MARGINAL PERCEPTRON FOR NON-LINEAR AND MULTI CLASS CLASSIFICATION

Hemant Panwar[1] and Surendra Gupta[2]

Computer Engineering Department
Shri Govindram Seksaria Institute of Technology and Science
23 Park Road, Indore, India
[1]hemant061018@gmail.com
[2]sgupta@sgsits.ac.in

ABSTRACT

Generalization error of classifier can be reduced by larger margin of separating hyperplane. The proposed classification algorithm implements margin in classical perceptron algorithm, to reduce generalized errors by maximizing margin of separating hyperplane. Algorithm uses the same updation rule with the perceptron, to converge in a finite number of updates to solutions, possessing any desirable fraction of the margin. This solution is again optimized to get maximum possible margin. The algorithm can process linear, non-linear and multi class problems. Experimental results place the proposed classifier equivalent to the support vector machine and even better in some cases. Some preliminary experimental results are briefly discussed.

KEYWORDS

Perceptron, Non-linear Classification, Multi Class Classification, Support Vector Machine & Generalization Error.

1. INTRODUCTION

A perceptron is a binary classifier, initially developed as a model of the biological neuron [1]. Internally, it computes a linear combination of real-valued labeled samples and predicts the classes for unlabeled samples. If two different set of samples can be separated by a straight hyperplane, they are called linearly separable. As a consequence, irrespective of the training algorithm used, linear classifiers like the perceptron cannot arrive at correct predictions for all potential instances unless the given problem is linearly separable. Classifiers generally face the problem of incorrect classification for those instances which are closer to separating hyperplane, known as generalization error. The number of generalization errors depends on the margin or distance between the positive and negative samples [2]. The hyperplane that maximizes the margin to the closest positive and negative sample is called the optimal hyperplane. The perceptron will not necessarily find the optimal hyperplane. However, the perceptron is an online learning classifier, means it process one sample at a time which allowed allows it to spare time and memory resources for handling large-scale classification problems.

Support Vector Machine (SVM) is a large margin classifier which is able to find an optimal hyperplane [3]. SVM produces large margin solutions by solving a constrained quadratic optimization problem using dual variables. Unlike perceptron, SVM is capable to process linearly non separable problems as well as multi class problems. But, quadratic dependence of its memory requirements in the number of training samples prohibits the processing of large-scale classification problems. Although SVMs have led to improved convergence rates, but in practice their super-linear dependence on the number of samples, lead to excessive runtimes, when large-scale datasets are processed.

The above considerations motivated research in a large margin classifier based on the perceptron, which can possess optimal hyperplane and can also process linearly non separable as well as multi class problems. Subsequently, various algorithms succeeded in attaining maximum margin approximately by employing modified perceptron like update rules. Such algorithms include Relaxed Online Maximum Margin Algorithm (ROMMA) [4], Approximate Large Margin Algorithm (ALMA) [5].

2. MOTIVATION OF THE ALGORITHM

Consider a linearly separable training set $\{(x_k, l_k)\}_{k=1}^{m}$, with vectors x_k as input samples vector and labels $l_k \in \{+1,-1\}$. An augmented space is constructed by placing x_k in the same position at a distance ρ in an additional dimension, i.e. extending x_k to $[x_k, \rho]$[6]. Following the augmentation, a reflection is performed with respect to the origin of the negatively labeled patterns by multiplying every pattern with its label. This allows a uniform treatment of both categories of patterns.

The relation characterizing optimally correct classification of the training patterns y_k by a weight vector u of unit norm in the augmented space is

$$u \cdot y_k \geq \gamma_d \equiv \max_{u':\|u'\|=1} \min_i \{u', y_i\} \forall k \tag{1}$$

where γ_d is the maximum directional margin. In proposed algorithm the augmented weight vector a_t is initially set to zero, i.e. $a_0 = 0$, and is updated according to the classical perceptron rule

$$a_{t+1} = a_t + y_k \tag{2}$$

each time an appropriate misclassification condition is satisfied by a training pattern y_k. Inner product of (2) with the optimal direction u and (1) gives

$$u \cdot a_{t+1} - u \cdot a_t = u \cdot y_k \geq \gamma_d$$

a repeated application of which gives [7]

$$\|a_t\| \geq u \cdot a_t \geq \gamma_d t$$

thus an upper bound can be obtain on γ_d provided $t > 0$

$$\gamma_d \leq \frac{\|a_t\|}{t} \tag{3}$$

Assume that satisfaction of the misclassification condition by a pattern y_k has as a consequence that $\|a_t\|^2 / t > a_t \cdot y_k$ (i.e., the normalized margin $u \cdot y_k$ of y_k (with $u_t \equiv a_t / \|a_t\|$) is smaller than the upper bound (3) on γ_d). Statistically, at least in the early stages of the algorithm, most updates do not lead to correctly classified patterns (i.e., patterns which violate the misclassification condition) and as a consequence $\|a_t\| / t$ will have the tendency to decrease. Obviously, the rate at which this will take place depends on the size of the difference $\|a_t\|^2 / t - a_t \cdot y_k$ which, in turn, depends on the misclassification condition.

For solutions possessing margin the most natural choice of misclassification condition is the fixed (normalized) margin condition

$$a_t \cdot y_k \leq (1-\varepsilon)\gamma_d \|a_t\| \tag{4}$$

with the accuracy parameter ε satisfying $0 < \varepsilon \le 1$. The perceptron algorithm with fixed margin condition converges in a finite number of updates to an ε-accurate approximation of the maximum directional margin hyperplane [8,9].

The above difficulty associated with the fixed margin condition may be remedied if the unknown γ_d is replaced for $t > 0$ with its varying upper bound $\|a_t\|/t$ [10]

$$a_t \cdot y_k \le (1-\varepsilon)\frac{\|a_t\|^2}{t} \tag{5}$$

Condition (5) ensures that $\|a_t\|^2/t - a_t \cdot y_k \ge \varepsilon\|a_t\|^2/t > 0$. Thus, it can be expected that $\|a_t\|/t$ will eventually approach γ_d close enough, thereby allowing for convergence of the algorithm to an ε-accurate approximation of the maximum directional margin hyperplane. It is also apparent that the decrease of $\|a_t\|/t$ will be faster for larger values of ε.

The proposed algorithm, now employs the misclassification condition (5) (with its threshold set to 0 for $t = 0$), which may be regarded as originating from (4) with γ_d replaced for $t > 0$ by its dynamic upper bound $\|a_t\|/t$.

The algorithm employing the misclassification condition above will attain maximum margin for linearly separable problems, but it can be further optimize, by moving optimum surface boundary [11]. Below are equations for two support vector (one from each class) with α and β as weight optimize factor for weights other than augmented weight and augmented weight, respectively.

$$\alpha\left(\sum_{i=0}^{n-1} a_i x_{1i}\right) + \beta(a_n x_{1n}) = +1 \tag{6}$$

$$\alpha\left(\sum_{i=0}^{n-1} a_i x_{2i}\right) + \beta(a_n x_{2n}) = -1 \tag{7}$$

both equation will give and values of α and β,

$$\alpha = \frac{2}{p_{t_1} + p_{t_2}}$$

$$\beta = \frac{2a_n + p_{t_2} - p_{t_1}}{a_n\left(p_{t_1} + p_{t_2}\right)}$$

values of α and β can be used to get new values of weight vector [11]. It provides a solution for optimization of surface boundary for linear classification.

3. PROPOSED ALGORITHM

3.1 Linearly non separable case

In most classification cases, the separating plane is non-linear. However, the theory of proposed classifier can be extended to handle those cases as well. The core idea is to map the input data **x** into a feature space of a higher dimension (a Hilbert space of finite or infinite dimension) [12] and then perform linear separation in that higher dimensional space.

$$x \to \varphi(x),.$$

$$x = (x_1, x_2, ..., x_n),$$

$$\varphi(x) = (\varphi_1(x), \varphi_2(x), ..., \varphi_n(x), ...),$$

where $\varphi(x)$ are some real functions. There is an optimal separating hyperplane in a higher dimension, which corresponds to a nonlinear separating surface in input space. A very simple example to illustrate this concept is to visualize separating a set of data in a 2-dimensional space whose decision surface is a circle. In figure 1, each data point is mapped into a 3 dimensional feature space.

Data inside the circle are mapped to points on the surface of the lower sphere whereas the ones outside the circle are mapped to points on the surface of the upper sphere. The decision surface is linear in the 3-dimensional sphere. Certain kernel functions can be used for mapping low dimension data into high dimensional data.

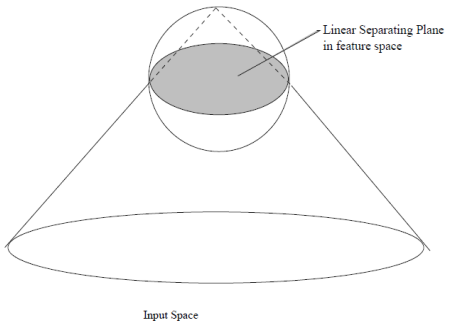

Figure 1. Linear separation in a high dimensional space

Thus, a non-linear kernel can be used to map non-linear input space into high dimensional linear feature space then classifier will process problem as linearly separable problem. Polynomial kernel is used for nonlinear case in the proposed classifier. Figure 2 shows the proposed algorithm using non-linear polynomial kernel.

Input : A linearly separable augmented dataset $S = (y_1, \dots, y_k, \dots, y_n)$ with reflection assumed.
Fix : ε
Define : $q_k = \|y_k\|^2$, $\varepsilon' = 1 - \varepsilon$
Initialize : $t = 0, a_0 = 0,\ l_0 = 0, \theta_0 = 0;$
repeat
  for $k = 0$ to n do
    $p_{t_k} = a_t \,.(y_k + 1)^d$
    if $p_{t_k} \leq \theta_t$ then
      $a_{t+1} = a_t + y_k$
      $l_{t+1} = l_t + 2p_{t_k} + q_k$
      $t \longleftarrow t + 1$
      $\theta_t = \varepsilon' l_t \,/\, t$
until no update made within the for loop
for $k = 0$ to n do
    $p_{s_1} = smallest\ in\ class + 1$
    $p_{s_n} = smallest\ in\ class - 1$
$\alpha = 2 \,/\, (p_{s_1} + p_{s_n})$
$\beta = (2a_n + p_{s_n} - p_{s_1}) \,/\, (p_{s_1} + p_{s_n})$
for only augmented weight in weight vector
  $a = \beta \,.a$
for other weights in weight vector
  $a = \alpha \,.a$

Figure 2. Proposed algorithm for the classifier

3.2 Multi class classification case

Perceptron is a binary classifier, it can process problems that have only two class classifications. Perceptron can be used for multi class classification by connecting multiple perceptron as single unit. Same concept is applied in proposed classifier for multi class classification. At the time of

learning, proposed classifier produces hyperplane equal to one less than number of classes problem have. Basic approach is to separate one class samples from rest, then considering only rest again separate one class samples from rest and so on.

4. EXPERIMENTAL EVALUATION

The proposed algorithm is implemented in c++ using an object oriented approach. The classifier was experimented on some benchmark data sets from the UCI machine learning repository [13]: Iris plants database (IRIS), monk's problems (MNK1, MNK2, MNK3), image segmentation data set (IMAGE), hill train data set (HILL), APECTS heart data set (SPA), connect-4 data set (CON4), hayes rothes data set (HAYES) and glass identification data set (GLASS). Some of these data sets have two class classification and some have multi classes. These data sets are then converted into classifier understandable data sets format. Only those data sets, which have two classes, are compared with SVMlight [14]. The machine used in the experiments was a Pentium D, 3.2 GHz computer running UBUNTU 10.10 computer.

Table 1. Classifier and SVMlight output on some two class classification data set

Data set	Classes / features	Training samples	Testing samples	Classifier results		SVMlight results	
				Misclassified samples	Accuracy (in %)	Misclassified samples	Accuracy (in %)
HAYES	2 / 4	102	27	0	100	13	51.85
MNK1	2 / 6	124	431	0	100	144	66.67
MNK2	2 / 6	169	432	59	86.3425	141	67.36
MNK3	2 / 6	122	432	51	88.1944	84	80.55
SPA	2 / 44	80	187	0	100	131	29.95

Table 2. Classifier output on some multi class classification data set

Data set	Classes	features	Training samples	Testing samples	Classifier results	
					Misclassified samples	Accuracy (in %)
IRIS	3	4	150	150	0	100
IMAGE	7	19	210	2100	1199	42.9048
CON4	3	42	1000	425	60	85.882
HAYES	3	4	132	28	1	96.4285
GLASS	7	10	164	50	14	72.00

Table 1 shows classifier and SVMlight output on some benchmark data sets. Only two class classification data sets are processed through SVMlight. Results show that for each data set the proposed classifier produces a better accuracy than SVMlight. Even for HAYES, MNK1 AND SPA data sets, classifier produces 100% accuracy, while SVMlight failed to do it. For SPA data set, in which number of features are more as compare to others, SVMlight produces very less accuracy. These results proved that proposed classifier is more accurate than SVMlight for two class classification data sets.

Table 2 shows classifier output on some multi class classification data sets. For IRIS data set classification is done on training samples and classifier produces 100% accuracy for it. This shows that classifier is able to generalize well. IMAGE data set is a seven class classification data set. The classifier produces less accuracy for this data set, major issue for this can be very less training samples as compared to testing samples. But for GLASS data set which has seven classes and different training and testing samples, classifier produces 72% accuracy. It is observed that classifier produces less accuracy, if the number of classes is more but it also depends on number and pattern of training samples. These issues can be taken for further research in this area. Based on above result it can be say that current implementation of proposed classifier fully withstand with SVM and also provide advantages of classical perceptron algorithm.

4. CONCLUSIONS

The proposed classifier for large margin employ the classical perceptron updates and converges, in a finite numbers of steps. It is implemented for linearly separable, non-linearly separable by mapping input space into high dimensional feature space as well as for multi class problems using the concept of multi perceptron. The perceptron based weight updating approach improves the performance of classifier as compared to SVM. The proposed classifier, on larger dimensional benchmark data sets, produces better results as compared to SVMlight in term of accuracy.

REFERENCES

[1] Rosenblatt, F.: The perceptron: A probabilistic model for information storage and organization in the brain. Psychological Review, 65(6) (1958) 386–408.

[2] V. Vapnik, Statistical Learning Theory. New York: Wiley, 1998

[3] Cristianini, N., Shawe-Taylor, J.: An introduction to support vector machines (2000) Cambridge, UK: Cambridge University Press.

[4] Y. Li and P. M. Long, "The relaxed online maximum margin algorithm," Mach. Learn., vol. 46, nos. 1– 3, pp. 361–387, Jan. 2002.

[5] C. Gentile, "A new approximate maximal margin classification algorithm," J. Mach. Learn. Res., vol. 2, pp. 213–242, Dec. 2001.

[6] Duda, R.O., Hart, P.E.: Pattern classsification and scene analysis (1973) Wiley.

[7] Novikoff, A.B.J.: On convergence proofs on perceptrons. In Proc. Symp. Math. Theory Automata, Vol. 12 (1962) 615–622.

[8] Tsampouka, P., Shawe-Taylor, J.: Perceptron-like large margin classifiers. Tech. Rep., ECS, University of Southampton, UK (2005).

[9] Tsampouka, P., Shawe-Taylor, J.: Analysis of generic perceptron-like large margin classifiers. ECML (2005) 750–758.

[10] Panagiotakopoulos, C., Tsampouka. P.: The perceptron with dynamic margin., ALT(2011), pp. 204-218.

[11] Panwar, H., Gupta, S.: Optimized large margin classifier based on perceptron., Advances in Computer Science, Engineering & Applications, AISC 2012, Volume 166/2012, 385-392

[12] Pontil, M., Verri, A.: Properties of Support Vector Machines. MIT AI Memo 1612, 1998.

[13] UCI's Machine Learning Repository: *http://archive.ics.uci.edu/ml/.*

[14] SVMlight: A Light Weight Support Vector Machine. *http://svmlight.joachims.org/.*

Authors

Hemant Panwar received his Bachelor of Engineering degree in Information Technology from RGPV University, India in 2010. He is currently pursuing Master of Engineering in Computer Engineering from SGSITS, Indore, India. His research interests include machine learning and system software design.

Surendra Gupta received the Bachelor of Engineering degree in computer science and engineering from Barkatullah University, India in 1997 and Master of Engineering degree in computer engineering from DAVV University, India in 2000. He is currently working as Assistance Professors in computer engineering department at SGSITS Indore, India. His interests are in machine learning and optimization. He is a member of the computer society of India.