Neurocomputing 278 (2018) 41–50

Contents lists available at ScienceDirect

Neurocomputing

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

Generalized Correntropy based deep learning in presence of non-Gaussian noises

Liangjun Chen[a,*], Hua Qu[a], Jihong Zhao[b,a]

[a] School of Electronic and Information Engineering, Xi'an Jiaotong University, Xi'an 710049, China
[b] School of Telecommunication and Information Engineering, Xi'an University of Posts and Telecommunications, Xi'an 710061, China

ARTICLE INFO

Article history:
Received 30 November 2016
Revised 17 June 2017
Accepted 19 June 2017
Available online 14 September 2017

Keywords:
Deep learning
Generalized Correntropy
Stacked autoencoders
Non-Gaussian noise
Network traffic classification

ABSTRACT

Deep learning algorithms are the hottest topics in machine learning area lately. Although deep learning has made great progress in many domains, the robustness of learning systems with deep architectures is still rarely studied and needs further investigation. For instance, the impulsive noises (or outliers) are pervasive in real world data and can badly influence the mean square error (MSE) cost function based deep learning algorithms. Correntropy based loss function, which uses Gaussian kernel, is widely utilized to reject the above noises, however, the effect is not satisfactory. Therefore, generalized Correntropy (GC) is put forward to further improve the robustness, which uses generalized Gaussian density (GGD) function as kernel. GC can achieve extra flexibility through the GC parameters, which control the behavior of the induced metric, and shows a markedly better robustness than Correntropy. Motivated by the enhanced robustness of GC, we propose a new robust algorithm named generalized Correntropy based stacked autoencoder (GC-SAE), which is developed by combining the GC and stacked autoencoder (SAE). The new algorithms can extract useful features from the data corrupted by impulsive noises (or outliers) in a more effective way. The good robustness of the proposed method is confirmed by the experimental results on MNIST benchmark dataset. Furthermore, we show how our model can be applied for robust network classification, based on Moore network data of 377,526 samples with 12 classes.

1. Introduction

Basically, machine learning acquires information from series of examples to guide computing procedures. There are various machine learning algorithms, such as Support Vector Machine (SVM) [1], Bayesian algorithms [2], K-means [3] and Decision Tree (DT) [4], which are generally used for solving diverse problems, for example, EEG signal classification [5], cancer diagnosis [6], text classification [7], speech transmission [8], facial identification and so on. Fundamentally, deep learning can be understood as an unsupervised feature extractor. With the recent progress, deep learning schemes have flourished in machine learning and many other areas [9–11]. Several algorithms are most popular in deep learning, such as restricted Boltzmann machines [12] based DBN, autoencoders [13,14], Convolution Neural Network (CNN) [15,16] and Recurrent Neural Network (RNN) [17,18]. Under Gaussian assumption, the smooth, convex, mathematical tractable, low computational cost and the optimality can be achieved by using mean square error (MSE) based cost functions. Thus, the MSE based cost function is generally applied in deep learning algorithms, such as autoencoder, RNN and so on. As the MSE performs poor in non-Gaussian situations, the desirable result of MSE can only be obtained under Gaussian distribution. So the deep learning systems with other cost functions need to be further investigated to improve the robustness of non-Gaussian noises.

Correntropy [19] is a special case of the cross entropy in information theoretic learning (ITL) [20]. It is a nonlinear and local similarity measure that shows the similarity between two random variables in a neighborhood of the joint space controlled by the kernel bandwidth. Compared with MSE, Correntropy is excellent for its insusceptibility of outliers with a proper kernel bandwidth [21–24]. Thus, Correntropy is widely applied in replacing the MSE in handling the heavy-tailed impulsive noises [25–28]. In [29], Chen et al. proved that the maximum Correntropy estimation is equivalent to a smoothed maximum a posteriori (MAP) estimation. Generally, a Gaussian kernel is employed as the kernel function of Correntropy, because Gaussian kernel is smooth and strictly positive. When using Gaussian kernel, Correntropy can induce a nonlinear metric called Correntropy induced metric (CIM), which is capable of approximating different norms (from L_0 to L_2) of data. Seth et al. [30] used CIM as an approximation of L_0 norm

[*] Corresponding author.
E-mail address: cljun27@stu.xjtu.edu.cn (L. Chen).

http://dx.doi.org/10.1016/j.neucom.2017.06.080
0925-2312/© 2017 Elsevier B.V. All rights reserved.

to achieve a better sparsity for the compressed signal reconstruction. Singh et al. [31,32] proposed a new Correntropy based cost function named C-loss for training neural network classifier. C-loss, a nonlinear measure of similarity, is robust to outliers and can also approximate different norms (from L_0 to L_2) of data. In essence, C-loss is an MSE in reproducing kernel Hilbert space (RKHS). In [33], Qi et al. used the maximum Correntropy criterion (MCC) in deep learning to reduce the bad influence from outliers. In [34], Chen et al. proposed a Correntropy based SAE (CSAE) which uses the C-loss based reconstruction loss term and the C-loss based sparsity penalty term to replace the original terms in SAE. Through combining the two Correntropy based terms, CSAE shows an obvious improvement of the robustness when the data contains outliers or impulsive noise. In [35], Ma et al. applied Correntropy in sparse adaptive filtering algorithms and used these algorithms to handle robust channel estimation problems. In addition, Correntropy is also used in Kalman filter in [36] to improve the robustness of Kalman filter against impulsive noises. However, Correntropy based loss function is not robust enough. When the impulsive noises become serious, the performance of Correntropy based methods deteriorates severely.

In [37], Chen et al. proposed a generalized Correntropy (GC), which is based on generalized Gaussian density (GGD) function, and imports additional shape parameters α to expand even further the range of possible induced metrics Correntropy possesses. Consequently, GC not only generalizes Correntropy, but also generalizes second-order statics metrics. Likewise, the order-α generalized Correntropy induced metric (GCIM) or generalized Correntropy loss (GC-loss) function also has similar behavior no matter how the norms (from L_0 to L_α) of data changes. By employing suitable α with GC, one can further enhance the robustness of outliers and impulsive noises. Because of the similar behavior, the GC can be applied in deep learning as a cost function as well. Especially, the GC based algorithms may outperform significantly the original Correntropy based algorithms.

In this paper, we propose a robust deep learning model based on SAE and GC-loss, called generalized Correntropy based SAE (GC-SAE). Different from conventional SAE and the CSAE, the reconstruction loss in GC-SAE is built on the basis of GC-loss, which can further enhance the robustness. Through simulation experiments on MNIST benchmark dataset [38], it is proved that the proposed model improves the robustness without increasing computational complexity. Furthermore, series of experiments on Moore network traffic dataset are also presented to show the good robustness of proposed method in handling unclean network traffic classification.

In addition, network traffic classification problem is drawn into this paper as the application case of the proposed GC-SAE algorithm. Network traffic classification plays an indispensable role in modern network management [39]. To effectively manage a network, it is necessary to know the current status of network through traffic classification. One can apply a much better resource allocation strategies through well classified traffic information to improve the QoS. Moreover, the detection of vicious attacks also relies on accurate traffic classification [40]. Therefore, since the invention of Internet, the technology of traffic classification has been drawing increasing attentions of academia and practitioners.

However, real-world traffic data often contains unclean samples, such as zero-day traffic [41], which will severely affect the performances of classifiers. The zero-day traffic is created by new applications which cannot be classified properly by a given classifier and will be mislabeled to a known class. Generally speaking, zero-day traffic is the major portion of unrecognized data making up to 60% of flows and 30% of bytes in a network traffic dataset. The mislabeled samples will not only negatively affect the identification accuracy of the known class, but also decrease the whole performance of the classifier. In addition, the network traffic data is always extracted from the massive real network traffic, which may contain a lot of error information. Most of existing traffic classification methods ignore these bad influences and consequently get severely compromised results. Hence, the robustness of network traffic classification methods needs further investigation. Here, we introduce the proposed GC-SAE algorithm to handle the traffic classification problem and improve the classification accuracy. Series of experiments on Moore network traffic dataset are presented to show the good robustness of proposed GC-SAE algorithm on classifying unclean network traffic data. Impulsive noises and zero-day traffic are added to the original network dataset to test the robustness of proposed GC-SAE in network traffic classification. GC-SAE achieves highest robustness in experiments with both kind of noises.

Particularly, this paper is an extension of the paper “Generalized Correntropy Induced Loss Function for Deep Learning” which is posted on International Joint Conference on Neural Networks (IJCNN 2016) [42].

The remainder of the paper is organized as follows. In the next section, the backgrounds of Correntropy, stacked autoencoders and network traffic classification are briefly described. The definitions and some properties of GC and GC-loss are concisely introduced in Section 3. The new model GC-SAE is presented in Section 4. In Section 5, experiments on MNIST dataset are carried out to show that, even when data is badly corrupted by impulse noises, the robustness of the proposed GC-SAE also performs well. The good classification results of Moore network traffic dataset under impulsive noises and mislabel samples are shown in Section 6. And finally, in Section 7, we give the conclusion.

2. Backgrounds

2.1. Correntropy and Correntropy based loss function

Correntropy is a similarity measure defined by mapping signals nonlinearly into a kernel space [19]. Given two random variables **S** and **T**, Correntropy is defined as

$$V(\mathbf{S},\mathbf{T}) = E[<\Phi(\mathbf{S}),\Phi(\mathbf{T})>] = E[\kappa_\sigma(\mathbf{S},\mathbf{T})], \tag{1}$$

where κ_σ is a radial kernel function with kernel size (or bandwidth) parameter σ, Φ is a nonlinear mapping induced by κ_σ which transforms data from the input space to a high (possibly infinite) dimensional RKHS, E denotes the mean (or sample mean for empirical case) operator and $<\cdot,\cdot>$ denotes the inner product.

Given two samples $S \in R^{M\times N}$ and $T \in R^{M\times N}$,

$$S = [s_1, s_2, \dots, s_N] \tag{2}$$

$$T = [t_1, t_2, \dots, t_N], \tag{3}$$

where M is the dimension of the samples s_i and t_i, and N is the samples number. So, the empirical Correntropy is defined as

$$\hat{V}(S,T) = \frac{1}{N}\sum_{i=1}^{N}\kappa_\sigma(s_i,t_i). \tag{4}$$

Kernel methods are commonly used in machine learning. Gaussian kernel is the most popular kernel in Correntropy, which is defined as follows,

$$\kappa_\sigma(s_i,t_i) = exp\left(-\frac{\| s_i - t_i \|^2}{2\sigma^2}\right) \tag{5}$$

where $\|\cdot\|$ is Euclidean norm. In functional analysis view, the kernel size determines the inner product, i.e., the metric of similarity in RKHS. All the data would look similar in the RKHS (with inner products all close to 1) with a large kernel size. On the contrary,

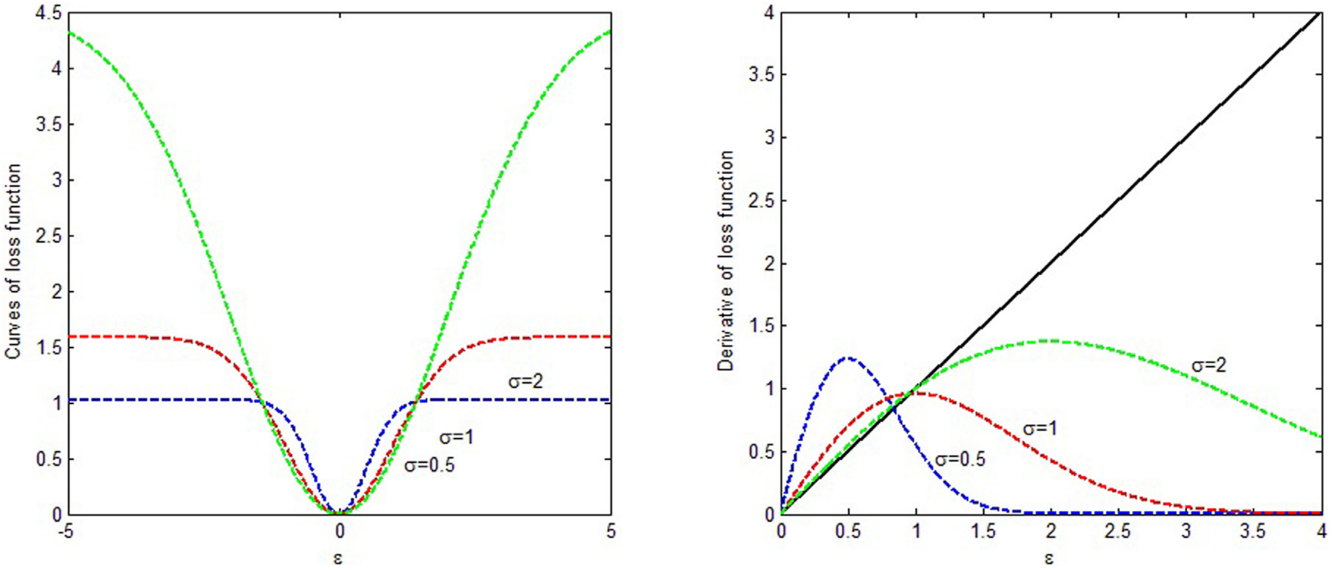

Fig. 1. (a) The function value of C-loss with different kernel size σ as the errorϵ increasing; (b) The derivative of C-loss with different kernel size σ. It is clear that the behavior of C-loss is tuned with the change of kernel size σ. If the MSE loss function is used, the linear derivative with samples which produce high errors will lead to severe effects of the weights. However, highly erroneous samples with C-loss have less influence on the weights.

with a small kernel size, all the data would look different (with inner products all close to 0) and inference cannot proceed on unseen samples that fall between the training points.

It is easy to induce C-loss from the definition of Correntropy. In [31,32], C-loss is proposed for classification tasks. In these tasks, our target is maximizing the similarity between the classifier output **S** and the label **T**. C-loss was defined as follows:

$$C_{\text{loss}}(\mathbf{S},\mathbf{T}) = \beta[1 - E(\kappa_\sigma(\mathbf{S},\mathbf{T}))]. \tag{6}$$

where $\beta = [1 - exp(-\frac{1}{2\sigma^2})]^{-1}$ is a positive scaling constant which is employed to keep $C_{\text{loss}}(0) = 1$. Similarly, the empirical C-loss between the classifier output samples S and the label samples T can be calculated as

$$C_{\text{loss}}(S,T) = \beta\left[1 - \frac{1}{N}\sum_{i=1}^{N}\kappa_\sigma(s_i,t_i)\right]. \tag{7}$$

Particularly, CIM is a special case of empirical C-loss when $\beta = 1$.

In Fig. 1, the curves and the derivative curves of C-loss are presented. In order to compare the difference between C-loss and MSE based loss function, the derivative curve of the MSE is also plotted as a linear curve, which is increasing with the error in Fig. 1(b). Thus, a large error $\epsilon = T - S$, which was always caused by outliers or impulsive noises, will severely affect the weights of the model when using MSE as the loss function. Conversely, the derivatives of C-loss are too small to influence the training when error is very large. As a result, C-loss can be applied to decrease the harmful impact of the outliers and improve the robustness of original system. However, limited by the definition, the metric induced by a Gaussian based Correntropy mapping only spans from L_2 to L_0. Thus, the shape of Correntropy based loss function may not optimally suit specific datasets, which leads to a sub-optimum result.

Additionally, with the ability of approximating different norms, when $M = 1$, $C_{\text{loss}}(X,\mathbf{0})$ (or $CIM(X,\mathbf{0})$) can be utilized to build an L_0 norm approximator of $\|X\|_0$ [30]:

$$\| X \|_0 \sim C_{\text{loss}}(X,\mathbf{0}) = \beta\left[1 - \frac{1}{N}\sum_{i=1}^{N}\kappa_\sigma(X,\mathbf{0})\right]. \tag{8}$$

Generally speaking, the sparsity is achieved through minimizing an L_0 norm of the original data. Unfortunately, L_0 norm minimization is a NP-hard problem. As a result, a common way is employing L_1 norm as an approximator of L_0 norm. Here, the C-loss based L_0 norm approximator can be arbitrarily closed to the L_0 norm, as $\sigma \to 0$. By minimizing the C-loss based L_0 norm approximator, one can get a much sparser representation when compared with the L_1 norm based L_0 norm approximator.

2.2. Stacked autoencoder

Auto-encoder is a popular method in deep learning. A compressive representation of the original data is learned by an autoencoder from encoding the input and outputting the reconstruction of its own input.

A typical autoencoder has three layers, the input layer, the hidden layer and the output layer. It can also be divided into two parts, the encoder and the decoder. The encoder and the decoder are a pair of mapping which are totally inverse. The encoder maps from the input layer to the hidden layer. After a linear mapping, the signal is projected with a non-linear function:

$$S' = f(W^{(1)}S + b^{(1)}), \tag{9}$$

where $f(\cdot)$ is usually a sigmoid function and S is input. The $\{W, b\}$ are corresponding parameters. Similarly, the decoder oppositely maps from the hidden layer to the output layer. Particularly, output layer and input layer have same number of nodes which is used to reconstruct S:

$$T = f(W^{(2)}S' + b^{(2)}). \tag{10}$$

Then, an optimized parameter set $\theta = \{W^{(1)}, b^{(1)}, W^{(2)}, b^{(2)}\}$ is obtained by minimizing the reconstruction loss. For handling classification tasks, a classifier is also needed to link after the autoencoder.

The classical cost function of auto-encoder is:

$$J_{cost}(\theta) = J_{MSE}(\theta) + J_{weight}(\theta) + J_{sparse}(\theta). \tag{11}$$

The first term $J_{cost}(\theta)$ is an MSE based reconstruction loss function, which is formulated as:

$$J_{MSE}(\theta) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{1}{2}\parallel t_i - s_i \parallel^2\right), \tag{12}$$

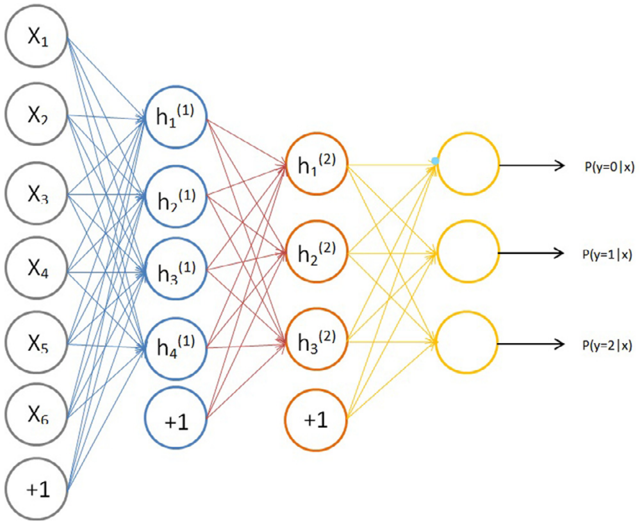

Fig. 2. A SAE with two hidden layers. The two hidden layers are trained layer-wisely. After training, a classifier is connected to the output of the second hidden layer for achieving the classified result.

The second term $J_{weight}(\theta)$ is a weight decay term, which is used to prevent over-fitting:

$$J_{weight}(\theta) = \frac{\xi}{2}\sum_{l=1}^{2}\sum_{i=1}^{s_l}\sum_{j=1}^{s_l+1}(w_{ji}^{(l)})^2 \tag{13}$$

where $w_{ji}^{(l)}$ is the weight between unit i in layer l and unit j in layer $l+1$, ξ is the parameter of weight decay term, and s_l represents the number of units in layer l. In order to avoid over-fitting, the weight decay term keeps the value of weights small.

The last term $J_{sparse}(\theta)$ is a sparsity penalty term, which is defined as follows:

$$J_{sparse}(\theta) = \tau\sum_{i=1}^{s_l}KL(\rho \parallel \hat{\rho}_i), \tag{14}$$

where $KL(\cdot\|\cdot)$ is the Kullback–Leibler (KL) divergence, which is commonly used to measure difference between two distributions. ρ is a small number called sparsity parameter and τ is the weight adjustment parameter. $\hat{\rho}_i$ is the activation value of the ith hidden layer unit . The value of $\hat{\rho}_i$ is constrained by sparsity penalty term to near ρ under KL divergence. The sparsity penalty term can improve robustness of the system.

Stacked autoencoder (SAE) is a deep architecture which uses auto-encoders as the building blocks. Generally, SAE is trained layer-wisely by using back propagation. With flexible deep architecture, SAE can learn the high level representation of the original data by extracting features. In Fig. 2, a SAE with two hidden layers and a classifier is presented. The first layer is the input layer. The second layer and third layer are the hidden layers. The last layer is a classifier, such as SVM based classifier or soft-max based multi-class classifier.

2.3. Network traffic classification

With the explosive increasing number of new Internet applications, the traffic classification technologies have also lived through three levels. In the first level, the port-based technology is used to classify the applications according to the certain port numbers. But, most of the current applications are using dynamic port. In the second level, the payload-based technology identifies the applications according to the payload of IP packets to avoid the problem of dynamic port. However, the encrypted applications make the payload-based technology useless. Hence, the machine learning classification methods based on network flows statistical features are wildly used.

The statistical features of the network flows are a set of feature vectors which can be used to represent the network traffic. All kinds of machine learning methods in classification are applied to deal with this problem, such as Naive Bayes [43], Bayesian Neural Networks [44], Support Vector Machines [45], K-means [46] and DBSCAN [47]. The deep learning methods outperform a lot of traditional, especially in classification tasks. Nevertheless, there are still little relative works in network traffic classification using deep learning. In this paper, we try to use deep learning methods to solve this problem.

Additionally, the noisy data training remains a challenging topic not only in network traffic classification, but also in universal classification problems. Particularly, zero-day traffic will seriously affect the performance of classifiers, because these traffic is mislabeled to a known class and badly influences the identification accuracy of the known classes. Moreover, if the network attack happens, which are flushing currently, are not recognized by the firewall, they will damage the network and create abnormal values in network traffic dataset. These abnormal values can be looked as the outliers, but they are ignored byc most of existing traffic classification methods. In this paper, robust network traffic classification is concerned carefully. It is notable that the features of Moore network traffic dataset are more varied and complex compared with MNIST dataset. As a consequence, the robustness of Correntropy based algorithms is not enough. Motivated by this, GC-SAE is naturally utilized here. With the extra flexibility, GC-SAE oughts to perform better and achieves higher accuracy under corrupted network traffic data.

3. Related work

3.1. C-loss based stacked autoencoders

In [34], CSAE employs a C-loss induced reconstruction loss and a C-loss based sparsity penalty term to replace original terms in autoencoder to improve the robustness under non-Gaussian noises:

$$J_{CSAE}(\theta) = J_{C_{loss}1}(\theta) + J_{weight}(\theta) + J_{C_{loss}2}(\theta). \tag{15}$$

where $J_{C_{loss}1}(\theta) = J_{C_{loss}}(S,T)$ is the C-loss induced reconstruction loss, $J_{C_{loss}2}(\theta) = J_{C_{loss}}(\theta, 0)$ is the C-loss based sparsity penalty term. Through minimizing the C-loss based sparsity penalty term, a sparser representation of the original data can be obtained. The sparser representation means a more essential information of the original data and can't be severely affected by outliers. Thus, the robustness of CSAE can be further improved with a C-loss based sparsity penalty term. The second term $J_{weight}(\theta)$ is not changed.

3.2. Generalized Correntropy

The GC is induced from the generalized Gaussian density(GGD) function. GGD with zero-mean is given by

$$G_{\alpha,\sigma}(e) = \frac{\alpha}{2\beta\Gamma(1/\alpha)}\exp\left(-|\frac{e}{\beta}|^{\alpha}\right) = \gamma_{\alpha,\beta}\exp\left(-\lambda|e|^{\alpha}\right) \tag{16}$$

where $\alpha > 0$ is the shape parameter, $\Gamma(.)$ is the gamma function, $\lambda = 1/\beta^{\alpha}$ is the kernel parameter, and $\gamma_{\alpha,\beta} = \alpha/(2\beta\Gamma(1/\alpha))$ is the normalization constant. When $\alpha \to \infty$, the GGD density converges to a uniform density on $(-\beta, \beta)$ point-wisely. Also, the Laplace $(\alpha = 1)$ and Gaussian $(\alpha = 2)$ distributions are the special cases of this parametric family of symmetric distributions.

In [37], the GGD density function is used as the kernel function of GC. So, the GC is defined as

$$V_{\alpha,\beta}(S,T) = \mathbf{E}[G_{\alpha,\beta}(S-T)] \tag{17}$$

Obviously, the Correntropy is a special case of GC when $\alpha = 2$. Moreover, the kernel function of GC cannot satisfy the Mercer's condition.

Then, the sample estimator of the GC is given by

$$\hat{V}_{\alpha,\beta}(S,T) = \frac{1}{N}\sum_{i=1}^{N} G_{\alpha,\beta}(s_i - t_i) \tag{18}$$

The detailed properties of the GC is depicted in [37].

3.3. GC-loss

As described before [32], C-loss is induced from Correntropy.

Similarly, GC-loss function, which is used as a measure in data analysis such as classification and regression, can be easily obtained from the definition of GC. A GC-loss between S and T is defined as [37]

$$J_{GC_{\text{loss}}}(S,T) = G_{\alpha,\beta}(0) - V_{\alpha,\beta}(S,T). \tag{19}$$

The GC-loss is always non-negative as $J_{GC_{\text{loss}}}(S,T) \geq 0$. When $0 < \alpha \leq 2$, with a nonlinear mapping $\varphi(.)$induced by κ,

$$J_{GC_{\text{loss}}}(S,T) = \frac{1}{2}\mathbf{E}[\|\varphi_{\alpha,\beta}(S) - \varphi_{\alpha,\beta}(T)\|^2] \tag{20}$$

which transforms its argument into a high-dimensional Hilbert space f_κ. Distinctly, minimizing the GC-loss is the same with maximizing the GC.

Assume that samples $(s_i, t_i)_{i=1}^{N}$ are drawn from the joint PDF $p_{ST}(s,\, t)$, then an estimator of the GC-loss can be achieved as

$$\hat{J}_{GC_{\text{loss}}}(S,T) = G_{\alpha,\beta}(0) - \hat{V}_{\alpha,\beta}(S,T) \tag{21}$$

$$= \gamma_{\alpha,\beta} - \frac{1}{N}\sum_{i=1}^{N} G_{\alpha,\beta}(s_i - t_i) \tag{22}$$

$$= \gamma_{\alpha,\beta} - \frac{1}{N}\sum_{i=1}^{N} G_{\alpha,\beta}(e_i) \tag{23}$$

Let $\bar{S} = [s_1, \cdots, s_N]^T, \bar{T} = [t_1, \cdots, t_N]^T$, generalized Correntropy induced metric (GCIM) is formulated as

$$GCIM(\bar{S},\bar{T}) = \sqrt{\hat{J}_{GC_{\text{loss}}}(S,T)}, \tag{24}$$

which defines a metric in the N-dimensional sample vector space when $0 < \alpha \leq 2$.

In Fig. 3, the surface of the GCIM is presented. Clearly, as the error ϵ increased, the behavior of GCIM transforms from the L_α norm (samples correctly classified with high confidence) to the L_1 norm, and then approaches the L_0 norm (misclassified samples) eventually.

Especially, it is feasible for GCIM, which has the additional degrees of freedom GGD provides, to behave like more different norms (from L_∞ to L_0) than a Gaussian based Correntropy (from L_2 to L_0). This extra flexibility can enhance the robustness of GC-loss with a most suitable shape.

Some optimization properties of the GC-loss is presented below [37].

Property 1: Let $\bar{e} = [e_1, \cdots, e_N]^T$. Then the following statements hold:

1) if $0 < \alpha \leq 1$, then the GC-loss $\hat{J}_{GC-loss}$ is concave at any $\bar{e}$ with $e_i \neq 0 (i = 1, \cdots, N)$;
2) if $\alpha > 1$, then the GC-loss $\hat{J}_{GC-loss}$ is convex at any $\bar{e}$ with $0 < |e_i| \leq [(\alpha - 1)/\alpha\lambda]^{1/\alpha} (i = 1, \cdots, N)$;
3) if $\lambda \to 0+$, then for any $\bar{e}$ with $e_i \neq 0 (i = 1, \cdots, N)$, the GC-loss $\hat{J}_{GC-loss}$ is concave at $\bar{e}$ for $0 < \alpha \leq 1$, and convex at $\bar{e}$ for $\alpha > 1$.

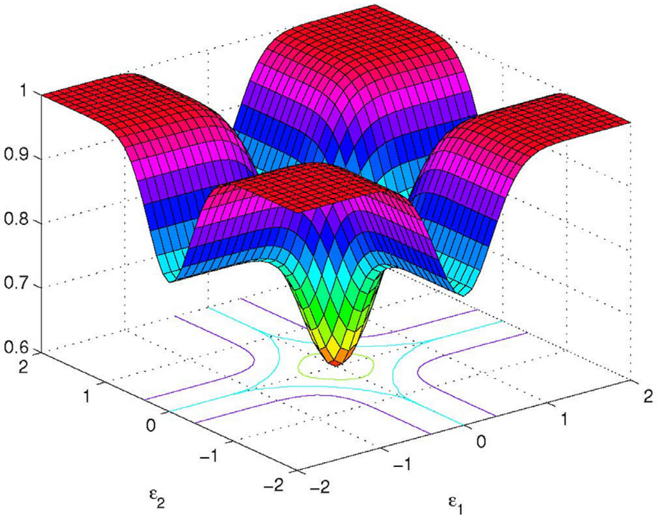

Fig. 3. Surface of the GCIM in 3D space ($\alpha = 4, \sigma = 1$). With the increase of error ϵ, its behavior changes from L_α norm to L_0 norm of the data in different regions.

Table 1
Models with different loss functions.

Abbreviation	Reconstruction loss	Weight penalty term	Sparsity penalty term
SAE	J_{MSE}	J_{weight}	J_{sparse}
CSAE	$J_{C_{\text{loss}}1}$	J_{weight}	$J_{C_{\text{loss}}2}$
GC-SAE	$J_{GC_{\text{loss}}}$	J_{weight}	$J_{C_{\text{loss}}2}$

Property 2: For $\alpha > 1$, the GC-loss $\hat{J}_{GC-loss}$ is a differentiable invex function of $\bar{e} = [e_1, \cdots, e_N]^T$ with $e_i \leq M (i = 1, \cdots, N)$, where M is an arbitrary positive number.

4. Generalized Correntropy based stacked autoencoders

4.1. Description of the GC-SAE

Because of the attractive property detailed above, it is advantageous to combine GC with SAE. Therefore, to further improve the robustness, we modify the cost function of CSAE based on GC-loss. The cost function of GC-SAE is defined as follows:

$$J_{GC-SAE}(\theta) = J_{GC_{\text{loss}}}(\theta) + J_{weight}(\theta) + J_{C_{\text{loss}}}(\theta). \tag{25}$$

The first term $J_{GC_{\text{loss}}}(S,T)$ is a GC-loss induced reconstruction loss. Through replacing the $J_{C_{\text{loss}}1}$ in CSAE with GC-loss based reconstruction loss $J_{GC_{\text{loss}}}$, the robustness can be further improved. By adjusting the shape parameter α, a proper shape can be achieved. With the input keeps same, shapes of reconstruction loss function changes with different gradients. Therefore, a proper shape means a suitable gradient which can maximumly reduce the baleful influence from outliers. Thus, the GC-SAE would be more robust.

The second weight decay term $J_{weight}(\theta)$ keeps same.

The third term is the sparsity penalty term, where the sparsity penalty term in CSAE is continued to use in GC-SAE and formulated as $J_{C_{\text{loss}}}(\theta, 0)$.

The optimal parameter θ is achieved by minimizing the cost function J_{GC-SAE}. The auto-encoders with cost function J_{GC-SAE} are stacked to build the GC-SAE to learn the robust high level representation. GC-SAE is also trained layer-wisely.

The three autoencoder based algorithms (SAE, CSAE and GC-SAE) are shown in Table 1. We use the abbreviations to indicate the algorithms with different loss functions. In Table 1, it is clear that all the three autoencoder based algorithms has three parts in loss functions. Especially, GC-SAE uses J_{GC-SAE} as reconstruction loss. GC-SAE and CSAE are using the same C-loss based sparsity penalty term. The weight penalty term keeps same.

4.2. Notations

We will use the following notations to denote the variables in GC-SAE.

w_{ji}^p: The weight between the unit i in current layer and the unit j in previous layer, at the pth iteration.

a_j^p: The output of the unit j in previous layer, at the pth iteration.

$z_i^p = \sum_j W_{ji}^p a_j^p$: The weighted sum of all output a_j^p of the previous layer, at the pth iteration.

y^p: The true labels of dataset, for the pth iteration.

y_o^p: The predicted labels of dataset in output layer, at the pth iteration.

4.3. Training algorithm

The new cost function equation (25) can be used to train the GC-SAE with gradient descent method. The training procedure is totally same with the traditional SAE. Generally, GC-SAE is trained layer-wisely. Each hidden layer of GC-SAE is trained independently as feature extraction. When previous hidden layer has been trained, its output will used as the input of next hidden layer. Then, all the hidden layers are connected to build a deep neural network for fine-tuning. After fine-tuning, the training of GC-SAE is done and can be applied for tests.

The weights update is computed as follows:

$$w_{ji}^{p+1} = w_{ji}^{p} + \eta \frac{\partial (J_{GC_{\text{loss}}}(y, y_o))}{\partial \epsilon^p} f^{'}(z_i^p) a_j^p \tag{26}$$

where η is the learning rate parameter. The above equation can be written simply as:

$$w_{ji}^{p+1} = w_{ji}^{p} + \eta \delta_i^p a_j^p \tag{27}$$

where ,

$$\delta_i^p = \frac{\partial (J_{GC_{\text{loss}}}(y, y_o))}{\partial \epsilon^p} f^{'}(z_i^p). \tag{28}$$

The computation of δ_i^p in output layer is:

$$\delta_i^p = \delta_o^p = \frac{\partial (J_{GC_{\text{loss}}}(y, y_o))}{\partial \epsilon^p} f^{'}(z_o^p) \tag{29}$$

Then, when computing the δ_i^p of the previous hidden layers, the equation becomes:

$$\delta_i^p = \delta_h^p = f^{'}(z_h^p) \sum_{i=1}^{C_i} \delta_i^p w_{hi}^p \tag{30}$$

where C_i is the number of nodes in the next layer. Then, Eqs. (29) and (30) can be employed to update the GC-loss based model.

The third term is also the C-loss based sparsity penalty term as an L_0 norm approximator. We can optimize the C-loss based sparsity penalty term by gradient descent algorithm as well. So, the gradient of the C-loss based sparsity penalty term is derived as follows:

$$\frac{\partial C_{\text{loss}}(\hat{\rho}_i^p, \mathbf{0})}{\partial \hat{\rho}_i^p} = \frac{\hat{\rho}_i^p}{C_i \sigma^2} exp\left(-\frac{(\hat{\rho}_i^p)^2}{2\sigma^2}\right) \tag{31}$$

Then, this gradient can be used to update the C-loss based sparsity penalty term in GC-SAE.

Finally, by combining GC-loss based reconstruction term and C-loss based sparsity penalty term, the weights update is:

$$w_{ji}^{p+1} = w_{ji}^{p} + \eta \delta_i^p a_j^p - \eta \frac{\hat{\rho}_i^p}{C_i \sigma^2} exp\left(-\frac{(\hat{\rho}_i^p)^2}{2\sigma^2}\right) \tag{32}$$

5. Experimental results

In this section, the comparison of the classification performance of the proposed GC-SAE and other five deep learning algorithms is shown on handwritten benchmark dataset MNIST [38]. We compare the classification accuracy of GC-SAE under various degrees of impulsive noises to present the better robustness of the proposed method. Then, the experiments which illustrate the performance of the GC-SAE with different values of α are also carried out for parameter optimization.

5.1. Dataset

MNIST is a handwritten benchmark dataset which contains 60000 samples of the training set and 10000 samples of the test set. The dataset has ten classes of handwritten images (from 0 to 9). The size of the gray scaled images is 28×28 and normalized to [0, 1].

In order to test the robustness of GC-SAE under outliers and non-Gaussian noises, the mix-Gaussian noise, as an observation noise, is employed to corrupt the original training samples. Following is the description of mix-Gaussian noise:

$$U(x) = (1 - A) * u(0, \upsilon_1^2) + A * u(0, \upsilon_2^2) \tag{33}$$

where $u(0, \upsilon_i^2)(i = 1, 2)$ denote the Gaussian distributions with mean values 0 and υ_i^2 variances. A is the mixture coefficient which controls the degree of noise. The mix-Gaussian noise is centered at 0. The mix-Gaussian noise can randomly generate abrupt large values to model the strong impulsive noises.

5.2. Algorithm settings

The experimental network, which is used in this section, is all constructed with three layers. The input layer has 784 nodes. Two hidden layers have 200 nodes and the output layer has 10 nodes. The parameters are set as: $\xi = 0.005$, $\iota = 0.3$ and $\rho = 0.2$. The kernel size $\sigma 1$ in CSAE is set to 2.2. The sparsity penalty term kernel size $\sigma 2$ of CSAE and GC-SAE is also set to 0.025. For different degrees of noises, the shape parameters α of GC-SAE are experimentally set. Moreover, the structure of H-ELM algorithm is set to 784-200-200-5000-10. The number of hidden nodes is kept same with the autoencoder based algorithms for easy comparison.

All the parameters of the network are randomly initialized and trained layer-wisely (i.e. the hidden layers are trained one by one and use output from the previous layer as the input). All the experiment results are obtained by averaging the values over 10 trials to avoid uncertainty of the random initialization and the noise generation.

5.3. Performance of GC-SAE

To evaluate the performance of extracting features, we compare the classification accuracies of GC-SAE with other five deep learning algorithms under mix-Gaussian noise.

In Table 2, the classification accuracy results with MNIST dataset under different degrees of noises are shown. Obviously, the proposed GC-SAE always shows the highest classification accuracies under different degrees of noises. CSAE and SDA have similar accuracy results. GC-SAE outperforms CSAE and SDA that the classification accuracy increases from 0.65% to 2.15%. When compared with the original SAE, the classification accuracy of GC-SAE is 25.42% higher at most. The classification accuracy curves are shown in Fig. 4. Clearly, in Fig. 4, the curve of GC-SAE decreases slowest as the degree of noise increasing. Hence, it is able to demonstrate that GC-SAE is able to effectively learn the useful features under impulsive noise scenarios.

Table 2
Results of classification accuracies on MNIST under different degrees of mix-Gaussian noises.

Method	$A = 0.05$	$A = 0.1$	$A = 0.15$	$A = 0.2$
GC-SAE	**95.68**	**94.04**	**91.71**	**87.80**
CSAE	94.20	92.11	89.14	83.44
SAE	93.41	88.71	79.64	62.38
SDA	95.03	91.54	89.10	85.65
H-ELM	54.94	40.94	39.79	44.28
MLP-BP	78.88	75.46	73.84	67.03

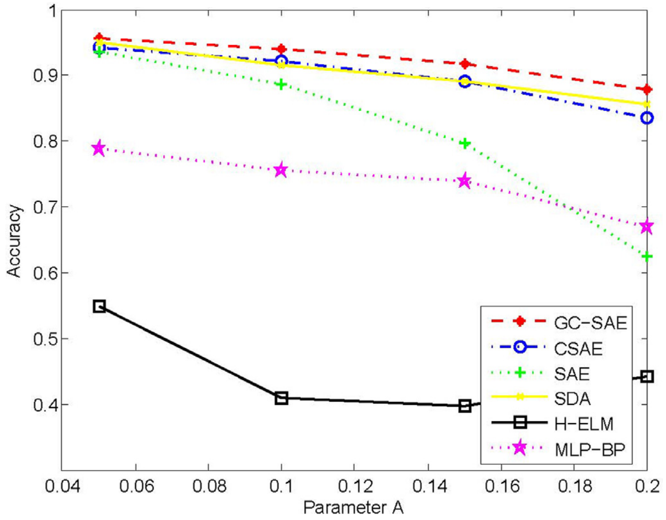

Fig. 4. Comparison of classification accuracies on MNIST under different degrees of mix-Gaussian noise. It is clear that GC-SAE is more robust than others when the noise becomes severer.

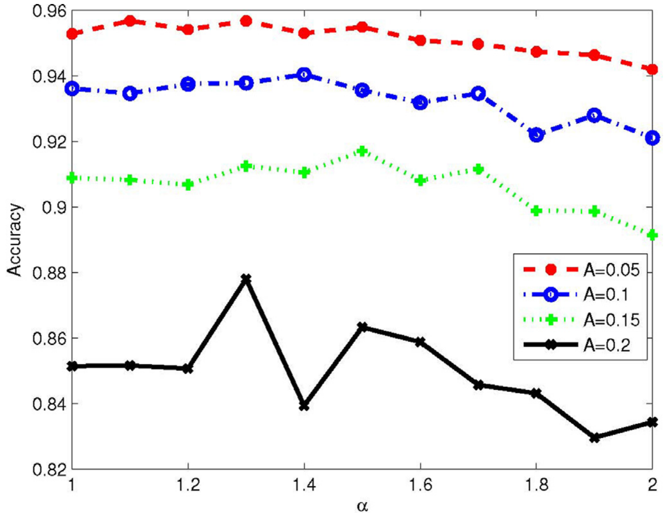

Fig. 5. Classification accuracies of GC-SAE with different values of α, when the noise is increasing.

5.4. Selection of α in GC-SAE

The improvement of GC is mainly reflected on the flexible shape of the reconstruction loss function which is controlled by the shape parameter α. Thus choosing a suitable value of α will markedly improve the performance of GC-SAE. The accuracies of classification with different values of α is shown in Fig. 5. This figure shows that with different noises, the most suitable values of α are different. It is easy to understand that there is only one shape, which is controlled by α, can maximally reduce the bad influence from a given noise.

Table 3
Classification accuracies of GC-SAE with different α.

α	$A = 0.05$	$A = 0.1$	$A = 0.15$	$A = 0.2$
1	95.28	93.64	90.89	85.14
1.1	**95.68**	93.46	90.82	85.16
1.2	95.41	93.75	90.68	85.07
1.3	95.67	93.78	91.25	**87.80**
1.4	95.30	**94.04**	91.05	83.94
1.5	95.49	93.56	**91.71**	86.33
1.6	95.08	93.18	90.81	85.88
1.7	94.97	93.46	91.16	84.57
1.8	94.74	92.20	89.88	84.31
1.9	94.63	92.80	89.87	82.96
2	94.20	92.11	89.14	83.44
2.5	93.47	89.34	84.48	73.19
3	90.99	81.14	61.75	52.18
3.5	85.54	54.69	42.21	22.04
4	78.04	21.58	20.69	17.02

Also, in Table 3, the classification accuracies of GC-SAE with different α are shown. Obviously, one can get a much better performance with a proper α.

6. GC-SAE for robust network traffic classification

Network traffic classification is an enduring problem in computer network management. As the Internet accelerating developing, a better network traffic classification is always needed for well managing the network. In this section, GC-SAE is used to classify the Moore network traffic dataset [43]. Particularly, there are two kinds of noises (mix-Gaussian noise and mislabeled samples) which are employed in the dataset. The GC-SAE method is also compared with other five deep learning algorithms to show the better robustness.

6.1. Dataset

In this section, all the experiments adopt the Moore network traffic dataset which is from the University of Cambridge and is named Moore-set in this paper. In Moore-set, there are 377526 net flow samples which are classified to 12 classes. The statistic information of Moore-set is shown in Table 4. There are 249 features in each samples. Some features are presented in Table 5. The last feature is the label. Obviously, the number of different classes are not averaging. This will make an illusory high classification accuracy. Actually, only the samples of several dominating classes, such as WWW and MAIL, are classified precisely. The rest samples are almost mislabeled. To avoid this problem, we extract part of the dataset to build a relative average data-set and name Moore-a. In Moore-a, except three classes (multimedia, interactive and games), there are 1500 samples for each class in training set. The number of samples in test set of Moore-a are not same. But, totally, there are 13500 samples for training and 9000 samples for test. The detailed number of different classes in Moore-a is shown in Table 6. The samples of the rest three classes are built as a 700-samples mislabeled noise which means all of them are randomly mislabeled with a ”known” label and will be used in the experiments of the mislabel samples corrupted data classification. Particularly, there are only 694 samples. The rest 6 samples are added by random selection of multimedia samples.

6.2. Algorithm settings

In this section, the architectures of GC-SAE, CSAE, SDA, SAE and MLP-BP are same and experimentally set. There are 5 hidden layers

Table 4
Statistic information of Moore-set.

Network flow class	Applications	Number of samples	Percentage (%)
GAMES	Microsoft Direct Play	8	0.0021
P2P	eDonkey, BitTorrent	2094	0.5547
MAIL	IMAP, POP,SMTP	28567	7.5669
WWW	Web browsers	328092	86.9058
INTERACTIVE	SSH,TELNET	110	0.0291
MULTIMEDIA	Windows Media Player,iTunes	576	0.1526
FTP-DATA	FTP, wget	5797	1.5355
SERVICES	X11, DNS, IDENT, LDAP	2099	0.5560
ATTACK	Port scans,worms,viruses	1793	0.4750
FTP-PASV	Skype	2688	0.7120
DATABASE	MySQL, dbase,Oracle	2648	0.7014
FTP-CONTROL	MSN Messenger	3054	0.8090

Table 5
Part of features.

Flow duration
TCP Port
Packet inter-arrival time (mean, variance, . . .)
Payload size (mean, variance, . . .)
Effective Bandwidth based upon entropy
Fourier Transform of the packet inter-arrival time%

Table 6
Statistic information of Moore-a.

Network flow class	Training data	Test data
GAMES	0	0
P2P	1500	594
MAIL	1500	1500
WWW	1500	1500
INTERACTIVE	0	0
MULTIMEDIA	0	0
FTP-DATA	1500	624
SERVICES	1500	599
ATTACK	1500	293
FTP-PASV	1500	1188
DATABASE	1500	1148
FTP-CONTROL	1500	1554

in this architecture and each hidden layer has 90 nodes. Every result is achieved by 3 steps. First, we use original SAE to choose the proper architecture. Then, CSAE is used to select the suitable kernel sizes $\sigma 1$ and $\sigma 2$ under different degrees of noises. Finally, the shape parameter α is also elected to adjust the different degrees of noises. Additionally, the structure of H-ELM is set to 248-100-100-3000-9. The methods are trained layer-wisely as well. Each result is obtained by averaging the values over 10 trials, too.

6.3. Classification results of mix-Gaussian noise corrupted dataset

The realistic network traffic data is gathered and featured from the massive original network flow data. So errors may occur in every step. Particularly, when some network attacks are not recognized and cut off, they will lead to a severe change of some network value. Here, we use mix-Gaussian noise to simulate this kind of noise.

In this section, different degrees of mix-Gaussian noises are employed in Moore-a dataset. The performances of the GC-SAE and others are compared in the Table 7. Obviously, with different degrees of noises, the GC-SAE can always achieve the highest classification accuracies. Even compared with CSAE, GC-SAE is still better about 4%. In Fig. 6, the accuracy curves of different algorithms are illustrated. Of all the cures, GC-SAE curve decreases slowest when the noise becomes severer. So the robust network traffic classification can be obtained through applying GC-SAE. Moreover, GC-SAE has the highest accuracy with the original data which means GC-SAE is effective for network traffic classification.

Table 7
Results of classification accuracies on Moore-a dataset under mix-Gaussian noise.

Method	$A = 0$	$A = 0.05$	$A = 0.1$	$A = 0.15$	$A = 0.2$
GC-SAE	**93.83**	**87.43**	**85.52**	**84.13**	**83.27**
CSAE	89.13	86.22	83.66	80.74	79.24
SAE	92.16	83.79	79.29	76.91	70.43
SDA	82.78	81.63	79.82	78.14	74.77
H-ELM	91.91	65.26	46.92	47.74	42.52
MLP-BP	82.51	80.33	76.01	69.37	63.62

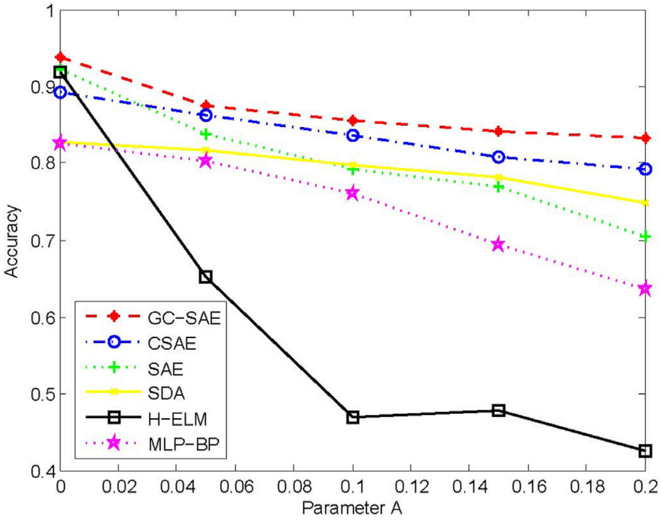

Fig. 6. Comparison of classification accuracies on Moore dataset with mix-Gaussian noise. As the degrees of noise increasing, the accuracy curve of GC-SAE is always higher than others, which demonstrates a better robustness of GC-SAE. Moreover, GC-SAE can achieve around 90% classification accuracies on Moore dataset. So, GC-SAE is suitable for network classification task.

6.4. Classification results of mislabeled sample corrupted dataset

With the swift development of current computer network, the number of applications increases explosively. The traffic which is created by the new applications and named zero-day traffic. In recent years, one third of the total traffic are zero-day traffic. Hence, the zero-day traffic cannot be ignored any more. However, the traditional network traffic classification methods are not flexible or robust. After training, the classes are settled. So, the zero-day traffic will be inevitably classified to wrong classes. These misclassified samples will in turn decrease the classification accuracies of the known classes. Thus, the robustness of the mislabeled samples is very crucial to the network traffic classification methods.

Table 8
Results of classification accuracies on Moore-a dataset with mislabeled samples.

Method	100	200	300	400	500	600	700
GC-SAE	**93.73**	**91.68**	**91.91**	**93.39**	**94.98**	**92.76**	**91.62**
CSAE	90.78	90.98	91.01	92.94	92.93	91.20	81.11
H-ELM	90.67	89.66	85.6	93.37	92.23	92.08	74.67
SAE	91.99	77.10	76.92	88.39	77.11	80.20	66.47
SDA	78.14	76.17	66.71	76.67	84.74	71.87	66.98
MLP-BP	80.66	79.54	81.93	80.31	81.04	82.63	61.02

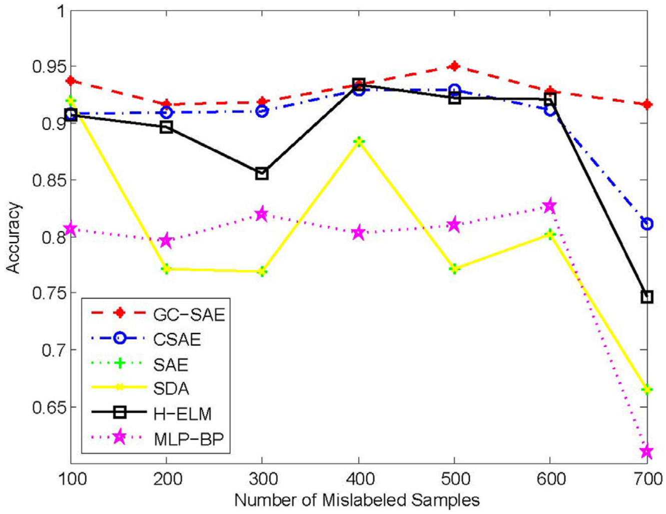

Fig. 7. Classification accuracy curves of GC-SAE and other algorithms with mislabeled samples corrupted Moore-a dataset. The GC-SAE shows a more robust performance than others when the noise becomes severer.

In this section, the mislabeled samples corrupted Moore-a dataset is used to show the robustness of GC-SAE. The samples of games, multimedia and interactive are mislabeled and used as noise. Apparently, the mislabeled samples can also be treated as the normal samples corrupted by outliers.

In Table 8, the experimental results of GC-SAE and other algorithms under different number of mislabeled samples are depicted. The GC-SAE still gets the highest results. But, the gap between GC-SAE and CSAE is smaller than the results under mix-Gaussian noises. In Fig. 7, the accuracy curves of GC-SAE and other five deep learning algorithms with different number of mislabeled samples are also illustrated. Obviously, GC-SAE is more robust than CSAE. But, I have to mention that as the number of mislabeled samples increasing, the accuracy grows up in some cases. In my opinion, the real mislabeled samples are not completely same as the mix-Gaussian noise. Although mislabeled samples would be harmful, they still contain much more useful information than the mix-Gaussian noise. So when the number of mislabeled samples is small, the additional number of training samples is helpful. As the number of mislabeled samples increasing, the bad influence will predominate and then the accuracy decreases very fast. These experiments demonstrate that the GC-SAE has a better robustness when the data includes mislabeled samples.

7. Conclusions

In this work, a new robust deep learning model, named generalized Correntropy (GC) based SAE (GC-SAE), is built by replacing the original reconstruction loss function in CSAE with a GC based loss function. By using the generalized Gaussian density (GGD) function as the kernel, GC is more robust to impulsive noises (outliers). The experimental results on the MNIST benchmark dataset under mix-Gaussian noise are carried out to confirm the better robustness of GC-SAE. The selection of shape parameter α is also illustrated. Moreover, GC-SAE is also applied to handle the network traffic classification tasks. Two kinds of noises are added in Moore network traffic dataset. The results of the mix-Gaussian noise corrupted dataset and mislabeled sample corrupted dataset show better robustness of GC-SAE. Therefore, it is proved that GC-SAE can be used appropriately in tackling the network traffic classification. Current work has been restricted in network traffic classification. The GC-loss can also be used to solve other computer network problems.

Acknowledgments

This work was supported by National High Technology Research and Development Program 863 (no. 2014AA01A701), the Research Fund of Ministry of Education-China Mobile (MCM20150102) and National Natural Science Foundation of China (nos. 61531013, 61372152).

References

[1] J.A. Suykens, J. Vandewalle, Least squares support vector machine classifiers, Neural Process. Lett. 9 (3) (1999) 293–300.
[2] Z. Ma, A. Leijon, Bayesian estimation of beta mixture models with variational inference, IEEE Trans. Pattern Anal. Mach. Intell. 33 (11) (2011) 2160–2173.
[3] T.K. Means, S. Wang, E. Lien, A. Yoshimura, D.T. Golenbock, M.J. Fenton, Human toll-like receptors mediate cellular activation by mycobacterium tuberculosis, J. Immunol. 163 (7) (1999) 3920–3927.
[4] S.R. Safavian, D. Landgrebe, A survey of decision tree classifier methodology, IEEE Trans. Syst. Man Cybern. 21 (3) (1991) 660–674.
[5] Z. Ma, J.H. Xue, A. Leijon, Z.H. Tan, Z. Yang, J. Guo, Decorrelation of neutral vector variables: theory and applications, IEEE Trans. Neural Networks Learn. Syst. PP (2016) 1–15.
[6] Z. Ma, A.E. Teschendorff, A. Leijon, Y. Qiao, H. Zhang, J. Guo, Variational Bayesian matrix factorization for bounded support data, IEEE Trans. Pattern Anal. Mach. Intell. 37 (4) (2015) 876–889.
[7] S. Tong, D. Koller, Support vector machine active learning with applications to text classification, J. Mach. Learn. Res. 2 (2001) 45–66.
[8] Z. Ma, P.K. Rana, J. Taghia, M. Flierl, A. Leijon, Bayesian estimation of Dirichlet mixture model with variational inference, Pattern Recognit. 47 (9) (2014) 3143–3157.
[9] G.E. Hinton, S. Osindero, Y.W. Teh, A fast learning algorithm for deep belief nets, Neural Comput. 18 (7) (2006) 1527–1554.
[10] Y. Bengio, Learning deep architectures for AI, Found. Trends Mach. Learn. 2 (1) (2009) 1–127.
[11] H. Larochelle, D. Erhan, A. Courville, J. Bergstra, Y. Bengio, An empirical evaluation of deep architectures on problems with many factors of variation, in: Proceedings of the 24th International Conference on Machine Learning, ACM, 2007, pp. 473–480.
[12] Y. Freund, D. Haussler, Unsupervised Learning of Distributions of Binary Vectors Using Two Layer Networks, 1994.
[13] Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle, Greedy layer-wise training of deep networks, Adv. Neural Inf. Process. Syst. 19 (2007) 153.
[14] C. Poultney, S. Chopra, Y.L. Cun, Efficient learning of sparse representations with an energy-based model, in: Advances in Neural Information Processing Systems, 2006, pp. 1137–1144.
[15] P.Y. Simard, D. Steinkraus, J.C. Platt, Best practices for convolutional neural networks applied to visual document analysis, in: ICDAR, 3, 2003, pp. 958–962.
[16] S. Ji, W. Xu, M. Yang, K. Yu, 3D convolutional neural networks for human action recognition, IEEE Trans. Pattern Anal. Mach. Intell. 35 (1) (2013) 221–231.
[17] M. LukoEvilus, H. Jaeger, Reservoir computing approaches to recurrent neural network training, Comput. Sci. Rev. 3 (3) (2009) 127–149.
[18] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436–444.
[19] W. Liu, P.P. Pokharel, J.C. Principe, Correntropy: a localized similarity measure, in: The 2006 IEEE International Joint Conference on Neural Network Proceedings, IEEE, 2006, pp. 4919–4924.
[20] J.C. Principe, D. Xu, J. Fisher, Information theoretic learning, Unsupervised Adaptive Filtering, 1, 2000, pp. 265–319.
[21] W. Liu, P.P. Pokharel, J.C. Principe, Correntropy: properties and applications in non-Gaussian signal processing, IEEE Trans. Signal Process. 55 (11) (2007) 5286–5298.
[22] R. He, W.S. Zheng, B.G. Hu, X.W. Kong, A regularized correntropy framework for robust pattern recognition, Neural Comput. 23 (8) (2011) 2074–2100.
[23] S. Zhao, B. Chen, J.C. Principe, Kernel adaptive filtering with maximum correntropy criterion, in: The 2011 International Joint Conference on Neural Networks (IJCNN), IEEE, 2011, pp. 2012–2017.
[24] B. Chen, L. Xing, J. Liang, N. Zheng, J.C. Principe, Steady-state mean-square error analysis for adaptive filtering under the maximum correntropy criterion, IEEE Signal Process. Lett. 21 (7) (2014) 880–884.

[25] Z. Wu, S. Peng, B. Chen, H. Zhao, Robust Hammerstein adaptive filtering under maximum correntropy criterion, Entropy 17 (10) (2015) 7149–7166.
[26] A. Gunduz, J.C. Principe, Correntropy as a novel measure for nonlinearity tests, Signal Process. 89 (1) (2009) 14–23.
[27] B. Chen, J. Wang, H. Zhao, N. Zheng, J.C. Principe, Convergence of a fixed-point algorithm under maximum correntropy criterion, IEEE Signal Process. Lett. 22 (10) (2015) 1723–1727.
[28] B. Chen, L. Xing, J. Liang, N. Zheng, J.C. Principe, Steady-state mean-square error analysis for adaptive filtering under the maximum correntropy criterion, IEEE Signal Process. Lett. 21 (7) (2014) 880–884.
[29] B. Chen, J.C. Principe, Maximum correntropy estimation is a smoothed MAP estimation, IEEE Signal Process. Lett. 19 (8) (2012) 491–494.
[30] S. Seth, J.C. Principe, Compressed signal reconstruction using the correntropy induced metric, in: 2008 IEEE International Conference on Acoustics, Speech and Signal Processing, IEEE, 2008, pp. 3845–3848.
[31] A. Singh, J.C. Principe, A Loss Function for Classification based on a Robust Similarity Metric, 2010, pp. 1–6.
[32] A. Singh, R. Pokharel, J. Principe, The c-loss function for pattern classification, Pattern Recognit. 47 (1) (2014) 441–453.
[33] Y. Qi, Y. Wang, X. Zheng, Z. Wu, Robust feature learning by stacked autoencoder with maximum correntropy criterion, in: 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2014, pp. 6716–6720.
[34] L. Chen, H. Qu, J. Zhao, B. Chen, J.C. Principe, Efficient and robust deep learning with correntropy-induced loss function, Neural Computing Appl. 27 (4) (2016) 1019–1031.
[35] W. Ma, H. Qu, G. Gui, L. Xu, J. Zhao, B. Chen, Maximum correntropy criterion based sparse adaptive filtering algorithms for robust channel estimation under non-Gaussian environments, J. Franklin Inst. 352 (7) (2015) 2708–2727.
[36] B. Chen, X. Liu, H. Zhao, J.C. Principe, Maximum correntropy Kalman filter, Automatica 76 (2017) 70–77.
[37] B. Chen, L. Xing, H. Zhao, N. Zheng, J.C. Principe, Generalized correntropy for robust adaptive filtering, IEEE Trans. Signal Process. 64 (13) (2016) 3376–3387.
[38] Y. Lcun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, in: Proceedings of the IEEE.
[39] T.T. Nguyen, G. Armitage, A survey of techniques for internet traffic classification using machine learning, IEEE Commun. Surv. Tutor. 10 (4) (2008) 56–76.
[40] Y. Xiang, W. Zhou, M. Guo, Flexible deterministic packet marking: an IP traceback system to find the real source of attacks, IEEE Trans. Parallel Distrib. Syst. 20 (4) (2009) 567–580. 2476–2490.
[41] J. Zhang, X. Chen, Y. Xiang, W. Zhou, J. Wu, Robust network traffic classification, IEEE/ACM Trans. Networking 23 (4) (2015) 1257–1270. 86(11), 2278–2324, 1998.
[42] L. Chen, H. Qu, J. Zhao, Generalized correntropy induced loss function for deep learning, in: 2016 International Joint Conference on Neural Networks (IJCNN), IEEE, 2016, pp. 1428–1433.
[43] A.W. Moore, D. Zuev, Internet traffic classification using Bayesian analysis techniques, in: ACM SIGMETRICS Performance Evaluation Review, Vol. 33, ACM, 2005, pp. 50–60.
[44] T. Auld, A.W. Moore, S.F. Gull, Bayesian neural networks for internet traffic classification, IEEE Trans. Neural Networks 18 (1) (2007) 223–239.
[45] A. Este, F. Gringoli, L. Salgarelli, Support vector machines for TCP traffic classification, Computer Networks 53 (14) (2009) 2476–2490.
[46] J. Eerman, A. Mahanti, M. Arlitt, Internet traffic identification using machine learning techniques, in: Proceedings of the 49th IEEE GLOBECOM, San Francisco, 2006.
[47] J. Erman, M. Arlitt, A. Mahanti, Traffic classification using clustering algorithms, in: Proceedings of the 2006 SIGCOMM Workshop on Mining Network Data, ACM, 2006, pp. 281–286.

Liangjun Chen received the bachelor degree from the Xi'an Jiaotong University in 2012, and is currently pursuing the Ph.D. degree in computer science at Xi'an Jiaotong University. His research interests include network traffic classification, deep learning and Information Theoretical Learning.

Hua Qu is a professor of Xi'an Jiaotong University, China. He received his B.A. degree from Nanjing University of Posts and Telecommunications, China, and his Ph.D. degree from Xi'an Jiaotong University. His research interests include mobile Internet, IP based network, network management and control, radio resource management in LTE-A system etc. He is a senior member of China Institute of Communications and also an editor of China Communications magazine.

Jihong Zhao is a professor of Xi'an Jiaotong University, China. She received her B.A. degree from Huazhong University of Science and Technology, China, and her Ph.D. degree in computer science from Xi'an Jiaotong University. Her current research is in broadband communication network, management and control of new generation network and machine learning for network management.