Classification using Multiplicative Update Rules in Non-negative Matrix Factorization

Gamit Dipali[1]
[1]Computer Science and Engineering, Gujarat Technological University

Abstract---Basically nonnegative matrix factorization (NMF) is performing on the original data for data transformation (dimensionality reduction) which is followed by the SVM classification. Using multiplicative update rules, there is a classification using SVM on two classes data which employ two sets of class one belongs to maximum margin classification constraints imposed into NMF and second is belongs to maximum margin classification constraints imposed into discriminant NMF. This paper aims to multiclass classification using two approaches multiclass SVM classifier and combining two class classifier. The previous framework two class which have also a very high accuracy of classification on various database. But this paper leads keeping the classification accuracy high with doing multiclass classification which employ maximum margin between the multiple classes of data.

Keywords: Nonnegative matrix factorization (NMF), multiclass SVM classifier, static reliability measure(SRM), dynamic reliability measure(DRM), one against all approach.

I. INTRODUCTION

Nonnegative matrix factorization is a group algorithm in multivariate analysis and linear algebra where matrix V is factorized into two matrices W and H with the property that all three matrices have no negative elements.

$$V = WH \quad (1.1)$$

Where V matrix is a data matrix whose jth column is the jth element vector of dimension N.W is a basis matrix whose ith element vector of dimension N. H is a coefficient matrix with dimension L

Nmf was first employ by Lee and Seung in [1] for learning facial image parts and semantic text features. They are also introduced two sets of multiplicative update rules for the estimation of V and W in[2]. These update rules are derived from the minimization of the cost function which represents the error of the factorization. The factorization error is computed by the Frobenius form

$$\left||X - ZH|\right|_{2F} \quad (1.2)$$

The cost function is nonconvex with respect to both variables Z and H. However, it is convex with respect to either Z or H. Therefore, the multiplicative update rules of [3] converge to a local minimum of the cost function. An extensive study on the convergence of the multiplicative update rules of NMF is presented in [4]. After NMF is performed on the original data, classification methods, such as support vector machines (SVMs) [5], can be applied on the projected data. SVMs find the hyperplane in the high-dimensional projection space, which has the maximum distance to the closest projected data of each class. This hyperplane is called a maximum-margin hyperplane. Consequently, SVMs are maximum-margin classifiers. As in the NMF case, SVMs optimize an objective function under certain constraints. . In the case of linear classification, the formulation of the SVM optimization problem depends only on dot products of the data. By applying the kernel trick, the data are projected on a transformed feature space and the dot products are substituted by a nonlinear kernel function

II. SUPPORT VECTOR MACHINES

A. Two-Class SVM

In this section, we give a very brief review of SVM and refer the the details to [6] and [7]. Consider N training samples: $\{x1; y1\}; \cdots; \{fxN; yN\}$, where $xi\ 2\ Rm$ is a m-dimensional feature vector representing the ith training sample, and $yi \in \{1; -1\}$ is the class label of xi. A hyperplane in the feature space can be described as the equation $wT\ x + b = 0$, where $w \in Rm$ and b is a scalar. When the training samples are linearly separable, SVM yields the optimal hyperplane that separates two classes with no training error, and maximizes the minimum distance from the training samples to the hyperplane. It is easy to find that the parameter pair $(w; b)$ corresponding to the optimal hyperplane is the solution to the following optimization problem:

$$\text{Minimize: } L(w) = ½||W||2 \quad (2.1)$$

$$\text{Subject to: } yi(wt\ xi + b) \geq 1, i = 1, \ldots N \quad (2.2)$$

For linearly nonseparable cases, there is no such a hyperplane that is able to classify every training sample correctly. However the optimization idea can be generalized by introducing the concept of *soft margin*. The new optimization problem thus becomes:

where »*i* are called slack variables that are related to the soft margin, and C is the tuning parameter used to balance the margin and the training error. Both optimization problems (2.1) and (2.2) can be solved by introducing the Lagrange multipliers ®*i* that transform them to quadratic programming problems. For the applications where linear SVM dose not produce satisfactory performance, nonlinear SVM is suggested. The basic idea is to map x by nonlinearl mapping $Á(x)$ to a much higher dimensional space in which the optimal hyperplane is found. The nonlinear mapping can be implicitly defined by introducing the so-called kernel function $K(xi; xj)$ which computes the inner product of vectors $Á(xi)$ and $Á(xj)$. The typical kernel functions include the radial basis function

$$K(x_i; x_j) = \exp(-||xi - xj||^2 \quad (2.3)$$

At the classification stage, the class label ySVM of a sample x is determined by the sign of the following decision function

$$\boldsymbol{F(x) = wt\ \varphi(x) + b} + \sum_{i=1}^{n} \alpha_i y_i K(x_i; x) + b: \quad (2.4)$$

III. RELIABILITY MEASURES

A. Static Reliability Measure

Using the training error *Remp* to estimate *R* is a straightforward method which has been adopted in many applications. However, as pointed out in [6] [7], when the number of the training samples is relative small with respect to the dimensionality of the feature vector *X*, a small Remp does not guarantee a small generalization error *R*. An upper bound of *R* is given in [6] [7] and one advantage of SVM is that minimizing the objective function will also minimize this upper bound [6] [7]. In other words, smaller objective function means smaller generalization error, or a more reliable classifier. Following this idea, we rewrite the objective function of SVM as

$$\text{OBJ} = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\left(1 - y_i f(x_i)\right) + \qquad (3.1)$$

Where $(u)+ = u$ if $u \geq 0$ and 0 if $u \cdot 1$, and propose a reliability measure as

$$\lambda\text{SRM} = \exp\left(-\frac{\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}(1 - y_i f(x_i))}{\sigma} +\right) \qquad (3.2)$$

Where $f(xi) = wT\, xi + b$. The parameter ¾ = CN is introduced as a normalization factor to offset the effect of the different regularization parameter C and training size N. For the linear separable case where $(1 - yif(xi))+ = 0$ for all training samples, the measure λSRM is reduced to

$$\lambda_{\text{SRM}} = \exp\left(-\frac{\|w\|^2}{2CN}\right) \qquad (3.3)$$

Recall that $\frac{2}{\|w\|^2}$ is the classification margin. The classifier with smaller $\|w\|$, which corresponds to larger margin, is considered to be more accurate in generalization and therefore its reliability measure λSRM is larger.

Note the test sample x does not appear in Eq. (3.3) and thus ¸SRM is the same for all samples. For this reason, it is named static reliability measure. The computational load of SRM is not high. When the number of support vectors is relatively smaller than the training size N, which happens most of the time, the complexity of SRM is O(N).

B. Dynamic Reliability Measure

SRM assumes the classifier to be equally effective throughout the entire feature space. In reality, the classifier's performance exhibits spatial variation [8], to accommodate which we extend SRM to DRM, a dynamic reliability measure. The basic idea is to estimate the classifier's reliability in a *local region* of feature space surrounding the test sample x. Here the local region, denoted as $Nf(x)$, is defined as the training samples that compose the k-nearest neighbours of x. Moreover, we are especially interested in the reliability of the classifier with respect to certain output (1 or -1 in this case).

Suppose $C(x)$ *2 f1;¡1g* is the class label assigned to x by the SVM classifier. Let $NC(x)$ k (x) denote the set of the training samples that locate among the k nearest neighbors of x and are classified to the same class .

we make the training sample xi contributes to the overall OBJ by OBJ(xi). In analogy to Eq. (3.2) which takes the summation of OBJ(xi) on all samples, we formulate the local version of OBJ as

$$\text{OBJlocal} = \sum OBJ(\hat{}xj) \qquad (3.4)$$

Where $\hat{}\, xj \in NC(x)\ k\ (x), (\hat{}\, xj\ ;\ \hat{}\, yj)$ is the training pair, and kx is the number of training samples in the set $NC(x)$ k (x). Now with OBJ local at hand, we can compute the reliability of the decision "x belongs to $C(x)$" by

$$\lambda_{\text{DRM}}(x) = \exp\left(-\left(\frac{\|w\|^2}{2CN} + \frac{\sum_{i=1}^{kx}\left(1 - y_i f(^{x_i}\right)}{kx} +\right)\right) \qquad (3.5)$$

From the derivations above we can see that, unlike λSRM, ¸λDRM(x) is a dynamic function of x, which varies depending on the location of the test sample and the classified label $C(x)$. For this reason, DRM has to be recomputed as new samples come in and is more expensive than SRM. The extra cost is consumed by finding the k nearest neighbors, which is O(N).

C. One Against All Approach

Based on the SRM and DRM discussed above, we propose a new decision rule for the one-against-all method.Suppose that we have trained M support vector machines SVM1, SVM1,., SVMM, each of which has the decision function fl. First, we evaluate fl at the given sample x using Eq. (3.1), and generate a soft decision yfl *2* [¡1 1] assuming the classifier is completely trustable

$$y_{\text{fi}} = \text{sign}(\text{fi}(x))(1 - \exp^{\text{jfi}(x)\text{j}}) \qquad (3.6)$$

Note that y_{fi} carries two kinds of information: the sign part encodes the hard decision on "x belongs to class l or not"and its absolute value represents how strong the decision is. The farther x locates away from the decision boundary which yields larger $fl(x)$, the stronger the decision. Then, instead of comparing yfl directly, we weight yfl by the liability measures as ¹yfl = yfl¸l, where ¸l denotes the SRM or DRM of SVM. Finally, the sample x is classified as in class l¤ that yields the largest ¹yfl

$$l¤ = = \arg\max_{l = 1 \ldots M^1:} y_{\text{fi}} \qquad (3.7)$$

It is can be shown that when ¸l are equal, Eq. (3.7) becomes

$$l¤ = \underset{l = 1;\,:\,:\,:;M}{arg\ max}\ y_{fl} = \underset{l = 1;\,:\,:\,:;M}{arg\ max}\ f_l(x); \qquad (3.8)$$

Which reduces to the decision rule of the conventional one against all method.

IV. REVIEW OF NMF AND SVM CLASSIFIER

A. NMF Algorithm

The objective of NMF is to find a pair of matrices Z ∈ _N×L , H ∈ _L × M, minimizing the cost function which measures the Frobenius norm of error between the initial data matrix X ∈ _N×M, xi j ≥ 0, i = 1, . . . , N, j = 1, . .. , M,and its approximation by a matrix product ZH

$$\arg\min \sum_{i=1}^{N}\sum_{j=1}^{M}(\text{xi j} - \sum_{i=1}^{L}\text{zil}\,\text{hl j}) \qquad (4.1)$$

zil , hl j

subject to the constraints

$$\text{zil} \geq 0, \text{hl j} \geq 0 \text{ and } \sum_{i=1}^{N} \text{zil} = 1\ \forall l = 1, \ldots, L. \qquad (4.2)$$

We notice that (4.1) is the element-wise formulation of (1.2).Lee and Seung in [6] solved the optimization problem (4.2),by using the expectation–maximization (EM)

algorithm [9], [10], leading to the following multiplicative update rules:

$$h(t+1) = h(t)\,\frac{\sum_{i=1}^{N} x_{ij}\, z(t)}{\sum_{i=1}^{N}\sum_{k=1}^{l} z_{il}\, z_{ik}\, h_{kj}} \qquad (4.3)$$

l j l j

where the upper scripts (t) and $(t+1)$ denote the tth and $(t+1)$th iterations. It is obvious from the aforementioned formulation that, in each iteration, the updates (4.3) are performed sequentially. Furthermore, another set of multiplicative update rules is presented in [6], for minimizing of the Kullback–Leibler divergence

$$\arg\min \sum_{i=1}^{N}\sum_{j=1}^{M}\left[x_{ij}\ln\left(\frac{x_{ij}}{\sum_{l=1}^{L} z_{il}h_{lj}}\right) + \sum_{l=1}^{L} z_{il}h_{lj} - x_{ij}\right] \qquad (4.4)$$

zil , hl j

subject to the constraints given in (16). The corresponding multiplicative update rules, which are also derived by the EM algorithm.

B. Support Vector Machines

Let us consider a set D = $\{\{\boldsymbol{x}_j, y_j\}, j = 1,\ldots,M, \boldsymbol{x}_j \in \_N$, y j $\in \{-1, 1\}\}$, of M training data, where **x**j denote the data points and y_j the corresponding labels. Our objective is to separate the positive data points from the negative ones, by finding the maximum-margin hyper plane, i.e., the hyper plane, whose distance from the nearest points of each class is maximal. Let us consider the vector **w** ∈ _N, which is normal to the hyper plane and the constant b, such as |b|/_**w**_is equal to the perpendicular distance between the hyper plane and the origin. The objective of SVM is the minimization of

$$\arg\min 1/2\,||w||$$

subject to the constraint

y j _wT x j − b_ − 1 ≥ 0 ∀ j = 1, . . . , M.

The mathematical analysis on the derivation of the objective function of SVM can be found in [8]. A point **x** $_j$ is called a support vector, if the equality in (4.5) holds. Intuitively, support vectors are the points whose removal from the training dataset would change the maximum-margin hyperplane. The solution to the problem of SVM is found by minimizing the Lagrangian function.

$$\arg\min\left\{\frac{1}{2}\sum_{j=1}^{M}\sum_{k=1}^{M} a_j\, a_k\, y_j\, y_k x_j\, x_k - \sum_{j=1}^{M} a_j\right\} \qquad (4.6)$$

We notice that (4.6) is a nonnegative quadratic programming problem, which depends only on the Lagrange multipliers a_j . After we find the Lagrange multipliers, the maximum-margin hyperplane **w** and the margin b are estimated by using the following equations:

$$W = \sum_{j=1}^{M} a_j\, y_j\, x_j \qquad (4.7)$$

$$b = \frac{1}{n\,(M_{SV})}\sum_{j\in M_{SV}}(w^T \mathbf{x}_j - y_j) \qquad (4.8)$$

Where MSV is the set of indices of the support vectors and n(MSV) is their number. Finally, the decision about the class of a testing sample **x** is computed by the decision function

$$\text{sign}\,(w^T x + b) \qquad (4.9)$$

V. RELATED WORKS

In this section, we present a new treatment for the decoding strategy when combining the outcomes of different base learners toward a multiclass solution. We propose a decoding approach based on the conditional probabilities of groups of correlated base learners. For that, we discuss the concept of learning outcome correlations among binary classifiers and present a principled way to find groups of correlated base learners. The rationale is that it is not always possible to consider the output of all base learners of an input as independent random variables (RV). Finally, we present one strategy to reduce the number of required base learners in the multiclass classification and another to find new base learners in order to replace less discriminative ones. We can use these two procedures iteratively to improve the overall multiclass classification performance.

A. Formalization

To classify an input, we use a set of trained base learners T . We call OT the set of outcomes or realization of the base learners in T . For simplicity, each element of T is a base learner that produces an outcome ∈ {−1,+1}. Therefore, a realization of T consists of the outcomes ∈ {−1,+1} for all the chosen binary classifiers over all the training examples. In addition, consider li as a class indicator (label) for a problem with Nc classes.

Given an input element x to classify, a realization OT contains the information available to determine the class of x and to find P(y = li|x) = P(y = li|OT). For instance, for one input example x and a set T with three base learners, a realization OT of T could be of the form OT (x) = $h + 1, -1, -1i$.

From Bayes theorem

$$P(y = li|OT) = \frac{P(O_T|y = l_i)P(y = l_i)}{P(O_T)} \qquad (19)$$

$$\propto P(OT|y = li)P(y = li) \qquad (5.1)$$

P(OT) is common to all classes and can be suppressed.

Previous approaches have solved the above model by considering the independence of the outcomes for the base learners in T [2]. If we consider independence among all the outcomes of base learners in T for a test example x. where Ot T refers to the outcome of a base learner t ∈ T . The class of the input x is given by

B. Relaxing The Independence Assumptions

Although the independence assumption simplifies the model, it comes with limitations and it is not the best choice in all cases [11]. In fact, it is possible that relaxing such constraint might lead to better classification performance.

We relax the assumption of independence among all binary classifiers. When the outcomes of two base learners overlap up to some point for the same training examples, it would be unwise to threat their results as independent random variables (RVs). In our approach, we find groups of correlated base learners and represent their outcomes for training examples as dependent RVs, using a single conditional probability table (CPT) as an underlying distribution model. Each group of correlated classifiers then has its own CPT, and we combine the groups as if they are

independent from each other — to avoid a dimensionality explosion.

We can view this technique as a Bayesian-Network-inspired approach for RV estimation. We decide the RV that represents the class of an input example based on the RVs that represent the outcomes of the base learners. We model the multiclass classification problem conditioned to groups of correlated base learners C. The model inEquation 1 then become

$$P(y = li|OT , C) \propto P(OT , C|y = li)P(y = li). \quad (5.2)$$

We assume independence only among the groups of highly correlated base learners ci ⊂ C. Therefore, the class of an input x is given by

$$class(x) = argmaxj \; Y \; ci \subset C \; P(OciT, ci|y = lj)P(y = lj), \quad (5.3)$$

Where Oci T refers to the outcomes of all base learners in the group of highly correlated base learners ci ⊂ C for training data points X. To find the groups of correlated base learners C, we define a correlation matrix A among the classifiers. The correlation matrix measures how correlated are two base learners when classifying a set of training examples X. In Section III-E, we show how to create such correlation matrix A. After calculating the correlation matrix A, we use a clustering algorithm to find the groups of correlated baselearners in A (Section III-F).

C. Strategies To Improve Efficiency And/or Effectiveness

The groups of correlated base learners can contain classifiers that do not contribute significantly to the overall classification specially if they are within a large interdependent group. We (1) identify and remove the less important base learners within a group, speeding 4 IEEE Transactions On Neural Networks And Learning Systems, Vol. X, No. Y, August 2013 up the overall classification process and making more robust CPTs estimations or (2) find new base learners that will improve the classification outcome.

In eliminating less important base learner, we show a consistent approach to eliminate the less important base learners within groups of correlated base learners. In addition, in replacing less important base learner, we discuss a simple idea to find new base learners that can be used, among other situations, to replace the ones considered as less discriminatives. We can use both procedures iteratively until a convergence criteria is satisfied. These two procedures are very fast as most of the information needed is already calculated during the earlier training steps.

VI. EXPERIMENTS

In this section we examine the framework Multiclass SVM classifier and analysis the results on various datasets like Cohn–Kanade database for facial expression recognition and on the AIIA/MOBISERV database for eating and drinking activity recognition.

We validate the proposed methods using two scenarios, small and large datasets. First, we consider datasets with a relative small number of classes (Nc < 30). For that, we use several UCI2, and one NIST3 datasets. Second, we consider two large scale multiclass applications:one for the Australian Sign Language (Auslan)4, and one for the Amsterdam Library of Objects (ALOI)5.

A. Scenario 1(10-26 classes)

From the experiments, we can see that the use of conditional probabilities and groups of correlated base learners to decode the binary classifications and create a multiclass prediction improves the performance of ECOC-based approaches. This is also true for other UCI datasets not shown here such as *abalone*, *covtype*, and *yeast*. For the LDA base learner, note that the normal multi-class LDA is not as effective as the proposed methods (e.g., Pendigits, MNist, Vowel, Letter-2.). For the SVM base learner and MNist, and Vowel datasets, B-SVM, SB-SVM, Crammer-Singer L1 and Crammer-Singer L2 are all worse than M-* methods as well as the OVA reference. Considering the Pendigits and Isolet datasets, the performances are comparable. However, SB-SVM, Crammer-Singer L1 and Crammer-Singer L2 are not easy to parallelize contrary to M-* and Passerini methods which can delegate the training and testing of their base learners to separate threads/cores easily.

For the UCI and NIST datasets, the M-* results are, in average, one standard deviation above Passerini's results when using SVM and, at least, two standard deviations above Passerini's when using LDA. We have found that Passerini's assumption of independence is not as robust as the proposed methods when the number of base learners and classes increase (c.f., Sec. IV-B). This is also true for the One-vs-One combinations. As the number of classes grows, we observe that these solutions become less discriminative.

B. Scenario 2 (95-1,000 classes)

Now we consider two large-scale applications: Auslan (Nc = 95), and ALOI (Nc = 1, 000) datasets.In such applications, OVO as well as B-SVM, SB-SVM, Crammer and Singer's L1, and Crammer and Singer's L2 are] computationally expensive, and ECOC-based approaches with a few base learners seem to be more appropriate. In Figures 5–6, we show results using the baseline method proposed (M-BAS) *vs.* ECOC Hamming decoding and Passerini et al. [12] approaches for LDA and SVM baselearners. For LDA base learners, we show its multi-class version for reference. For SVM, we show B-SVM, SB-SVM, Crammer and Singer's L1, and Crammer and Singer's L2 as references.

Here, we emphasize the performance for a small number of base learners in comparison with the number of all possible separation choices. As we increase the number of base learners, all approaches fare steadily better. The experiments show clearly that, in scenarios with more than 30 classes, the independence restriction [12] does not yield the best performance.

VII. CONCLUSION

In this paper, we addressed two key issues of multiclass classification: the choice of the coding matrix and the decoding strategy for using when reducing multiclass problems to binary problems. For that, we presented a new treatment for the decoding strategy.

The advantages of the solution we present in this paper are: (1) it works independent of the number of

selected base learners; (2) it can be associated with each one of the previous approaches in the literature such as OVO, OVA, ECOC, and their combinations; (3) it does not rely on the independence restriction among all base learners; (4) its implementation is simple and it uses only basic probability theory; (5) it is fast and does not impact the multiclass procedure; and (6) it is highly parallelizable being suitable to be implemented in current GPUs or multi-core processors7

Future work include the design of alternative ways to store the conditional probability tables other than sparse matrices and hashes and the implementation of the current methods using GPUs and multicore machines. As the methods of this paper start from the choice of dichotomies, we also plan to further investigate using these methods on OVO and OVA sets of classifiers.

REFERENCES

[1] D. D. Lee and H. S. Seung, "Learning the parts of objects by nonnegative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, Oct. 1999
[2] P. Smaragdis and J. Brown, "Non-negative matrix factorization for polyphonic music transcription," in *Proc. IEEE Workshop Appl. Signa Process. Audio Acoust.*,
[3] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *Advances in Neural Information Processing Systems 13.*
[4] C.-J. Lin, "On the convergence of multiplicative update algorithms for nonnegative matrix factorization," *IEEE Trans. Neural Netw.*, vol. 18, no. 6, pp. 1589–1596, Nov. 2007.
[5] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining Knowl. Disc.*, vol. 2, pp. 121–167, Jun. 1998.
[6] V.N. Vapnik, "An Overview of Statistical Learning Theory", *IEEE Trans. on Neural Networks*, vol. 10, no. 5, pp. 988-999, Sept. 1999.
[7] C. Cortes and V.N. Vapnik, "Support Vector Networks", *Machine Learning*, vol. 20, no. 3, pp. 273-297, 1995.
[8] K. Woods, W.P. Kegelmeyer, and K. Bowyer, "Combination of Multiple Classifier Using Local Accuracy Estimates", *IEEE Tran. on Pattern Analysis and Machine Intelligence*, vol. 19, no. 4, pp. 405-410, April 1997
[9] A.P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. Royal Stat. Soc. B*, vol. 39,
[10] L. Saul, F. Pereira, and O. Pereira, "Aggregate and mixed-order Markov models for statistical language processing," in *Proc. 2nd Conf. Empirical Methods Natural Lang. Process.*, 1997, pp. 81–89.
[11] A.Narasimhamrthy, "Theoretical bounds of majority voting performance for a binary classification problem," *IEEE Transactions on Pattern Analysis and Machine Intelligence (T.PAMI)*, vol. 27, no. 12, pp. 1988– 1995, Dec 2005.
[12] A.Passerini, M. Pontil, and P. Frasconi, "New results on error correcting output codes of kernel machines," *IEEE Trans. on Neural Networks and Learning Systems (T.NNLS)*, vol. 15, no. 1, pp. 45–54, 2004.

