Incremental Training of Multiclass Support Vector Machines

Symeon Nikitidis [1], Nikos Nikolaidis [2] and Ioannis Pitas [3]
Informatics and Telematics Institute, Centre for Research and Technology Hellas, Greece

Department of Informatics, Aristotle University of Thessaloniki, Greece
[1] *nikitidis@aiia.csd.auth.gr*, [2] *nikolaid@aiia.csd.auth.gr*, [3] *pitas@aiia.csd.auth.gr*

Abstract

We present a new method for the incremental training of multiclass Support Vector Machines that provides computational efficiency for training problems in the case where the training data collection is sequentially enriched and dynamic adaptation of the classifier is required. An auxiliary function that incorporates some desired characteristics in order to provide an upper bound of the objective function which summarizes the multiclass classification task has been designed and the global minimizer for the enriched dataset is found using a warm start algorithm, since faster convergence is expected when starting from the previous global minimum. Experimental evidence on two data collections verified that our method is faster than retraining the classifier from scratch, while the achieved classification accuracy is maintained at the same level.

1. Introduction

Support Vector Machines (SVMs) [9] have become popular in pattern recognition problems due to their excellent generalization performance. Usually, SVMs are trained using a batch approach which requires all training data to be available at once, so that training is performed in one batch. If more training data are available later on, the SVM classifier should be retrained from scratch.

In this paper, we investigate the scenario where we have obtained the optimal Lagrange multipliers that define the normal vector of each decision surface for n base training samples and we seek the new minimizer over an enriched dataset, where m new training pairs were added. In such a case, assuming that the classifier was initially well trained, resolving the optimization problem from scratch is computationally inefficient. An alternative approach is to use the initial solution and the optimum Lagrange multipliers, obtained from the base training dataset, as an advanced starting point to warm-start the new optimization process. The computational advantage of such an approach is maximized, especially if we are adding a small amount of new training samples in a large dataset over which the SVM classifier has already been well trained.

The main motivation in applying a warm-start strategy is the expectation that two such closely related optimization problems should in general share similar characteristics. More precisely, the new decision surface is expected to have minimal disturbances with respect to its previous form.

Although various papers have been published on SVM training, relatively few have considered the problem of incremental training and even less have treated multiclass problems. An active set approach which involves a warm start algorithm to incrementally train SVMs was proposed in [8], while a method for incrementally updating the parameters of an SVM classifier has been applied in [10] for dynamic visual category learning.

2. Multiclass SVMs

Crammer and Singer in [2] proposed an approach for multiclass classification problems, by solving a single optimization schema. Given a set of n training data $\mathcal{X} = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_n, y_n)\}$ where $\mathbf{x}_i \in R^d, i = 1, ..., n$ are the input feature vectors, $y_i \in \{1, ..., k\}$ is the class label associated with sample $\mathbf{x}_i$ and d is the dimensionality of the input feature vectors, the idea is to consider all available training data at once and construct k two-class categorization rules where k is the number of classes. Solving this single optimization problem leads to the construction of k decision functions where the m-th decision surface $\mathbf{w}_m^T \phi(\mathbf{x})$, determined by its normal vector $\mathbf{w}_m \in R^d$, separates the training vectors of the m-th class from all the others, by minimizing the following primal problem:

$$\min_{\mathbf{w}_m, \xi_i} \frac{1}{2} \sum_{m=1}^{k} \mathbf{w}_m^T \mathbf{w}_m + C \sum_{i=1}^{n} \xi_i, \qquad (1)$$

subject to the constraints:

$$\mathbf{w}_{y_i}^T \phi(\mathbf{x}_i) - \mathbf{w}_m^T \phi(\mathbf{x}_i) \geq b_i^m - \xi_i, \quad i = 1, \ldots, n \quad (2)$$

Here, $\phi(.)$ is a function that maps the input feature vector $\mathbf{x}_i$ to an arbitrary-dimensional space $\mathcal{F}$ which usually has the structure of a Hilbert space [6], where the data are supposed to be linearly or near linearly separable, C is the term that penalizes the training errors, $\boldsymbol{\xi} = [\xi_1, ..., \xi_l]^T$ is the slack variable vector and $\mathbf{b}$ is a bias vector defined for $m = 1, ..., k$ as:

$$b_i^m = 1 - \delta_{y_i}^m = \begin{cases} 1, & \text{if } y_i \neq m \\ 0, & \text{if } y_i = m \end{cases} \quad (3)$$

where $\delta_{y_i}^m$ is the Kronecker delta function which is 1 for $y_i = m$ and 0 otherwise. The decision function is:

$$\arg \max_{m=1,\ldots,k} \left(\mathbf{w}_m^T \phi(\mathbf{x}) \right). \quad (4)$$

Switching to the dual formulation and seeking for the saddle point of the Lagrangian, while, at the same time, requiring that the minimum over the primal variables $\mathbf{w}$ and $\boldsymbol{\xi}$ should satisfy the Karush-Kuhn-Tucker (KKT) conditions, the multiclass classification task is summarized in the following single optimization problem:

$$\min_{\boldsymbol{\alpha}} \quad W(\boldsymbol{\alpha}) = \frac{1}{2} \boldsymbol{\alpha}^T \mathbf{H} \boldsymbol{\alpha} + \mathbf{b}^T \boldsymbol{\alpha} \quad (5)$$

under the following linear constraints:

$$\begin{array}{ll} \alpha_i^m & \leq 0 \quad \text{, if } y_i \neq m \\ \alpha_i^m & \leq C \quad \text{, if } y_i = m \end{array}$$
$$i = 1, \ldots, n \quad m = 1, \ldots, k, \quad (6)$$

where $\mathbf{b} = \left[b_1^1, \ldots, b_1^k, \ldots, b_n^1, \ldots, b_n^k \right]^T$ and $\boldsymbol{\alpha} = \left[\alpha_1^1, \ldots, \alpha_1^k, \ldots, \alpha_n^1, \ldots, \alpha_n^k \right]^T$ are the Lagrange multipliers associated with the constraints in (2). Furthermore, $\mathbf{H}$ is the Hessian matrix defined as $\mathbf{H} = \mathbf{K} \otimes \mathbf{I}$, $\mathbf{I}$ is a k by k identity matrix, $\otimes$ denotes the Kronecker product and $\mathbf{K}$ is the Kernel matrix, whose elements are equal to $K_{i,j} = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$. The Hessian matrix $\mathbf{H}$ is symmetric and positive semidefinite, since it has been derived by a direct product operation on the kernel matrix $\mathbf{K}$, which is also symmetric and positive definite. This property reveals a so called Quadratic Program (QP) since the objective function $W(\boldsymbol{\alpha})$ in (5) is a convex quadratic function with linear constraints and consequently, its minimization problem has a global minimum and no local minima. Finally, the decision surface is given by:

$$\arg \max_{m=1\ldots k} \quad \sum_{i=1}^{l} \alpha_i^m K(\mathbf{x}_i, \mathbf{x}). \quad (7)$$

3. Problem Formulation

In this section, we describe how we extend the previous SVM formulation in order to cope with modifications on the data collection used for training the SVM classifier. We also demonstrate the design of an auxiliary function that provides an upper bound of the objective function which summarizes the multiclass classification task as a single optimization problem.

3.1 Extending the multiclass SVM formulation

We investigate the incremental training task by examining the scenario where the SVM classifier has been trained over a base dataset $\mathcal{X}_n$ of n training pairs $(\mathbf{x}_i, y_i)$, $i = 1, \ldots, n$ and the optimum Lagrange multipliers $\boldsymbol{\alpha}_{n,o}$ that minimize the objective function in (5) have been evaluated. When m new training samples $\mathcal{X}_m = \{\mathbf{x}_s, y_s\}$, $s = n+1, \ldots, n+m$ are added in the original dataset, creating the so called augmented training dataset $\mathcal{X}_{n+m} = \mathcal{X}_n \bigcup \mathcal{X}_m$ we want to update the current SVM configuration and obtain a new SVM classifier that incorporates the new training data.

In order to facilitate a dynamic adaptation of the SVM classifier to the augmented training dataset $\mathcal{X}_{n+m}$ we express the new training task with respect to its initial form, along with an update term corresponding to the new training samples $\mathcal{X}_m$. However, since the initial and the augmented classification problems do not have the same number of constraints or variables, it is required to expand vectors $\boldsymbol{\alpha}_{n,o}$, $\mathbf{b}_n$ and the Hessian matrix $\mathbf{H}_n$ accordingly:

$$\boldsymbol{\alpha}_{n+m} = \begin{bmatrix} \boldsymbol{\alpha}_{n,o} \\ \boldsymbol{\alpha}_s \end{bmatrix}, \quad \mathbf{b}_{n+m} = \begin{bmatrix} \mathbf{b}_n \\ \mathbf{b}_s \end{bmatrix} \quad (8)$$

$$\mathbf{H}_{n+m} = \begin{bmatrix} \mathbf{H}_n & \mathbf{K}(\mathbf{x}_i, \mathbf{x}_s) \otimes \mathbf{I} \\ \mathbf{K}(\mathbf{x}_i, \mathbf{x}_s)^T \otimes \mathbf{I} & \mathbf{K}(\mathbf{x}_s, \mathbf{x}_s) \otimes \mathbf{I} \end{bmatrix}$$
$$i = 1, ..., n \quad , \quad s = n+1, ..., n+m \quad (9)$$

where $\boldsymbol{\alpha}_{n,o} = \left[\alpha_1^1, ..., \alpha_1^k, ..., \alpha_n^1, ..., \alpha_n^k \right]^T$, $\boldsymbol{\alpha}_s = \left[\alpha_{n+1}^1, ..., \alpha_{n+1}^k, ..., \alpha_{n+m}^1, ..., \alpha_{n+m}^k \right]^T$, $\mathbf{b}_n = \left[b_1^1, ..., b_1^k, ..., b_n^1, ..., b_n^k \right]^T$ and $\mathbf{b}_s = \left[b_{n+1}^1, ..., b_{n+1}^k, ..., b_{n+m}^1, ..., b_{n+m}^k \right]^T$. $\mathbf{H}_n$ is the $nk \times nk$ Hessian matrix computed over the initial n base training samples of $\mathcal{X}_n$, while $\mathbf{K}(\mathbf{x}_i, \mathbf{x}_s)$ and $\mathbf{K}(\mathbf{x}_s, \mathbf{x}_s)$ are the $n \times m$ and $m \times m$ kernel matrices used in order to expand the initial Hessian matrix $\mathbf{H}_n$ to $\mathbf{H}_{n+m}$, and are computed over both the initial and the new training pairs and over the samples of set $\mathcal{X}_m$, respectively.

The Hessian matrix $\mathbf{H}_{n+m}$ can be expressed as the sum of the Hessian $\mathbf{H}'_n$ computed over the base dataset

$\mathcal{X}_n$ and an update term matrix $\mathbf{H}_m$, evaluated using the m new training samples of $\mathcal{X}_m$. Both matrices are non-negative and are defined as:

$$\mathbf{H}'_n = \begin{bmatrix} \mathbf{H}_n & \mathbf{0}_{nk\times mk} \\ \mathbf{0}_{mk\times nk} & \mathbf{0}_{mk\times mk} \end{bmatrix},$$

$$\mathbf{H}_m = \begin{bmatrix} \mathbf{0}_{nk\times nk} & \mathbf{K}(\mathbf{x}_i,\mathbf{x}_s)\otimes\mathbf{I} \\ \mathbf{K}(\mathbf{x}_i,\mathbf{x}_s)^T\otimes\mathbf{I} & \mathbf{K}(\mathbf{x}_s,\mathbf{x}_s)\otimes\mathbf{I} \end{bmatrix},$$

$$i=1,...,n \quad s=n+1,...,n+m, \tag{10}$$

where $\mathbf{0}$ is an all-zeros array of appropriate dimensions. Subsequently, the objective function $W(\boldsymbol{\alpha}_{n+m})$ for the augmented minimization problem could be formulated as:

$$W(\boldsymbol{\alpha}_{n+m}) = \frac{1}{2}\boldsymbol{\alpha}_{n+m}^T\left(\mathbf{H}'_n+\mathbf{H}_m\right)\boldsymbol{\alpha}_{n+m}+\mathbf{b}_{n+m}^T\boldsymbol{\alpha}_{n+m}. \tag{11}$$

3.2 Auxiliary function

We define an auxiliary function in order to identify the global minimizer of the objective function $W(\boldsymbol{\alpha}_{n+m})$. The derivation of the auxiliary function we have followed is similar with the one presented in [7]. Similar techniques have been also used in order to establish the convergence of many statistical learning algorithms e.g., the Expectation-Maximization algorithm [3] for maximum likelihood estimation and nonnegative matrix factorization [4]. Since the minimization problem is a QP problem and has a global and no local minima, we seek to define an appropriate convex auxiliary function F, which will provide an upper bound of the objective function. Due to space limitations we will just sketch the auxiliary function and will not provide an analysis about how convergence to the global minimizer is achieved. This auxiliary function should satisfy the following properties:

1. It should bound the objective function from above:

$$W(\mathbf{u}) \le F(\mathbf{u},\boldsymbol{\alpha}_{n+m}). \tag{12}$$

2. The following equation should hold:

$$W(\boldsymbol{\alpha}_{n+m}) = F(\boldsymbol{\alpha}_{n+m},\boldsymbol{\alpha}_{n+m}). \tag{13}$$

Our goal is to use this auxiliary function F in order to derive the update rule $\boldsymbol{\alpha}' = \arg\min_{\mathbf{u}} F(\mathbf{u},\boldsymbol{\alpha})$, which will never increase the objective function, since the following inequality is valid:

$$W(\boldsymbol{\alpha}') \le F(\boldsymbol{\alpha}',\boldsymbol{\alpha}) \le F(\boldsymbol{\alpha},\boldsymbol{\alpha}) = W(\boldsymbol{\alpha}). \tag{14}$$

The minimizer $\boldsymbol{\alpha}'$ can be found by computing the derivative of the auxiliary function and setting it to zero. By iterating this update, a series of minimizers $\boldsymbol{\alpha}'$ are generated that improve the objective function and will lead to the global minimum, since the convexity property of $W(\boldsymbol{\alpha}_{n+m})$ implies that any reached local minimum is also global. After some involved derivations one can prove that the auxiliary function F is formed as the sum of second order functions f_i of u_i as: $F(\mathbf{u},\boldsymbol{\alpha}_{n+m}) = \sum_i f_i(u_i,[\boldsymbol{\alpha}_{n+m}]_i)$. Where:

$$\begin{aligned} f_i(u_i,[\boldsymbol{\alpha}_{n+m}]_i] \quad &= \tfrac{1}{2}\tfrac{[\mathbf{H}'_n\boldsymbol{\alpha}_{n+m}]_i}{[\boldsymbol{\alpha}_{n+m}]_i}u_i^2 + \tfrac{1}{2}\tfrac{[\mathbf{H}_m\boldsymbol{\alpha}_{n+m}]_i}{[\boldsymbol{\alpha}_{n+m}]_i}u_i^2 \\ &+[\mathbf{b}_{n+m}]_i u_i. \end{aligned} \tag{15}$$

We have selected to warm start the solution process by using the initial solution and the optimum Lagrange multipliers $\boldsymbol{\alpha}_{n,o}$ as a starting point and to initialize only the related to the new training samples Lagrange multipliers $\boldsymbol{\alpha}_s$. With this approach faster convergence is expected since in general, the new training samples of $\mathcal{X}_m$ modify the decision hyperplane in a relatively smooth manner. As a result, only a small portion of the SVM parameters should be evaluated and only few of the old Lagrange multipliers in $\boldsymbol{\alpha}_{n,o}$ would require an update.

4. Experimental Results

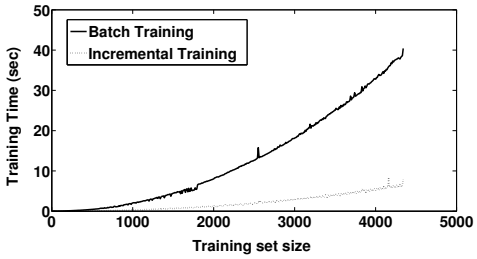
In our first experiment we provide evidence regarding the computational efficiency of the proposed incremental SVM training algorithm by comparing the computational cost of updating the SVM parameters through incremental training with the cost of retraining the classifier from scratch. For our second experiment, we have considered incremental learning applied to the frontal face view recognition problem, where the classifier is incrementally trained on a training set which is sequentially enriched by adding a single new example and testing is performed on the complete test set. Our aim is to investigate the behavior of the classifier, with respect to the classification accuracy, as the number of training samples increases and to derive useful insights regarding the effect of the new samples to the decision surface.

The experiment regarding the achieved computational efficiency has been conducted on the multiclass Satimage database, obtained from the UCI Repository of machine learning databases [1], while for the incremental learning scenario, on facial instances derived form the XM2VTS database [5]. Training and test data for both experiments were randomly ordered and normalized, such as to be in the $[-1,1]$ range with zero mean, before applied to the proposed method.

Figure 1 shows a comparison of the measured train-

ing time for the batch training model and the incremental training approach over the same training samples. The increment size is $m = 10$ and is kept constant. On average, the required computational time is 83.8% less for the incrementally training approach than the equivalent time required for the batch training model while, the achieved classification performance using an RBF kernel $K(\mathbf{x}_i, \mathbf{x}_j) = e^{\gamma||\mathbf{x}_i - \mathbf{x}_j||^2}$ with a fixed cost parameter value $C = 0$ and a Gaussian spread parameter $\gamma = 2^4$ was 91.1%.

Figure 1. SVM batch/incremental training times on Satimage dataset.

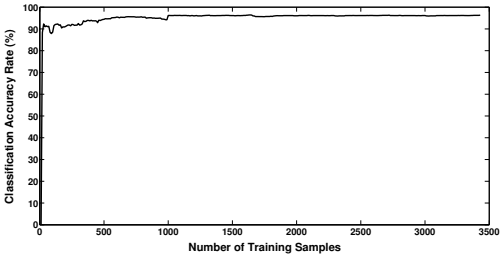


For the incremental learning scenario the task at hand is to distinguish the frontal views of a person face from the non frontal ones, while he is performing various head poses. In order to form the training and test sets, face detection and tracking were applied on the frames of the video sequences and the resulting Regions Of Interest (ROIs) were anisotropically scaled, so as to have fixed size of 30×40 pixels, converted to grayscale and was scanned row-wise so as to form a feature vector $\mathbf{x} = [f_1 \dots f_{1200}]^T$ (f_i being the luminance of the i-th pixel) which was used to compose the training and test sets that are fed to the SVM classifier. In total 6862 facial images were extracted and divided in two equally sized parts for training and testing. As it can be observed from Figure 2 the classifier can effectively classify the facial images even when only a few tens of samples from each class are used for training. Moreover, the decision surface essentially remains static, as we keep augmenting the training set beyond 1000 training samples, since the achieved classification accuracy rate remains constantly over 96%. The achieved classification accuracy percentage using the entire 3431 training samples is 96.33%.

5. Conclusions

We described an incremental training method which can efficiently update parameters of a multiclass SVM classifier. We demonstrated the performance improvement on a multiclass dataset and showed that our method is much faster than retraining the classifier from scratch while the achieved classification accuracy is

Figure 2. Frontal face view recognition rate versus the training set size.



maintained at the same level.

Acknowledgment

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 211471 (i3DPost).

References

[1] A. Asuncion and D. Newman. *UCI Machine Learning Repository*, 2007.

[2] K. Crammer and Y. Singer. On the learnability and design of output codes for multiclass problems. *Machine Learning*, 47(2-3):201–233, May 2002.

[3] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

[4] D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *Advances in Neural Information Processing Systems (NIPS)*, pages 556–562. MIT Press, 2001.

[5] K. Messer, J. Matas, J. Kittler, J. Luttin, and G. Maitre. XM2VTSDB: The extended M2VTS database. In *Second International Conference on Audio and Video-based Biometric Person Authentication (AVBPA)*, pages 72–77, 1999.

[6] B. Scholkopf, S. Mika, C. J. C. Burges, P. Knirsch, K. R. Muller, G. Ratsch, and A. J. Smola. Input space versus feature space in kernel-based methods. *IEEE Transactions on Neural Networks*, 10(5):1000–1017, September 1999.

[7] F. Sha, Y. Lin, L. K. Saul, and D. D. Lee. Multiplicative updates for nonnegative quadratic programming. *Neural Computation*, 19(8):2004–2031, 2007.

[8] A. Shilton, M. Palaniswami, S. Member, D. Ralph, and A. C. Tsoi. Incremental training of support vector machines. *IEEE Transactions on Neural Networks*, 16(1):114–131, 2005.

[9] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, New York, 1995.

[10] T. Yeh and T. Darrell. Dynamic visual category learning. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pages 1–8, 2008.