Multi-class Classification: A Coding Based Space Partitioning

Sohrab Ferdowsi[1], Svyatoslav Voloshynovskiy[1], Marcin Gabryel[2], and Marcin Korytkowski[2]

[1]University of Geneva, Centre Universitaire d'Informatique,
Battle Bât. A, 7 route de Drize, 1227 Carouge, Switzerland
[2]Institute of Computational Intelligence, Częstochowa University of Technology
Al. Armii Krajowej 36, 42-200 Częstochowa, Poland

Abstract. In this work we address the problem of multi-class classification in machine learning. In particular, we consider the coding approach which converts a multi-class problem to several binary classification problems by mapping the binary labeled space into several partitioned binary labeled spaces through binary channel codes. By modeling this learning problem as a communication channel, these codes are meant to have error correcting capabilities and thus performance improvement in classification. However, we argue that conventional coding schemes designed for communication systems do not treat the space partitioning problem optimally, because they are heedless of the partitioning behavior of underlying binary classifiers. We discuss an approach which is optimal in terms of space partitioning and advise it as a powerful tool towards multi-class classification. We then review the LDA, a known method for multi-class case and compare its performance with the proposed method. We run the experiments on synthetic data in several scenarios and then on a real database for face identification.

Keywords: Multi-Class Classification, Error Correcting Output Codes, Support Vector Machines, Linear Discriminant Analysis

1 INTRODUCTION

The general multi-class classification is a fundamentally important problem that can find many different applications in different domains. Classification of alphabet letters from a handwritten document, phoneme segmentation in speech processing, face identification and content identification of multimedia segments are among the application examples of this problem. Although studied for many years, it is still considered as an open issue and none of the many proposed methods achieve superior performance under all conditions for different applications.

In binary classification where there are only two classes, many methods exist that are well developed to tackle different classification scenarios. However, they are designed naturally to address the binary case and their extension to multi-class is either not possible or is not straightforward. One way to accommodate for the multi-class case and still use the existing powerful binary algorithms is to break a multi-class problem to several binary problems and combine the results together. This method is done in several ways.

Among the existing methods that convert the multi-class problem to several binary ones there is the famous one-vs-all approach which considers each of the classes in each binary classification problem to be classified against the rest. This requires to run the binary classification algorithm $M-1$ times. In fact, because the situation is not symmetric, it might be the case that these binary problems are very different from each other in nature which might be problematic in practice. Another method would be to consider each pair of classes together. This method, known as one-vs-one approach, requires $\frac{M(M-1)}{2}$ binary classifications to capture all the combinations between the data.

These methods, while could work for small M's, will seriously fail as M grows because the number of binary classifications to be carried out would be intractable. The fact that in order to identify M objects would naturally require $\log_2 |\mathcal{M}|$ binary descriptors has motivated an extensive research in the field. A significant achievement was due to Dietterich [1] who made an analogy between the problem of classification and communication channel and thus suggested to consider it as a channel coding problem. This idea was followed by further research and study, however, no significant results have been reported.

An essential element of Dietterich approach is the selection of coding matrix. A coding matrix design was proposed in [2] which was shown to be optimal in terms of space partitioning for the problem of content identification of objects. In this paper we relax the assumption of identification where in each class there is only one instance and instead we address the general problem of multi-class classification where there is an arbitrary number of classes and arbitrary number of instances in each class as the training examples. We compare it with the Linear Discriminant Analysis (LDA), an efficient multi-class classification schemes in different classification scenarios and show that it could be efficiently used also for the general problem of multi-class classification.

LDA is a powerful technique in statistics that aims at finding a set of transformed features in a reduced dimension space such that they give best discrimination among classes. The use of LDA in machine learning, although not popularized, but has been extensively studied at least for the binary case. In an experimental study in [3], the use of LDA for multi-class classificaion has been investigated. It is shown that this method, although with certain limitations regarding its assumption on the distributions of classes, can serve as an efficient classification method also for the multi-class case.

The paper is organized as follows: In Section 2 we study the problem of multi-class classification. We review the coding approach for multi-class classification and introduce a method to optimally solve it. We then consider the LDA method towards multi-class classification. The experimental results are reported in Section 3. We finally summarize the paper in Section 4.

2 Multi-class Classification

In this section we first consider the binary classification and the SVM approach to solve it. We then define the problem of multi-class classification and explain an important approach towards solving this problem known as Error Correcting Output Codes (ECOC)[1]. Finally, we consider the design method introduced in [2] as the basis for our later discussions.

2.1 Binary Classification Using Support Vector Machines

Here we consider the general formulation of Support Vector Machine (SVM) for binary case. SVM is considered to be one of the most successful machine learning approaches for the binary supervised learning classification. When the two classes are linearly separable, the SVM algorithm tries to find a separating hyperplane such that it has the maximum margin from the closest instances of the two classes. The closest instances from each class are also called support vectors. This criterion, under certain conditions, is proved to guarantee the best generalization performance.

Concretely, given the training instances as $\mathbf{x}(i)$'s with $1 \leq i \leq s$ and $\mathbf{x}(i) \in \mathbb{R}^N$ and their binary labels as $g(\mathbf{x}(i)) \in \{-1, +1\}$ for each of $\mathbf{x}(i)$'s, we seek to find a hyperplane in $\mathbb{R}^N$ characterized by $\mathbf{w} \in \mathbb{R}^N$ and b such that it correctly classifies all the examples while it has the largest margin. Solving this problem is formulated as the optimization problem of (1). The objective function is maximizing the margin and the constraints try to ensure that the training instances are correctly classified, i.e., their predicted labels are the same as their true labels.

$$\begin{aligned} &\underset{\mathbf{w}}{\text{minimize}} && \frac{1}{2}\mathbf{w}^T\mathbf{w} \\ &\text{subject to} && g(\mathbf{x}(i))\left(\mathbf{w}^T\mathbf{x}(i) + b\right),\ i = 1, \ldots, s. \end{aligned} \tag{1}$$

This optimization problem, or its other variants are treated using quadratic programming techniques and many powerful packages are designed to practically handle it.

For the case where the training instances are not linearly separable, the above formulation can be modified by using the kernels technique. The idea is to use kernels which are pre-designed nonlinear functions that map the input space to a higher dimensional space. It could be the case that the instances in the new space are linearly separable. Therefore, (1) can be modified and used to find the best separating boundary between the instances in the higher dimensional space.

A useful method to gain insight into the spatial geometry of the instances could be derived from Voronoi diagrams. A Voronoi diagram is a way of partitioning the space into convex cells where in each cell there is only one instance from the given examples. The cells are chosen such that all the points in the cell are closest to the corresponding instance than other instances in space. Based on this definition, one could easily confirm that the SVM boundaries, when trained with enough parameters, should in fact follow the Voronoi cells of the set of support vectors. Figure 1 shows this phenomenon.

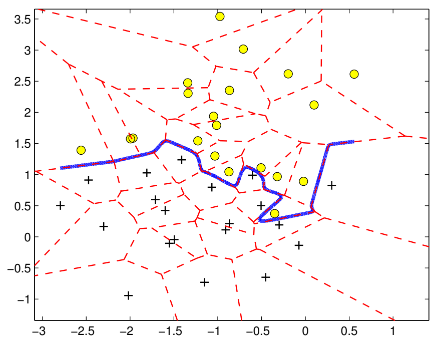


Fig. 1. SVM boundaries (solid lines) following the Voronoi cells (dashed lines) of the support vectors

It is important to mention that in all of these analyses, we are using the l_2-norm as the measure of distance. Other norms could also be studied which could offer advantages for some applications where the distortions have a non-Gaussian character.

2.2 Multi-class Classification: Problem Formulation

A set of training instances are given which contain features, $\mathbf{x}(i) \in \mathbb{R}^N$ and their labels $g(\mathbf{x}(i))$'s belonging to any of the M classes. These labels are assumed to have been generated through an unknown hypothetical mapping function $g : \mathbb{R}^N \longmapsto \{1, 2, ..., M\}$ that we want to approximate based on the given labeled training examples. The labels, unlike the common cases in machine learning, belong to a set of $M \geq 2$ members rather than only two classes. Because most of the existing classification algorithms are naturally designed for $M = 2$ classes, to generalize them for multi-class cases usually requires to consider the problem as several binary classification problems.

2.3 Coding Approach for Multi-Class Classification

The main idea behind the Error Correcting Output Codes (ECOC) approach to solve the multi-class problem is to consider it as a communication system where the identity of the correct output class for a given unlabeled example is being transmitted over a hypothetical channel which, due to the imperfections of the training data, the errors in the learning process and non-ideal choice of features is considered to be noisy [1]. Therefore, it would make sense to try to encode the classes using error correcting codes and transmit each of the bits through the channel, i.e., to run the learning algorithm, so that we would be able to cope with the errors in each individual binary classifier. Figure 2 illustrates this idea.

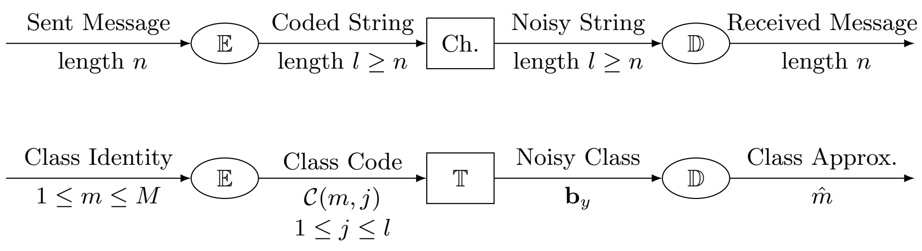


Fig. 2. Communication system and its classification equivalent: $\mathbb{E}$ and $\mathbb{D}$ are the Encoding and Decoding stages in communication, respectively. $\mathbb{T}$ is the training procedure, $\mathcal{C}$ is the coding matrix, $\mathbf{b}_y$ is the derived binary code for the test example.

Concretely, we assign randomly to each of the M classes a row of a coding matrix $\mathcal{C}_{(M \times l)}$. Then we run a binary learning algorithm on all the training samples for every column of $\mathcal{C}$ so that we will have l mappings from $\mathbb{R}^N$, the original data space to the one dimensional binary space $\{0, 1\}$, or equivalently, one mapping rule from $\mathbb{R}^N$ to the l-dimensional binary space $\{0, 1\}^l$.

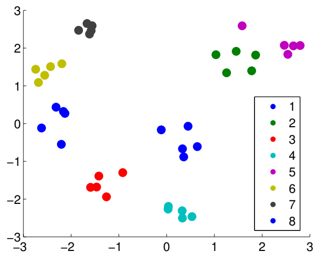
Given a new unlabeled example $\mathbf{y}$, we map it from $\mathbb{R}^N$ to the l-dimensional binary space through the same mapping rule learned as above. We then compare this binary representation $\mathbf{b}_y$ with the items in the database, or equivalently the rows of $\mathcal{C}$ by minimum Hamming distance rule or any other relevant decoding method.

Coding Matrix Design An important concern in this method is the choice of the coding matrix. Using the techniques from coding theory the main focus of the researchers has been to design elements of $\mathcal{C}$ optimally to be able to combat against noise while keeping the rate of communication as close to 1 as possible, which implies fewer number of binary classifications. Through this end, most design strategies were in essence aiming to maximize the Hamming distance between the rows of $\mathcal{C}$[4].

Another approach in the design of this matrix is to assign the elements randomly. It is shown that this random assignment approach could do as good as the previous coding-oriented designs [5, 6].

It is important to mention that in the design of channel codes, e.g., famous codes like BCH or LDPC, it is generally assumed that the channel noise, in our case the bit flippings of $\mathbf{b}_y$ due to the classification process, is *i.i.d.*, or at least is independent of the input codewords. In the case of classification where we have a hypothetical channel representing the behavior of the learning algorithm, however, we can observe that these bit flippings, or equivalently the channel noise samples are highly correlated with the input codewords. Therefore, many of the information theoretic explanations based on notions like typicality could no longer be valid. To address this issue, we consider this problem from a learning theoretic viewpoint.

Optimal Coding As mentioned earlier in section 1, a coding design was suggested in [2] which was shown to be optimal in terms of space partitioning. An intuitive explanation for this fact is presented below.


(a) instances ($M = 8$ and $p = 5$)

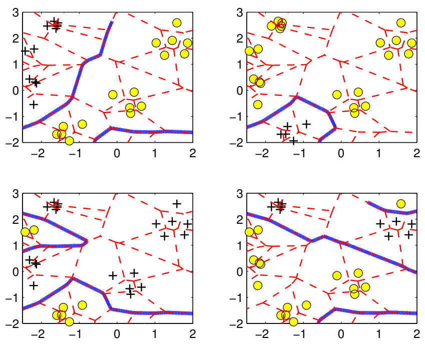

(b) binary classifications: SVM boundaries (solid) and Voronoi regions (dashed)

Fig. 3. Instances of the original multi-class problem and the corresponding binary classifications

Figure 3(a) shows the class instances and figure 3(b) shows the the corresponding decision boundaries of a binarized multi-class classification problem with $M = 8$ and $p = 5$ two dimensional instances for every class. As is clear from figure 3(b) and was discussed earlier in 1, the decision boundaries follow the outer Voronoi cells of each class. Notice that the choice of $\mathcal{C}$ dictates that in each of the l classifications, some of the adjacent classes be grouped together and labeled as '0' and some be grouped as '1' and also this labeling and grouping could be changed in each classification.

Therefore, in order to design an optimal coding matrix, we ask the question: Which choice of the elements of $\mathcal{C}$ guaranties learning at least one time these relevant Voronoi regions while avoiding redundant boundary learning? The first requirement links with having good performance via optimal class separability and the latter implies maintaining the number of columns (classifiers) as small as possible, i.e, having maximal rate of communication.

Equivalently, the optimal assignment of codewords should work as a unique and non-redundant encoding of each class with the decision boundaries of SVM classifier following the Voronoi regions between the corresponding classes.

This optimal design, in fact happens when the rows of $\mathcal{C}$ are chosen simply as the modulo-2 equivalent of the the class numbers. Therefore, as an example, in a 20-class problem, the codewords will be of length $l = \lceil \log_2 M \rceil = 5$, the 1^{st} class should be encoded as the binary string $'00000'$, and the 6^{th} class should be encoded as $'00101'$.

Intuitively, considering each of the two adjacent classes, this approach assigns to each of them a codeword where they differ in at least one position. This satisfies our first objective because it guaranties that each pair of rows of $\mathcal{C}$ that could be assigned to two adjacent classes finds at least one position where the bit values are different. Therefore, they will be mapped into two different classes at least once which implies that the decision boundary will cross between them at least once.

With this approach, the second requirement is also fully satisfied. Since the length of each codeword, i.e, the number of columns of $\mathcal{C}$ equals $l = \lceil \log_2 M \rceil$ which is minimal and the rate of communication is exactly 1.

In essence, the optimality of this approach is based on the fact that we learn the relevant Voronoi regions optimally. In fact learning the Voronoi regions of the support vectors is also equivalent to the maximum likelihood rule under Gaussian assumption for the classes. It is important to mention that we are using SVM classifiers with nonlinear kernels that are properly tuned. Other learning rules do not necessarily learn these Voronoi regions accurately and also the improper choice of kernel parameters does not guarantee learning these regions properly.

While ML decoding rule requires the knowledge of Voronoi regions for all classes, in our approach, these regions are distributed among the l binary classifiers and are learned implicitly. The fusion of the results of all binary classifiers, equivalently produces the entire coverage for all classes defined by their Voronoi regions. Therefore, the results of these fused binary decoding scheme should coincide with the optimal ML decoder.

It is also very important to mention that given a test example to be classified, unlike any other method, the complexity of decoding in this approach comprises only the SVM functional evaluations and does not incur any Hamming distance computation or decoding computations as in LDPC or other coding methods. The reason is due to the fact that a produced codeword is directly referring to a storage point in memory, hence no computation at all. However, the SVM functional evaluations involved could be very high if the nonlinear kernels are too complex.

2.4 Linear Discriminant Analysis

In this par,t we review the Discriminant Analysis approach which is a collection of techniques in statistical pattern recognition that try to find a set of discriminating features from the data by transforming them into a lower dimensional space. With their small complexity, their powerful behaviour in many classification scenarios and their theoretical importance, they are considered to be a good method of choice for many problems. Here we consider the famous LDA for the multi-class case.

Multi-class LDA Formulation Given the training dataset as $\mathcal{X} = [\mathbf{x}(1), \mathbf{x}(2), \cdots, \mathbf{x}(s)]^T$ with instance $\mathbf{x}(i) \in \mathbb{R}^N$, labeled as belonging to any of the M classes, c_i's, the objective is to find a transformation $\mathbb{W} \in \mathbb{R}^{N \times (M-1)}$ such that it maps the data into an $M-1$ dimensional space $\tilde{\mathbb{X}} = \mathbb{X}\mathbb{W}$ such that the data could be discriminated more easily. Equivalently, it draws $M-1$ hyperplanes between the M classes.

Given a testing sample $\mathbf{y}$, we transform it to the new space as $\tilde{\mathbf{y}} = \mathbf{y}^T\mathbb{W}$. We then find the transformed class center $\tilde{\mu}_i = \mu_i^T\mathbb{W}$ where $\mu_i = \frac{1}{p_i}\sum_{\mathbf{x}\in c_i}\mathbf{x}$, such that it has the smallest Euclidean distance to $\tilde{\mathbf{y}}$. The sample $\mathbf{y}$ is then classified to c_i.

The transform $\mathbb{W}$ should be found such that in the transformed space, the inter-class distances are the highest while the intra-class distances between the instances are the lowest, which is equivalent to having the maximal distinguishability between the classes.

Concretely, we define the intra-class scatter matrix for class c_i as:

$$\hat{\mathbb{S}}_i = \sum_{\mathbf{x}\in c_i}(\mathbf{x}-\mu_i)(\mathbf{x}-\mu_i)^T \tag{2}$$

and the total intra-class scatter matrix as:

$$\hat{\mathbb{S}}_{intra} = \sum_{i=1}^{M}\hat{\mathbb{S}}_i. \tag{3}$$

The inter-class scatter matrix is defined as:

$$\hat{\mathbb{S}}_{inter} = \sum_{i=1}^{M}p_i(\mu_i-\mu)(\mu_i-\mu)^T, \tag{4}$$

where $\mu = \frac{1}{s}\sum_{i=1}^{M}p_i\mu_i$. Jointly maximizing the inter-class scatter of the data in the transformed space and minimizing their intra-class scatters is equivalent to solving the generalized eigenvalue problem in (5),

$$\hat{\mathbb{S}}_{inter}\mathbb{W} = \lambda\hat{\mathbb{S}}_{intra}\mathbb{W} \tag{5}$$

Based on the $\mathbb{W}$ derived from Eq. 5, the test data $\mathbf{y}$ is classified as:

$$\hat{c} = \underset{i}{\operatorname{argmin}}\,||\mathbf{y}\mathbb{W} - \mu_i\mathbb{W}||_2, \tag{6}$$

which is searching for the closest class center to the test example in the transformed domain.

LDA for Multi-Class Classification LDA, along with its other variants could be considered as reliable methods for multiclass classification. The computational complexity of this approach could be significantly lower than other methods. However, there are certain limitations with this method.

Unlike the SVM classifier discussed above which only considers the boundary instances, LDA assumes an underlying distribution for the data. Therefore, in order for the distribution estimation to be accurate, the number of training instances, compared to the data dimension should not be small. This could be a serious limiting factor, as it could be the case in many applications that the training data are limited. Moreover, the underlying distribution under which the method is successful is assumed to be Gaussian. Therefore, deviations from Gaussianity could seriously affect the performance. We investigate this fact in our experimental results.

It is also important to point out that LDA tries to maximize distinguishability and does not necessarily consider minimizing the training error. For the multiclass case, in the eigenvalue decomposition problem of (5), pairs of classes between which there are large distances, completely dominate the decomposition. Therefore, in the transformed space, the classes with smaller distances will be overlapped [3].

3 Experimental Results

In this section we experimentally study the discussed approaches under different practical scenarios. Since there are many different parameters involved, especially for the multi-class case, to compare different algorithms is not trivial. The data dimension, the size of the training set, the way the instances of different classes are mixed with each other and the underlying distributions of the classes could significantly affect the classification scenario. Especially, when we generalize from binary to the multi-class case, we could see that these parameters and their relative values could radically change the scenario.

Therefore, instead of using only the current available databases which are very limited in terms of being rich in having different combinations of the parameters and therefore capturing a meaningful generality of an arbitrary scenario, we also experimented with synthetic databases. This way we can have the possibility to arbitrarily change these parameters.

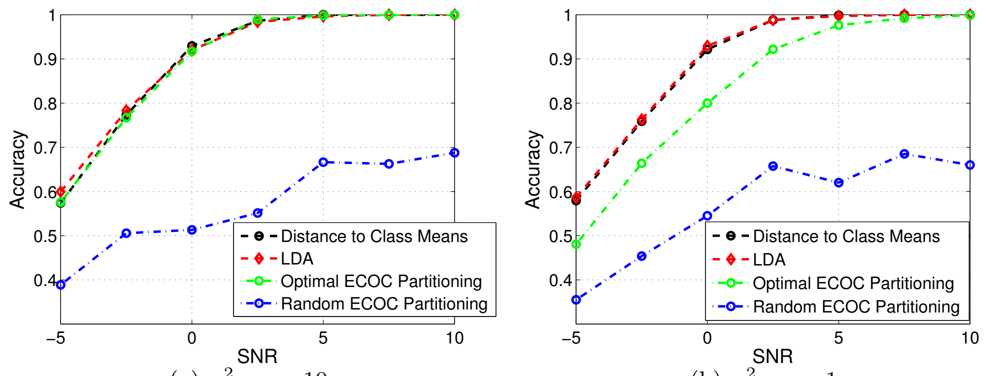
We first experiment the case where we generate the data from a Gaussian distribution where we can have control over inter-class and intra-class variances. We then experiment the case where each class is generated as a mixture of Gaussians. We finally report the result of comparison over the *ExtendedYaleB* [7] database for face recognition.

3.1 Synthetic Data

We perform the experiments on the synthetic data in two parts. First we consider Gaussian distribution for the classes. In order to imitate more realistic solutions, we then experiment with the cases where the classes follow mixtures of several Gaussians.

Gaussian Data We generate centers of classes as $\mathbf{x}_c(i) \in \mathbb{R}^N$ with $\mathbf{X}_c \sim \mathcal{N}(\mathbf{0}, \sigma^2_{inter}\mathbb{I}_N)$ and $1 \leq i \leq M$. We then generate the j^{th} instance of class c_i as $\mathbf{x}(j) = \mathbf{x}_c(i) + \mathbf{Z}_{intra}$ with $\mathbf{Z}_{intra} \sim \mathcal{N}(\mathbf{0}, \sigma^2_{intra}\mathbb{I}_N)$. The test data are generated as Additive White Gaussian Noise with covariance matrix $\sigma_z^2\mathbb{I}_N$ added to the centers of each class. We measure the performance of algorithms with accuracy which is defined as the ratio of all the correctly classified items to the whole test set size.

Figure 4(a) depicts the results of classification when the number of classes were chosen as $M = 16$, the data dimension as $N = 15$ and the inter-class and intra-class scatters were chosen as $\sigma^2_{inter} = 10$ and $\sigma^2_{intra} = 1$, respectively. The independent variable was the SNR which is defined as $SNR = 10\log_{10}\frac{\sigma^2_{inter}}{\sigma^2_Z}$. The comparison was made between four methods. The first method is to classify a test item based on its Euclidean distance closeness to the centers of each of the classes, estimated from the training data. This approach, given the Gaussian assumption for each of the classes, is optimal in terms of Maximum-Likelihood rule. The second method is the multi-class LDA. The third method is the discussed ECOC-based space partitioning with optimal coding matrix design. The forth method is ECOC when the matrix was randomly chosen. For the two last methods, the number of classifiers were $l = 4$ and the binary classifiers were SVM with Gaussian kernels. The number of train and test instances per each of the classes were chosen as 100 in all experiments. Figure 4(b) is the result of the same experiment but with a different ratio for inter-class and intra-class scatters. They were chosen as $\sigma^2_{inter} = 1$ and $\sigma^2_{intra} = 1$. As can be seen from figure 4(b),



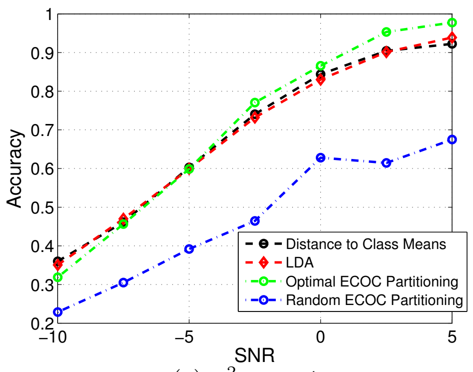
(a) $\sigma^2_{inter} = 10$ (b) $\sigma^2_{inter} = 1$

Fig. 4. Accuracy vs. SNR for Gaussian Data ($M = 16$, $N = 15$)

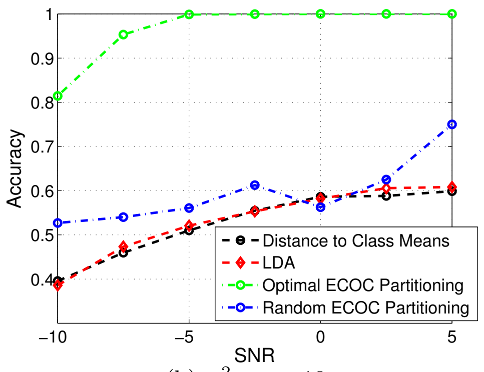
because the instances of each class are very scattered within the instances of the neighboring classes, the underlying binary SVM with Gaussian kernel tends to hugely overtrain on these training data, thus they fail to generalize on the test set. It is important to mention that the good behavior of the first two methods is due to the unrealistic Gaussian setup considered. We will see in the next experiments that this performance could radically decrease as we change the data distribution to more realistic ones.

Gaussian Mixtures In this part every class is assumed to have 5 Gaussian clouds from which the instances are generated. The class centers are generated as before. The centers of Gaussian clouds for class i, centered on $\mathbf{x}(i)$ are generated from another Gaussian distribution with covariance matrix $\sigma^2_{cloud}\mathbb{I}_N$. The class instances are then generated as Gaussians which are centered on the cloud centers with covariance σ^2_{intra}.

Figure 5(a) is the result of experiment when there were $M = 16$ classes with dimension $N = 15$. The variances were $\sigma^2_{inter} = 1$, $\sigma^2_{cloud} = 1$ and $\sigma^2_{intra} = 1$ and each of the 5 clouds there were 25 training and test examples.



(a) $\sigma^2_{cloud} = 1$



(b) $\sigma^2_{cloud} = 10$

Fig. 5. Accuracy vs. SNR for Mixture of Gaussian Data ($M = 16$, $N = 15$)

Figure 5(b) is the result of the same experiment with a more scattered clouds than the previous experiment, i.e., $\sigma^2_{cloud} = 10$ was changed.

As can be seen from 5(b), the first two methods are obviously sub-optimal for this scenario. However, the discussed ECOC-based space partitioning method can successfully cope with it. Also, in all of the experiments, it is clearly seen that random coding, when the number of its binary classifiers is small, cannot provide a good performance. To resolve this, also to avoid the fluctuating results seen in all the figures, one should add more columns to the coding matrix, or equivalently increase the number of binary classifications, while the optimal coding approach achieves the best performance with the least number of binary classifications. This optimality is of course limited and dictated by the choice of underlying classifications.

3.2 Face Identification

To compare the performance of the discussed methods on the real data, we used the *ExtendedYaleB* [7] database for face identification. There were 28 people, from each of them 500 face images were taken under different lighting conditions. We randomly chose 250 images from each for training and 250 for testing. The images were resized to 128×128 pixels and then directly vectorized. Random Projections were used to reduce the dimension of the features from $N = 16384$ to $L = 50$. The instances were then fed to the same four methods. For the last two methods, $l = \lceil log_2 28 \rceil = 5$ binary SVM's with Gaussian kernels were used. Table 1 illustrates the results.

Method	Euclidean Distance to Class Means	LDA	Optimal ECOC Partitioning	Random ECOC Partitioning
Accuracy	0.8771	0.8944	**0.9967**	0.7141

Table 1. Accuracy of different methods on *ExtendedYaleB* Database

4 Summary

In this paper we studied the problem of multi-class classification. We considered a general approach which was based on coding theory in communication systems that converts a multi-class problem to several binary problems by assigning a codeword to each class. We argued that conventional channel codes are not optimal for this problem as they do not take into account the partitioning properties of their underlying binary classifiers. We then proposed an approach for coding and showed that, if proper binary classifiers are used, it is optimally partitioning the multi-label space to several binary spaces. We tested the performance of the proposed methodology under different datasets. We first generated synthetic data that could simulate different real scenarios and then compared the performances under a real face identification database. The method was shown to be superior under more realistic classification scenarios. However, it was seen that because complex binary SVM's were used, one should be careful to avoid over-training of the underlying classifiers.

Acknowledgments

The work presented in this paper was supported by a grant from Switzerland through the Swiss Contribution to the enlarged European Union (PSPB-125/2010).

References

1. Dietterich, T.G., Bakiri, G.: Solving multiclass learning problems via error-correcting output codes. J. Artif. Intell. Res. (JAIR) **2** (1995) 263–286
2. Ferdowsi, S., Voloshynovskiy, S., Kostadinov, D.: Content identification: binary content fingerprinting versus binary content encoding. In: Proceedings of SPIE Photonics West, Electronic Imaging, Media Forensics and Security V, San Francisco, USA (January, 23 2014)
3. Li, T., Zhu, S., Ogihara, M.: Using discriminant analysis for multi-class classification: an experimental investigation. Knowledge and information systems **10**(4) (2006) 453–472
4. Murphy, K.P.: Machine Learning: A Probabilistic Perspective (Adaptive Computation and Machine Learning series). The MIT Press (8 2012)
5. James, G., Hastie, T.: The error coding method and picts. J. Computational and Graphical Statistics **7:3** (1998) 377–387
6. Voloshynovskiy, S., Koval, O., Beekhof, F., Holotyak, T.: Information-theoretic multiclass classification based on binary classifiers. Signal Processing Systems **65** (2011) 413–430
7. Lee, K., Ho, J., Kriegman, D.: Acquiring linear subspaces for face recognition under variable lighting. IEEE Trans. Pattern Anal. Mach. Intelligence **27**(5) (2005) 684–698