Kernel Classifier with Correntropy Loss

Rosha Pokharel and José. C. Príncipe

Department of Electrical and

Computer Engineering

University of Florida

Gainesville, Florida, USA

Email: {rosha, principe}@cnel.ufl.edu

Abstract—Classification can be seen as a mapping problem where some function of x_n predicts the expectation of a class variable y_n . This paper uses kernel methods for the prediction of class variable, together with a recently proposed cost function for classification, called Correntropy-loss(C-loss) function. C-Loss is a non-convex loss function based on a similarity measure called correntropy and is known to closely approximate the ideal 0 – 1 loss function for classification. This paper shows via experimental results that, by replacing the cost function - Mean Square Error (MSE) in a conventional kernel based functional mapping, by a non-convex loss function C-Loss, a non-overfitting, and hence, a better classifier can be obtained. Since gradient descent can still be used with the C-loss and the kernel mapper, the classifier can be easily trained without performance penalty, compared to the SVM, which makes the approach very practical.

I. INTRODUCTION

Classification is a broad and well researched topic in Machine learning field and is most popularly used in today's electronic world of rapidly increasing various forms of data. A classifier learns a correct way of predicting labels or a decision rule for an incoming data, based on some sort of learning algorithm that focuses on minimizing the empirical risk. Given n observations, where each observation consists of a pair: a vector $\mathbf{x}_i \in R^m, i = 1, \dots, n$ and the associated label $y_i \in \{-1, 1\}$, assumed to be i.i.d realization of a random pair $(\mathbf{X}, Y)$, a learning machine is defined by a set of possible mappings $\mathbf{x} \rightarrow f(\mathbf{x}, w)$, where the functions $f(\mathbf{x}, w)$ themselves are labeled by the adjustable parameters w [1]. Since a particular choice of w generates a trained machine, a Radial Basis Function(RBF) network with w which corresponds to the weights, is a learning machine in this sense and requires some form of learning rule, which depends on a loss function. The difference between the true label y and the predicted label $f(\mathbf{x}, w)$, is called loss and the ideal loss function for classification is the misclassification error rate or the 0 – 1 loss,

$$l_{0-1}(f(\mathbf{x}, w), y) = \frac{1}{2} \|y_i - f(\mathbf{x}_i, w)\|_0 \quad (1)$$

where $\|\cdot\|_0$ denotes the L_0 norm. The expected value of this measured error on the training set, for a finite number of observations, is called the empirical risk, $R_{emp}(w)$ [1] [2] i.e.,

$$R_{emp}(w) = \frac{1}{2l} \sum_{i=1}^n \|y_i - f(\mathbf{x}_i, w)\|_0 \quad (2)$$

Minimization of this fixed number $R_{emp}(w)$ for a particular choice of w and for a particular training set $\{\mathbf{x}_i, y_i\}$, is called empirical risk minimization, and is a basis for learning the decision rule or the discriminant function f for classification.

The 0 – 1 loss is however, non-convex and dis-continuous and hence, convex surrogate functions are used in order to achieve minimization of $R_{emp}(w)$ [3]. Most widely used surrogate loss functions are square loss (used in neural network and RBF network), $(y - f(\mathbf{x}))^2$ and hinge loss, $|1 - yf(\mathbf{x})|$ (used in Support Vector Machine). Convex loss functions are widely preferred and used because of their computational advantages such as, unique optima, easy to use and readily available optimization tools, etc. However, such loss functions are poor approximators of the 0 – 1 loss. Hence, this paper replaces the square loss in the kernel method, by a recently proposed non-convex loss function for classification, called correntropy loss function [4], which is based on the similarity measure proposed in [5]. Use of correntropy induced cost function instead of MSE in Kernel adaptive filtering [6] yielding a robust regression method, has been already successfully introduced in [7]. It is called Kernel Adaptive Maximum Correntropy Criterion (KMC), and is inspired by Kernel Least Squares(KLMS) algorithm [8]. It has shown an improved performance with respect to accuracy and robustness in presence of impulsive noise. This paper extends the methodology to design a kernel classifier trained with the C-loss function.

C-loss in [4] has been used in a neural network classifier as a modification of the back propagation algorithm and shows an improved performance over square loss. However, the non-convexity introduced by the neural network architecture impacts the overall performance and hence, the algorithm has to incorporate a global loss function like the square loss, in order to first approach the vicinity of the global optimum and then switches to the C-loss function. Thus, to avoid this complexity brought upon by the neural network architecture, in this paper the C-loss function will be used in conjunction with a kernel classifier that can still be adapted with gradient descent approach. It uses gradient descent approach and replaces the conventional square loss function. The classifier thus obtained, shows a very good performance in terms of accuracy and almost no overfitting even after prolonged training.

The paper proceeds as follows: Section II gives a brief summary of C-loss function and KMC algorithm, section III

Fig. 1. C-loss function for different values of kernel parameter σ , plotted along with 0 – 1 loss.

explains the formulation of kernel method with C-loss to train the classifier using gradient descent technique, section IV includes the results obtained on two real world datasets and section V concludes the paper with possible future work.

II. RELATED WORK

A. C-loss function

Given the classification problem setting described earlier in section I, the goal of classification is to select a function f , such that (2) is minimized. Due to the non-convexity and discontinuity of the 0–1 loss function, choosing a correntropy based loss function gives rise to the C-loss function. In a classification setting, the optimization goal is to maximize the similarity between the classifier output and the true label in correntropy sense and so the loss function should be chosen such that minimization of the expected risk is equivalent to maximization of correntropy [4]. Therefore, the C-loss function or the correntropy induced loss function is:

$$l_C(y, f(\mathbf{x})) = \beta[1 - K_\sigma(y, f(\mathbf{x}))] \quad (3)$$

where β is a positive scaling constant chosen such that for classification margin $\alpha = yf(\mathbf{x})$, $l_C(\alpha = 0) = 1$ and thus, $\beta = [1 - \exp(\frac{-1}{2\sigma^2})]^{-1}$. Also, K_σ is a Gaussian kernel with the kernel size parameter σ . From the definition of correntropy, the σ parameter governs the region where similarity measure is computed and hence is a very crucial factor in case of C-loss as well [5]. [4] shows that C-loss with $\sigma = 0.5$ gives a close approximation to 0-1 loss while C-loss with $\sigma = 1$ is almost linear, though non-convex, and closely approximates the hinge loss function. Fig.1 compares different values of σ with 0–1 loss. The derivative of C-loss function given by (4) also has a very desirable property, governed by σ parameter.

$$\frac{\partial l_C(e_n)}{\partial e_n} = \frac{\beta e_n}{\sigma^2} \exp\left(\frac{-e^2}{2\sigma^2}\right) \quad (4)$$

From the plot of the derivatives as function of the error e_n , as shown in Fig.2, it is clear that if $1 < |e_n| < 2$, the sample is misclassified, and if $0 < |e_n| < 1$, the sample is correctly classified. Compared to the square loss, it gives very less

Fig. 2. Derivative of C-loss function for different values of σ (dotted line), compared with derivative of square loss (solid line).

weight to the high errors, remaining less sensitive to outliers or noisy samples.

B. KMC

KMC is a kernel based Maximum Correntropy Criterion(MCC) learning algorithm for adaptive filtering, which is inspired by KLMS algorithm. This algorithm maps the input data $\mathbf{u}_i$ into a higher dimensional feature space as $\varphi(\mathbf{u}_i)$ through a kernel induced mapping and then uses MCC as the cost function to minimize the difference between the desired data d and the filtered output y . With the use of the *kernel trick* and gradient descent approach, the output is expressed in terms of inner products between the new input and previous inputs, weighted by prediction error. Given the new pairwise sample $\{\varphi(\mathbf{u}_n), d_n\}$, and the weight function Ω at sample n ,

$$y_{n+1} = \Omega_{n+1}^T \varphi_{n+1} \quad (5)$$

which simplifies to

$$y_{n+1} = \eta \sum_n^{i=1} \left[\exp\left(\frac{-e_i^2}{2\sigma^2}\right) e_i \kappa(\mathbf{u}_i, \mathbf{u}_{n+1}) \right] \quad (6)$$

κ = positive definite kernel

φ_i = simplified notation for $\varphi(\mathbf{u})_i$

η = learning rate

$e_n = d_n - \Omega_n^T \varphi_n$

$$\Omega_{n+1} = \eta \sum_n^{i=1} \left[\exp\left(\frac{-e_i^2}{2\sigma^2}\right) e_i \varphi_i \right]$$

In [7] it is shown that KMC is a self regularized method under an appropriate step size and initial condition. This paper utilizes the methodology applied to the problem of classification and uses C-loss for two-class classification.

III. KERNEL CLASSIFIER WITH C-LOSS

The kernel method, understood as the algorithmic implementation of statistical learning theory [9], is a very popular approach in many fields including SVMs, regularization networks, boosting and Gaussian processes, etc. [10], [11] and [12] provide good literature in kernel based classifiers with different types of learning algorithms for training the learning machine. This paper is also based on the kernel method for classification, but uses a gradient descent approach and C-loss function for training the classifier. Only the two-class classification problem has been addressed in this paper, but it generalizes without any problem to the multi class case, like in neural network training. By selecting a Gaussian function as kernel, the kernel method implements an RBF network, where every training sample is regarded as the center of the a node for RBF network. Gaussian kernel has been used in this case, to utilize its advantage of being a universal approximator [13]. Thus, by placing a kernel at every sample, the classifier maps the input samples to the desired response $y \in \{-1, 1\}$. Following the argument in Section II, the output of an RBF network can be represented as:

$$y(\mathbf{x}) = \sum_{i \in N} w_i \langle \varphi(\mathbf{x}_i), \cdot \rangle = \sum_{i \in N} w_i \kappa(\mathbf{x}_i, \cdot) \quad (7)$$

where the approximating function $y(\mathbf{x})$ is represented as a sum of N radial basis functions, each associated with a different center $\mathbf{x}_i$, and weighted by a coefficient w_i obtained from the training data, and κ is a positive definite kernel. Following the usual notation in adaptive filters, coefficient w_i is expressed as the weight function Ω . Then, using the stochastic gradient approximation to the new pairwise sample $\{\varphi(\mathbf{x}_n), d_n\}$:

$$\begin{aligned} \Omega_0 &= 0 \\ \Omega_{n+1} &= \Omega_n + \eta \frac{\partial}{\partial \Omega_n} \left[\beta \left\{ 1 - k_\sigma \left(d_n, \Omega_n^T \varphi(\mathbf{x}_n) \right) \right\} \right] \\ &= \Omega_n + \eta \frac{\beta}{\sigma^2} \exp \left(\frac{-e_n^2}{2\sigma^2} \right) e_n \varphi_n \\ &= \Omega_{n-1} + \eta \frac{\beta}{\sigma^2} \sum_{i=1}^{i=n-1} \left[\exp \left(\frac{-e_n^2}{2\sigma^2} \right) e_i \varphi_i \right] \\ &= \eta \frac{\beta}{\sigma^2} \sum_{i=1}^{i=1} \left[\exp \left(\frac{-e_n^2}{2\sigma^2} \right) e_i \varphi_i \right] \end{aligned} \quad (8)$$

Using the approach of (5), y_{n+1} can be readily written as:

$$y_{n+1} = \eta \frac{\beta}{\sigma^2} \sum_{i=1}^{i=1} \left[\exp \left(\frac{-e_i^2}{2\sigma^2} \right) e_i \kappa(\mathbf{x}_i, \mathbf{x}_{n+1}) \right] \quad (9)$$

It is not surprising to see that both the equations (6) and (9) turn out to be the same, except for the constant term $\eta \frac{\beta}{\sigma^2}$ and let it be represented by γ . It is clear from (9) that γ serves the same purpose as the learning parameter η while it is inversely proportional to the kernel size.

C-loss function is a non-convex loss function for some of the values of the kernel size, and has been used with neural network classifier in [4]. In [4] it has been shown that, there

is a need of a global loss function that guides the C-loss function to reach the optimal solution. More specifically, the paper proposes training the classifier by square loss for certain number of epochs and then switching to C-loss to get the optimal solution. This could guarantee the optimality of the classifier, but the supposedly *optimal* solution could also be entirely guided by the square loss function. Moreover, knowing when to switch between the two loss function is in itself a challenging task to achieve [4].

In this case, where the C-loss is used with the kernel mapper, only the C-loss is potentially non-convex for proper selection of the kernel size in the kernel mapper. Hence, the training has been simplified by using C-loss alone and the results will be shown with C-loss training. It can be observed from Fig.1 that C-loss with $\sigma = 1$ is almost linear and closely resembles the hinge loss. So, C-loss with $\sigma = 1$ has been chosen to train the classifier, which should yield a close match to the SVM performance.

IV. EXPERIMENTS AND RESULTS

Unlike kernel regression, kernel classification uses a much smaller number of samples and hence, the classifier needs to be passed through the training data multiple times. The training samples are said to be passed through the classifier in *epochs*, and each epoch contains a fixed number of training samples. Hence, the corresponding coefficients are added at every epoch to finally obtain a coefficient vector. This paper uses the same idea to train the classifier while using gradient descent approach. The classifier has been tested with real world dataset obtained from the UC Irvine machine learning repository [14]. Two different datasets have been used : Pima Diabetes data set and Wisconsin Breast cancer dataset. The classifier is trained using C-loss and square loss individually and the mean classification performance in both the cases have been compared with each other. From the classification performance, it can be observed that the performance of the C-loss function is better than that of square loss function even after prolonged training.

The classifier has been trained with different number of training epochs and with different kernel size for the mapper. As every training point serves as the center for the kernel, and because there are very few number of samples in a classification problem, the mapper kernel size makes a big difference in the classification performance. Hence, the experiment has been done for different kernel sizes in order to check its impact on this classification setting.

A. Pima Diabetes Dataset

With this dataset, the classifier classifies if a test subject falls in a diabetic or a non-diabetic group, based on the 8 physiological measurements provided as features. The dataset consists of 768 subjects or samples, out of which, 300 samples have been used for training the classifier while the rest have been used for testing. The classifier performance has been obtained after averaging the 100 Monte Carlo runs. The result thus obtained at different training epochs is presented in figure

Fig. 3. Mean performance of classifier with C-loss (solid line) and MSE (dotted line) on Pima Diabetes dataset in terms of percentage accuracy for different values of mapper kernel size, number of epochs and fixed C-loss kernel size $\sigma = 1$.

Fig. 4. Mean performance of classifier with C-loss (solid line) and MSE (dotted line) on Wisconsin Breast cancer dataset in terms of percentage accuracy for different values of mapper kernel size, number of epochs and fixed C-loss kernel size $\sigma = 1$.

Fig.3 and tabulated in Table I. Table I tabulates the the average percentage accuracy of classification obtained at the end of training, for different number of epochs and different values of classifier kernel sizes. The table also shows the variance of accuracy along the 100 Monte Carlo runs at the end of training epochs, for both the loss functions. The best performance are shown in bold numbers. From the figure Fig.3, it is clear that the performance of the classifier with C-loss is much better than that with square loss, in terms of the accuracy and overfitting after prolonged training. The result shows that by using C-loss in the kernel based classifier, a smooth, non-overfitting result can be obtained. Also, larger classifier kernel size seems to give a better performance, which is clear from the figure as well as the table.

B. Wisconsin Breast Cancer Dataset

The dataset consists of samples from 683 breast cancer patients and the classification problem in this dataset is to separate the malignant or benign sample based on the 9 features such as size, shape, clump thickness, and cell properties, etc. 200 samples have been chosen for training, the rest for testing the classifier. As earlier, the classifier is trained with C-loss and square loss, for different number of epochs and different values of classifier kernel size. The result obtained from the average of 100 Monte Carlo runs have been shown in Fig. 4 and tabulated in Table II. The table also shows the variance of accuracy along the 100 Monte Carlo runs at the end of training epochs, for both the loss functions and marks the consistency of C-loss. Similar to the result obtained in table

TABLE I
MEAN PERFORMANCE OF CLASSIFIER WITH C-LOSS AND MSE ON PIMA DIABETES DATASET FOR DIFFERENT VALUES OF MAPPER KERNEL SIZE, NUMBER OF EPOCHS AND FIXED C-LOSS KERNEL SIZE = 1

Kernel size	2			1		
Number of epochs	25	50	75	25	50	75
C-loss	74.13±0.61	75.0±0.63	74.98±0.74	74.35±0.72	74.01±0.81	73.86±0.75
MSE	69.03±0.81	55.01±0.75	51.91±0.87	71.39±0.78	65.31±0.82	62.81±0.88

TABLE II
MEAN PERFORMANCE OF CLASSIFIER WITH C-LOSS AND MSE ON WISCONSIN BREAST CANCER DATASET FOR DIFFERENT VALUES OF MAPPER KERNEL SIZE, NUMBER OF EPOCHS AND FIXED C-LOSS KERNEL SIZE = 1

Kernel size	2			1		
Number of epochs	25	50	75	25	50	75
C-loss	97.22±0.61	96.68±0.64	97.25±0.76	96.96±0.74	96.88±0.76	96.84±0.68
MSE	94.19±0.61	63.78±0.64	56.23±0.83	96.25±0.90	78.39±0.75	72.55±0.87

in Pima data set, the classifier has a better performance when C-loss is used and the performance deteriorates with square loss. Table II shows the average percentage accuracy of at the end of training for each of six cases and the best performance is shown in bold in case of both the loss functions. The variance of the accuracy at the end of each epoch shown in table, marks the consistency of the result through runs.

C. Comparison with SVM

As the method used in this paper uses a different cost function in a kernel based classifier and out-performs the square loss function, it would be interesting to compare the result with that of SVM. Table III compares the performance of SVM with the best result obtained from C-loss, tabulated in table I and II for Pima diabetes and Wisconsin breast cancer dataset respectively. The best SVM performance for a particular box-constraint parameter, C , has been tabulated from the result presented in [4]. Table III shows that the performance of C-loss and SVM is very close in both the datasets.

TABLE III
COMPARISON OF BEST PERFORMANCE OF C-LOSS FROM TABLE I AND II AND THAT OF SVM ON BOTH DATASETS

Dataset	Pima Diabetes	Wisconsin Breast cancer
C-loss	75.00	97.25
SVM	74.46	96.92

V. CONCLUSION

Kernel methods have been widely used in the field of classification and clustering, but there is always a search for an efficient learning method and an appropriate loss function. This paper tries to put light in training a Kernel method for classification using a different non-convex loss function and shows that it gives an improved result compared to the

conventional way of using square loss function. The paper shows that, by training the kernel adaptive filter inspired kernel based classifier with non-convex C-loss function, a more accurate and non-overfitting classifier can be obtained. Moreover, results show that the performance of such classifier is as good as that of SVM. Also, with very less variance across the 100 Monte Carlo runs, the loss function supports the fact that it does not undergo the effect of local minima. The regularization property of such a classifier, effect of the classifier kernel size and the effect of using C-loss with $\sigma = 0.5$ are some of interesting future work in this topic, that are worth exploring.

REFERENCES

- [1] C. Burges, "A tutorial on support vector machines for pattern recognition," *Data mining and knowledge discovery*, vol. 2, no. 2, pp. 121–167, 1998.
- [2] V. Vapnik, S. Golowich, and A. Smola, "Support vector method for function approximation, regression estimation, and signal processing," in *Advances in Neural Information Processing Systems 9*. Citeseer, 1996.
- [3] C. Bishop and SpringerLink, *Pattern recognition and machine learning*. springer New York, 2006, vol. 4.
- [4] A. Singh and J. Principe, "A loss function for classification based on a robust similarity metric," in *Neural Networks (IJCNN), The 2010 International Joint Conference on*. IEEE, 2010, pp. 1–6.
- [5] W. Liu, P. Pokharel, and J. Principe, "Correntropy: A localized similarity measure," in *Neural Networks, 2006. IJCNN'06. International Joint Conference on*. IEEE, 2006, pp. 4919–4924.
- [6] W. Liu, J. Principe, and S. Haykin, *Kernel Adaptive Filtering: A Comprehensive Introduction*. Wiley Publishing, 2010.
- [7] S. Zhao, B. Chen, and J. Principe, "Kernel adaptive filtering with maximum correntropy criterion," in *Neural Networks (IJCNN), The 2011 International Joint Conference on*. IEEE, 2011, pp. 2012–2017.
- [8] W. Liu, P. Pokharel, and J. Principe, "The kernel least-mean-square algorithm," *Signal Processing, IEEE Transactions on*, vol. 56, no. 2, pp. 543–554, 2008.
- [9] V. Vapnik, "An overview of statistical learning theory," *Neural Networks, IEEE Transactions on*, vol. 10, no. 5, pp. 988–999, 1999.
- [10] D. Lowe, "Similarity metric learning for a variable-kernel classifier," *Neural computation*, vol. 7, no. 1, pp. 72–85, 1995.
- [11] P. Paclík, J. Novovicová, P. Pudil, and P. Somol, "Road sign classification using laplace kernel classifier," *Pattern Recognition Letters*, vol. 21, no. 13, pp. 1165–1174, 2000.

- [12] X. Hong, S. Chen, and C. Harris, "A kernel-based two-class classifier for imbalanced data sets," *Neural Networks, IEEE Transactions on*, vol. 18, no. 1, pp. 28–41, 2007.
- [13] C. Micchelli, Y. Xu, and H. Zhang, "Universal kernels," *The Journal of Machine Learning Research*, vol. 7, pp. 2651–2667, 2006.
- [14] C. Blake and C. Merz, "Uci repository of machine learning databases. university of california, irvine, dept. of information and computer sciences," 1998.