Physica A 534 (2019) 122168

Contents lists available at ScienceDirect

Physica A

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

Radial basis function network using Lambert–Tsallis W_q function

J.L.M. da Silva, F.V. Mendes, R.V. Ramos *

Lab. of Quantum Information Technology, Department of Teleinformatic Engineering – Federal University of Ceara - DETI/UFC, C.P. 6007 – Campus do Pici, 60455-970 Fortaleza-Ce, Brazil

HIGHLIGHTS

- Lambert–Tsallis W_q function.
- Disentropy and quantum relative disentropy.
- Radial basis function network.
- Probability density function estimation.

ARTICLE INFO

Article history:
Received 18 April 2019
Received in revised form 29 July 2019
Available online 31 July 2019

Keywords:
Lambert–Tsallis W_q function
Quantum relative disentropy
Radial basis function network

ABSTRACT

The present work brings two applications of the Lambert–Tsallis W_q function in radial basis function networks (RBFN). Initially, an RBFN is used to discriminate between entangled and disentangled bipartite of qubit states. The kernel used is based on the Lambert–Tsallis W_q function for $q \in \{1/2, 3/2, 2\}$ and the quantum relative disentropy is used as a distance measure between quantum states. Entangled states with concurrence larger than 0.1 were correctly classified by the proposed RBFN in at least 97% of the cases. Following, a RBFN with the same kernel is used to estimate the probability density function of a set of data sampled according to Normal and Cauchy distributions.

1. Introduction

The Lambert W function has found several applications in physics and engineering, providing analytical solutions where traditionally numerical methods were used [1–3]. Its generalization, the recently proposed Lambert–Tsallis W_q function [4] was used to define the generalized disentropy [4]. The classical and quantum disentropy have been used, respectively, in classical and quantum information theory [5]. In the present work we show that W_q can be used to construct a useful radial basis function that can be successfully used in a radial basis function network (RBFN). Two RBFN were constructed. The first one is a classifier trained to identify entangled and disentangled bipartite of qubit states. It uses the quantum relative disentropy [5] as distance measure between quantum states. The Wootters' concurrence [6] is used to measure the error rate of the RBFN. Following, a second RBFN is used to estimate the probability density function (PDF) of a set of data samples. The test of this RBFN was done considering two situations: data samples from normal and Cauchy distributions.

This work is outlined as follows: In Section 2, a brief review of RBFN, Lambert–Tsallis function and disentropy is provided; In Section 3 the proposed classifier is introduced and the numerical results are shown. In Section 4 the proposed PDF estimator is presented and the numerical results are shown. At last, the conclusions are drawn in Section 5.

* Corresponding author.
E-mail addresses: Jorgemouta95@gmail.com (J.L.M. da Silva), fernandovm@gmail.com (F.V. Mendes), rubens.ramos@ufc.br (R.V. Ramos).

https://doi.org/10.1016/j.physa.2019.122168
0378-4371/

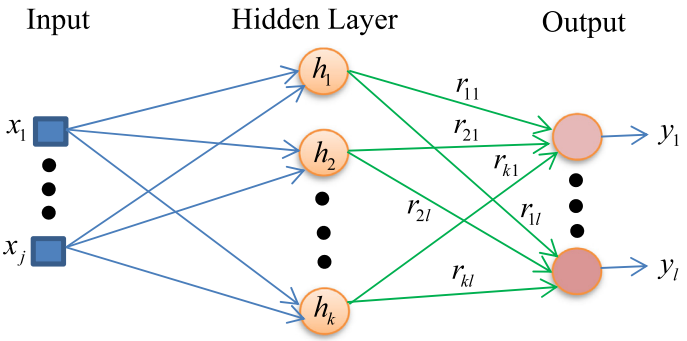

Fig. 1. RBFN with one hidden layer.

2. Radial basis function network, Lambert–Tsallis W_q function and disentropy

Radial basis function network can successfully be used to realize function approximation and pattern recognition, for example. Its basic structure with only one hidden layer is shown in Fig. 1.

The t-th output, y_t, is given by

$$y_t\left(\vec{x}\right) = \sum_{n=1}^{k} r_{nt} h_n \left[d\left(\vec{x}, \vec{x}_n^c\right); \delta_n; c_n\right]. \tag{1}$$

In (1), the input variable $\vec{\boldsymbol{x}}$ is a function of the input data $\{x_1,\ldots, x_j\}$, h_n is the radial basis function of the nth neuron. It has three parameters: $\vec{\boldsymbol{x}}_{\boldsymbol{n}}^{\boldsymbol{c}}$ (the center), δ_n (the width) and c_n (the type of RBF used). The value of h_n is maximum when the distance given by the function $d(\cdot)$ between its center and the input variable is zero and it decreases when the distance increases. Functions like Gaussian, q-Gaussian and Cauchy function have been used as RBF [7,8]. The distance function d can be, for example, an Euclidean distance or an entropic distance. A RBFN can use different RBFs, in this case the parameter c_n identifies the type of RBF used. At last, r_{nt} are the coefficients found during the training stage. The training consists of finding out the best values of $\vec{\boldsymbol{x}}_{\boldsymbol{n}}^{\boldsymbol{c}}$, δ_n, c_n, r_{nt} and the number of neurons in the hidden layer, such that the error during the training using known examples is minimal. The size and quality of the training set are also important for a good performance of the classifier. It is common to use a genetic algorithm or another heuristic to train the RBFN.

The RBF proposed in this work is

$$h\left[d\left(x, x_c\right); \delta; q\right] = \frac{C_1^q}{1 + W_q\left[\delta d\left(x, x_c\right)\right]} - C_2^q, \tag{2}$$

where C_1^q and C_2^q are parameters used to normalize the function h and W_q is the Lambert–Tsallis function that was introduced in [4]. Basically, it is the function that solves the equation

$$W_q\left(z\right) e_q^{W_q(z)} = z. \tag{3}$$

In (3) q is the Tsallis non-extensivity parameter [9] and $e_q(x)$ is the q-exponential. When $q = 1$, one has $e_q(x) = e^x$ and, hence, $W_{q=1}$ is the famous Lambert W function. For example, for $q \in \{1/2, 3/2, 2\}$ one has

$$W_{1/2}(x) = \frac{\left[3\sqrt[3]{2x + \sqrt{\left(2x + \frac{8}{27}\right)^2 - \frac{64}{729}}} + \frac{8}{27} - 2\right]^2}{9\sqrt[3]{2x + \sqrt{\left(2x + \frac{8}{27}\right)^2 - \frac{64}{729}}} + \frac{8}{27}} \quad \text{for } x > -0.2950 \tag{4}$$

$$W_{3/2}\left(x\right) = \frac{2\left((x+1) - \sqrt{2x+1}\right)}{x} \quad \text{for } x > -1/2 \tag{5}$$

$$W_2\left(x\right) = \frac{x}{x+1} \quad \text{for } x > -1. \tag{6}$$

The respective radial basis functions are

$$h\left(\delta d\left(x, x_c\right); q = 1\right) = 1 \Big/ \left[1 + W\left(\delta d\left(x, x_c\right)\right)\right] \tag{7}$$

$$h\left(\delta d\left(x, x_c\right); q = 1/2\right) = 1 \Big/ \left[1 + W_{1/2}\left(\delta d\left(x, x_c\right)\right)\right] \tag{8}$$

$$h\left(\delta d\left(x, x_c\right); q = 3/2\right) = \frac{3/2}{1 + W_{3/2}\left[\delta d\left(x, x_c\right)\right]} - \frac{1}{2} \tag{9}$$

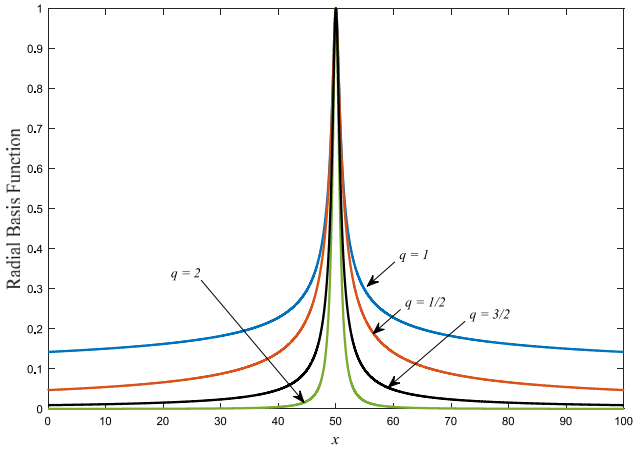

Fig. 2. Radial basis functions (7)–(10) for $\delta = 1$ and $d = (x - 50)^2$.

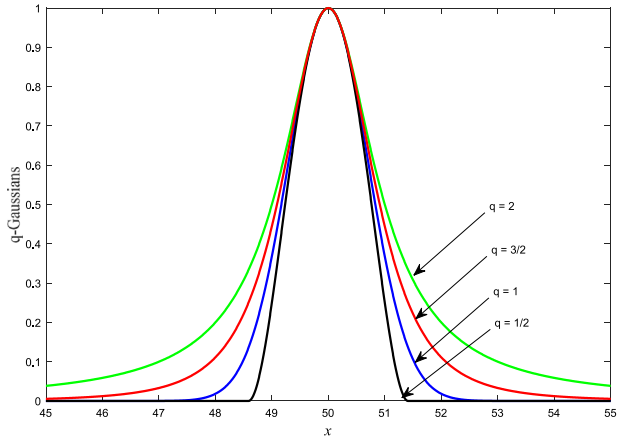

Fig. 3. q-Gaussians as RBFs for $q \in \{1/2, 1, 3/2, 2\}$.

$$h\left(\delta d\left(x, x_c\right); q = 2\right) = \frac{2}{1 + W_2\left[\delta d\left(x, x_c\right)\right]} - 1. \tag{10}$$

In Fig. 2 it is shown the plots of (7)–(10) versus x for $\delta = 1$ and $d = (x - 50)^2$.

As one can see in Fig. 2, The RBFs using the Lambert–Tsallis functions are sharply peaked functions. It is interesting to compare them with RBFs having rounded peak, like the q-Gaussians [7]. The q-Gaussian function is given by

$$e_q^{-x^2} = \begin{cases} e^{-x^2} & q = 1 \\ \left[1 - (1-q)\, x^2\right]^{1/(1-q)} & q \neq 1 \,\&\, 1 + (1-q)\, z \geq 0. \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

The plot of q-Gaussians for $q \in \{1/2, 1, 3/2, 2\}$ can be seen in Fig. 3.

The quantum disentropy, by its turn, is defined as [4]

$$D_q\left(\rho\right) = \sum_n \lambda_n^q W_q\left(\lambda_n\right), \tag{12}$$

where λ_i is the ith eigenvalue of the density matrix ρ. The quantum relative disentropy, by its turn, is given by [5]

$$D_q^R\left(\rho \| \Gamma\right) = \sum_n \lambda_n^q \left| W_q\left(\lambda_n\right) - W_q\left(\gamma_n\right)\right|. \tag{13}$$

where λ_n and γ_n are, respectively, the eigenvalues of the density matrices ρ and Γ.

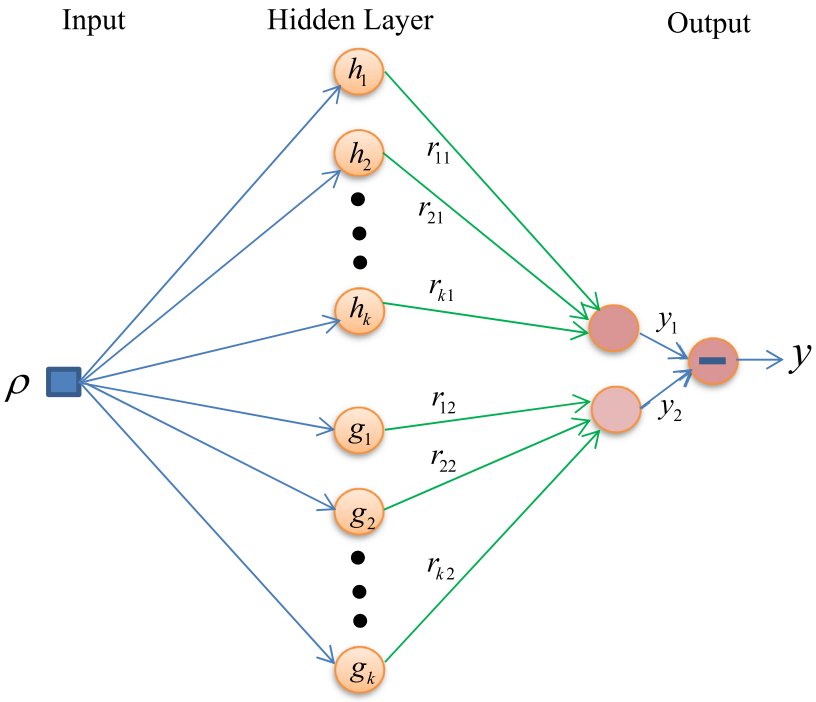

Fig. 4. RBFN used to discriminate between entangled and disentangled two-qubit states.

At last, the Wootters' concurrence (an entanglement measure) of the bipartite state ρ, is given by $C(\rho) = \max\{0, \lambda_1\text{-}\lambda_2\text{-}\lambda_3\text{-}\lambda_4\}$, where $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \lambda_4$ are the eigenvalues of the Hermitean matrix $R = \sqrt{\sqrt{\rho}\left(\sigma_y \otimes \sigma_y\right)\rho^*\left(\sigma_y \otimes \sigma_y\right)\sqrt{\rho}}$. The star denotes the complex conjugate and σ_y is the spin–flip Pauli gate.

3. The classifier of $C_2 \otimes C_2$ quantum states using RBFN with Lambert–Tsallis function

Since the goal is to show that W_q can be used to construct a useful radial basis function, initially we chose to implement a classifier able to discriminate between entangled and disentangled two-qubit states. This problem is easily solved using Wootter's concurrence [6], hence we used it in order to measure the error rate of the RBFN implemented. The classifier proposed has one input (the input quantum state), one hidden layer and a single output value. It is shown in Fig. 4.

The distance function used is the quantum relative disentropy with $q = 2$,

$$d\left(\rho, \Gamma_n^c\right) = D_2\left(\rho \| \Gamma_n^c\right). \tag{14}$$

In (14) Γ_n^c are the 'central' quantum states. Using (14) and (2) in (1) one gets the equations that describe the proposed classifier

$$y_1(\rho) = \sum_{n=1}^{k} r_{n1} h_n\left[\delta_{n1} D_2^2\left(\rho \| \Gamma_n^c\right); q\right] = \sum_{n=1}^{k} \frac{r_{n1} C_1^q}{1 + W_q\left[\delta_{n1} D_2^2\left(\rho \| \Gamma_n^c\right)\right]} - C_2^q \tag{15}$$

$$y_2(\rho) = \sum_{n=1}^{k} r_{n2} h_n\left[\delta_{n2} D_2^2\left(\rho \| \Phi_n^c\right); q\right] = \sum_{n=1}^{k} \frac{r_{n2} C_1^q}{1 + W_q\left[\delta_{n2} D_2^2\left(\rho \| \Phi_n^c\right)\right]} - C_2^q \tag{16}$$

$$y(\rho) = y_1(\rho) - y_2(\rho). \tag{17}$$

The central states $\mathbf{\Gamma_n^c}$ are entangled while $\mathbf{\Phi_n^c}$ are disentangled states. The input state ρ is considered entangled (disentangled) by the RBFN if $y(\rho) > 0$ ($y(\rho) \leq 0$). The heuristic used to training the RBFN (to find the values of $\mathbf{\Gamma_n^c}$, $\mathbf{\Phi_n^c}$, r_{n1}, r_{n2}, δ_{n1} and δ_{n2} that minimizes the error rate) was the *Differential Evolution* (*DE*), since the solutions are directly coded in real values instead of binary strings. Initially, we used $q = 2$ and 20 neurons in the hidden layer, $k = 10$ in (15)–(16). The initial training set was composed by 2500 entangled states and 2500 disentangled states chosen randomly. The *DE* algorithm used a population with 20 individuals and crossover and mutation rates equal to 0.75 and 0.25, respectively. The training took 1000 generations. After the training, we used a test set with 1 000,000 states (500,000 entangled and 500,000 disentangled) chosen randomly and the classifier discriminated them correctly in 89.42% of the cases. The correct

Table 1
Training sets. C is the Wootter's concurrence.

Training set – 5000 states randomly chosen	
Str_0	2500 disentangled states and 2500 entangled states without any restriction.
Str_1	2500 disentangled states and 2500 entangled states with $C \leq 0.1$.
Str_2	2500 disentangled states and 2500 entangled states with: $C \leq 0.1$ (75%) and $0.1 < C \leq 0.2$ (25%).
Str_3	2500 disentangled states and 2500 entangled states with: $C \leq 0.1$ (50%) and $0.1 < C \leq 0.2$ (50%).
Str_4	2500 disentangled states and 2500 entangled states with: $C \leq 0.1$ (25%) and $0.1 < C \leq 0.2$ (75%).
Str_5	2500 disentangled states and 2500 entangled states with: $C \leq 0.1$ (25%); $0.1 < C \leq 0.2$ (25%); $0.2 < C \leq 0.3$ (25%); $0.3 < C \leq 0.4$ (25%).

Table 2
Test sets.

Test set – 1,000,000 states randomly chosen	
$Stst_1$	Entangled and disentangled states without any restriction
$Stst_2$	Only disentangled states
$Stst_3$	Only entangled states
$Stst_4$	Only entangled states with $C \leq 0.1$
$Stst_5$	Only entangled states with $0.1 < C \leq 0.2$
$Stst_6$	Only entangled states with $0.2 < C \leq 0.3$
$Stst_7$	Only entangled states with $0.3 < C \leq 0.4$
$Stst_8$	Only entangled states with $0.4 < C \leq 0.5$
$Stst_9$	Only entangled states with $0.5 < C \leq 0.6$

Table 3
Success rate when the training and test sets used are Str_i and $Stst_j$.

$q = 2$	$Stst_1$	$Stst_2$	$Stst_3$	$Stst_4$	$Stst_5$	$Stst_6$	$Stst_7$	$Stst_8$	$Stst_9$
Str_0	87%	85.31%	86.93%	75.56%	99.65%	100%	100%	100%	100%
Str_1	**89.42%**	75.69%	**89.34%**	**87.80%**	**100%**	100%	100%	100%	100%
Str_2	87.60%	83.69%	87.51%	78.06%	99.99%	100%	100%	100%	100%
Str_3	86.06%	86.07%	85.98%	72.62%	99.92%	100%	100%	100%	100%
Str_4	84.01%	89.08%	84%	65.80%	99.50%	100%	100%	100%	100%
Str_5	81.86%	**91.76%**	81.82%	59.45%	97.70%	100%	100%	100%	100%

Table 4
Success rate when the training and test sets used are $Str_{0,1}$ and $Stst_j$, $j = 1,\ldots,9$, for the classifier using $q = 1/2$ and $q = 3/2$.

	$Stst_1$	$Stst_2$	$Stst_3$	$Stst_4$	$Stst_5$	$Stst_6$	$Stst_7$	$Stst_8$	$Stst_9$
Str_0 $q = 1/2$	86.59%	**86.14%**	86.55%	74.23%	99.62%	100%	100%	100%	100%
Str_1 $q = 1/2$	**89.55%**	76.14%	**89.45%**	**87.77%**	100%	100%	100%	100%	100%
Str_0 $q = 3/2$	87.52%	83.81%	87.43%	77.78%	99.99%	100%	100%	100%	100%
Str_1 $q = 3/2$	89.05%	78.82%	88.97%	84.88%	100%	100%	100%	100%	100%

discrimination is more complicated when the input is an entangled state with a very low value of entanglement. In order to check the performance of the classifier in this region, we considered the scenarios described in Tables 1 and 2:

The success rates for the different scenarios using $q = 2$ are shown in Table 3.

Table 4 shows the results of the classifier using $q = 1/2$ and $q = 3/2$. Here, only two training sets were used Str_0 e Str_1.

As it can be noted in Tables 3 and 4, entangled states with concurrence larger than 0.2 are always correctly discriminated by the RBFN. Moreover, there is no statistically significant change between the different values of q used.

At last, a classifier using q-Gaussians was also simulated. The equations that describe the q-Gaussian-based classifier are

$$y_1(\rho) = \sum_{n=1}^{k} r_{n1} e_q^{-\delta_{n1} D_2^2\left(\rho \| \Gamma_n^c\right)} = \sum_{n=1}^{k} r_{n1} \left[1 - (1-q)\,\delta_{n1} D_2^2\left(\rho \| \Gamma_n^c\right)\right]_+^{1/(1-q)} \tag{18}$$

$$y_2(\rho) = \sum_{n=1}^{k} r_{n2} e_q^{-\delta_{n2} D_2^2\left(\rho \| \Phi_n^c\right)} = \sum_{n=1}^{k} r_{n2} \left[1 - (1-q)\,\delta_{n2} D_2^2\left(\rho \| \Phi_n^c\right)\right]_+^{1/(1-q)} \tag{19}$$

$$y(\rho) = y_1(\rho) - y_2(\rho)\,. \tag{20}$$

In (18)–(19) $[A]_+ = \max\{A,0\}$. Table 5 shows the results of the q-Gaussian-based classifier for $q \in \{1/2, 3/2, 2\}$.

Comparing Tables 3–5, one can see a slightly better performance of the classifier using the Lambert–Tsallis function, however, based only on these results, it is not possible to claim that in general sharp peaked RBFs show better performance than RBFs having rounded peak.

Table 5
Success rate when the training and test sets used are $Str_{0,1}$ and $Stst_j$, $j = 1,\ldots,9$, for the q-Gaussian-based classifier.

	$Stst_1$	$Stst_2$	$Stst_3$	$Stst_4$	$Stst_5$	$Stst_6$	$Stst_7$	$Stst_8$	$Stst_9$
Str_0 $q = 2$	87.03%	85.38%	86.98%	75.69%	99.81%	100%	100%	100%	100%
Str_1 $q = 2$	**89.07%**	78.41%	**88.99%**	**85.18%**	100%	100%	100%	100%	100%
Str_0 $q = 1/2$	85.08%	**88.10%**	85.05%	69.22%	99.66%	99.90%	99.76%	99.41%	98.58%
Str_1 $q = 1/2$	88.73%	80.28%	88.63%	83.09%	100%	100%	100%	100%	100%
Str_0 $q = 3/2$	87.26%	84.86%	87.17%	76.48%	99.96%	100%	100%	100%	100%
Str_1 $q = 3/2$	88.99%	78.49%	88.92%	84.92%	100%	100%	100%	100%	100%

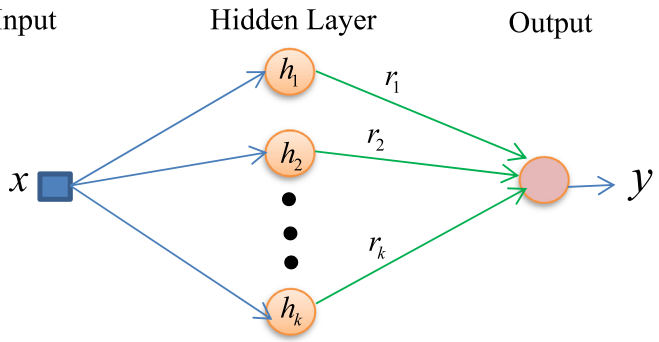

Fig. 5. RBFN for PDF estimation.

4. Probability density function estimator using RBFN and W_2

Sometimes one is interested in the determination of the probability density function of a physical process, like in quantum state tomography [10]. In this cases only the data sample whose probability density function is not known in advance are available. Hence, one has to estimate the PDF using the available set of data. There are several PDF estimators in the literature [11]. The easiest way to estimate the PDF from available data is to construct the histogram. However, this estimation is coarse. Here, we propose a RBFN for PDF estimation using W_2 in the kernel. The proposed RBFN is shown in Fig. 5.

The RBFN output is

$$y(x) = \sum_{n=1}^{k} r_n \left[\frac{2}{1 + W_2\left[\delta_n \left(x - x_n^c\right)^2\right]} - 1 \right]. \tag{21}$$

In order to get the parameters r_n and x_n^c, we firstly construct the histogram with number of bins much smaller than the number of samples. The parameter r_n is the relative frequency and x_n^c is the center of the nth bin. The values of δ_n depends on the real PDF and, therefore, it is the variable to be optimized. We chose to use the same width for all neurons in order to avoid a numerical optimization (as the one used to training the RBFN in Section 3). The width was varied linearly in a fixed range. The best width is the one that minimizes the absolute value of the difference between the log-likelihood of the initial histogram and the log-likelihood of the PDF estimated by the RBFN. The number of neurons in the hidden layer is equal to the number of bins of the histogram. The first test was realized with a normal distribution

$$p(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}. \tag{22}$$

Twenty thousand numbers were randomly chosen according to the PDF in Eq. (22) and a histogram with 2000 bins was constructed, hence, there were two thousand neurons in the hidden layer. The real and estimated PDF's can be seen in Fig. 6. The value of δ_n used was 30 for all neurons in the hidden layer.

A second estimation was also realized. Ten thousand numbers were randomly chosen according to the PDF in Eq. (22) and a histogram with 200 bins was constructed, hence, there were two hundred neurons in the hidden layer. Two RBFN were tested. The RBFN I used Eq. (10) as RBF while RBFN II used the q-Gaussian with $q = 2$ as RBF. The real and estimated PDF's can be seen in Fig. 7. The value of δ_n used was 50 for all neurons in the hidden layer.

The third test uses a standard Cauchy distribution:

$$p(x) = 1/\left[\pi\left(1 + x^2\right)\right]. \tag{23}$$

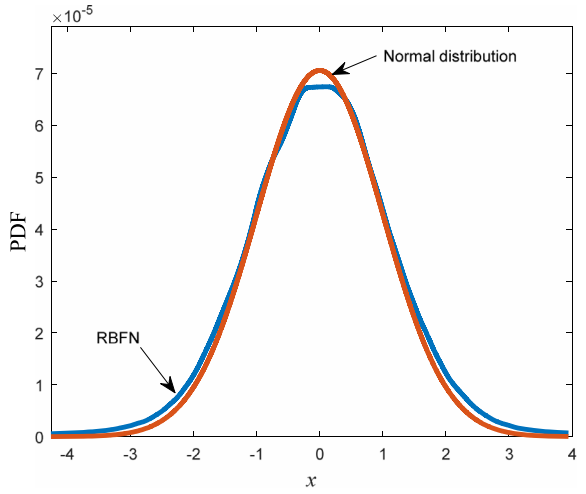

Fig. 6. Normal distribution and its estimation using a RBFN with $q = 2$, twenty thousand samples and 2000 neurons.

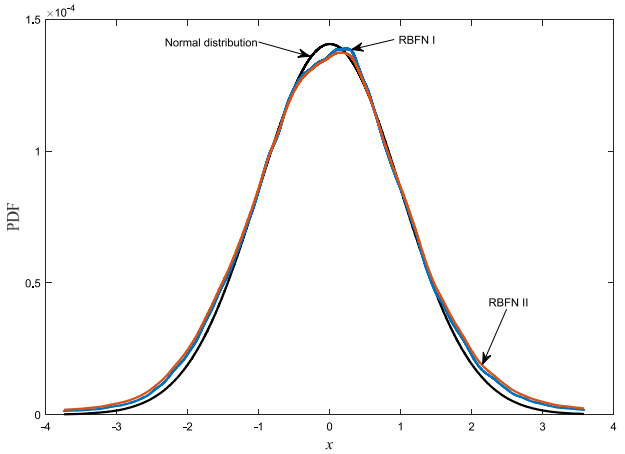

Fig. 7. Normal distribution and its estimations using the RBFN I (Lambert–Tsallis) and the RBFN II (q-Gaussian). Ten thousand samples and two hundred neurons were used.

Ten thousand numbers were randomly chosen according to the PDF in Eq. (23) and a histogram with 2000 bins was constructed. The real and estimated PDF's can be seen in Fig. 8. The value of δ_n used was 0.5 for all neurons in the hidden layer.

As one can note, in all cases the PDF estimation was not perfect but it was good enough for, for example, to provide a good estimation of the entropy or disentropy of the data sample.

The problem of estimating a PDF using a histogram as start point is, in some sense, a function fitting task. Let us consider, now, the Lambert–Tsallis function for $q = 1/2$ given by Eq. (4). In a defined range, using the same topology shown in Fig. 5, its normalized version $W_{1/2}/\max(W_{1/2})$ can be well fitted by

$$y(x) = \sum_{n=1}^{k} \frac{W_{1/2}\left(x_n^c\right)}{W_{1/2}\left(x_k^c\right)} \left[\frac{2}{1 + W_2\left[\delta_n \left(x - x_n^c\right)^2\right]} - 1 \right] \tag{24}$$

in the range $\left[x_1^c, x_k^c\right]$. Fig. 9 shows the plot of $W_{1/2}(x)$ and $y(x)$ given by (24) in the range [0, 98.99]. The plot in Fig. 9 has 20,000 points and we used 1000 neurons ($k = 1000$ in (24)) with $\delta_n = 50.15$ for all of them.

5. Conclusions

The good performances of the RBFNs presented show that the proposed kernel based on the Lambert–Tsallis W_q function, Eq. (2), is useful and it can be used in RBFNs. Here, we were restricted to $q \in \{1/2, 3/2, 2\}$ because of their

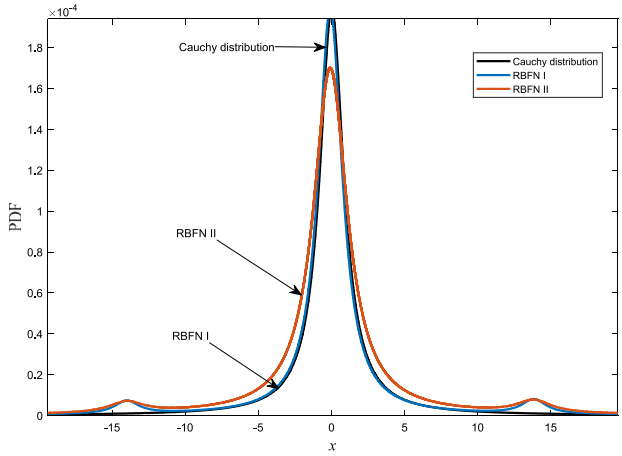

Fig. 8. Cauchy distribution and its estimations by RBFN I (Lambert–Tsallis) and an RBFN II (q-Gaussian). Ten thousand samples and two thousand neurons were used.

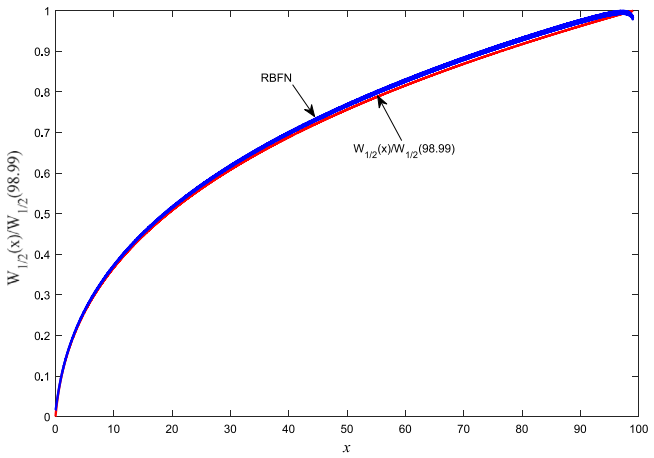

Fig. 9. $W_{1/2}(x)$ and its approximation using a RBFN (Eq. (24)) versus x.

easy and fast numerical calculations, but other values of q can also be tested using the numerical procedure shown in [4] for calculation of the Lambert–Tsallis function. In conclusion, the high rate success of the classifier also shows that the quantum relative disentropy is a useful measure of the distance between quantum states.

Acknowledgments

The authors gratefully acknowledge many helpful discussions with Guilherme Barreto and Francesco Corona of Federal University of Ceara. This study was financed in part by the Brazilian agencies CAPES - Finance Code 001, and CNPq via Grant no. 307184/2018-8. Also, this work was performed as part of the Brazilian National Institute of Science and Technology for Quantum Information.

References

[1] R.M. Corless, G.H. Gonnet, D.E.G. Hare, D.J. Jeffrey, D.E. Knuth, On the Lambert W function, Adv. Comput. Math. 5 (1996) 329–359.
[2] S.R. Valluri, D.J. Jeffrey, R.M. Corless, Some applications of the Lambert W function to physics, Can. J. Phys. 78 (9) (2000) 823–831.
[3] D.C. Jenn, Applications of the Lambert W function in electromagnetics, IEEE Antennas Propag. Mag. 44 (3) (2002).
[4] G.B. da Silva, R.V. Ramos, The Lambert-Tsallis W_q function, Physica A 525 (2019) 164–170.
[5] R.V. Ramos, Quantum and classical information theory with disentropy, arXiv:1901.04331, 2019.
[6] W.K. Wootters, Entanglement of formation of an arbitrary state of two qubits, Phys. Rev. Lett. 80 (1998) 2245.
[7] R. Tinós, L.O.M. Júnior, Use of the q-Gaussian function in radial basis function networks, in: A. Abraham, A.E. Hassanien, V. Snášel (Eds.), Foundations of Computational Intelligence Volume 5, in: Studies in Computational Intelligence, vol. 205, Springer, Berlin, Heidelberg, 2009.
[8] F. Fernández-Navarroa, C. Hervás-Martíneza, M. Cruz-Ramíreza, P.A. Gutiérreza, A. Valerob, Evolutionary q-Gaussian radial basis function neural network to determine the microbial growth/no growth interface of Staphylococcus aureus, Appl. Soft Comput. 11 (2011) 3012–3020.

[9] C. Tsallis, Possible generalization of Boltzmann–Gibbs statistics, J. Stat. Phys. 52 (1988) 479.
[10] J.L.E. Silva, S. Glancy, H.M. Vasconcelos, Quadrature histograms in maximum-likelihood quantum state tomography, Phys. Rev. A 98 (2018) 022325/1–7.
[11] J. Beirlant, E.J. Dudewicz, L. Gyorfi, E.C. van der Meulen, Onparametric entropy estimation: An overview, Int. J. Math. Stat. Sci. 6 (1997) 17–39.