Nearest Class Vector Classification for Large-Scale Learning Problems

Alexandros Iosifidis, Anastasios Tefas and Ioannis Pitas
Department of Informatics, Aristotle University of Thessaloniki, Thessaloniki, Greece
Email: {tefas,pitas}@aiia.csd.auth.gr

***Abstract*—In this paper, we describe a method for combined metric learning and classification, that is based on logistic discrimination for the determination of a low-dimensional feature space of increased discrimination power. An iterating optimization process is applied to this end, where the probability of correct classification rate is increased at each optimization step. Extensions of the method that allow richer class representations and non-linear feature space determination and classification are also described. The described optimization schemes are solved by following (stochastic or mini-batch) gradient descent optimization, which is well suited for large-scale learning problems.**

***Keywords*-Nearest Class Vector classification; Logistic Discrimination;**

I. Introduction

Metric learning approaches have been widely adopted in many Computer Vision tasks, including image and action classification [1], [2], [3], [4], [5]. Among them, the Nearest Class Centroid (NCC) classification scheme has been widely adopted due to its simplicity and low computational and storage cost in both the training and the test phases. In addition, it has been recently shown that the in Big Data applications the performance of the NCC classifier, when combined with a metric learning approach, is competitive to that of more sophisticated classification schemes, like Support Vector Machine (SVM)-based classifier, while at the same time its training process is much more efficient [3]. Roughly, the idea of metric learning approaches is the determination of a data projection process to a feature space of increased discrimination power, where simple classification schemes can be exploited.

Metric learning approaches include LDA and its variances [1], [2], LESS [6], Taxonomy Embedding [7], the Sift-bag kernel [8], nearest class centroid classifier based on Logistic Regression [3] and sample-to-class metric learning [9]. LDA determines an optimal discriminant subspace by maximizing the between-class to within-class scatter ratio assuming unimodal class probability distributions having the same covariance structure and employing the mean class vectors for classes representation. The LESS model [6], is used to learn a diagonal scaling matrix for the modification of the Euclidean distance by scaling the data dimensions and includes an l_1 penalty term in order to perform feature selection. Taxonomy Embedding [7] exploits a hierarchical cost function in order to map the samples to a lower dimensional feature space where each class is represented by the class mean vector. The Sift-bag kernel [8] determines a lower dimensional feature space that is orthogonal to the subspace with the maximal within-class variance, which is evaluated by employing the class mean vectors. The nearest class centroid classifier of [3] employs logistic regression for the determination of a Mahalanobis metric using the class mean vectors and is closely related to our work. Finally, the sample-to-class metric learning [9], learns a Mahalanobis metric by employing a Naive-Bayes Nearest Neighbor (NN) approach and, thus, requires the storage of all training samples.

In this paper, we describe a metric learning algorithm based on multi-class logistic discrimination, where a sample is enforced to be closer to its class representation than to any other class representation in the projection space. The algorithm determines both the optimal projection matrix and the optimal class representation that can be, subsequently, used for classification. A learning process that is based on (stochastic or mini-batch) gradient descent optimization is applied, where updates of the projection matrix and the class representation are performed iteratively. Such a learning process is well-suited for large-scale learning problems [3]. In order to distinguish our approach from the NCC classifier, it is referred to as the Nearest Class Vector (NCV) classifier hereafter. In order to overcome the unimodality assumption that is inherently set by all the NCC, including NCV, classifiers, we describe an extension, namely Nearest Subclass Vector (NSV) classifier, which exploits multiple representations per class. Finally, since kernel methods have been found to be very effective for the classification of human actions [10], [11], [12], we describe an extension of both the NCV and the NSV classification schemes that is able to determine an optimal data projection and optimal class representations in arbitrary-dimensional Hilbert spaces [13].

Compared to the NCC classifier, NCV has the advantage that, by optimizing the adopted criterion with respect to the data projection and the class representation(s), increased class discrimination in the projection space can be achieved. Compared with metric learning approaches exploiting NN-based classification schemes, NCV requires the storage of only few class vectors (the representative ones) and, thus, has lower computation and storage costs in the test phase.

II. NCV-BASED CLASSIFICATION

The NCV classifier assigns a sample $\mathbf{x}_i \in \mathbb{R}^D$ to the class $c^* \in \{1, \dots, C\}$ of the closest class vector, i.e.:

$$c^* = \arg\min_c d(\mathbf{x}_i, \boldsymbol{\mu}_c), \tag{1}$$

where $\boldsymbol{\mu}_c \in \mathbb{R}^D$ is the representation of class c and $d(\mathbf{x}_i, \boldsymbol{\mu}_c) = (\mathbf{x}_i - \boldsymbol{\mu}_c)^T \mathbf{M} (\mathbf{x}_i - \boldsymbol{\mu}_c)$ is the (squared) Mahalanobis distance between $\mathbf{x}_i$ and $\boldsymbol{\mu}_c$ in $\mathbb{R}^D$. By setting $\mathbf{M} = \mathbf{W}^T\mathbf{W}$, where $\mathbf{W} \in \mathbb{R}^{d \times D}$, $d(\mathbf{x}_i, \boldsymbol{\mu}_c)$ is given by:

$$\begin{aligned} d_{\mathbf{W}}(\mathbf{x}_i, \boldsymbol{\mu}_c) &= (\mathbf{x}_i - \boldsymbol{\mu}_c)^T \mathbf{W}^T\mathbf{W} (\mathbf{x}_i - \boldsymbol{\mu}_c) \\ &= \|\mathbf{W}\mathbf{x}_i - \mathbf{W}\boldsymbol{\mu}_c\|_2^2. \end{aligned} \tag{2}$$

In the case where $d < D$, equation (2) is equivalent to calculating the (squared) Euclidean distance of the image of $\mathbf{x}_i$ and $\boldsymbol{\mu}_c$ in $\mathbb{R}^d$, i.e., $\tilde{\mathbf{x}}_i = \mathbf{W}\mathbf{x}_i$ and $\tilde{\boldsymbol{\mu}}_c = \mathbf{W}\boldsymbol{\mu}_c$. That is, $\mathbf{W}$ can be considered as a projection matrix mapping the data in a d-dimensional feature space, where classification is performed by using the minimal Euclidean distance from the class vectors $\boldsymbol{\mu}_c$, $c = 1, \dots, C$.

NCV classifier exploits a probabilistic model based on multi-class logistic regression. The probability of observing class c given a vector $\mathbf{x}_i$ is defined by:

$$p(c|\mathbf{x}_i) = \frac{e^{-\frac{1}{2}d_{\mathbf{W}}(\mathbf{x}_i, \boldsymbol{\mu}_c)}}{\sum_{l=1}^{C} e^{-\frac{1}{2}d_{\mathbf{W}}(\mathbf{x}_i, \boldsymbol{\mu}_l)}}. \tag{3}$$

The optimal parameters $\{\mathbf{W}^*, \boldsymbol{\mu}_c^*\}$, $c = 1, \dots, C$ are calculated by exploiting a training data set formed by N vectors $\mathbf{x}_i \in \mathbb{R}^D$ followed by the corresponding class labels y_i. $\{\mathbf{W}^*, \boldsymbol{\mu}_c^*\}$, $c = 1, \dots, C$ are defined as those maximizing the mean log-likelihood of all the N training samples:

$$\mathcal{J}(\mathbf{W}, \boldsymbol{\mu}_c) = \frac{1}{N} \sum_{i=1}^{N} ln\, p(y_i|\mathbf{x}_i). \tag{4}$$

In the case where the distribution of training samples is not representative of the real class distributions, their contribution to $\mathcal{J}$ calculation can be weighted.

$\mathcal{J}$ is minimized with respect to both $\mathbf{W}$ and $\boldsymbol{\mu}_c$ by applying an Expectation Maximization-like iterative optimization process. That is, for a given set of class vectors $\boldsymbol{\mu}_{c,t}$ calculated at iteration t, the data projection matrix is updated by following the gradient of $\mathcal{J}$ with respect to $\mathbf{W}$, i.e. $\mathbf{W}_{t+1} = \mathbf{W}_t + \eta_{\mathbf{W}} \nabla_{\mathbf{W}} \mathcal{J}$. By using $\mathbf{W}_{t+1}$ the class vectors are, subsequently, updated by following the gradient of $\mathcal{J}$ with respect to $\boldsymbol{\mu}_c$, i.e., $\boldsymbol{\mu}_{c,t+1} = \boldsymbol{\mu}_{c,t} + \eta_{\boldsymbol{\mu}} \nabla_{\boldsymbol{\mu}_c} \mathcal{J}$. $\eta_{\mathbf{W}}$ and $\eta_{\boldsymbol{\mu}}$ are the update rate parameters used to adapt $\mathbf{W}$ and $\boldsymbol{\mu}_{c,t}$, respectively. The derivatives of $\mathcal{J}$ with respect to $\mathbf{W}$ and $\boldsymbol{\mu}_c$ are given by:

$$\nabla_{\mathbf{W}} \mathcal{J} = \frac{1}{N} \sum_{i,c=1}^{N,C} \Big(p(c|\mathbf{x}_i) - \alpha_i^c \Big) \mathbf{W} \mathbf{q}_i^c \mathbf{q}_i^{c\,T}, \tag{5}$$

$$\nabla_{\boldsymbol{\mu}_c} \mathcal{J} = \frac{1}{N} \sum_{i=1}^{N} \alpha_i^c \Big(1 - p(c|\mathbf{x}_i) \Big) \mathbf{W}^T \mathbf{W} \mathbf{q}_i^c, \tag{6}$$

where $\mathbf{q}_i^c = \boldsymbol{\mu}_c - \mathbf{x}_i$ and α_i^c is an index denoting if $\mathbf{x}_i$ belongs to class c, i.e., $\alpha_i^c = 1$ if $y_i = c$ and $\alpha_i^c = 0$ otherwise. The above described iterative optimization scheme is performed until $(\mathcal{J}_{t+1} - \mathcal{J}_t)/\mathcal{J}_t < \epsilon$, where ϵ is a small positive value (equal to 10^{-8} in our experiments). The class representations are initialized to the class mean vectors, i.e., $\boldsymbol{\mu}_{c,1} = \mathbf{m}_c$, $c = 1, \dots, C$, where $\mathbf{m}_c = \frac{1}{N_c} \sum_{i:y_i=c} \mathbf{x}_i$.

In the case where the classes forming the classification problem consist of multiple subclasses C_c, $c = 1, \dots, C$, each represented by a set of class vectors $\boldsymbol{\mu}_{cj}$, $j = 1, \dots, C_c$, the probability of observing class c given a vector $\mathbf{x}_i$ is given by:

$$p(c|\mathbf{x}_i) = \sum_{j=1}^{C_c} p(c_j|\mathbf{x}_i) \tag{7}$$

$$p(c_j|\mathbf{x}_i) = \frac{e^{-\frac{1}{2}d_{\mathbf{W}}(\mathbf{x}_i, \boldsymbol{\mu}_{cj})}}{\sum_{l=1}^{C} \sum_{k=1}^{C_l} e^{-\frac{1}{2}d_{\mathbf{W}}(\mathbf{x}_i, \boldsymbol{\mu}_{lk})}}. \tag{8}$$

$\{\mathbf{W}^*, \boldsymbol{\mu}_{cj}^*\}$, $c = 1, \dots, C$, $j = 1, \dots, C_c$ are calculated by applying the above-described iterative optimization process using the following gradients of $\mathcal{J}$ with respect to $\mathbf{W}$ and $\boldsymbol{\mu}_{cj}$:

$$\nabla_{\mathbf{W}} \mathcal{J} = \frac{1}{N} \sum_{i,c,j=1}^{N,C,C_c} \Big(p(c_j|\mathbf{x}_i) - \alpha_i^c \beta_i^{cj} \Big) \mathbf{W} \mathbf{q}_i^c \mathbf{q}_i^{cj\,T}, \tag{9}$$

$$\nabla_{\boldsymbol{\mu}_{cj}} \mathcal{J} = \frac{1}{N} \sum_{i=1}^{N} \alpha_i^c \beta_i^{cj} \Big(1 - p(c_j|\mathbf{x}_i) \Big) \mathbf{W}^T \mathbf{W} \mathbf{q}_i^{cj}, \tag{10}$$

where $\mathbf{q}_i^{cj} = \boldsymbol{\mu}_{cj} - \mathbf{x}_i$ and $\beta_i^{cj} = \frac{p(c_j|\mathbf{x}_i)}{\sum_{l=1}^{C_c} p(c_l|\mathbf{x}_i)}$. That is, each training sample $\mathbf{x}_i$ contributes to the adaptation of $\boldsymbol{\mu}_{cj}$ according to its membership value β_i^{cj}. Since the subclasses are not a priori known, the subclass vectors $\boldsymbol{\mu}_{cj}$ are initialized by applying a clustering technique (e.g. K-Means) on the training samples $\mathbf{x}_i$ belonging to class c.

In order to perform nonlinear classification, the input space $\mathbb{R}^D$ is mapped to a feature space $\mathcal{F}$ of arbitrary dimensions (usually having the structure of a Hilbert space [13]). NCV-based classification is subsequently applied in $\mathcal{F}$, leading to nonlinear classification in the input space $\mathbb{R}^D$. Let us denote by $\phi : \mathbf{x}_i \in \mathbb{R}^D \rightarrow \phi(\mathbf{x}_i) \in \mathcal{F}$ a nonlinear mapping from $\mathbb{R}^D$ to $\mathcal{F}$. The application of NCV in $\mathcal{F}$ leads to the calculation of a data projection matrix $\mathbf{W}_\phi \in \mathbb{R}^{d \times |\mathcal{F}|}$ and a set of class vectors $\boldsymbol{\mu}_c^\phi \in \mathbb{R}^{|\mathcal{F}|}$, $c = 1, \dots, C$. In this case, the adopted distance function used for the calculation of the conditional class probabilities is given by:

$$d_{\mathbf{W}_\phi}\Big(\phi(\mathbf{x}_i), \boldsymbol{\mu}_c^\phi\Big) = \|\mathbf{W}_\phi \phi(\mathbf{x}_i) - \mathbf{W}_\phi \boldsymbol{\mu}_c^\phi\|_2^2. \tag{11}$$

However, since $\mathbf{W}_\phi$ is a matrix of arbitrary (even infinite) dimensions, the distance in (11) cannot be directly calculated.

In order to proceed, we express $\mathbf{W}_\phi$ and $\boldsymbol{\mu}_c^\phi$ as linear combinations of the training vectors (represented in $\mathcal{F}$) [13], i.e. $\mathbf{W}_\phi = (\boldsymbol{\Phi}\mathbf{A})^T$ and $\boldsymbol{\mu}_c^\phi = \boldsymbol{\Phi}\mathbf{b}_c$, where $\boldsymbol{\Phi} = [\phi(\mathbf{x}_1), \ldots, \phi(\mathbf{x}_N)]$ and $\mathbf{A}$, $\mathbf{b}_c$ are a matrix and a vector containing the reconstruction weights for $\mathbf{W}_\phi$ and $\boldsymbol{\mu}_c^\phi$ respectively, Equation (11) can now be written as:

$$\begin{aligned} d_{\mathbf{A}}\Big(\phi(\mathbf{x}_i), \boldsymbol{\mu}_c^\phi\Big) &= \|\mathbf{A}^T\boldsymbol{\Phi}^T\phi(\mathbf{x}_i) - \mathbf{A}^T\boldsymbol{\Phi}^T\boldsymbol{\mu}_c^\phi\|_2^2 \\ &= \|\mathbf{A}^T\mathbf{K}_i - \mathbf{A}^T\mathbf{K}\mathbf{b}_c\|_2^2, \qquad (12) \end{aligned}$$

where $\mathbf{K}$ is the kernel matrix, having elements equal to $k_{ij} = \phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)$, $i,j = 1,\ldots,N$, and $\mathbf{K}_i$ is the i-th column of $\mathbf{K}$, having elements equal to $k_{ji} = \phi(\mathbf{x}_j)^T\phi(\mathbf{x}_i)$, $j = 1,\ldots,N$.

By observing Equations (2),(3),(4) and (12), it can be seen that the problem to be solved has been transformed to the determination of the reconstruction weights $\mathbf{A}^*$ and $\mathbf{b}_c^*$ for optimal non-linear data projection and optimal class representation, respectively. In this case, the gradient of $\mathcal{J}$ with respect to $\mathbf{A}$ is given by:

$$\nabla_{\mathbf{A}}\mathcal{J} = \frac{1}{N}\sum_{i,c=1}^{N,C}\Big(p(c|\phi(\mathbf{x}_i)) - \alpha_i^c\Big)\mathbf{K}(\mathbf{z}_i^c\mathbf{z}_i^{c\,T})\mathbf{K}\mathbf{A}, \quad (13)$$

while the gradient of $\mathcal{J}$ with respect to $\mathbf{b}_c$ is given by:

$$\nabla_{\mathbf{b}_c}\mathcal{J} = \frac{1}{N}\sum_{i=1}^{N}\alpha_i^c\Big(1 - p(c|\phi(\mathbf{x}_i))\Big)\mathbf{K}\mathbf{A}\mathbf{A}^T\mathbf{K}\mathbf{z}_i^c. \quad (14)$$

$\mathbf{z}_i^c = \mathbf{b}_c - \mathbf{1}_i$, where $\mathbf{1}_i$ is a vector having all its elements equal to zero, except of the i-th element, which is equal to one.

By using the same analysis for the case of multimodal classes, $\mathbf{A}$ and $\mathbf{b}_{c_j}$ are updated by using the following gradients:

$$\nabla_{\mathbf{A}}\mathcal{J} = \frac{1}{N}\sum_{i,c,j=1}^{N,C,C_c}\Big(p(c_j|\phi(\mathbf{x}_i)) - \alpha_i^c\beta_i^{cj}\Big)\mathbf{K}(\mathbf{z}_i^{cj}\mathbf{z}_i^{cj\,T})\mathbf{K}\mathbf{A}, \quad (15)$$

$$\nabla_{\mathbf{b}_{cj}}\mathcal{J} = \frac{1}{N}\sum_{i=1}^{N}\alpha_i^c\beta_i^{cj}\Big(1 - p(c_j|\phi(\mathbf{x}_i))\Big)\mathbf{K}\mathbf{A}\mathbf{A}^T\mathbf{K}\mathbf{z}_i^{cj}. \quad (16)$$

$\mathbf{z}_i^{cj} = \mathbf{b}_{cj} - \mathbf{1}_i$ and $\beta_i^{cj} = \frac{p(c_j|\phi(\mathbf{x}_i))}{\sum_{l=1}^{C_j} p(c_l|\phi(\mathbf{x}_i))}$. That is, each training sample $\mathbf{x}_i$ contributes to the adaptation of $\mathbf{b}_{cj}$ according to its membership value β_i^{cj}, evaluated on $\mathcal{F}$.

In the above, in the case where each class is represented by one class vector, $\mathbf{b}_c$ is initialized by setting all its elements equal to zero, expect of the elements corresponding to the training samples belonging to class c which are set equal to $1/N_c$, where N_c is the number of training samples belonging to class c. That is, each class representation is initialized to the class mean vector in $\mathcal{F}$. In the case of multiple class representations per class, $\mathbf{b}_{cj}$ are initialized by applying clustering on the training data belonging to class c. However, in this case clustering should be performed on the training data representations in $\mathcal{F}$, e.g. by applying kernel K-Means [14], by using the kernel matrix of the training samples belonging to each class separately.

III. Experiments

We have employed three publicly available human action recognition data sets, namely the Hollywood2, the Olympic Sports and the ASLAN data sets. As baseline approaches, we use the state-of-the-art methods proposed in [15], [16]: on the ASLAN data set we employ a set of 12 similarity values calculated for histogram similarity measure between pairs of videos, represented by using the Bag of Words (BoW) model for HOG, HOF and HNF descriptors evaluated on STIP video locations [17]. This video pair similarity representation is employed for classification using a linear Support Vector Machine (SVM) classifier. We employ this baseline to evaluate the performance of the linear version of the NCV classifier. For the remaining data sets we employ the Bag of Words (BoW)-based video representation by using HOG, HOF, MBH and Trajectory descriptors evaluated on the trajectories of densely sampled interest points [15]. Classification is performed by employing a kernel SVM classifier and the χ^2 kernel. We employ this baseline to evaluate the performance of the kernel version of the NCV and NSV classification schemes.

In our experiments we have used an adaptive optimization process where $\eta_{\mathbf{W}}$, $\eta_{\boldsymbol{\mu}}$ are dynamically determined by following a linear search strategy. That is, in each iteration of the optimization process the criterion $\mathcal{J}$ is evaluated by using $\eta_{\mathbf{W},0} = 0.1$ (or $\eta_{\boldsymbol{\mu},0} = 0.1$). In the case where $\mathcal{J}_{t+1} > \mathcal{J}_t$, the criterion $\mathcal{J}$ is evaluated by using an update rate parameter equal to $\eta_{\mathbf{W},n+1} = 2\eta_{\mathbf{W},n}$ (or $\eta_{\boldsymbol{\mu},n+1} = 2\eta_{\boldsymbol{\mu},n}$). This process is followed until $\mathcal{J}_{t+1} < \mathcal{J}_t$ and the value providing the maximal increase in $\mathcal{J}$ is employed. In the case where, by using an update rate parameter equal to $\eta_{\mathbf{W},0} = 0.1$ (or $\eta_{\boldsymbol{\mu},0} = 0.1$), $\mathcal{J}_{t+1} < \mathcal{J}_t$, the criterion $\mathcal{J}$ is evaluated by using an update rate parameter equal to $\eta_{\mathbf{W},n+1} = \eta_{\mathbf{W},n}/2$ (or $\eta_{\boldsymbol{\mu},n+1} = \eta_{\boldsymbol{\mu},n}/2$). This process is followed until $\mathcal{J}_{t+1} > \mathcal{J}_t$ and the value increasing the criterion $\mathcal{J}$ is employed. We evaluate $\mathcal{J}$ after introducing all the training samples for the adaptation of $\mathbf{W}$, $\boldsymbol{\mu}_{c,t}$. However, (5,6) can be also employed by stochastic gradient ascent algorithms for the adoption of the NCV in large-scale classification problems [4]. For the initialization of the data projection matrices $\mathbf{W}$ and $\mathbf{A}$ we use random projections [18].

A. Data sets

The Action Similarity Labeling (ASLAN) data set [16] consists of thousands of videos collected from the web, in over 400 complex action classes. A "same/not-same" benchmark is provided, which addresses the action recognition problem as a video pair similarity problem. Example video frames of the data set are illustrated in Figure 1. We use the standard partitioning provided by the database. The database consists of ten splits of video pairs, each containing 300 pairs of same actions and 300 pairs of not-same actions. The splits contain mutually exclusive action classes. That is, action classes appearing in one split do not appear in any other split. Performance is evaluated by applying the ten-fold cross-validation procedure. In each fold, nine of the splits are used to train the algorithms and performance is measured on the remaining one. An experiment consists of ten folds, one for each test split, and performance is calculated by using the mean accuracy and the standard error of the mean (SE) over all folds.

The Hollywood2 data set [19] consists of 1707 videos depicting 12 actions. It has been collected from 69 different Hollywood movies. The actions appearing in the data set are: answering the phone, driving car, eating, fighting, getting out of car, hand shaking, hugging, kissing, running, sitting down, sitting up, and standing up. Example video frames of the data set are illustrated in Figure 2. We use the standard training-test split provided by the database (823 videos are used for training and performance is measured in the remaining 884 videos). Training and test videos come from different movies. The performance is evaluated by computing the average precision (AP) for each action class and reporting the mean AP over all classes (mAP), as suggested in [19]. This is due to the fact that some videos of the data set depict multiple actions.

The Olympic Sports data set [20] consists of 783 videos depicting athletes practicing 16 sports, which have been collected from YouTube and annotated using Amazon Mechanical Turk. The actions appearing in the data set are: high jump, long jump, triple jump, pole vault, basketball lay-up, bowling, tennis serve, platform, discus, hammer, javelin, shot put, springboard, snatch, clean-jerk and vault. Example video frames of the data set are illustrated in Figure 3. We use the standard training-test split provided by the database (649 videos are used for training and performance is measured in the remaining 134 videos). The performance is evaluated by computing the mean Average Precision (mAP) over all classes, as suggested in [20].

B. *Results*

The mean accuracy and the standard error values obtained by applying the NCV classifier on the ASLAN data set are illustrated in Table I. In this Table we also provide the mean accuracy and standard error values obtained by applying classification using linear SVM, NCC [3], LDA and the method in [21] that determines the optimal class vectors for the LDA criterion (referred to as RCVLDA). It can be seen that the NCV classifier outperforms LDA in all the cases, while SVM outperforms both LDA and NCC in all cases. The determination of the optimal class representation for the LDA criterion leads to an increase of the performance of

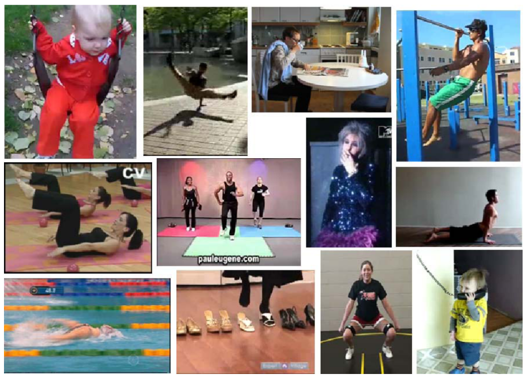

Figure 1. Video frames of the ASLAN data set.

Figure 2. Video frames of the Hollywood2 data set.

LDA. Specifically, RCVLDA outperforms LDA and NCC in all the cases, while it outperforms SVM in three out of four cases. Finally, the NCV algorithm outperforms SVM, LDA and NCC in all cases, while it outperforms RCVLDA in three out of four cases. Overall, the NCV classifier provides the best performance, equal to 61.4% (for $d = 5$), by using all the similarity values of the database. By using 10 subclasses per class, the NCV classifier further increases the performance to 61.66%.

The classification rates obtained by applying the kernel version of the NCV classifier on the Olympic Sports data set

Figure 3. Video frames of the Olympic Sports data set.

Table I
PERFORMANCE (ACCURACY ± SE) ON THE ASLAN DATA SET.

	HOG	HOF	HNF	ALL
SVM	57.78 ± 0.82 %	56.68 ± 0.56 %	59.47 ± 0.66 %	60.88 ± 0.77 %
LDA	50.33 ± 0.38 %	50.28 ± 0.27 %	49.82 ± 0.31 %	51.20 ± 0.43 %
NCC	56.83 ± 0.98 %	55.83 ± 0.73 %	57.83 ± 0.93 %	60.08 ± 0.92 %
RCVLDA	59.70 ± 0.91 %	**56.93 ± 0.63** %	59.17 ± 0.72 %	60.95 ± 0.81 %
NCV	**59.95 ± 0.6** %	56.58 ± 0.81 %	**60.08 ± 0.68** %	**61.4 ± 0.82** %

Table II
CLASSIFICATION RATES ON THE OLYMPIC SPORTS DATA SET FOR DIFFERENT TARGET DIMENSIONS d.

50	100	200	300	400
57.46 %	58.2 %	58.96 %	**61.94** %	**61.94** %

Table IV
PERFORMANCE (MAP) ON THE OLYMPIC SPORTS AND HOLLYWOOD2 DATA SETS.

	SVM	KDA	**NCV**
Olympic Sports	74.4 %	75.3 %	**76.85** %
Hollywood2	58.23 %	58.31 %	**58.85** %

for different dimensions d are illustrated in Table II. In this Table, we also provide the mean classification rates obtained by applying kernel SVM-based classification and Kernel Discriminant Analysis (KDA) [22] on the Olympic Sports data set. The NCV classifier outperforms the other two classification schemes. By using two subclasses per action class, the NSV classifier outperformed NCV, providing a classification rate equal to 63.43%.

Table III
MEAN CLASSIFICATION RATES ON THE OLYMPIC SPORTS DATA SET.

SVM	KDA	**NCV**
61.19 %	60.44 %	**61.94** %

The mean average precision values obtained by applying the kernel version of the NCV classifier on the Olympic Sports and the Hollywood2 data sets are illustrated in Table IV. In this Table, we also provide the mean average precision values obtained by applying kernel SVM and KDA on the two data sets. As can be seen, NCV outperformed both SVM and KDA in both data sets providing mAP equal to 78.85% and 58.85% for the Olympic Sports and the Hollywood2 data sets, respectively. Overall, the distance-based classification from optimized class representation(s) provides comparable (or even better) performance with that other classifiers, like KDA-based and SVM-based classification schemes.

IV. CONCLUSION

In this paper we described a metric learning method for distance-based classification. Learning is achieved by maximizing the log-likelihood of correct class prediction, which is calculated in a low-dimensional feature space of increased discrimination power by using an optimized class representation. Extensions that exploit multiple representations per class, as well as that operate in arbitrary-dimensional Hilbert spaces for non-linear data projection and classification have also been described.

ACKNOWLEDGMENT

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement number 316564 (IMPART).

REFERENCES

[1] H. Wang, S. Yan, D. Xu, X. Tang, and T. Huang, "Trace ratio vs. ratio trace for dimensionality reduction," *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[2] Y. Jia, F. Nie, and C. Zhang, "Trace ratio problem revisited," *IEEE Transactions on Neural Networks*, vol. 20, no. 4, pp. 729–735, 2009.

[3] T. Mensink, J. Verbeek, F. Perronnin, and G. Csurka, "Distance-based image classification: Generalizing to new classes at near-zero cost," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 11, pp. 2624–2637, 2013.

[4] ——, "Metric learning for large scale image classication: Generalizing to new classes at near-zero cost," *European Conference on Computer Vision*, 2012.

[5] D. Tran and A. Sorokin, "Human activity recognition with metric learning," *European Conference on Computer Vision*, 2008.

[6] C. Veenman and D. Tax, "Less: a model-based classifier for sparse subspaces," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 9, pp. 1496–1500, 2005.

[7] K. Weinberger and O. Chapelle, “Large margin taxonomy embedding for document categorization,” *Neural Information Processing Systems*, 2009.

[8] X. Zhou, X. Zhang, Z. Yan, S. Chang, M. Jonhson, and T. Huang, “Sift-bag kernel for video event analysis,” *ACM Multimedia*, 2008.

[9] Z. Wang, Y. Hu, and L. Chia, “Image-to-class distance metric learning for image classification,” *European Conference on Computer Vision*, 2010.

[10] H. Wang, M. Ullah, A. Klaser, I. Laptev, and C. Schmid, “Evaluation of local spatio-temporal features for action recognition,” *British Machine Vision Conference*, 2009.

[11] H. Wang, A. Klaser, C. Schmid, and C. Liu, “Action recognition by dense trajectories,” *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[12] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld, “Learning realistic human actions from movies,” *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[13] B. Scholkpf and A. Smola, “Learning with kernels,” 2001, mIT Press.

[14] J. Taylor and N. Cristianini, “Kernel methods for pattern analysis,” 2004, cambridge University Press.

[15] H. Wang, A. Klaser, C. Schmid, and C. Liu, “Dense trajectories and motion boundary descriptors for action recognition,” *International Journal of Computer Vision*, vol. 103, no. 60-79.

[16] O. Gross, T. Hassner, and L. Wolf, “The action similarity labeling challenge,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 3, pp. 615–621, 2013.

[17] I. Laptev, “On space-time interest points,” *International Journal of Computer Vision*, vol. 64, no. 2, pp. 107–123, 2005.

[18] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma, “Robust face recognition via sparse representation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, pp. 210–227, 2009.

[19] M. Marszalek, I. Laptev, and C. Schmid, “Actions in context,” *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[20] J. Niebles, C. Chend, and L. Fei-Fei, “Modeling temporal structure of decomposable mition segemnts for activity classification,” *European Conference on Computer Vision*, 2010.

[21] A. Iosifidis, A. Tefas, and I. Pitas, “On the optimal class representation in linear discriminant analysis,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 24, no. 9, pp. 1491–1497, 2013.

[22] G. Baudat and F. Anouar, “Generalized discriminant analysis using a kernel approach,” *Neural Computation*, vol. 12, pp. 2385–2404, 2000.