Model Selection with Support Vector Machines

Trinh Minh Tri Do and Thierry Artières

LIP6, Université Pierre et Marie Curie
8 rue du capitaine Scott, 75015, France
{Do@poleia.lip6.fr, Thierry.Artieres@lip6.fr }

Abstract. *This paper focuses on learning recognition systems able to cope with sequential data for classification and segmentation tasks. It investigates the integration of discriminant power in the learning of generative models, which are usually used for such data. Based on a procedure that transforms a sample data into a generative model, learning is viewed as the selection of efficient component models in a mixture of generative models. This may be done through the learning of a Support Vector Machine. We propose a few kernels for this and report experimental results for classification and segmentation tasks.*

1. Introduction

In order to perform classification, one can estimate either posterior class probabilities or class conditional probability density functions. Roughly speaking, the former corresponds to discriminant methods – Neural Networks, Support Vector Machines (SVM) – where training focuses on the differences between classes. The latter corresponds to independent learning of class conditional probability density models (e.g. generative models such as Gaussian model, Markov model) with e.g. a Maximum Likelihood criterion. Discriminant systems are usually more powerful than generative models. The latter are however very popular for many tasks. Indeed, most discriminant techniques are adapted to vectorial data, i.e. fixed dimension data, and cannot be used with variable sized (e.g. sequential) data like speech or handwriting signals. Hence most speech or handwriting recognition systems are based on Markovian models. Furthermore, due to the multimodal characteristic of such data (for example a handwritten character "b" may be written in various ways) mixture models are particularly popular in these fields. Hence many signal recognition and segmentation tasks are tackled by learning class models that are mixture of generative models. This paper proposes a way to increase the discriminant power of generative based systems and more specifically of mixture of generative models. Some techniques were proposed for discriminant learning of generative models, in particular for sequential data. For example, the use of the Fisher score [4] made it possible to use a discriminant technique, namely Support Vector Machines, for sequential data classification. This technique has a main drawback. It does not allow performing segmentation, which consists of segmenting an input signal into sub units (e.g. phonemes or letters) and simultaneously recognizing these units. Segmentation is generally the most interesting task for sequential data (e.g. in speech and handwriting recognition). Our goal is to explore techniques to combine the efficiency of the discriminant methods with the flexibility of mixtures of generative models. We aim at learning models that both exhibit improved discriminant power and are able to perform sequence recognition as well as segmentation. We then seek to use discriminant methods to build powerful generative models which perform better than generative models trained traditionally with a non discriminant criterion such as Maximum Likelihood. In the following, we will consider the case of generative model with the following form:

$$P(x/c) = \sum_{i}^{N_c} w_i^c P(x/\lambda_i^c) \qquad (1)$$

where c denotes a class, $P(x/c)$ is the probability of a sample data x (e.g. a sequence) by the model of class c, λ_i^c are component (generative) models for class c, and w_i^c are prior probabilities of the components of the mixture and verify $\sum_{i=1}^{N_c} w_i^C = 1$ for any c. $N_c$is called in the following the size of the model of the class c.

The paper is organized as follows. First, we discuss of related works that have been proposed for increasing discriminant power of generative models, mainly kernel methods in sequence recognition tasks. Then we describe our approach. First we present the principle of the method, then the learning of component models then the learning of prior weights. Finally, we report experimental results on sequence classification and segmentation tasks. We report experimental results for artificial data and for on-line handwriting data.

2. Related work

There have been a number of studies for increasing discriminant power of generative models. Most of these

techniques either rely on replacing usual Maximum Likelihood training criteria with a discriminant criterion such as Maximum Mutual Information or changing of representation space in order to use standard discriminant techniques such as neural networks or SVM. These latter studies mainly rely on the definition of kernel methods and SVM and have been proposed for sequence recognition tasks. SVM have become a traditional and powerful approach for classification and regression tasks [10],[11]. A SVM classifier has the following general form for a two classes classification problem:

$$f(x) = \sum_i y_i \alpha_i K(x_i, x) + b \quad (2)$$

where x is an input pattern to be classified, x_i are training samples, whose classes are identified by labels $y_i \in \{-1,+1\}$. The coefficients α_i and b are the parameters of the function f. K is the kernel function, it is the dot product of the projections of data in a high dimension space:

$$K(x_i, x) = \phi(x_i).\phi(x) \quad (3)$$

The classification of a test sample x is performed according to the sign of $f(x)$. Training of SVM consists of determining the parameters of the function f (α_i and b) such that the margin is maximized. Coefficients α_i are non null for a subset of the training points which are called "support vectors" (SV). Various approaches were proposed to apply SVM to sequential data. In [4],[5] the idea consists of transforming data into a fixed dimension representation, the Fisher Score. One uses a generative model learned from the data, λ_θ, and defined by a set of parameters θ. Then one defines the new representation of a sequence x as the gradient of the log likelihood of x by λ_θ:

$$U(x) = \vec{\nabla}_\theta \log(P(x/\lambda_\theta) \quad (4)$$

$U(x)$ includes information on how the model λ_θ should change to produce signal x with high probability. By doing this, one changes the space of representation of all training data, which are then represented in a fixed dimension space (the number of parameters of λ_θ). This makes possible the use of any discriminant classical technique such as Neural Networks or SVM on these representations.

Bahlmann [2] uses a system based on prototypes (representative samples of each class) and defines a kernel on sequential data which is based on a distance between sequences that has been very much used in the speech recognition field, the Dynamic Time Warping (DTW) distance. The kernel between two sequences is defined by:

$$K(x, y) = e^{-A.d_{dtw}(x,y)+B} \quad (5)$$

where A and B are parameters that are tuned empirically. Similar techniques are discussed in [12]. Note that the system proposed by the author is a SVM discriminant function that does not allow performing segmentation but exhibit improved classification performance over purely generative models.

Moreno [9] explores a third technique which consists of using a correspondence between data and model. In the speaker identification task he studied in his paper each sample data consists of a speech signal of a few tens of seconds. It is transformed into a generative model (a mixture of Gaussian laws) by learning with a Maximum Likelihood criterion. The Gaussian model is simple enough to be learned on one test sample (i.e. signal) only. From this construction of a model λ_x from a data x, [9] proposes to use a probabilistic kernel between two data x and y, which reflects the difference between associated generative models λ_x and λ_y. This kernel is based on the Kullback Leibler divergence between the probability distributions implemented by the two models. Then he builds a discriminant SVM function. This approach can, of course, only be applied if the considered generative model is simple enough to be learned with only one training or test sample.

3. Model selection with Support Vector Machines

First, we discuss the principle of our approach, then we present kernels for generative models. These kernels are used to select component models λ_i^c in a mixture model defined according to Eq. (1). Finally we discuss the learning of prior weights w_i^c in such a mixture model.

3.1. Principle

Figure 1 schematically presents our framework which relies on the change of representation space. This change exploits a correspondence between data and model, which we will discuss in more detail later. The idea is that, using a transformation λ, any sample x may be transformed into a generative model $\lambda(x)$ or λ_x in such a way that this model gives high likelihood to x and to similar samples. We will call such a model a *local generative model*. For instance a real feature vector x may be transformed into a Gaussian distribution with mean x and identity covariance matrix.

During training, one changes the representation space by transforming, using λ, any training sample x in a local generative model $\lambda(x)$. Next, one can use SVM on these new "data" (i.e. local generative models) by defining a kernel function on these generative models.

Learning of a SVM with such a kernel leads to a discriminant function whose support vectors are *local generative models*. From this procedure, we investigated two kinds of discriminant systems.

First, one can build a discriminant SVM system that performs sequence classification. Training is done as just described. At test time, a test sample is first transformed into a local generative model, then the SVM function is used to determine the corresponding sample class.

Second, one can design a system based on mixture of generative models. Training starts as just described. Then, mixture models are built as follows: component models are defined as support vectors in the SVM system since these local generative models are good candidates for component models λ_i^c in Eq. (1). Next, prior weights are learned using a discriminant criteria. In case of sequence data, such generative models may be used for classifying or segmenting input signals.

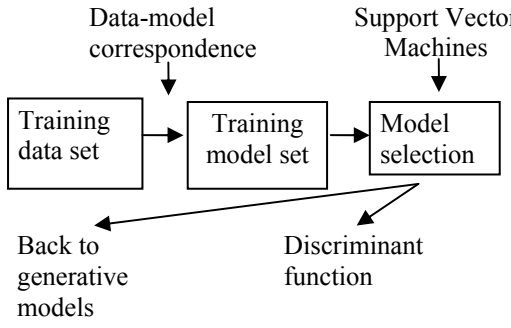

Figure 1. Using Support Vector Machines for learning mixture of generative models.

3.2. Kernels for model selection

Our work is inspired by [9] in that we seek to use Support Vector Machines by exploiting a correspondence between data and model. Our work differs in two points. First, the data-model correspondence does not involve training but exploits prior information about the task. Our approach can thus seem less generic than the one in [9] but one should note that the training of a model from a single data is generally not feasible without prior information. Second, our method may be used in two ways: for building a powerful discriminant function for sequence classification; and for learning efficient generative models able to perform segmentation (cf. Fig. 1).

In the following, we assume that there exists a data-model association. Hence, training is performed based on a training sample database $\{(x_i, y_i)\}$ and a database of associated generative models $\{(\lambda_{x_i}, y_i)\}$. We discuss now the kernels that we used on these models.

A first method that we considered consists of defining explicitly the projection function $\phi(x)$. The idea here is to represent a data *x* by the probabilities of this data computed by the set of all models of all the classes. In the two-class case, noting λ_j^i the model corresponding to the j^{th} training sample for class *i*, $\phi(x)$ is defined by:

$$\phi(\mathrm{x}) = \left[\log\left(\mathrm{p}(\mathrm{x}/\ \lambda_{\mathrm{i}}^{\mathrm{j}})\right), i,\ j\right] \tag{6}$$

One can use standard kernel functions, for example Gaussian, between $\phi(x)$ and $\phi(y)$:

$$\mathrm{K}_{\Phi}(\mathrm{x},\mathrm{y}) = e^{-\gamma\|\phi(x)-\phi(y)\|^2} \tag{7}$$

where γ is a constant. In order to simplify the procedure, one can build $\phi(x)$ from the scores computed by a limited subset of models of each class only. In our experiments, we randomly choose 10 models per class and define $\phi(x)$ as a vector of *Nx10* probabilities, where *N* is the number of classes. We will call this method the *Φ Kernel*.

We also used the kernel proposed by Moreno, exploiting the symmetric Kullback-Leibler divergence between two generative models λ_x and λ_y, built from two data *x* and *y*, we call this method *KL Kernel*. The symmetric divergence is written as:

$$KL_{sym}(\lambda_x, \lambda_y) = \frac{1}{2}(KL(\lambda_x|\lambda_y) + KL(\lambda_y|\lambda_x)) \tag{8}$$

In our experiments, we estimated KL divergences on the training data with:

$$KL(\lambda_x|\lambda_y) = \sum_z P(\mathrm{z}/\lambda_X)\log\frac{P(\mathrm{z}/\lambda_x)}{P(\mathrm{z}/\lambda_y)} \tag{9}$$

where *z* denotes samples in training data set. We used the following kernel:

$$K_{KL}(\lambda_x, \lambda_y) = e^{-A.KL_{sym}(\lambda_x,\lambda_y)+B} \tag{10}$$

At last, we used the Fisher Score as in [5] with a Gaussian kernel. This will be called the *Fisher Score kernel*.

3.3. Data to model transformation procedure

Our approach relies on a procedure that transforms a sample in a local generative model. In [9] this transformation is done through the learning of a statistical model (a mixture of Gaussian distributions) from a sample. This is possible in this case since each sample data consists of a speech signal of a few tens of seconds from which such a generative model may be learned. Of course, this approach stands for simple enough generative models that can be learned with only one training or test sample.

More generally, this transformation may be the result of a, possibly optimal but not necessarily, training step. Such a transformation may indeed be viewed as a non linear mapping from the original input space into a feature space. In our experiments we investigated both a very simple transformation for artificial data and a more sophisticated transformation for on-line handwriting signals that is based on prior knowledge about on-line handwriting signals and on-line handwriting recognition. As we will see our framework may be viewed as a way to exploit efficient kernels based on prior knowledge about the data and the task. This is, in our mind, an interesting idea since first, designing efficient kernels is generally a complex task and

second, there is often enough knowledge to allow designing an efficient data-to-model transformation.

3.4. Learning prior component weights

As discussed above, the learning of a SVM with a kernel defined on generative models allows selecting the components of the mixture models, i.e. the λ_i^c in Eq. (1). To complete the learning requires learning mixture coefficients, w_i^c. We propose here to seek coefficients optimizing the discriminating criterion:

$$V = \prod_{(x_i,y_i)\in TS} p(y_i/x_i) \tag{11}$$

where *TS* denotes the Training Set. This criterion is the product of the posterior class probabilities of the training data, the λ_i^c models remaining fixed. Taking the logarithm, one thus seeks to maximize:

$$J = \sum_{x\in TS_1} \log p(C_1/x) + \ldots + \sum_{x\in TS_N} \log p(C_N/x) \tag{12}$$

where TS_i denotes the subset of the training samples for class i in *TS*. We optimize the criterion with a gradient algorithm by deriving the criterion *J* with respect to w_i^c. One effect of optimizing this criterion is that a significant part of the weights converge towards 0, then identifying useless models for discrimination.

4. On-line handwritten digits recognition

We report experiments that we carried out on online handwriting signals (temporal sequence of points captured via an electronic tablet). We present the database, discuss the data-to-model correspondence and provide results.

4.1. Database

We worked on a part of the UNIPEN database [3], an international benchmark database in the handwriting recognition field. Our experiments have been performed with signals corresponding to the 10 digits written by more than 200 writers. We use about 16000 samples, 33% for training and 66% for testing. Each experimental result is an average result obtained on 3 experiments.

4.2. Data to model transformation

Handwriting signals are much variable (there are a few allograph for a character), so that the use of mixture of models or systems based on typical writings is very popular. A character is often modelled with a mixture of models, most often left right Markov models, each corresponds to an allograph. Training such a mixture model is not easy since the number of allograph as well as the topology of Markovian models must be tuned by hand. A number of studies were carried out to completely learn character models from the data [1],[7],[8]. They all rely on the building of one model (e.g. HMM) from one sample signal and exploit a representation of handwriting signals as a sequence of direction codes. We use in this study the procedure proposed in [1]. The idea is to build, from one original signal, one HMM giving high likelihood to signals that look like the original one and giving low likelihood to signals that do not. Figure 2 illustrates this procedure schematically. A handwriting signal is first represented as a sequence of basic strokes belonging to a set of 36 elementary strokes (represented in figure 3). There are straight lines uniformly distributed between 0° and 360° as well as slightly convex or concave strokes. The output of this preprocessing is a representation of the original signal as a sequence of elementary stroke sequence (e.g. es_1, es_{21} and es_1 in figure 3). From this representation, one can build a left right HMM with as many states as there are basic strokes in the sequence (e.g. three states on the right in figure 5). An emission probability distribution is associated to each state; it is derived from the corresponding elementary stroke. For example, the "ideal" elementary stroke in the first and in the third states of the model in Figure 2 should be es_1, so that the pdf in these states give high likelihoods to elementary strokes that are similar to this stroke.

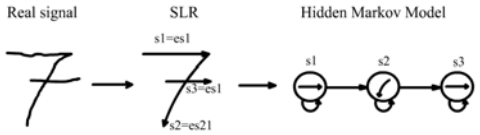

Fig. 2. Building a left right HMM from a sample. An online handwriting signal is segmented into a SLR, then a left right Markov mode is built from this sequence.

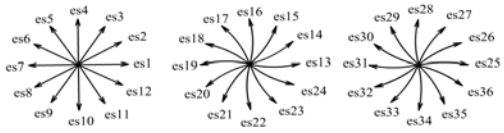

Fig. 3. Set 36 elementary strokes used to represent handwriting signals- from left to right 12 straight lines (named *es1* to *es12*), 12 convex strokes (named *es13* to *es24*) and 12 concave strokes (named *es25* to *es36*).

4.3. Results

Table 1 summarizes the performances of various methods for the classification of digits. First rows correspond to generative models learned with our discriminant approach. Component models λ_i^c are selected using SVM with different kernels while prior coefficients w_i^c are learned using the discriminant criterion discussed in §3.4. We also report accuracies for the purely discriminant SVM

functions, note that these latter systems cannot be used for segmentation tasks. First, the performances of the pure discriminant SVM functions much depend on the kernel used, since they range from 86% to almost 99%. Note here that the performance of the SVM function with K_{KL} kernel (98.8%) is the best performance ever achieved on these data, to our knowledge. Note here that whatever the kernel used, performances of the generative systems are very close, about 95% with a non-discriminant training of priors and about 98% with a discriminant training of priors. Thus, although these kernels are not all well adapted to build a discriminant function, they seem equivalent for the design of generative systems. Second, the generative systems whose elementary models are selected via SVM outperform the reference system in [8] dedicated to on-line handwriting. This system obtains a performance about 97.2% on these data. Compared to these performances, the approaches proposed here reduce the error rate by 20% for the generative systems, and by about 60% for the SVM system with the K_{KL} kernel.

Table 1. Performance of SVM for the selection of the elementary models and as a discriminant function, compared to a non-discriminant reference system [8].

Method	K_{KL}	K_{Φ}	K_{FS}
Generative Models	97.9	98	98.1
Discriminant fonction (SVM)	98.8	97.5	96.9
Reference method [8]	97.2		

5. Two dimensional data

We report here additional experiments performed on two-dimensional data in order to put in evidence the behaviour of the method. We use here a database extracted from the *pbvowel* database [6], containing speech signals (vowel). These signals were preprocessed and are represented, after feature extraction, as two dimensional vectors (two first formants). There are 10 classes and we use approximately 600 samples in training and 600 in test. We used a very simple transformation between sample data and model. The model associated with a sample is a Gaussian distribution whose mean is the sample and whose covariance matrix is proportional to the identity matrix. For a sample x, the associated model λ_x is a Gaussian law $N(x, \sigma^2. Id)$, where σ is shared by all the models built from samples. Its value was fixed empirically according to the average distance between two samples in the training database.

5.1. Classification and segmentation results

Table 2 reports classification results for generative models learned with our discriminant approach and pure discriminant SVM functions. As observed with on-line handwritten signals, the performance varies depending on the kernel. Best kernels for designing discriminant generative models are not necessarily the best kernels for designing a accurate SVM. Second, the generative systems learned with our discriminant approach perform similarly or better than pure SVM function, whatever the kernel. Also, the average model size of systems for generative systems is significantly less than the required number of support vectors necessary to achieve best results with the SVM system (10 instead of ~40). We also implemented a benchmark method by training generative models (as in Eq. (1)) to maximize training data Likelihood using an *EM* optimization algorithm, it achieves a recognition rate from 73 to 78% when the number of components in the mixture ranges from 5 to 50. Compared to these results, the generative systems whose models are selected by SVM are better in average.

Table 2. Classification accuracy on two dimensional data for generative models whose component models have been selected using SVM, and for pure SVM discriminant functions on generative models.

Method	K_{KL}	K_{Φ}	K_{FS}
Discriminant Generative Models			
Performance	77.5	78.3	78.5
Model size	11	10	12
SVM function			
Performance	76.2	73.3	78.5
#Support Vectors per class	39	39	40

Fig. 4 and Fig. 5 illustrate the idea behind model selection. These figures plot the training samples of the ten classes. They also plot for two generative systems the mean vectors of component Gaussian models in bold. Fig. 4 shows mean vectors of the non discriminant system while Fig. 5 shows support vectors of a discriminant system, where component models have been selected with the *KL kernel*. In Fig. 4 selected models correspond to typical samples whereas in Fig. 5 selected models are at the border between classes. One can imagine that, at least, such a discriminant selection of generative models leads to smaller class models since there is no use selecting a model corresponding to non ambiguous data as is done by a non discriminant training technique.

In order to evaluate the segmentation power of our generative systems, we used a Markov Model for generating artificial sequences of the vowel data. The model has ten states, one state for each of the ten classes. Each state emits randomly a feature vector corresponding to its class. We generated about 400 sequences for testing, with an average length of 60.

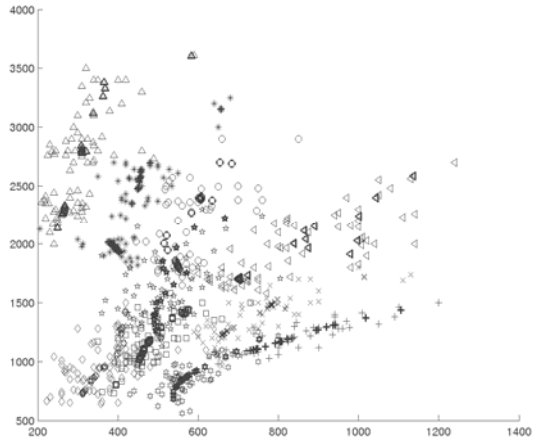

Figure 4. Models learned with a non discriminant criterion (ML). Mean vectors of component Gaussian models are in bold.

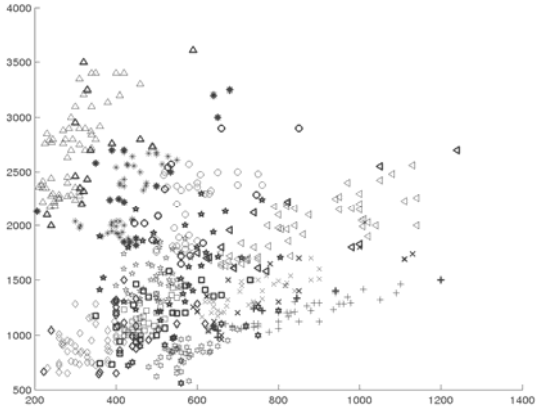

Figure 5. Model selection with *KL Kernel*. Samples corresponding to Support vector models are in bold.

Table 3. Segmentation results (%error) on sequences for discriminant generative models learned with the K_{KL}, the K_{Φ} and the K_{FS} kernels and for the standard non discriminant approach (Maximum Likelihood, ML).

Method	K_{KL}	K_{Φ}	K_{FS}	ML
Performance	18.9	18.1	18.1	22.9

Generative systems are learned with the training data as in previous section. For testing, we use a Hidden Markov Model with ten states. The emission probability distribution in a state is defined as the mixture of generative models learned for this class. The evaluation of segmentation is based on the string edit distance between the output label sequence and the original label sequence. As is usually done we consider as errors deletions and substitutions. Table 3 reports segmentation results as percentages of segmentation errors for various methods. As a reference, note that pure Maximum Likelihood estimation of generative models lead to 22.9% error rate. One can see that all systems whose models have been selected using SVM outperform *ML* estimated systems.

6. Conclusion

We studied in this paper discriminant methods for the training of mixtures of generative models. We considered the possibility of transforming the learning problem in a model selection problem that may be addressed using Support Vector Machines. The performance obtained demonstrate the ability of the method to build discriminant generative models able to cope efficiently with sequence classification as well as with segmentation. We plan to validate this work on on-line handwriting word recognition.

7. References

[1] Artières T., Marukatat S., Gallinari P., (2007), "Online Handwritten Shape Recognition Using Segmental Hidden Markov Models," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 29, no. 2, pp. 205-217.

[2] Bahlmann C., Haasdonk B., Burkhardt H. (2002), On-line Handwriting Recognition using Support Vector Machines - A kernel approach. In Proceedings of the 8th Int. Workshop on Frontiers in Handwriting Recognition(IWFHR), pp. 49-54

[3] Guyon I., Schomaker L., Plamondon R., Liberman M., Janet S. (1994), UNIPEN project of on-line data exchange and recognizer benchmark, International Conference on Pattern Recognition, pp. 29-33

[4] Jaakkola T., Diekhans M., Haussler D. (1998), Exploiting generative models in discriminative classifiers, Advances in Neural Information Processing Systems 11, San Mateo, CA, pp. 487-493

[5] Jaakkola T., Diekhans M., Haussler D. (1999), Using the Fisher kernel method to detect remote protein homologies, International Conference on Intelligent Systems for Molecular Biology, pp. 149-158

[6] Klautau A. (2002), Classification of Peterson & Barney's vowels using Weka, Technical report, UFPA.

[7] Lee J.J., Kim J. and Kim J.H. (2001), Data-driven design of HMM toplogy for on-line handwriting recognition, International Journal of Pattern Recognition and Artificial Intelligence, Vol. 15, n° 1, pp. 107-121.

[8] Marukatat S., Artières T., Gallinari P (2004). A generic approach for on-line handwriting recognition, in IWFHR, Tokyo, pp. 401-406

[9] Moreno Pedro J., Ho Purdy P., Vasconcelos N. (2003), A Kullback-Leibler Divergence Based Kernel for SVM classification in Multimedia applications, NIPS.

[10] Vapnik V, Ben-Hur A., Horn D., Siegelmann H.T (2001), A Support Vector Method for Hierarchical Clustering, NIPS 13, pp 367-273.

[11] Vapnik V. (1995), The Nature of Statistical Learning Theory, Eds Springer-Verlag, New York.

[12] Watkins V, (2000), Dynamic alignment kernels. In A. Smola, P. Bartlett, and B. Scholkopf, editors, Advances in large margin classifiers, pages 39--50. The MIT Press.