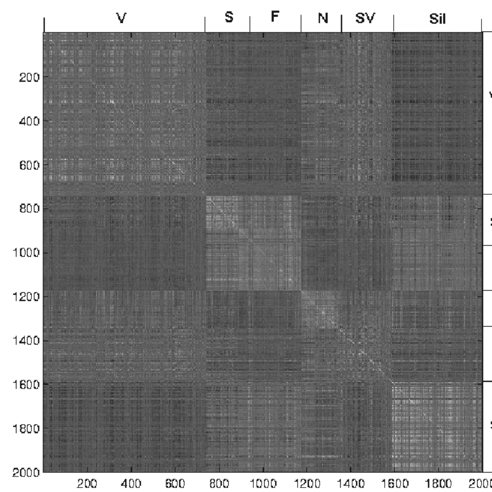

Figure 5: *Image of the kernel matrix showing the extent of static information in the training data. There are 6 distinct classes, and the grayscale factor depicts the similarity between distinct classes in the feature space.*

A second order polynomial kernel $K(\mathbf{x},\mathbf{y}) = [1 + (\mathbf{x}.\mathbf{y})^2]$ was chosen for this experiment. Figure 5 depicts the gray scale plot of the kernel matrix. Several distinct regions can be identified from the plot, indicating that the classes belonging to these distinct region can be easily discriminated based on static information.

Table 2 illustrates the improvement in FDKM performance obtained by the EM procedure. The initial step, corresponding to kernel logistic regression trained

Table 2: *Confusion matrix evaluated on TIMIT test set for $\mu = 0.6$ Top: EM-FDKM. Bottom: Initial step (hard transitions between training labels).*

Class	V	S	F	N	SV	Sil
V	73%	2%	1%	10%	12%	2%
S	6%	90%	0%	1%	2%	2%
F	2%	3%	94%	0%	0%	1%
N	8%	0%	0%	84 %	6%	2%
SV	22%	3%	1%	12%	61%	0%
Sil	1%	1%	3%	1%	1%	94%
V	62%	2%	3.5%	14%	25%	2%
S	1%	81%	13%	1%	1%	3%
F	3.0%	6.5%	89%	0%	0%	1.5%
N	10%	2%	3%	78%	6%	1%
SV	24%	3%	2%	4%	60%	7%
Sil	5.5%	6%	3%	1%	2%	82.5%

with hard transitions between labels, serves as a base of comparison.

5 Conclusion

FDKM merges large-margin classification techniques into an HMM framework for robust forward decoding MAP sequence estimation. FDKM improves decoding and generalization performance for data with embeded-ded sequential structure, providing an elegant tradeoff between learning temporal versus spatial dependencies. Experiments with TIMIT and other sequential data demonstrated the ability of EM training to reduce or mask the effect of noisy or missing labels $y_j[n]$.

Acknowledgements

This work is supported by a grant from the Catalyst Foundation, New York.

References

[1] L. Rabiner and B-H Juang, *Fundamentals of Speech Recognition,* Englewood Cliffs, NJ: Prentice-Hall, 1993.

[2] Boser, B., Guyon, I. and Vapnik, V., "A training algorithm for optimal margin classifier," in *Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory*, pp 144-52, 1992.

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

[4] Girosi, F., Jones, M. and Poggio, T. "Regularization Theory and Neural Networks Architectures," *Neural Computation*, vol. **7**, pp 219-269, 1995.

[5] Dempster, A.P., Laird, N.M and Rubin, D. B. "Maximum-likelihood from incomplete dat via the EM algorithm," *J. Roy. Statist. Soc. B*, vol. **39**, pp 1-38, 1977.

[6] Chakrabartty, S. and Cauwenberghs, G. "Sequence Estimation and Channel Equalization using Forward Decoding Kernel Machines," *IEEE Int. Conf. Acoustics and Signal Proc. (ICASSP'2002)*, Orlando FL, 2002.

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

[8] Bahl, L.R., Cocke J., Jelinek F. and Raviv J. "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Transactions on Inform. Theory*, vol. **IT-20**, pp. 284-287, 1974.

[9] Jaakkola, T. and Haussler, D. "Probabilistic kernel regression models," *Proceedings of Seventh International Workshop on Artificial Intelligence and Statistics* , 1999.

[10] Schölkopf, B., Burges, C. and Smola, A., Eds., *Advances in Kernel Methods-Support Vector Learning,* MIT Press, Cambridge, 1998.

[11] Wahba, G. *Support Vector Machine, Reproducing Kernel Hilbert Spaces and Randomized GACV*, Technical Report 984, Department of Statistics, University of Wisconsin, Madison WI.

[12] Zhu, J and Hastie, T., "Kernel Logistic Regression and Import Vector Machine," *Adv. IEEE Neural Information Processing Systems (NIPS'2001)*, Cambridge, MA: MIT Press, 2002.

[13] Fisher, W., Doddington G. et al *The DARPA Speech Recognition Research Database: Specifications and Status.* Proceedings DARPA speech recognition workshop, pp. 93-99, 1986.

[14] Baum, L. E, Petrie T. et al *A maximization technique occuring in the statistical analysis of probabilistic functions of Markov Chains.* Ann. Math. Statistics, vol. 41, pp. 164-171, 1970.

[15] Fosler-Lussier, E. Greenberg, S. Morgan, N., "Incorporating contextual phonetics into automatic speech recognition," *Proc. XIVth Int. Cong. Phon. Sci.*, 1999.

[16] Chakrabartty, S., and Cauwenberghs, G. "Forward Decoding Kernel Machines: A Hybrid HMM/SVM approach to Sequence Recognition", *International Workshop of Pattern Recognition with Support Vector Machines*, ICPR 2002.

Appendix A: Dual Formulation of Kernel Logistic Regression

The regularized log-likelihood/cross entropy for kernel logistic regression is given by [11, 12]

$$\begin{aligned} H &= \sum_k \frac{1}{2}|\mathbf{w}_k|^2 && (18)\\ &- C[n]\sum_n^N [\sum_k^M y_k[n] f_k(\mathbf{x}[n]) - \log(\sum_p^M e^{f_p(\mathbf{x}[n])})] \,. \end{aligned}$$

First order conditions with respect to parameters $\mathbf{w}_k$ and b_k in $f_k(\mathbf{x}) = \mathbf{w}_k.\mathbf{x} + b_k$ yield

$$\begin{aligned} \mathbf{w}_k &= C[n]\sum_n^N [y_k[n] - \frac{e^{f_k(\mathbf{x}[n])}}{\sum_p^M e^{f_p(\mathbf{x}[n])}}]\mathbf{x}[n] \\ 0 &= C[n]\sum_n^N [y_k[n] - \frac{e^{f_k(\mathbf{x}[n])}}{\sum_p^M e^{f_p(\mathbf{x}[n])}}] \,. \qquad (19) \end{aligned}$$

Denote

$$\lambda_k^n = C[n][y_k[n] - \frac{e^{f_k(\mathbf{x}[n])}}{\sum_p^M e^{f_p(\mathbf{x}[n])}}] \qquad (20)$$

in the first-order conditions (19) to arrive at the kernel expansion (12) with linear constraint

$$\begin{aligned} f_k(\mathbf{x}) &= \sum_n \lambda_k^n K(\mathbf{x}[n], \mathbf{x}) + b_k && (21)\\ 0 &= \sum_n \lambda_k^n \,. && (22) \end{aligned}$$

Note also that $\sum_k \lambda_k^n = 0$.

Legendre transformation of the primal objective function (18) in $\mathbf{w}_k$ and b_k leads to a dual formulation directly in terms of the coefficients λ_k^n [9]. Define $z_n = \log(\sum_p^M e^{f_p(\mathbf{x}[n])})$, and $Q_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$. Then (20) and (21) transform to

$$\sum_l Q_{nl}\lambda_k^l - \log[y_k[n] - \lambda_k^n/C[n]] + b_k - z_n = 0 \qquad (23)$$

which correspond to first-order conditions of the convex dual functional

$$\begin{aligned} H_d &= \sum_k^M [\frac{1}{2}\sum_n^N \sum_l^N \lambda_k^n Q_{nl} \lambda_k^l && (24)\\ &+ C[n]\sum_n^N (y_k[n] - \frac{\lambda_k^n}{C[n]}) \log(y_k[n] - \frac{\lambda_k^n}{C[n]})] \end{aligned}$$

under constraints

$$\begin{aligned} \sum_n \lambda_k^n &= 0 && (25)\\ \sum_k \lambda_k^n &= 0 && (26)\\ \lambda_k^n &\leq C[n] y_k[n] && (27) \end{aligned}$$

where b_k and z_n serve as Lagrange parameters for the equality constraints (25) and (26).

Appendix B: Derivation of EM-FDKM Forward-Backward recursions

We derive the factorization of EM auxiliary function (4) in terms of forward-backward variables (6). For convenience of notation, the normalization constant Z and prior $P(\mathbf{w})$ are omitted and we expand

$$I(\mathbf{w}, \mathbf{w}^p) = \sum_{\overline{\mathbf{Q}}} P(\overline{\mathbf{Q}}, \overline{\mathbf{I}}|\overline{\mathbf{X}}, \mathbf{w}^p) \log P(\overline{\mathbf{Q}}, \overline{I}|\overline{\mathbf{X}}, \mathbf{w}) \quad (28)$$

The conditioned variables $\overline{\mathbf{X}}$,$\mathbf{w}$ and $\mathbf{w}^p$ are assumed implicitly were omitted. The joint probability $P(\overline{\mathbf{Q}}, \overline{\mathbf{I}}|\overline{\mathbf{X}}, \mathbf{w})$ can be factored as

$$\begin{aligned} P(\overline{\mathbf{Q}}, \overline{\mathbf{I}}|\overline{\mathbf{X}}, \mathbf{w}) &= P_{q[1]q[0]}[1] P(I[1]|q[1], q[0]) ... && (29)\\ & \quad P_{q[N]q[N-1]}[N] P(I[N]|q[N], .., q[0]) \end{aligned}$$

where the states $q[n]$ follow the first order Markovian property shown in Figure 1. The training label $I[n]$ in (29) depend on its context, with strong dependence on its previous state $q[n-1]$. To reduce the dependency of $I[n]$ into a tractable first order Markovian form, a parameter μ is introduced in the approximation

$$P(I[n]|q[n], q[n-1], \ldots, q[0]) \approx (P_{I[n]q[n-1]}[n])^{\mu} \quad (30)$$

The parameter μ controls the degree to which states $q[n]$ and context affect the training labels $I[n]$. Increasing μ biases the estimates of $q[n]$ strongly towards $I[n]$. $I(\mathbf{w}, \mathbf{w}^p)$ is then expanded, using definitions (6) as

$$\begin{aligned} I(\mathbf{w}, \mathbf{w}^p) &= \sum_n^N \sum_{ij}^S \gamma_j[n-1] (\widehat{P}_{I[n]j}[n])^{\mu} \widehat{P}_{ij}[n] \\ & \quad \beta_i[n+1](1 + \mu\delta_{I[n],i}) \log P_{ij}[n] \quad (31) \end{aligned}$$

from which equation (8) can be readily identified, with regularization parameters $C_j[n]$ in (9) and normalization constant $Z = \sum_j \gamma_j[N]$.