The publication is available at https://doi.org/10.1007/978-3-319-68612-7_15. Cite as:
K. Gajowniczek, L. J. Chmielewski, A. Orłowski, and T. Ząbkowski. Generalized entropy cost function in neural networks. In A. Lintas, S. Rovetta, P.F.M.J. Verschure, and A.E.P. Villa, editors, Artificial Neural Networks and Machine Learning – ICANN 2017. Proc. 26th Int. Conf. on Artificial Neural Networks ICANN 2017, Part II, volume 10614 of Lecture Notes in Computer Science, pages 128–136, Alghero, Sardinia, Italy, 11-14 Sep 2017. Springer, Cham. doi:10.1007/978-3-319-68612-7_15.

Generalized Entropy Cost Function in Neural Networks

Krzysztof Gajowniczek, Leszek J Chmielewski, Arkadiusz Orłowski,
Tomasz Ząbkowski

Faculty of Applied Informatics and Mathematics – WZIM
Warsaw University of Life Sciences – SGGW
Nowoursynowska 159, 02-787 Warsaw, Poland
krzysztof_gajowniczek@sggw.pl

Abstract. Artificial neural networks are capable of constructing complex decision boundaries and over the recent years they have been widely used in many practical applications ranging from business to medical diagnosis and technical problems. A large number of error functions have been proposed in the literature to achieve a better predictive power. However, only a few works employ Tsallis statistics, which has successfully been applied in other fields. This paper undertakes the effort to examine the q-generalized function based on Tsallis statistics as an alternative error measure in neural networks. The results indicate that Tsallis entropy error function can be successfully applied in the neural networks yielding satisfactory results.

Keywords: Neural networks, Tsallis entropy error function, Classification.

1 Introduction and problem statement

Artificial neural networks (ANNs) are flexible and powerful statistical learning models used in many applications. They have been extensively and successfully applied in areas such as signal processing, pattern recognition, machine learning, system control, and many business problems including marketing and finance [1-5]. Several features of artificial neural networks make them very popular and attractive for practical applications. First, they possess an ability to generalize, even in the case of incomplete or noisy data. Second, neural networks are non-parametric which means that they do not require any a-priori assumptions about the distribution of the data. Third, they are good approximators able to model continuous function to a desired accuracy.

From a pattern recognition perspective, the goal is to find the required mapping from input to output variables in order to solve the classification or the regression problem. The main issue in neural networks application is to find the correct values for the

A. Lintas et al. (Eds.): ICANN 2017, Part II, LNCS 10614, pp. 128–136, 2017.
https://doi.org/10.1007/978-3-319-68612-7_15

weights between the input and output layer using a supervised learning paradigm (training). During the training process the difference between the prediction made by the network and the correct value for the output is calculated, and the weights are changed in order to minimize the error.

The form of the error function is one of the factors in the weight update process. For the successful application it is important to train the network with an error function that reflects the objective of the problem. The mean square error (MSE) is the most commonly used function although it has been suggested that it is not necessarily the best function to be used, especially in classification problems [6-8]. A number of alternative error functions have been proposed in the literature and the maximum likelihood (cross entropy) function was particularly reported as a more appropriate function for classification problems [7, 8].

In this paper we undertake the effort to examine an alternative error function such as the q-generalized function based on Tsallis statistics. In particular the properties of the function and its impact on the neural network classifiers is analyzed as well as a careful analysis of the way in which the error function is introduced in the weight update equations is presented. To the best of our knowledge the proposed error function was never examined before in the context of the neural network learning.

The rest of this paper is organized as follows. In the second section the literature review on similar problems is presented. An analysis of the way the error function is incorporated in the training algorithm is presented in section three. The fourth section deals with the experiments carried out and their results are presented. The paper ends with concluding remarks in the last section.

2 Literature review on similar problems

The research on neural networks is considerable and the literature around this filed is growing rapidly. While the method becomes a more and more substantial part of the state-of-the-art automatic pattern recognition systems applicable in a variety of fields, different questions arise considering the network architecture and the fundamentals of training process.

Usually, the works include modifications and improvements of the neural network structure, weights initialization [9], weights updating procedure [10], error functions [11], [12] and activation functions [13], [14]. The training of artificial neural networks usually requires that users define an error measure in order to adapt the network weights to meet certain model performance criteria. The error measure is very important and in certain circumstances it is essential for achieving satisfactory results. Different error measures have been used to train feedforward artificial neural networks, with the mean-square error measure (and its modifications) and cross-entropy being the most popular ones.

It can be shown that the true posterior probability is reaching a global minimum for both the cross-entropy and squared error criteria. Thus, in the theory an ANN can be trained equally well by minimizing each of the functions, as long as it is capable of approximating the true posterior distribution arbitrarily close. When it comes to the modelling of distribution, squared error is bounded and the optimization is therefore

more robust to outliers than minimization of cross-entropy. However, in practice, cross-entropy mostly leads to quicker convergence resulting better quality in terms of classification error rates. Hence, squared error became less popular over the last years [8, 15]. In the literature, the previous works on the error functions have usually been evaluated on rather small datasets.

When it comes to applications under the nonextensive statistics with Tsallis distributions, called q-distributions, formed by maximizing Tsallis entropy with certain constraints – such distributions have applications in physics, astronomy, geology, chemistry and finance [16]. However, these q-distributions remain largely unnoticed by the computer science audience, with only a few works applying them to ANNs, not necessary as the error functions [17]. For instance, [17] introduces q-generalized RNNs (random neural network) for classification where parametrized q-Gaussian distributions are used as activation functions. These distributions arise from maximizing Tsallis entropy and have a continuous real parameter q – the entropic index – which represents the degree of nonextensivity.

In this paper, in order to address the identified literature gap, we present an investigation on the properties of the q-entropic error criteria for training of ANNs. The theoretical analysis of the error bounds was supported by experimental evaluation with properly trained networks taking into account classification accuracy measures.

3 Theoretical framework

As a general measure of diversity of objects, a Shannon entropy is often used which is defined as:

$$H_S = -\sum_{i=1}^{n} t_i \log t_i, \tag{1}$$

where t_i is the probability of occurrence of an event x_i being an element of the event X that can take values $x_i, \dots, x_n$. The value of the entropy depends on two parameters: (1) disorder (uncertainty) and it is maximum when the probability t_i for every x_i is equal; (2) the value of n. Shannon entropy assumes a tradeoff between contributions from the main mass of the distribution and the tail. To control both parameters a generalizations was proposed by Tsallis:

$$H_{T_q} = \frac{1}{q-1}\left(1 - \sum_{i=1}^{n} t_i^q\right). \tag{2}$$

With Shannon entropy, events with high or low probability have equal weights in the entropy computation. However, using Tsallis entropy, for $q > 1$, events with high probability contribute more to the entropy value than those with low probabilities. Therefore, the higher is the value of q, the higher is the contribution of high probability events in the final result.

It can be shown that q-Tsallis relative entropy is a generalization of the Kullback-Leibler entropy (in the limit of $q \to 1$ the q-Tsallis relative entropy becomes the Kullback-Leibler entropy). It refers to two probability distributions $\{t_i\}$ and $\{y_i\}$, $i = 1$ to n, over the same alphabet, and is defined as [18]:

$$H_{T_q}(t_i||y_i) = \frac{1}{1-q}\left(1 - \sum_{i=1}^{n} t_i^q y_i^{1-q}\right). \quad (3)$$

At the limit $q = 1$, one has $H_{T_1}(t_i||y_i) = \sum_{i=1}^{n} t_i \log(t_i/y_i)$, i.e., the Kullback-Leibler relative entropy [19]. For any order $q \neq 0$, the Tsallis relative entropy $H_{T_q}(t_i||y_i)$ of the above equation (3) vanishes if and only if $t_i = y_i$ for all $i = 1$ to n. For any $q > 0$, the Tsallis relative entropy $H_{T_q}(t_i||y_i)$ is always nonnegative. In this regime of $q > 0$, the Tsallis relative entropy $H_{T_q}(t_i||y_i)$ behaves much like the conventional Kullback-Leibler relative entropy, yet in a generalized form realized by the additional parameterization by q.

In the present paper we have used a variant of gradient descent method known as Broyden-Fletcher-Goldfarb-Shanno (BFGS) optimization algorithm [20], in order to train a neural network (multilayer perceptron). To make use of BFGS, the function being minimized should have an objective function that accepts a vector of parameters, input data, output data, and should return both the cost and the gradients. In these circumstances the cost function which implements the Tsallis entropy is defined as:

$$C(t, y) = \frac{1}{1-q}(1 - t^q y^{1-q} - (1-t)^q (1-y)^{1-q}), \quad (4)$$

where t and y stand for the true value and output of the neural network, respectively.

Fig. 1 shows the general behavior of the error function for different values of parameter q.

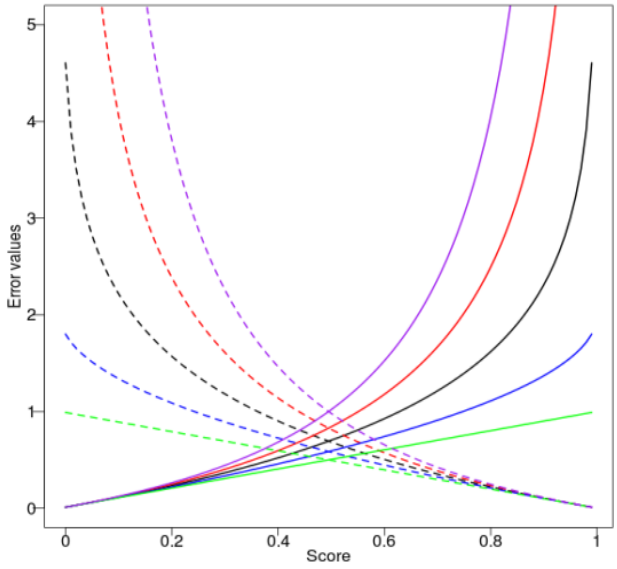

Fig. 1. Error entropy values for different q values in terms of different score values. Color denotes the value of the parametr q as follows: 0 – green, 0.5 – blue, 1 – black, 1.5 – red, 2 – purple. The difference between the solid and dashed lines is explained in the main text.

The solid lines show entropy values for the growing score (outcome of ANN) in case if the true value is 0. It could be clearly seen that if the score tends to 1 the error function increases; in other words, this is an undesirable situation for a given case. In contrast, the dashed lines show entropy values for the growing score in case if the true value is 1. Once again, entropy values increase when the ANN network outcome is inconsistent with the true value.

4 Numerical experiment

4.1 Implementation

In our case, all the numerical calculations were performed on a personal computer with the following parameters: Ubuntu 16.04 LTS operating system and Intel Core i5-2430M 2.4 GHz, 2 CPU * 2 cores, 8 GB RAM. R-CRAN [21], which is an advanced statistical package, as well as an interpreted programming language, was used as the computing environment. For training neural networks we used the BFGS algorithm, available in the *nnet* library [21]. A logistic function was used to activate all of the neurons in the neural network and initial weights vector was chosen randomly using a uniform distribution.

To compare the neural networks obtained for different values of q we define two measures. These are: (1) AUC (area under the ROC curve) and (2) classification accuracy. Those measures are related to efficiency and effectiveness of the ANN and they have been often used for evaluation of classification models in the context of various practical problems such as credit scoring, income and poverty determinants or customer insolvency and churn [22, 23].

The starting point for the numerical experiments was the randomly selected split of the examined datasets into two parts, which corresponded to the training and validation with the following proportions: training 70%, validation 30%. The main criterion taken into account while learning the models was to gain good generalization of knowledge with the least error. The most commonly used measure to assess the quality of binary classification problem is AUC. Therefore, to find the best parameters for all models and to assure their generalization, the following function was maximized:

$$\mathrm{f}(\mathrm{AUC_T}, \mathrm{AUC_W}) = -\frac{1}{2}|\mathrm{AUC_T} - \mathrm{AUC_W}| + \frac{1}{2}\mathrm{AUC_W}, \tag{5}$$

where $\mathrm{AUC_T}$ and $\mathrm{AUC_W}$ stand for the training and validation errors, respectively.

In contrast to other machine learning algorithms, ANN required special preparation of the input data. The vector of continuous variables were standardized, while the binary variables were converted such that the value of 0 was transformed into -1.

Each time, 15 neural networks were learned with various parameters (the number of neurons in the hidden layer from 1 to 15). To avoid overfitting, after the completion of each learning iteration (with a maximum of 50 iterations), the models were checked for the error measure defined in equation (5). At the end, the ANN characterized by the smallest error was chosen as the best model. In order to achieve robust estimation of models' error, for each number of hidden neurons, ten different ANN were learned with

different initial weights vector. Final estimation of the error was computed as the average value over ten models and for each number of hidden neurons.

4.2 Results

Our research was conducted on several benchmarking data sets which are freely available. However, due to limited room, results for two datasets only are shown. We conducted the simulations using the dataset known as Churn (3333 observations 19 predictors [24]) and Hepatitis [25] (155 observations and 19 predictors). Moreover, only relevant results related to the best performance in terms of equation 5 are discussed. For both datasets the networks with at least 10 hidden units delivered the robust results as provided in Tables 1 and 2.

Table 1. The results for the Churn dataset.

q-value	Number of hidden neurons	Avg number of iterations	Training sample		Validation sample		AUC Equation No 5
			Accuracy	AUC	Accuracy	AUC	
1.0	10	20.3	0.904	0.895	0.888	0.874	0.852
1.2	10	14.8	0.913	0.906	0.899	0.886	0.867
1.4	10	13.5	0.906	0.893	0.898	0.880	0.867
1.0	11	17.7	0.921	0.915	0.904	0.891	0.868
1.2	11	14.8	0.922	0.912	0.905	0.891	0.869
1.4	11	14.9	0.909	0.896	0.901	0.884	0.871
1.0	12	18.3	0.919	0.909	0.903	0.891	0.872
1.2	12	15.0	0.913	0.906	0.901	0.894	0.882
1.4	12	16.2	0.900	0.904	0.902	0.891	0.878

Table 2. The results for the Hepatitis dataset.

q-value	Number of hidden neurons	Avg number of iteration	Training sample		Validation sample		AUC Equation No 5
			Accuracy	AUC	Accuracy	AUC	
1.0	11	5.4	0.773	0.841	0.762	0.837	0.832
1.2	11	4.1	0.772	0.838	0.755	0.835	0.832
1.4	11	4.7	0.774	0.837	0.758	0.834	0.829
1.0	12	6.0	0.778	0.844	0.756	0.837	0.830
1.2	12	5.5	0.775	0.841	0.756	0.836	0.829
1.4	12	5.1	0.770	0.836	0.754	0.833	0.827
1.0	13	4.6	0.769	0.838	0.748	0.832	0.825
1.2	13	4.3	0.771	0.836	0.746	0.830	0.824
1.4	13	4.1	0.768	0.835	0.745	0.831	0.826

In particular, the results can be summarized as follows:

- The best results were obtained for q-value equal to 1.0, 1.2 and 1.4; significant drop in classification accuracy was in general observed for $q > 2$ (due to the limit of pages and the goal of the article set for the best results, data not shown).
- Behavior of error function with q-parameter greater than 1 is more non-linier (see Fig. 1) than behavior of error function based on Shannon entropy ($q \approx 1$);
- ANN learned with $q > 1$ required less iteration to achieve convergence in terms of Eq. (5);
- Overall performance depends on number of input variables (input neurons) and number of hidden neurons; The choice of the proper network structure should be based on solid experiments, since this may lead to unwanted effects influencing the stability and performance of the training algorithm and the trained network as a whole.

5 Summary and concluding remarks

The results of this study indicate that in classification problems, Tsallis entropy error function can be successfully applied in the neural networks yielding satisfactory results in terms of the number of iterations required for training, and the generalization ability of the trained network.

The contribution of this study provides the proof that the q-entropy can substitute other standard entropic error functions like the Shannon's one with satisfactory results, leading to less epochs and delivering the same percentage of correct classifications. The choice of the error function is indeed an important factor to be examined with great care when designing a neural network for a specific classification problem.

Possible future research on this topic could consider two streams. Firstly, comparative study on the impact of various error functions, including mean square error and the mean absolute error, used for various classification problems [26-28], should be made. Secondly, the effect of the proposed error functions on other types of neural network architectures, including application on a variety of real datasets, should be studied.

References

1. Paliwal, M, Kumar, UA: Neural networks and statistical techniques: A review of applications, Expert Systems with Applications 36(1), 2–17 (2009)
2. Bishop CM: Neural Networks for Pattern Recognition Oxford, UK: Clarendon Press, (1995)
3. Szupiluk, R, Wojewnik, P, Ząbkowski, T: Prediction improvement via smooth component analysis and neural network mixing, In Kollias, S, Stafylopatis, A, Duch, W, Oja E (eds), LCNS, Theoretical Computer Science and General Issues Springer, pp 133-140, Springer, Heidelberg (2006)
4. Gajowniczek, K, Ząbkowski, T, Data Mining Techniques for Detecting Household Characteristics based on Smart Meter Data, Energies 8(7), 7407–7427 (2015)

5. Ząbkowski, T, Szczesny, W: Insolvency modeling in the cellular telecommunication industry, Expert Systems with Applications 39, 6879–6886 (2012)
6. Kalman, BL, Kwasny, SC: A Superior Error Function for Training Neural Networks, In: International Joint Conference on Neural Networks, pp 49–52 (1991)
7. White, H: Artificial Neural Networks: Approximation and Learning Theory, Blackwell, Cambridge (1992)
8. Golik, P, Doetsch, P, Ney, H: Cross-entropy vs squared error training: a theoretical and experimental comparison, In: 14th Annual Conference of the International Speech Communication Association "Interspeech-2013", pp 1756–1760, France (2013)
9. Waghmare, LM, Bidwai, NN, Bhogle, PP: Neural network weight initialization, In: Proc. IEEE International Conference on Mechatronics and Automation, pp 679–681, (2007)
10. Ramos, EZ, Nakakuni, M, Yfantis, E: Quantitative measures to evaluate neural network weight initialization strategies, In: IEEE Computing and Communication Workshop and Conference (CCWC), 1–7, (2017)
11. Falas, T, Stafylopatis, AG: The impact of the error function selection in neural network-based classifiers, In: International joint conference on neural networks, pp 1799–1804, (1999)
12. Shamsuddin, SM, Sulaiman, MN, Darus, M: An improved error signal for the backpropagation model for classification problems, International Journal of Computer Mathematics 76(3), 297–305 (2001)
13. Narayan, S: The generalized sigmoid activation function: competitive supervised learning, Information sciences 99(1-2), 69–82 (1997)
14. Kamruzzaman, J, Aziz, SM: A note on activation function in multilayer feedforward learning, In: Proceedings of the 2002 International Joint Conference on Neural Networks IJCNN'02, 519–523, (2002)
15. Kline, DM, Berardi, VL: Revisiting squared-error and cross-entropy functions for training neural network classifiers, Neural Computing and Applications 14(4), 310–318 (2005)
16. Picoli, S, Mendes, RS, Malacarne, LC, Santos, RPB: Q-distributions in complex systems: A brief review, Brazilian J Phys 39(2A), 468–474 (2009)
17. Stosic, D, Stosic, D, Zanchettin, C, Ludermir, T, Stosic, B: QRNN: q-generalized random neural network, IEEE Transactions on Neural Networks and Learning Systems 28(2), 383–390 (2016)
18. Tsallis, C: Introduction to Nonextensive Statistical Mechanics, Springer, New York (2009)
19. Cover, TM, Thomas, JA: Elements of Information Theory, Wiley, New York (1991)
20. Dai, YH: Convergence properties of the BFGS algorithm, SIAM Journal on Optimization, 13(3), 693–701 (2002)
21. R Core Team: A language and environment for statistical computing, R Foundation for Statistical Computing, Vienna, Austria, (2015)
22. Gajowniczek, K, Ząbkowski, T, Szupiluk, R: Estimating the ROC curve and its significance for classification models' assessment, Quantitative Methods in Economics 15(2), 382–391 (2014)
23. Chrzanowska, M, Alfaro, E, Witkowska, D: The Individual Borrowers Recognition: Single and Ensemble Trees, Expert Systems with Applications 36(3), 6409–6414 (2009)
24. Churn dataset, http://www.dataminingconsultant.com/DKD.htm, last accessed 2017/01/12
25. Hepatitis dataset, https://archive.ics.uci.edu/ml/datasets/Hepatitis, last accessed 2017/01/12

26. Gajowniczek, K, Ząbkowski, T, Orłowski, A: Entropy Based Trees to Support Decision Making for Customer Churn Management, Acta Physica Polonica A 129(5), 971–979 (2016)
27. Gajowniczek, K, Karpio, K, Łukasiewicz, P, Orłowski, A, Ząbkowski, T: Q-entropy approach to selecting high income households, Acta Physica Polonica A127(3A), 38–44 (2015)
28. Gajowniczek, K, Ząbkowski, T, Orłowski, A: Comparison of Decision Trees with Renyi and Tsallis Entropy Applied for Imbalanced Churn Dataset, Annals of Computer Science and Information Systems 5, 39–43 (2015)