Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

The C-loss function for pattern classification

Abhishek Singh [a,*], Rosha Pokharel [b], Jose Principe [b]

[a] Department of Electrical & Computer Engineering, University of Illinois at Urbana-Champaign, United States
[b] Department of Electrical & Computer Engineering, University of Florida, Gainesville, United States

ARTICLE INFO

Article history:
Received 3 October 2012
Received in revised form
16 July 2013
Accepted 24 July 2013
Available online 3 August 2013

Keywords:
Classification
Correntropy
Neural network
Loss function
Backprojection

ABSTRACT

This paper presents a new loss function for neural network classification, inspired by the recently proposed similarity measure called Correntropy. We show that this function essentially behaves like the conventional square loss for samples that are well within the decision boundary and have small errors, and L_0 or counting norm for samples that are outliers or are difficult to classify. Depending on the value of the kernel size parameter, the proposed loss function moves smoothly from convex to non-convex and becomes a close approximation to the misclassification loss (ideal 0–1 loss). We show that the discriminant function obtained by optimizing the proposed loss function in the neighborhood of the ideal 0–1 loss function to train a neural network is immune to overfitting, more robust to outliers, and has consistent and better generalization performance as compared to other commonly used loss functions, even after prolonged training. The results also show that it is a close competitor to the SVM. Since the proposed method is compatible with simple gradient based online learning, it is a practical way of improving the performance of neural network classifiers.

1. Introduction

Classification aims at assigning class labels to data using an ‘optimal’ decision rule that is learnt using a set of pre-labeled training samples. This ‘optimal’ decision rule or discriminant function f is learnt by minimizing the empirical risk, which is a sample average of a loss function. The loss (function of the prediction $f(\mathbf{x})$, and the true label y) is essentially the price we pay for predicting the label to be $f(\mathbf{x})$, instead of y. This procedure for learning the discriminant function is called the Empirical Risk Minimization, and is a widely used principle for classification and statistical learning [1,2].

The most natural loss function for classification is the misclassification error rate (or the 0–1 loss)

$$l_{0-1}(f(\mathbf{x}),y) = \|(-yf(\mathbf{x}))_+\|_0, \tag{1}$$

where $(.)_+$ denotes the positive part and $\|.\|_0$ denotes the L_0 norm. This essentially is a count of the number of incorrect classifications made by the decision rule f. Therefore, the 0–1 loss function directly relates to the probability of misclassification. Optimization of the risk based on such a loss function, however, is computationally intractable due to its non-continuity and non-convexity [1,2]. Therefore, a surrogate loss function is applied to many classification procedures. For example, well known loss functions for training the weights of a neural network or a radial basis function (RBF) network are the squared loss, $(y-f(\mathbf{x}))^2$, or $(1-yf(\mathbf{x}))^2$, and the logistic loss, $\log(1+e^{-yf(\mathbf{x})})$. The Support Vector Machine (SVM) [3,4] uses the hinge loss, $[1-yf(\mathbf{x})]_+$.

Within the statistical learning community, convex surrogates of the 0–1 misclassification loss are highly preferred because of the virtues that convexity brings – unique optima, efficient optimization using convex optimization tools and amenability to theoretical analysis of error bounds [5]. However, convex functions are still poor approximations to the 0–1 loss function. They tend to be boundless and offer poor robustness to outliers [2]. Another important limitation is that the complexities of convex optimization algorithms grow very fast with more data [6]. Some non-convex loss functions have been proposed recently with the aim of addressing these issues [7,8].

There is a large class of problems where optimization cannot be done using convex programming techniques. For example, training of deep networks for large scale AI problems primarily rely on online, gradient-based methods [9,10]. Such neural network based learning machines can benefit from non-convex loss functions, as they can potentially offer better scalability, robustness and generalization performance. Although non-convex optimization and loss functions do not offer many theoretical guarantees, the empirical evidence that they work better in engineering applications is becoming overwhelming [6].

A loss function for classification that is inspired by the statistical measure called Correntropy [11] was proposed in [12]. Correntropy

* Corresponding author. Tel.: +1 3526154195.
E-mail addresses: abhishek486@gmail.com, asingh18@illinois.edu, abhishek_singh@ieee.org (A. Singh), rosha@cnel.ufl.edu (R. Pokharel), principe@cnel.ufl.edu (J. Principe).

0031-3203/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.patcog.2013.07.017

between two random variables is a generalized correlation function or a robust measure of statistical similarity, which makes use of second and higher order statistics. It has been successfully applied to problems like robust regression [13], adaptive filtering [14–17], pitch detection in speech [18,19], MACE (Minimum Average Correlation Energy) filtering for object recognition [20], etc. In a classification setting, maximizing the similarity between the prediction $f(\mathbf{x})$ and the target y in the Correntropy sense, effectively induces a non-convex, smooth loss function (which we call C-loss) that can be used to train a classifier using an online gradient based technique.

This paper extends our earlier work in [12] and further characterizes the C-loss function for classification. We examine the performance of a single hidden layer perceptron trained with the C-loss function (using backpropagation) over different system parameters such as training epochs and network size. We obtain better generalization results on several synthetic and real world datasets, when compared to the traditional squared loss function.

Furthermore, we demonstrate the performance of the C-loss function while training RBF networks as well. We obtain superior results using the C-loss function when compared to the logistic loss function, while training an RBF classifier.

We also compare the performance of the proposed loss function with SVMs, that use the hinge loss function.

Conventional neural network based classifiers suffer from the problem of overfitting due to overtraining, which often causes poor generalization. In all the abovementioned experiments, we show that classifiers trained using the proposed C-loss function are more robust to overfitting even on prolonged training, and are able to maintain consistent generalization performance. The proposed online method of training classifiers using the C-loss function has the overall practical appeal that it offers more consistent and better generalization performance at no additional computational cost.

The next section formalizes the pattern classification problem from the point of view of statistical learning. Section 3 introduces the C-loss function, along with some of its properties. In Section 4 we discuss how the C-loss function can be used to train neural network based classifiers. Section 5 presents our experimental results. We compare the performance of the C-loss function to the square loss (on MLPs), the logistic loss (on RBF networks), and SVMs on several real world datasets obtained from UCI Machine Learning Repository [21], using different neural network architectures, and several different system parameters. In Sections 6 and 7 we present some important discussions and insights and draw conclusions.

2. Statistical theory of classification

2.1. Loss functions and risk

Suppose we are given a training set of observations $D_n = \{(\mathbf{x}_i,\ y_i),\ i = 1, 2, \ldots, n\}$, assumed to be i.i.d. realizations of a random pair $(\mathbf{X}, Y)$. Here, $\mathbf{X} \in \mathcal{X}$ is the input vector and $Y \in \{-1, 1\}$ is the class label (we consider a binary classification problem for now). The goal of classification is to select a function f from a class of functions $\mathcal{F}$, such that the sign of $f(\mathbf{X})$ is an accurate prediction of Y under an unknown joint distribution $P(\mathbf{X}, Y)$. In other words, we want to select $f \in \mathcal{F}$ that minimizes the risk $R(f)$ given by

$$R(f) = E[l_{0-1}(Yf(\mathbf{X}))] = P(Y \neq \text{sign}(f(\mathbf{X}))). \tag{2}$$

The product $yf(\mathbf{x})$ is called the margin (denoted by α) and can be treated as a measure of correctness of the decision for the sample $\mathbf{x}$. Given a sample set D_n of realizations, it is natural to consider the empirical risk, or the sample average of the 0–1 loss

$$\hat{R}(f) = \frac{1}{n}\sum_{i=1}^{n} l_{0-1}(y_i f(\mathbf{x}_i)). \tag{3}$$

Optimization of the empirical risk as above, however, is computationally intractable, primarily because of the discontinuity of the 0–1 loss function. The optimization procedure therefore involves choosing a surrogate $\phi(\alpha) = \phi(yf(\mathbf{x}))$ as the loss function. The result is the minimization of the ϕ−risk and empirical ϕ−risk defined by the following:

$$R_\phi(f) = E[\phi(Yf(\mathbf{X}))] \tag{4}$$

$$\hat{R}_\phi(f) = \frac{1}{n}\sum_{i=1}^{n} \phi(y_i f(\mathbf{x}_i)). \tag{5}$$

Fig. 1 shows three commonly used surrogate loss functions – the hinge loss used in SVMs, the square loss and the logistic loss used in training neural networks and RBF networks.

In addition to making the optimization of the risk tractable, choosing a surrogate loss function has another motivation. Minimizing the sample average of an appropriately well behaved loss function may have a regularizing effect [22].

2.2. Bayes' Optimal decision rule

Let $p(\mathbf{x}) = P(Y = 1|\mathbf{X} = \mathbf{x})$ be the conditional probability of the positive class given $\mathbf{X} = \mathbf{x}$. Then, the decision-theoretic optimal classification rule with the smallest generalization error is $\text{sign}[p(\mathbf{x}) - 1/2]$. This is called the Bayes' optimal rule. The risk associated with the Bayes' optimal rule is called the Bayes' optimal risk $R^* = R(f^*)$.

2.3. Fisher consistency

Definition 1. A margin-based loss function $\phi(yf(\mathbf{x}))$ is said to be Fisher consistent or 'classification calibrated' if the population minimizer f^* of the expected risk $E[\phi(Yf(\mathbf{X}))]$ has the same sign as the Bayes' optimal decision rule $\text{sign}[p(\mathbf{x}) - 1/2]$.

Fisher consistency simply provides the reassurance that optimizing a surrogate loss does not ultimately hinder the search for a discriminant function that achieves the Bayes' optimal risk. Lin [23] states a theorem that can be used to easily check if a given function is Fisher consistent.

Theorem 1. *If V is a function satisfying the following two assumptions*:

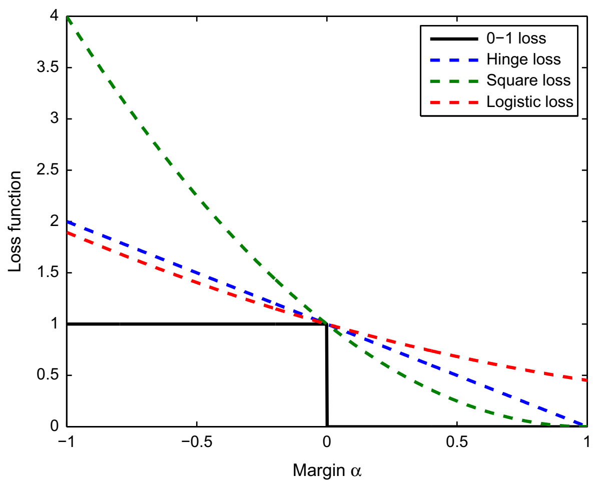

Fig. 1. The hinge and square loss functions, plotted along with the 0–1 loss.