AKSDA-MSVM: A GPU-accelerated Multiclass Learning Framework for Multimedia

Stavros Arestis-Chartampilas
CERTH-ITI
Thermi 57001, Greece
stav_ares@iti.gr

Nikolaos Gkalelis
CERTH-ITI
Thermi 57001, Greece
gkalelis@iti.gr

Vasileios Mezaris
CERTH-ITI
Thermi 57001, Greece
bmezaris@iti.gr

ABSTRACT

In this paper, a combined nonlinear dimensionality reduction and multiclass classification framework is proposed. Specifically, a novel discriminant analysis (DA) technique, called accelerated kernel subclass discriminant analysis (AKSDA), derives a discriminant subspace, and a linear multiclass support vector machine (MSVM) computes a set of separating hyperplanes in the derived subspace. Moreover, within this framework an approach for accelerating the computation of multiple Gram matrices and an associated late fusion scheme are presented. Experimental evaluation in five multimedia datasets, on tasks such as video event detection and news document classification, shows that the proposed framework achieves excellent results in terms of both training time and generalization performance.

Keywords

Discriminant analysis; GPU; multiclass classification; SVM

1. INTRODUCTION

In high-dimensional machine learning problems such as multimedia classification, the curse-of-dimensionality becomes a major challenge: the complexity of the pattern recognition model grows exponentially with the dimensionality of the space [8, 25]. A key observation for solving the curse-of-dimensionality problem is that most physical processes observed in some high-dimensional ambient space adhere to another, low-dimensional manifold. Hence, the curse-of-dimensionality can be alleviated by first identifying and then operating in this "intrinsic" low-dimensional space. Based on this fact, a major research direction is the investigation of frameworks that combine manifold learning with classification approaches [3, 29, 12, 15, 22, 19].

Kernel discriminant analysis (KDA) is a powerful class of manifold learning techniques (e.g. KFD [20], GDA [2]) that combined with linear (or piece-wise linear) classifiers have provided very good results in very challenging tasks. Moreover, further performance improvements have been achieved by subclass extensions of conventional KDA approaches, which by imposing a less strict requirement for the feature mapping are able to identify a more discriminant subspace (e.g. KSDA [28], KMSDA [13]). In the resulting subspace, it is then possible to use lower-capacity classifiers, such as linear SVMs (LSVMs), improving the generalization performance of the overall system [25].

Despite the above advances, the application of subclass-based KDA approaches to large-scale problems remains computationally challenging. The core of this limitation lies in the computation of the Gram matrix and the solution of the generalized eigenvalue problem. Recently, accelerated generalised subclass discriminant analysis (AGSDA) and its GPU implementation have been proposed, alleviating the above drawbacks and achieving state-of-the-art results in a variety of problems [1]. For example, its combination with LSVM (AGSDA-LSVM) has resulted in increased precision and orders of magnitude faster training times over LSVM and KSVM [1] in the problems of concept and event detection in video. Furthermore, earlier approaches such as SRKDA [4] and GSDA [10] (upon which AGSDA was based) were previously already shown to provide improved performance over many other KDA approaches.

The AGSDA-LSVM described above resolves many problems of subclass-based KDA approaches, but still suffers from the following limitations: i) it has been developed only for two-class problems and zero-mean datasets, ii) the GPU acceleration of AGSDA-LSVM has been proposed only in conjunction with the Gaussian RBF kernel. To this end, in this paper we propose AKSDA-MSVM that extends AGSDA-LSVM for multiclass classification, and datasets that are not necessarily normalized to have zero mean. Moreover, by carefully reformulating the Gram matrices of three popular kernels, we show how they can be simultaneously computed, exploiting both GPU and multi-core CPU acceleration. The experimental evaluation of AKSDA-MSVM on several large-scale datasets for both multimedia classification and retrieval tasks shows that the proposed approach achieves excellent performance in terms of both training time and generalization accuracy. Summarizing, the main contributions of this paper are:

- To the best of our knowledge, this is the first work that combines nonlinear discriminant analysis with multiclass support vector machines for multimedia classification.
- A method for the parallel computation of multiple Gram matrices is proposed, exploiting GPU and multicore CPU acceleration.

MM '16, October 15-19, 2016, Amsterdam, Netherlands
© 2016 ACM. ISBN 978-1-4503-3603-1/16/10...$15.00
DOI: http://dx.doi.org/10.1145/2964284.2967263

- The proposed approach achieves very good performance in terms of training time and generalization performance. Moreover, its software implementation is made freely available to the scientific community[1].

The rest of the paper is structured as follows: In Section 2, the proposed AKSDA-MSVM approach is described. Experimental results are presented in Section 3, while Section 4 concludes the paper.

2. AKSDA-MSVM

2.1 AKSDA

Let $\mathcal{X} = \{(\mathbf{x}_n, (\omega_n, \upsilon_n)), n = 1, \dots, N\}$ be a subclass partition of an annotated training set, where $\mathbf{x}_n \in \mathbb{R}^L$ is the n-th training observation in the L-dimensional input space, $\omega_n \in \{1, \dots, \Omega\}$, $\upsilon_n \in \{1, \dots, \Upsilon_{\omega_n}\}$ are its class and subclass labels, Υ_{ω_n} is the number of subclasses of class ω_n, and Ω, N are the total number of classes and observations respectively. The partitioning of classes to subclasses can be performed fully automatically (in our experiments this is done using k-means). Moreover, the training set is ordered in ascending order according to the class and subclass labels (i.e., $\omega_n \leq \omega_{n+1}$, and within a given class it is $\upsilon_n \leq \upsilon_{n+1}$). Given $\mathcal{X}$, AKSDA solves the generalized eigenvalue problem:

$$\mathbf{K}_c\mathbf{A}\mathbf{K}_c\mathbf{\Psi}_c = \mathbf{K}_c\mathbf{B}\mathbf{K}_c\mathbf{\Psi}_c\mathbf{\Lambda}_c, \tag{1}$$

where, $\mathbf{K}_c$, $[\mathbf{K}_c]_{r,q} = k_c(\mathbf{x}_r, \mathbf{x}_q)$ is the Gram matrix of the training set associated with a Mercer kernel function $k_c(\cdot,\cdot) : \mathbb{R}^L \times \mathbb{R}^L \to \mathbb{R}$, $\mathbf{A}$ is the between-subclass factor matrix, whose element $[\mathbf{A}]_{r,q}$ corresponding to samples $\mathbf{x}_r$, $\mathbf{x}_q$ is defined as

$$[\mathbf{A}]_{r,q} = \begin{cases} \frac{\hat{P}(\omega_r,\upsilon_r)(1-\hat{P}(\omega_r))}{N^2_{\omega_r,\upsilon_r}}, & \text{if } (\omega_r,\upsilon_r) = (\omega_q,\upsilon_q), \\ 0 & \text{if } \omega_q = \omega_r, \upsilon_q \neq \upsilon_r, \\ -\frac{\hat{P}(\omega_r,\upsilon_r)\hat{P}(\omega_q,\upsilon_q)}{N_{\omega_r,\upsilon_r}N_{\omega_q,\upsilon_q}}, & \text{otherwise,} \end{cases}$$

N_{ω_r}, $\hat{P}(\omega_r)$, N_{ω_r,υ_r}, $\hat{P}(\omega_r,\upsilon_r)$ are the number of observations and estimated prior probabilities of class ω_r and subclass (ω_r,υ_r) respectively, $\mathbf{B} = \mathbf{I} - \frac{1}{N}\mathbf{J}$ is the total factor matrix, $\mathbf{I}$ and $\mathbf{J}$ are $N \times N$ identity and all-ones matrices, $\mathbf{\Psi}_c \in \mathbb{R}^{N\times D}$ is the column-orthogonal eigenvector matrix, $\mathbf{\Lambda}_c \in \mathbb{R}^{D\times D}$ is the diagonal matrix with sorted eigenvalues in its diagonal, $D = \Upsilon - 1$ is the dimensionality of the projection subspace[2], and Υ is the total number of subclasses. Following [11], the above problem can be solved in two steps: a) identifying the eigenpairs $(\mathbf{V}, \mathbf{\Delta})$, $\mathbf{V} \in \mathbb{R}^{N\times D}$, $\mathbf{\Delta} \in \mathbb{R}^{D\times D}$ of $\mathbf{A}$, b) obtaining $\mathbf{\Psi}_c$ by solving the following linear matrix system

$$\mathbf{K}_c\mathbf{\Psi}_c = \mathbf{V}. \tag{2}$$

The eigenvector matrix $\mathbf{V}$ in the first step of AKSDA can be efficiently computed following [11]. The projection $\mathbf{z}$ of an observation $\mathbf{x}$ in the discriminant subspace can then be computed as $\mathbf{z} = \mathbf{\Psi}_c^T[k_c(\mathbf{x}_1,\mathbf{x}), \dots, k_c(\mathbf{x}_N,\mathbf{x})]^T$.

[1]Software: http://mklab.iti.gr/project/aksda

[2]$\mathbf{K}_c$ may be semidefinite positive matrix and in this case $D \leq \Upsilon - 1$. However, we assume that $\mathbf{K}_c$ is positive definite, which can be easily accomplished through regularization [11].

2.2 GPU-accelerated computation of multiple Gram matrices

One of the most computationally expensive parts of AKSDA (and of most kernel-based approaches) in terms of both memory consumption and learning time is the calculation of the Gram matrix $\mathbf{K}_c$. In [1], a tiled general matrix multiplication (GEMM) approach was proposed for the efficient computation of the Gram matrix with Gaussian RBF kernel. Inspired from [1], we propose a method for accelerating the computation of multiple Gram matrices. For illustration purposes, we examine the application of the proposed method in accelerating the computation of the Gram matrices associated with the Gaussian RBF, t-student and Cauchy kernels

$$\begin{aligned} k_1(\mathbf{x}_r,\mathbf{x}_q) &= \exp(-\gamma\|\mathbf{x}_r - \mathbf{x}_q\|^2), \\ k_2(\mathbf{x}_r,\mathbf{x}_q) &= (1 + \|\mathbf{x}_r - \mathbf{x}_q\|^d)^{-1}, \\ k_3(\mathbf{x}_r,\mathbf{x}_q) &= (1 + \|\mathbf{x}_r - \mathbf{x}_q\|^2/\sigma)^{-1}, \end{aligned} \tag{3}$$

where exp() is the exponential function, and γ, d, σ are the respective kernel parameters. Rearranging the above we get

$$k_1(\mathbf{x}_r,\mathbf{x}_q) = \exp(-\gamma u_{rq}), \qquad k_2(\mathbf{x}_r,\mathbf{x}_q) = \mathrm{sgm}_{d/2}(u_{rq}),$$
$$k_3(\mathbf{x}_r,\mathbf{x}_q) = \mathrm{sgm}_1(u_{rq}/\sigma),$$

where $\mathrm{sgm}_\iota(\upsilon) = (1+\upsilon^\iota)^{-1}$ is a sigmoid scalar function, and $u_{rq} = \|\mathbf{x}_r - \mathbf{x}_q\|^2$. Considering exp(), $\mathrm{sgm}_\iota()$ as element-wise matrix operators, the respective Gram matrices can be expressed as

$$\mathbf{K}_1 = \exp(-\gamma\mathbf{D}),\ \mathbf{K}_2 = \mathrm{sgm}_{d/2}(\mathbf{D}),\ \mathbf{K}_3 = \mathrm{sgm}_1(\mathbf{D}/\sigma). \tag{4}$$

The most computationally expensive part above is the computation of matrix $\mathbf{D}$, defined as $\mathbf{D} = \mathbf{X}^T\mathbf{X} + \mathbf{E}$, where $\mathbf{X}$ is the training data matrix, $\mathbf{E} = \mathbf{F} + \mathbf{C}$, $\mathbf{C} = \mathbf{F}^T$, and the elements of $\mathbf{F}$ are defined as $[\mathbf{F}]_{rq} = \mathbf{x}_r^T\mathbf{x}_r$, $\forall\, r, q$. As shown in [1], $\mathbf{D}$ can be easily partitioned to an arbitrary number of tiles, which can be computed in parallel by exploiting the GEMM function of CUDA's cuBLAS library. Furthermore, the element-wise matrix operations in (4) can also be parallelized by exploiting a multi-core CPU. Note that by appropriately formulating other kernel functions (e.g. the Inverse Multi-quadric kernel[3]) an arbitrary number of different Gram matrices can be parallelized.

2.3 Classification

Let $\mathcal{Z} = \{\mathcal{Z}_c, c = 1, \dots, C\}$ be a training dataset of C subsets, where each subset $\mathcal{Z}_c = \{(\mathbf{z}_{n,c}, \omega_n), n = 1, \dots, N\}$, $\mathbf{z}_{n,c} = \mathbf{\Psi}_c^T[k_c(\mathbf{x}_1,\mathbf{x}_n), \dots, k_c(\mathbf{x}_N,\mathbf{x}_n)]^T \in \mathbb{R}^D$ is derived from $\mathcal{X}$ using AKSDA and a specified kernel function $k_c(\cdot,\cdot)$. Given the above dataset, we define a linear model for each class and kernel function

$$f_{i,c}(\mathbf{z}_{n,c}) = \mathbf{w}_{i,c}^T\mathbf{z}_{n,c}, \tag{5}$$

where $\mathbf{w}_{i,c} \in \mathbb{R}^D$ is the weight vector referring to i-th class and c-th kernel function[4]. For the identification of the weight vectors we utilize the linear MSVM approach [16, 6]. That is, the following optimization problem is solved

[3]More details in http://crsouza.blogspot.com/2010/03/kernel-functions-for-machine-learning.html

[4]Note that a bias term is not included in (5). However, this can be easily solved by using augmented weight vectors or appropriate kernel functions as explained in [6].

using the respective training subset $\mathcal{Z}_c$

$$\min_{\mathbf{w}_{i,c},\boldsymbol{\xi}_{n,c}} \frac{1}{2}\sum_{i=1}^{\Omega} \mathbf{w}_{i,c}^T\mathbf{w}_{i,c} + R\sum_{n=1}^{N} \xi_{n,c},$$

subject to the constraints

$$(\mathbf{w}_{\omega_n,c} - \mathbf{w}_{i,c})^T\mathbf{z}_{n,c} \geq e_{i,n} - \xi_{n,c}, \ \forall i, n$$

where, $\xi_{n,c} \geq 0$ is the slack variable corresponding to $\mathbf{z}_{n,c}$, $R > 0$ is the penalty term, $e_{i,n} = 1-\delta_{i,n}$, and $\delta_{i,n}$ is the class indicator function, i.e., $\delta_{i,n} = 1$ if $\omega_n = i$, $\delta_{i,n} = 0$ otherwise. Given the learned models in (5) a test observation $\mathbf{x}_t$ can be then classified according to the following rule

$$\omega_t = \arg\max_{i=1,\dots,\Omega}(f_{i,c}(\mathbf{z}_{t,c})),$$

where $\mathbf{z}_{t,c} = \boldsymbol{\Psi}_c^T[k_c(\mathbf{x}_1,\mathbf{x}_t),\dots,k_c(\mathbf{x}_N,\mathbf{x}_t)]^T$. However, the above rule does not combine the different kernel modalities. To this end, following [18] the parameters α, β of the sigmoid function below are estimated in order to be able to map class similarities to posterior class probabilities

$$\hat{P}_c(i|\mathbf{x}_n) = \frac{1}{1+\exp(\alpha f_{i,c}(\mathbf{z}_{n,c}) + \beta)}.$$

Then, assuming equiprobable priors for each class, the sum rule [17] can be applied to yield the overall posterior probability for the i-th class

$$\dot{P}(i|\mathbf{x}_t) = \sum_{c=1}^{C} \hat{P}_c(i|\mathbf{x}_t),$$

and the following rule is used for classifying test observations

$$\omega_t = \arg\max_{i=1,\dots,\Omega}(\dot{P}(i|\mathbf{x}_t)). \quad (6)$$

3. EXPERIMENTAL RESULTS

3.1 Compared methods

We experimentally compare the following methods:

i) MSVM-1: Multiclass LSVM implementation of libsvm [5]. We use the Matlab version (compiled again in our evaluation workstation, so that all cores of the machine are exploited[5]). Matlab makes use of Intel's state-of-the-art numerical libraries, which speed up this method significantly.

ii) MSVM-2: Same as MSVM-1, with multiclass KSVM instead of multiclass LSVM.

iii) MSVM-3: Multiclass LSVM implementation of liblinear (out-of-the-box Matlab multi-core version) [7], which is among the fastest MSVM implementations.

iv) AKSDA-MSVM-1: The proposed method, using only the Gaussian RBF kernel. The liblinear MSVM [7] is employed for learning the separating hyperplanes.

v) AKSDA-MSVM-2: Same as AKSDA-MSVM-1, but with four kernels. Specifically, the Gaussian RBF, T-Student, Cauchy and Inverse Multi-quadric.

3.2 Datasets

One TRECVID, two UCI[6] (Amazon, YouTube) and two libsvm[7] (News20, Sector) datasets are used for the evaluation of the proposed method, described in the following:

i) MED12: This is an annotated subset of the TRECVID MED 2012 video corpus, where each video is represented in $\mathbb{R}^{101376}$ using improved dense trajectories [1, 14, 21, 26]. It consists of 25 target events and 13249 videos. The overall set is divided to 8824 training and 4425 evaluation videos.

ii) Amazon: It consists of 1500 reviews posted by 50 Amazon customers. Each review is represented by a 10000-dimensional vector. A random partition to 1200 training and 300 test observations is used.

iii) YouTube: It consists of 30 video game classes and up to 13 feature types per observation. We only use the audio MFCC features (2000-dimensional), and a subset of 20000 observations from the overall training set. For evaluation, the entire test set of 17177 instances is employed.

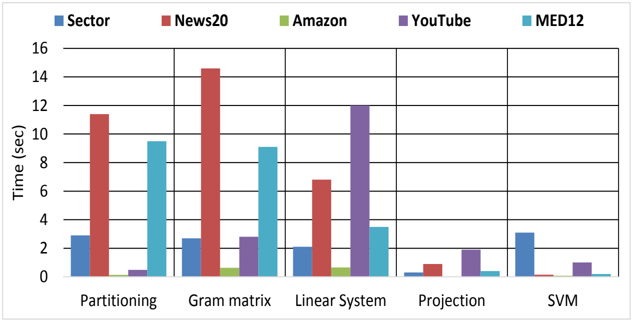
iv) News20: A collection of approximately 20000 news documents from 20 different newsgroups, used for text classification. A partition of 15935 training and 3993 test observations in $\mathbb{R}^{62061}$ is provided, already scaled. It is a very sparse dataset with approximately 0.13% non-zero elements.

v) Sector: A collection of 9619 corporate web pages organized into 105 categories based on each company's commercial activity. The scaled version of the dataset, partitioned to 6412 training and 3207 test observations in $\mathbb{R}^{55197}$, is used. It is very sparse, with only 0.3% non-zero elements.

3.3 Experimental setup

For all tested methods, model selection is done using 3-fold cross-validation, where at each fold the training set is split to 30% learning set and 70% validation set. For both AKSDA-MSVM (-1 & -2) and the KSVM of libsvm (MSVM-2), the kernel and SVM penalty parameters are searched in {0.5, 1, 1.5, ..., 10} and {0.1, 1, 10, 100}, respectively. For the penalty term of the LSVMs a broader set of candidate values is used according to relevant literature, i.e., {0.01, 0.03, 0.1, 0.3, 1, 3, 10} for the News20 and Sector, and {0.1, 1, 10, 100} for the YouTube and Amazon datasets. Moreover, class balancing is applied to each SVM classifier using a weight $(N - N_i)/N_i$, where N_i, N are the number of observations in i-th class and the total number of observations, respectively. The PC used for the evaluation is an Intel i7 3770K@3.5Ghz with 32 GB RAM, Windows 7, and a low-end Nvidia GPU (GTX 650). The experiments with AKSDA-MSVM were also repeated on a high-end GPU (GTX TITAN).

Figure 1: Training time along different parts of AKSDA-MSVM-1 on GTX TITAN



3.4 Results

The retrieval evaluation results in terms of mean average precision (MAP) and the respective times on GTX 650 are

[5]As advised in http://www.csie.ntu.edu.tw/~cjlin/libsvm/faq.html#f8032

[6]http://archive.ics.uci.edu/ml

[7]https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/

Table 1: Performance evaluation on YouTube

Method	MAP	Train/Test(minutes)
MSVM-1 [5]	38.94%	81.14/7.3
MSVM-2 [5]	49.14%	118.7/7.58
MSVM-3 [7]	28.14%	0.78/**0.018**
AKSDA-MSVM-1	49.28%	**0.57**/0.087
AKSDA-MSVM-2	**51.6%**	2.45/0.337

Table 2: Performance evaluation on MED12

Method	MAP	Train/Test(minutes)
AGSDA-LSVM [1]	41.20%	23.4/18.6
LSVM (from [1])	38.73%	75.5/2
AKSDA-MSVM-1	41.20%	**0.85/0.66**
AKSDA-MSVM-2	**41.87%**	1.01/0.80

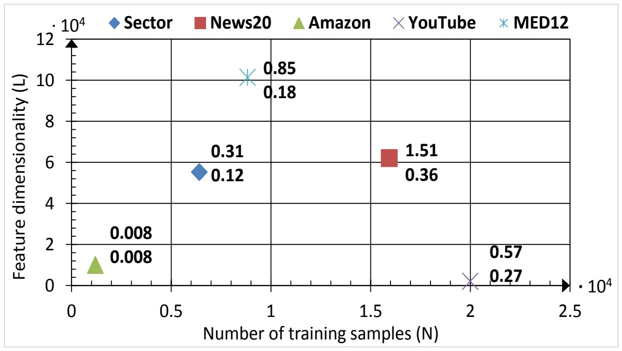
Table 3: Performance evaluation on three datasets (CCR %, Train/Test time in minutes)

Method	Amazon	News20	Sector
MSVM-1 [5]	66%	84.52%	91.89%
	0.65/0.020	3.1/0.16	2.12/0.18
MSVM-2 [5]	66.33%	84.62%	91.64%
	0.657/0.020	4.8/0.18	2.14/0.19
MSVM-3 [7]	72.3%	84.59%	93.73%
	0.08/**0.001**	**0.71/0.04**	4.75/**0.03**
AKSDA-MSVM-1	77.33%	86.3%	94.17%
	0.008/0.002	1.51/0.49	**0.31**/0.15
AKSDA-MSVM-2	**79.3%**	**86.6%**	**94.44%**
	0.023/0.003	2.48/0.58	0.65/0.16

Table 4: Performance evaluation on GTX TITAN (Train/Test time in minutes)

Dataset	AKSDA-MSVM-1	AKSDA-MSVM-2
Amazon	0.008/ 0.001	0.011/0.0016
News20	0.36/0.073	0.79/0.1
Sector	0.12/0.03	0.45/0.045
YouTube	0.27/0.05	0.93/0.2
MED12	0.18/0.09	0.3/0.11

Figure 2: Scalability analysis of AKSDA-MSVM-1 along different datasets and GPU cards. Training times on GTX 650 (top value) and GTX TITAN (bottom value) are reported, in minutes.



shown for the YouTube and MED12 video datasets, in Tables 1 and 2. It should be noted that due to higher RAM size requirements, MSVM-1 to -3 could not run for the MED12 dataset. The classification results in terms of correct classification rate (CCR) and the respective times (on GTX 650) for the Amazon, News20 and Sector datasets, are presented in Table 3. The training and test times of AKSDA-MSVM-1 & -2 on GTX TITAN, for all datasets, are shown in Table 4. Finally, the distribution of the training time of AKSDA-MSVM-1 along its individual components for each dataset, and the overall training time on GTX 650 and GTX TITAN, are shown in Figures 1 and 2, respectively. From the obtained results we observe the following:

i) AKSDA-MSVM-2, followed by AKSDA-MSVM-1, achieve the best generalization performance in all datasets.

ii) In terms of training time (even on GTX 650), AKSDA-MSVM-1 outperforms the state-of-the-art liblinear SVM implementation (MSVM-3) in three out of four datasets, while AKSDA-MSVM-2 (despite the computation of four Gram matrices) is still much faster than the MSVMs of libsvm (MSVM-1 & -2) in all datasets (Tables 1 & 3). Moreover, an impressive training time speedup (more than one order of magnitude) of the proposed method over the GPU accelerated AGSDA-LSVM [1] is observed in the MED12 dataset (Table 2). This is mainly due to the fact that AKSDA-MSVM solves one large eigenproblem (which is done efficiently using the proposed GPU-accelerated framework) to obtain a transformation matrix $\mathbf{\Psi}_c$ common for all (Ω) classes (see Sections 2.1, 2.2), while AGSDA-LSVM computes the GPU-accelerated solution of Ω eigenproblems to obtain one transformation matrix for each class. Finally, as expected, further improvement in training/test times is observed with the high-end GTX TITAN GPU (Table 4).

iii) From the analysis in Figure 1 we observe that the subclass partitioning, Gram matrix computation and linear system solver are the most intensive parts of the proposed method. Moreover, we can see that the linear system solver and the Gram matrix computation are mostly affected by the number of training observations and the feature vector dimensionality, respectively.

iv) In Figure 2, it is shown that in terms of training time the proposed method scales very well both with the number of training observations N and with feature dimensionality L. Specifically, considering the training times required for Sector and News20 when using GTX TITAN (thus overlooking any hardware limitations of the GTX 650), we see that for a similar feature vector size, doubling the number of training samples results in a $\times 3$ increase in training time, instead of the expected $\times 8$ increase of conventional KDA (due to its $O(N^3)$ complexity). Additionally, comparing the results for Sector and MED12, we see that despite a +100% increase in feature vector size and +50% increase in number of samples, the training time increases only by +50%.

4. CONCLUSION

A GPU-accelerated multiclass learning framework was presented providing very good performance in several multimedia-centered machine learning tasks. Future work directions include the extension of the proposed framework for multiple kernel learning [27, 9, 23] and the combination of AKSDA and MSVM in a single optimization problem [24].

5. ACKNOWLEDGEMENTS

This work was supported by the EU's FP7 and Horizon 2020 research and innovation programmes under grant agreements FP7-600826 ForgetIT and H2020-687786 InVID.

6. REFERENCES

[1] S. Arestis-Chartampilas, N. Gkalelis, and V. Mezaris. GPU accelerated generalised subclass discriminant analysis for event and concept detection in video. In *Proc. ACM Multimedia*, pages 1219–1222, Brisbane, Australia, 2015.

[2] G. Baudat and F. Anouar. Generalized discriminant analysis using a kernel approach. *Neural Comput.*, 12(10):2385–2404, Oct. 2000.

[3] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman. Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 19(7):711–720, Jul. 1997.

[4] D. Cai, X. He, and J. Han. Speed up kernel discriminant analysis. *VLDB J.*, 20(1):21–33, Mar. 2011.

[5] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Trans. on Intelligent Systems and Technology*, 2:27:1–27:27, 2011.

[6] K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *J. Mach. Learn. Res.*, 2:265–292, Mar. 2002.

[7] R.-E. Fan, K.-W. Chang, C.-J. Hsieh et al. LIBLINEAR: A library for large linear classification. *J. Mach. Learn. Res.*, 9:1871–1874, 2008.

[8] K. Fukunaga. *Introduction to statistical pattern recognition (2nd ed.)*. Academic Press Professional, Inc., San Diego, CA, USA, 1990.

[9] M. G. Genton. Classes of kernels for machine learning: A statistics perspective. *J. Mach. Learn. Res.*, 2:299–312, Mar. 2002.

[10] N. Gkalelis and V. Mezaris. Video event detection using generalized subclass discriminant analysis and linear support vector machines. In *Proc. ACM ICMR*, pages 25:25–25:32, Glasgow, UK, 2014.

[11] N. Gkalelis and V. Mezaris. Accelerated nonlinear discriminant analysis. *arXiv preprint arXiv:1504.07000*, 2015.

[12] N. Gkalelis, V. Mezaris, and I. Kompatsiaris. Mixture subclass discriminant analysis. *IEEE Signal Process. Lett.*, 15(5):319–332, May 2011.

[13] N. Gkalelis, V. Mezaris, I. Kompatsiaris, and T. Stathaki. Mixture subclass discriminant analysis link to restricted gaussian model and other generalizations. *IEEE Trans. Neural Netw. Learn. Syst.*, 24(1):8–21, Jan 2013.

[14] A. Habibian et al. Recommendations for video event recognition using concept vocabularies. In *Proc. ACM ICMR*, pages 89–96, Dallas, Texas, USA, 2013.

[15] T. Hofmann, B. Schölkopf, and A. J. Smola. Kernel methods in machine learning. *Ann. Statist.*, 36(3):1171–1220, 2008.

[16] S. S. Keerthi, S. Sundararajan et al. A sequential dual method for large scale multi-class linear SVMs. In *Proc. 14th ACM SIGKDD*, pages 408–416, Las Vegas, USA, Aug. 2008.

[17] J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas. On combining classifiers. *IEEE Trans. Pattern Anal. Mach. Intell.*, 20(3):226–239, Mar. 1998.

[18] H.-T. Lin, C.-J. Lin, and R. C. Weng. A note on platt's probabilistic outputs for support vector machines. *Mach. Learn.*, 68(3):267–276, oct 2007.

[19] Y. Liu, Y. Liu, S. Zhong, and K. C. Chan. Semi-supervised manifold ordinal regression for image ranking. In *Proc. ACM Multimedia*, pages 1393–1396, 2011.

[20] S. Mika, G. Ratsch, J. Weston, B. Scholkopf, and K. R. Mullers. Fisher discriminant analysis with kernels. In *Neural Networks for Signal Processing IX, 1999. Proc. of the 1999 IEEE Signal Processing Society Workshop*, pages 41–48, 1999.

[21] P. Over, G. Awad et al. TRECVID 2012 - an introduction to the goals, tasks, data, evaluation mechanisms and metrics. In *Proc. TRECVID 2012 Workshop*, Gaithersburg, MD, USA, Nov. 2013.

[22] F. Radenović, H. Jégou, and O. Chum. Multiple measurements and joint dimensionality reduction for large scale image search with short vectors. In *Proc. ACM ICMR*, pages 587–590, Shanghai, China, 2015.

[23] S. Sonnenburg, G. Rätsch, C. Schäfer, and B. Schölkopf. Large scale multiple kernel learning. *J. Mach. Learn. Res.*, 7:1531–1565, Dec. 2006.

[24] A. Tefas, C. Kotropoulos, and I. Pitas. Using support vector machines to enhance the performance of elastic graph matching for frontal face authentication. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23(7):735–746, Jul. 2001.

[25] V. Vapnik. *Statistical learning theory*. New York: Willey, 1998.

[26] H. Wang and C. Schmid. Action recognition with improved trajectories. In *Proc. IEEE Int. Conf. on Computer Vision (ICCV)*, pages 3551–3558, Sydney, Australia, Dec. 2013.

[27] J. Ye, S. Ji, and J. Chen. Multi-class discriminant kernel learning via convex programming. *J. Mach. Learn. Res.*, 9:719–758, Jun. 2008.

[28] D. You, O. C. Hamsici, and A. M. Martinez. Kernel optimization in discriminant analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(3):631–638, Mar. 2008.

[29] M. Zhu and A. M. Martinez. Subclass discriminant analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(8):1274–1286, Aug. 2006.