*supervised learning, Bayesian inference,
face recognition*

Alina MOMOT, Michał KAWULOK [*]

SPARSE BAYESIAN LEARNING IN CLASSIFYING FACE FEATURE VECTORS

The Relevance Vector Machine (RVM), a Bayesian treatment of generalized linear model of identical functional form to the Support Vector Machine (SVM), is the recently developed machine learning framework capable of building simple models from large sets of candidate features. The paper describes the application of the RVM to a classification algorithm of face feature vectors, obtained by Eigenfaces method. Moreover, the results of the RVM classification are compared with those obtained by using both the Support Vector Machine method and the method based on the Euclidean distance.

1. INTRODUCTION

The aim of the machine learning is extracting the structure from the data. Anyhow it is often difficult to solve problems like classification in the space, in which the underlying observations have been made. Kernel-based learning methods like the Support Vector Machine (SVM) [7] or the Relevance Vector Machine (RVM) [4] use implicit mapping of the input data into a high dimensional feature space defined by a kernel function and the learning takes place in the feature space. An interesting property of kernel-based systems is that, once a valid kernel function has been selected, one can practically work in spaces of any dimension without paying any computational cost, since feature mapping is never effectively performed.

A human face recognition is a complex problem which utilizes various techniques of image and data processing. Its most general aim is to provide a method of measuring similarity between any pair of images containing human faces. In this paper a possibility of applying the mentioned RVM learning method to a problem of face recognition is analysed and the results of face feature vectors classification are presented.

2. GENERALIZED LINEAR MODELS

In supervised learning it is given a set of examples of input vectors $\{x_n\}_{n=1}^N$ along with corresponding targets $\{y_n\}_{n=1}^N$, the latter of which might be real values (in regression) or class label (in classification). From this training set a model of the dependency of the targets on the inputs is learned with the objective of making accurate predictions of *y* for previously unseen *x*.

[*] Silesian University of Technology, Institute of Computer Science, 16 Akademicka St., 44-101 Gliwice, Poland

Generalised linear models, commonly used form of models for both classification and regression problems, take the form:

$$y(x;\boldsymbol{w}) = \sum_{i=1}^{M} w_i \varphi_i(x) + w_0 \quad (1)$$

where the output is linearly-weighted sum of *M*, generally nonlinear and fixed, basis functions $\{\varphi_i\}$ and the learning is the process of finding some weights which offer a good fit to the provided training data. When it is considering binary classification problems, following statistical convention the logistic link function is applied to the model output:

$$\sigma(x) = \frac{1}{1+e^{-y(x;\boldsymbol{w})}} \quad (2)$$

This renormalizes the model output such as $0 \le \sigma(x) \le 1$, and can be interpreted as a probability that *x* is a member of the "positive" class for the classification problem.

3. THE RELEVANCE VECTOR CLASSIFICATION

The Relevance Vector Machine (RVM) [4] has been fashioned from a Sparse Bayesian Learning (SBL) framework [8] for learning generalized linear models, whose predictors are sparse in that they contain relatively few non-zero w_i parameters. The RVM is named by analogy to the better known Support Vector Machine method [7], which is also a kind of sparse generalized linear models trainer.

Initially the RVM was presented as an alternative and direct competitor to the SVM. The SVMs can only be applied to learning a restricted subset of generalized linear models – those that can be defined by kernel functions (basis functions mentioned above, one basis function for each example in the training set) satisfying Mercer's condition [2] – while the RVM can learn a model with any collection of basis functions. Another disadvantage of the SVM is 'hard' binary decision while the RVM provides the conditional distribution *p*(*t*|*x*) in order to retrieve information about uncertainty in prediction. The key feature of the Support Vector Classification method is that its predicting function attempts to minimize a measure of error on the training set while simultaneously maximizing the "margin", in the feature space, between the two classes. It leads to necessity to estimate the error/margin trade-off parameter *C*, which generally entails a cross-validation procedure. On the other hand in the RVM learning process only the parameters of kernel function need to be estimated.

Briefly, in the Relevance Vector Machine method for solving a binary classification problem [5], where each training datum x_n has a label t_n (either 0 or 1), the probability that the data set is correctly labelled given some classifier model $\sigma(x)$ can be represented as:

$$p(\boldsymbol{t} \mid \boldsymbol{w}) = \prod_{n=1}^{N} \sigma(x)^{t_n} (1-\sigma(x))^{1-t_n} \quad (3)$$

Assuming that the training data is correctly labelled, Bayes' theorem allows to turn this expression around and infer likely values of the weights given some labelled data:

$$p(\boldsymbol{w} \mid \boldsymbol{t}) \propto p(\boldsymbol{t} \mid \boldsymbol{w}) p(\boldsymbol{w} \mid \boldsymbol{\alpha}), \tag{4}$$

where prior distribution over weight vector $\boldsymbol{w}$ is a zero-mean Gaussian:

$$p(\boldsymbol{w} \mid \boldsymbol{\alpha}) = \prod_{i=0}^{N} N(w_i; 0, \alpha_i^{-1}), \tag{5}$$

with $\boldsymbol{\alpha}$ a vector of $N+1$ hyperparameters. Importantly, there is an individual hyperparameter associated independently with every weight, moderating the strength of the prior. The most probable values of the hyperparameters are iteratively estimated from the training set data in manner described in [5]. Sparsity in this model is achieved because in practice one can find that the posterior distributions of many of the weights are sharply (indeed infinitely) peaked around zero [8]. The training vectors associated with the remaining non-zero weights are called relevance vectors. It is interesting that, unlike for the SVM, the relevance vectors are some distance from the decision boundary (in input data vectors space), appearing more "prototypical" or even "anti-boundary" in character [5].

4. HUMAN FACE RECOGNITION

Human face recognition is a complex problem which utilizes various techniques of image and data processing. Its most general aim is to provide a method of measuring similarity between any pair of images containing human faces. When a face is detected in the image and normalized (Figure 1), feature extraction must be applied to create a feature vector describing the face. The feature vectors should be easily comparable with each other in order to measure similarity between the images, from which they have been derived.

One of the feature extraction methods is based on the Principal Component Analysis and is called the Eigenfaces method [6]. This method creates a face space and makes it possible to project a normalized face image from the input image space into a less dimensional face space. Therefore, after the feature extraction an image is represented by a point in the face space. Similarity between the Eigenfaces feature vectors can be measured based on Euclidean or Mahalanobis distance between two points in the face space. The smaller the distance, the greater value of similarity.

Other approach is to use a classifier based on machine learning for assessing the similarity. It is possible to learn such a classifier to discriminate between all the classes of a given data set, in which one class consists of feature vectors belonging to one person. The main drawback of this method is that a classifier is fitted to a certain case and if there emerges a need for adding a new class, it is necessary to retrain the classifier.

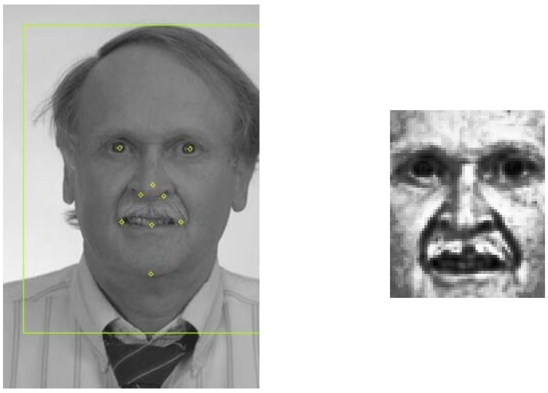

Fig. 1. Original image with detected face and facial features (left) and the same image after the normalisation (right)

There is also a more universal approach in which a difference between two feature vectors is analyzed. If there are two feature vectors $\boldsymbol{v}^{1}$ and $\boldsymbol{v}^{2}$ to be compared, at first an absolute difference vector $\boldsymbol{v}^{d}$ is created:

$$\forall i \in \{1,2,...,n\} \qquad v_i^d = \left|v_i^1 - v_i^2\right|, \qquad (6)$$

where n is the number of components of the vectors $\boldsymbol{v}^{1}$, $\boldsymbol{v}^{2}$ and $\boldsymbol{v}^{d}$. The difference vector $\boldsymbol{v}^{d}$ can have either intra-personal or extra-personal nature depending on classes, from which the feature vectors have been derived. If two feature vectors belong to one class, their difference vector has an intra-personal nature. Otherwise, if two feature vectors from different classes are subtracted from each other, an extra-personal difference vector is created. Hence, a classifier can be trained to distinguish between these two classes of difference vectors and decide whether two feature vectors describe the same face or not.

5. RESULTS

All the experiments for RVM were run in the MATLAB environment with using "SparseBayes V1.0" (Matlab code to implement sparse Bayesian regression and classification models)[1] written by M. Tipping. For the numerical experiments there were used images from the Feret face image database [3].

During the experiment 128-dimensional feature vectors were generated by the Eigenfaces method mentioned above. For the training stage, as the input vectors, there were randomly chosen 400 vectors of absolute differences in subset of input image of the same person (called internal difference vectors) for class labelled “1” and 400 vectors of absolute differences corresponding with different persons (called external difference vectors) for class labelled “0”. Means and standard deviations of input vectors of training data set are given in Figure 2 and Figure 3.

[1] available at *http://www.research.microsoft.com/mlp/rvm/SparseBayesV1.00.tar.gz*

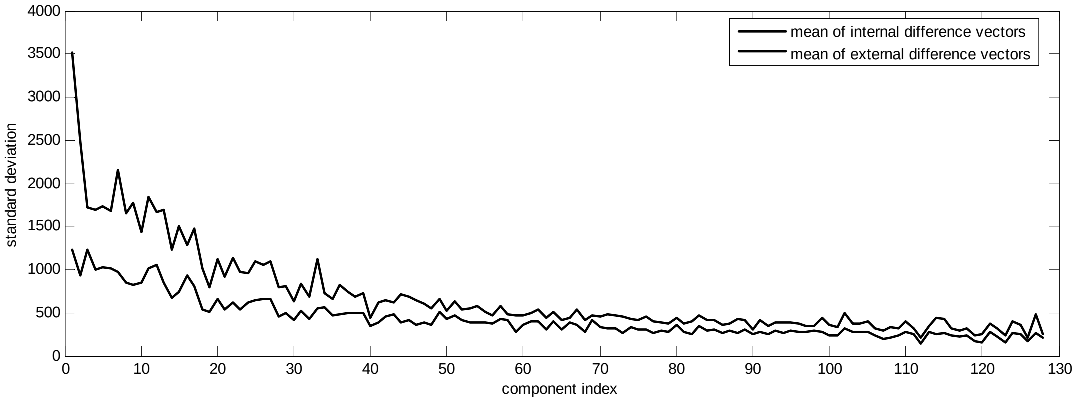

Fig. 2. Means of feature vectors of training data set.

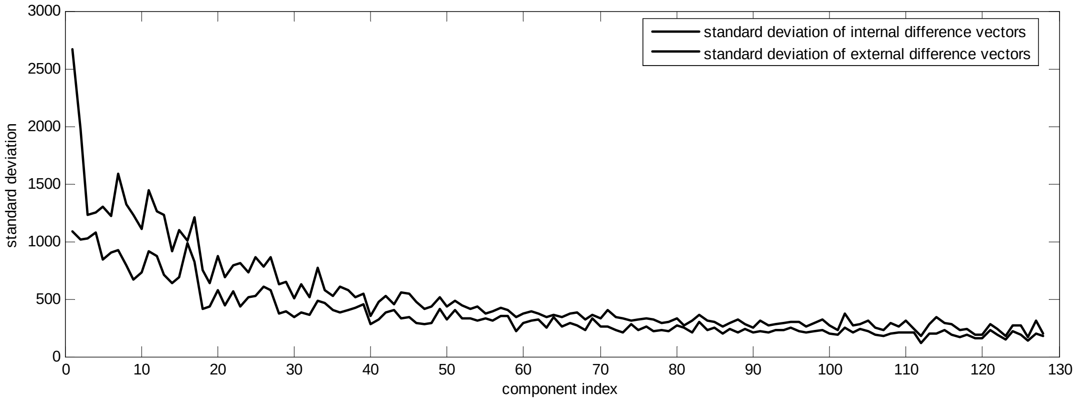

Fig. 3. Standard deviations of feature vectors of training data set.

For testing stage two sets, containing 1000 images each, were created. The first one will be denoted FeretA and consisted mainly of images taken in good lighting conditions which were relatively easy to recognize. The second set, which will be denoted FeretC, had more difficult cases for recognition and this could be the main reason for a significant difference in the classification results. In FeretA there were images of 395 different persons and in FeretC – 237. From both sets a template subset was distinguished which contained one image per person. During the experiment every 128-dimensional feature vector which was generated by the Eigenfaces method from the query set, FeretA or FeretC respectively, was compared with all the vectors from its corresponding template set. Absolute difference vectors for each pair of vectors, 395 or 237 for FeretA or FeretC respectively, were classified by the RVM method.

The RVM was trained for three different kernel functions: Gaussian with width parameter *s* defined as

$$K(x, x_i) = \exp\left(-\frac{\|x - x_i\|_2^2}{s^2}\right), \tag{7}$$

Laplacian with width parameter *s* defined as

$$K(x,x_i)=\exp\left(-\frac{\|x-x_i\|_2}{s}\right) \tag{8}$$

and Cauchy (heavy tailed) kernel function with width parameter *s* defined as

$$K(x,x_i)=\left(1+\frac{\|x-x_i\|_2^2}{s^2}\right)^{-1} \tag{9}$$

Experiments were performed for parameters *s* equal 1, 10 and 100.

For the classification of images of FeretA, there were taken 1000 128-dimensional feature vectors obtained from PCA transformation described above and 395 128-dimensional feature vectors representing 395 classes. For each vector *x* from the 1000-element set there were computed 395 vectors of absolute difference with the vectors from the 395-element set. Classifying these vectors a 395-dimensional vector *y* was obtained. The maximal positive value of *y* components gives, by its index, information about the class the element *x* is classified to. Most desired values of components of *y* are all negative but one positive, nevertheless such a situation appears rarely.

The results of the described experiment are presented in Table 1. The first column of the table contains only type of kernel function, since for all considered parameters *s* the results are exactly equal. It is worth mentioning that all the classifiers give 100% classification accuracy (fraction of correctly classified vectors in percents) on the training data and all the classifiers have 412 non-zero parameters in the prediction function (411 relevance vectors).

Table 1. The results of classification for test set FeretA.

kernel fuction	classification accuracy	PN vectors	AN vectors
Gaussian	51.9 %	519	481
Laplacian	51.9 %	519	481
Cauchy kernel	85.7 %	519	481

The column labelled “PN vectors” gives the information about the number of resulting vectors that have all but one components negative and one positive and the column labelled “AN vectors” gives the information about the number of resulting vectors that have all components negative. The numbers in the two last columns indicate that in all cases of classification resulting vectors have either all components negative or all but one components negative and one positive. Moreover, the classifier based on Cauchy kernel, unlike the two others, has the ability of correct classification of some resulting vectors having all components negative.

The results of classification of FeretC images are presented in Table 2. For the test set there were taken 1000 128-dimensional feature vectors obtained from PCA transformation described above and 237 128-dimensional feature vectors representing 237 classes. As before for each vector *x* from the 1000-element set there were computed 237 vectors of absolute difference with the vectors from the 237-element set. Classifying these vectors a 237-dimensional vector *y* was obtained. The maximal positive value of the *y* components gives, by its index, information about the class the element *x* is classified to. Also in this case all considered parameters *s* gave the exactly equal results for each type of kernel function.

Table 2. The results of classification for test set FeretC.

kernel fuction	classification accuracy	PN vectors	AN vectors
Gaussian	24.3 %	237	763
Laplacian	24.3 %	237	763
Cauchy kernel	65.7 %	237	763

Analyzing the two last columns in Table 2, as in experiments for the FeretA dataset, one can see that in all cases of classification the resulting vectors have either all components negative or all components negative but one positive. However now the classifiers based on Gaussian or Laplacian kernel have the ability of correct classification of some resulting vectors having all components negative. Nevertheless the classification accuracy of classifiers based on Cauchy kernel is greater than in case for the others.

In numerical experiments also there was considered prediction function with the parameter w0 was equal zero for classifiers based on Cauchy kernel with width parameter equal 1. The classifier gives 78.8% classification accuracy on the training data and has 611 non-zero parameters in prediction function (611 relevance vectors). However, the classifier gives 85.6% classification accuracy for test set FeretA and 65.7% for test set FeretC. Nevertheless it is worth mentioning that in this case classification resulting vectors have always at least two components positive.

For comparison classification results, a few experiments were made using the Support Vector Machines [1] method and method based on Euclidean distance. In the case of the SVM, the Laplacian kernel (8) with s=1.45 and polynomial kernel (10) with s=1 and d=2, defined as

$$K(x, x_i) = \left(x\, x_i^T + s\right)^d \tag*{(10),}$$

were tested.

The results of the classification for FeretA and FeretC, obtained for the best SVM classifier parameters (chosen in cross-validation procedure) C = 1 and $\varepsilon = 0.001$ in case of Laplacian kernel and C = 1 and $\varepsilon = 0.001$ in case of Polynomial kernel, are presented in Table 3. This table also contains classification results for method based on the Euclidean distance.

Table 3 The results of the SVM and Euclidean distance classification.

comparison method		classification accuracy	
		FeretA	FeretC
SVM	Laplacian	83.5 %	63.5 %
	Polynomial	85.4 %	68.4 %
Euclidean distance		82.9 %	66.6 %

6. CONCLUSIONS

In this paper an application of the Relevance Vector Machine to face feature vectors classification has been presented. The results of the RVM classification were compared to those obtained by using both the Support Vector Machine method and the method based on Euclidean distance.

The experiments have confirmed that the RVM classification of difference vectors can be used instead of calculating the Euclidean distance between two feature vectors to judge whether they belong to the same person or to different ones. Moreover, the examples presented in this paper have effectively demonstrated that the Relevance Vector Machine can attain a comparable level of generalization accuracy as the well-established Support Vector approach. Importantly, it benefits from absence of any additional nuisance parameters to set, apart from the need to choose the type of kernel and any associated parameters. Furthermore it should be noted that the RVM methodology is applicable to arbitrary basis functions, not limited to Mercer kernel as in the SVM. However, the principal disadvantage of Relevance Vector methods is in the complexity of the training phase requiring $O(N^{2})$ storage and $O(N^{3})$ computation. Thus for large datasets, this makes training considerably slower than for the SVM.

BIBLIOGRAPHY

[1] CORTES C., VAPNIK V., Support vector networks. Machine Learning, 20:1–25, 1995.

[2] MERCER J., Functions of positive and negative type and their connection with the theory of integral equations. Philos. Trans. Roy. Soc. London, A 209, pp. 415 - 446, 1909.

[3] PHILLIPS P. J, WECHSLER H., HUANG J., AND RAUSS P., “The FERET database and evaluation procedure for face recognition algorithms,” Image and Vision Computing J, Vol. 16, No. 5, pp 295-306, 1998.

[4] TIPPING M., The Relevance Vector Machine. In Advances in Neural Information Processing Systems 12, pp. 652 - 658, MIT Press, Cambridge, 2000.

[5] TIPPING M., Sparse Bayesian learning and the relevance vector machine. Journal of Machine Learning Research, 1(2), pp. 211 - 244, 2001.

[6] TURK M., PENTLAND A., Face Recognition Using Eigenfaces, in: Proceedings of Computer Vision and Pattern Recognition 1991, p.586 – 591.

[7] VAPNIK V.N., The nature of statistical learning theory. Springer, New York, 1995.

[8] WIPF D.P., PALMER J.A., RAO B.D., Perspectives on Sparse Bayesian Learning, Neural Information Processing Systems, Vol. 16, pp. 249 – 256, MIT Press, 2004.