Neuro-Fuzzy Modelling Based on Kolmogorov's Superposition: a New Tool for Prediction and Classification

Vitaliy Kolodyazhniy[1], Peter Otto[2]

[1]Control Systems Research Laboratory
Kharkiv National University of Radioelectronics
14, Lenin Av., Kharkiv, 61166, Ukraine
kolodyazhniy@ukr.net

[2]Department of Informatics and Automation
Technical University of Ilmenau,
PF 100565, 98684 Ilmenau, Germany
peter.otto@tu-ilmenau.de

Abstract: In the paper, a novel *Neuro-Fuzzy Kolmogorov's Network* (NFKN) is considered. The NFKN is based on the famous Kolmogorov's superposition theorem (KST). The network consists of two layers of neo-fuzzy neurons (NFNs) and is linear in both the hidden and output layer parameters, so it can be trained with very fast and simple procedures without any nonlinear operations. The validity of theoretical results and the advantages of the NFKN are confirmed by application examples: electric load forecasting, and classification of data from medical and banking domains.

1 Introduction

According to the Kolmogorov's superposition theorem (KST) [Kol57], *any* continuous function of d variables can be *exactly* represented by superposition of continuous functions of one variable and addition:

$$f(x_1,\ldots,x_d) = \sum_{l=1}^{2d+1} g_l\left[\sum_{i=1}^{d} \psi_{l,i}(x_i)\right], \tag{1}$$

where $x \in [x_1^{min}, x_1^{max}] \times \ldots \times [x_d^{min}, x_d^{max}]$, $g_l(\bullet)$ and $\psi_{l,i}(\bullet)$ are some continuous univariate functions, and $\psi_{l,i}(\bullet)$ are independent of *f*. Aside from the exact representation, the KST can be used as the basis for the construction of parsimonious universal approximators, and has thus attracted the attention of many researchers in the field of soft computing. Hecht-Nielsen was the first to propose a neural network implementation of KST [Hec87], but did not consider how such a network can be constructed. Computational aspects of approximate version of KST were studied by Sprecher [Spr96], [Spr97] and Kůrková [Kur91]. Yam *et al* [YNK99] proposed the multi-resolution approach to fuzzy control, based on the KST, and proved that the KST representation can be realized by a two-stage rule base, but did not demonstrate how such a rule base could be created from data. Lopez-Gomez and Hirota developed the Fuzzy Functional Link Network (FFLN) [LYH02] based on the fuzzy extension of the Kolmogorov's theorem. The FFLN is trained via fuzzy delta rule, whose convergence can be quite slow. A novel KST-based universal approximator called Fuzzy Kolmogorov's Network (FKN) with simple structure and training procedure with high rate of convergence was proposed in [KB04, KBO04]. The training of the FKN is based on the alternating linear least squares technique for both the output and hidden layers.

In this paper we consider a modification of the FKN, called *Neuro-Fuzzy Kolmogorov's Network* (NFKN), in which inputs can have variable number of membership functions. This provides more flexibility for the NFKN model and enables it to deal with both numerical and categorical variables. We also propose an efficient and computationally simple learning algorithm, whose complexity depends linearly on the dimensionality of the input space. The algorithm is a combination of the gradient descent procedure for the tuning of the hidden layer weights, and linear least squares method for the output layer.

2 Network Architecture

The NFKN (Fig. 1) is comprised of two layers of neo-fuzzy neurons (NFNs, Fig. 2) [Yam92] and is described by the following equations:

$$\hat{f}(x_1,\dots,x_d)=\sum_{l=1}^{n} f_l^{[2]}(o^{[1,l]}), \quad o^{[1,l]}=\sum_{i=1}^{d} f_i^{[1,l]}(x_i), \quad l=1,\dots,n, \tag{2}$$

where n is the number of hidden layer neurons, $f_l^{[2]}(o^{[1,l]})$ is the l-th nonlinear synapse in the output layer, $o^{[1,l]}$ is the output of the l-th NFN in the hidden layer, $f_i^{[1,l]}(x_i)$ is the i-th nonlinear synapse of the l-th NFN in the hidden layer.

The equations for the hidden and output layer synapses are

$$f_i^{[1,l]}(x_i)=\sum_{h=1}^{m_{1,i}} \mu_{i,h}^{[1]}(x_i) w_{i,h}^{[1,l]}, \quad f_l^{[2]}(o^{[1,l]})=\sum_{j=1}^{m_{2,l}} \mu_{l,j}^{[2]}(o^{[1,l]}) w_{l,j}^{[2]}, \quad l=1,\dots,n, \quad i=1,\dots,d, \tag{3}$$

where $m_{1,i}$ and $m_{2,l}$ is the number of membership functions (MFs) per input in the hidden and output layers respectively, $\mu_{i,h}^{[1]}(x_i)$ and $\mu_{l,j}^{[2]}(o^{[1,l]})$ are the MFs, $w_{i,h}^{[1,l]}$ and $w_{l,j}^{[2]}$ are tunable weights. We assume that the MFs are fixed, triangular, and equidistantly spaced over the range of each NFN input. The parameters of the MFs are not tuned.

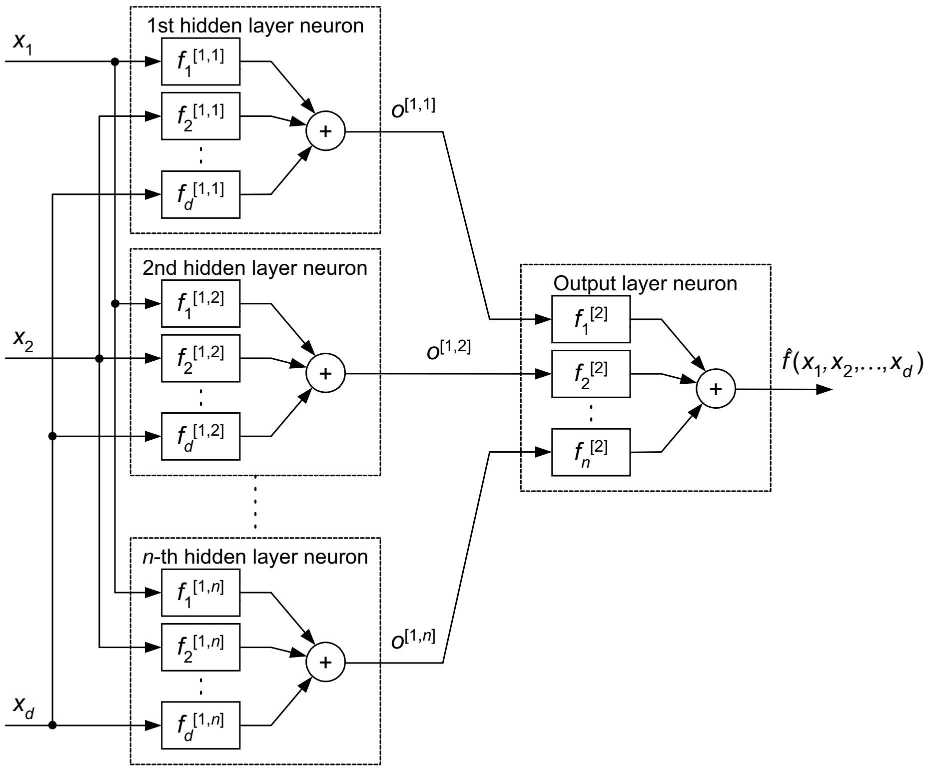

Fig 1: NFKN architecture with *d* inputs and *n* hidden layer neurons

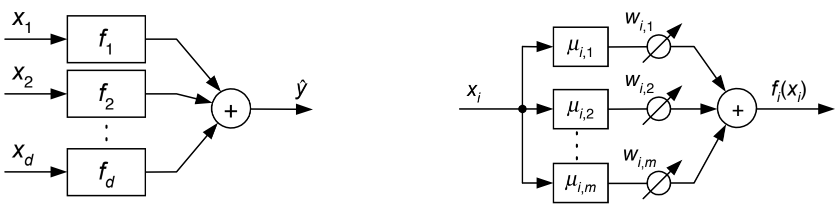

Fig. 2: Neo-fuzzy neuron (*left*) and its nonlinear synapse (*right*)

Nonlinear synapse is a single input-single output fuzzy inference system, and is thus a universal approximator [Kos92] of univariate functions, including $g_l(\bullet)$ and $\psi_{l,i}(\bullet)$ in (1). An example of approximation of a univatiate function is shown in Fig. 3. So the NFKN, in turn, can approximate any function $f(x_1,\ldots,x_d)$.

As in the FKN, the MFs in the NFKN at each input in the hidden layer are shared between all neurons (see Fig. 4). However, in the NFKN architecture we allow for different number of membership functions at each input.

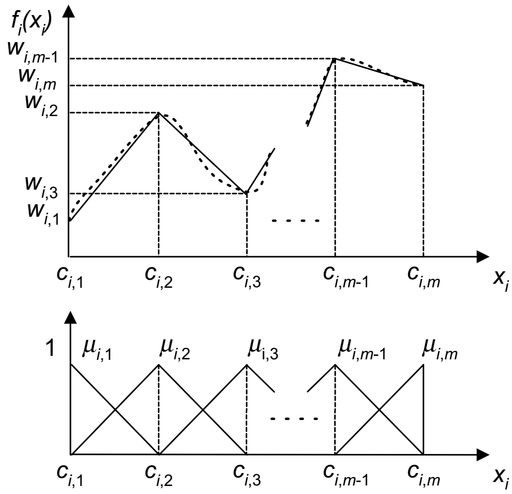

Fig. 3**:** Approximation of a univariate nonlinear function by a nonlinear synapse

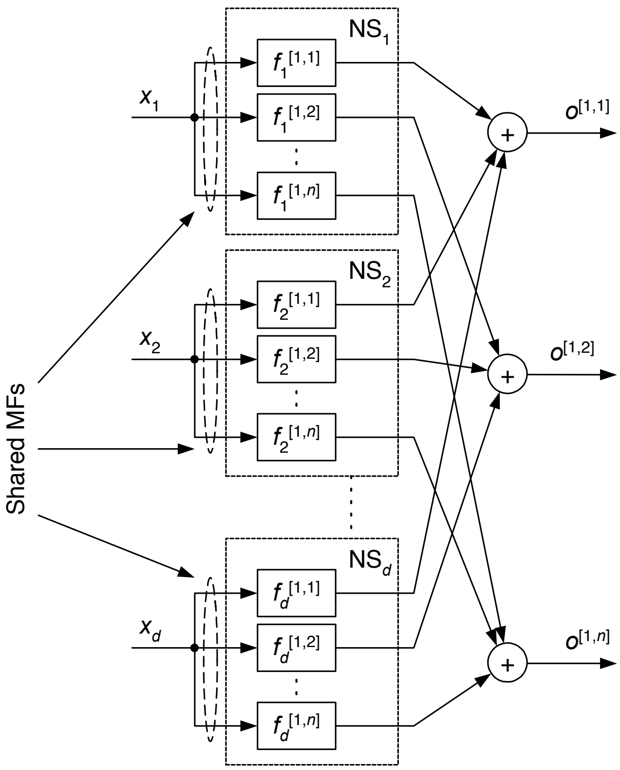

Fig. 4**:** Representation of the hidden layer of the NFKN with shared MFs

This property is essential for the processing of data with mixed numerical and categorical inputs, such that each category value of a categorical input corresponds to one MF and is encoded with a numerical value corresponding to the center of that MF. This is a more parsimonious and convenient approach than the conventional binary coding of categories, because we do not have to introduce additional inputs to the classifier.

3 Learning algorithm

The weights of the FKN are determined by means of a batch-training algorithm as described below. A training set containing *N* samples is used. The minimized error function is

$$E(t)=\sum_{k=1}^{N}\left[y(k)-\hat{y}(t,k)\right]^2=\left[Y-\hat{Y}(t)\right]^T\left[Y-\hat{Y}(t)\right], \qquad (4)$$

where $Y=\left[y(1),\ldots,y(N)\right]^T$ is the vector of target values, and $\hat{Y}(t)=\left[\hat{y}(t,1),\ldots,\hat{y}(t,N)\right]^T$ is the vector of network outputs at epoch *t*.

Since the nonlinear synapses (3) are linear in parameters, we can employ direct linear least squares (LS) optimization to find the ouput layer weights:

$$W^{[2]}=\left(\Phi^{[2]^T}\Phi^{[2]}\right)^{-1}\Phi^{[2]^T}Y^{[2]},\quad Y^{[2]}=Y\ , \qquad (5)$$

$$W^{[2]}=\left[w_{1,1}^{[2]},w_{1,2}^{[2]},\ldots,w_{n,m_2}^{[2]}\right]^T,\ \Phi^{[2]}=\left[\varphi^{[2]}(o^{[1]}(1)),\ldots,\varphi^{[2]}(o^{[1]}(N))\right]^T,$$

$$\varphi^{[2]}(o^{[1]})=\left[\mu_{1,1}^{[2]}(o^{[1,1]}),\mu_{1,2}^{[2]}(o^{[1,1]}),\ldots,\mu_{n,m_2}^{[2]}(o^{[1,n]})\right]^T.$$

Using the linearization technique for the output layer, we can find the hidden layer weights in a similar way, as is done in the FKN training method [KB04, KBO04]. In order to reduce the computational complexity, we can also find the hidden layer weights through the well-known gradient descent method:

$$W^{[1]}(t+1)=W^{[1]}(t)-\gamma(t)\frac{\nabla_{W^{[1]}}E(t)}{\left\|\nabla_{W^{[1]}}E(t)\right\|}, \qquad (6)$$

where $\gamma(t)$ is the adjustable learning rate. The norm in the denominator is to speed up convergence as proposed in [Jan92].

4 Applications

We have applied the NFKN to the problem of electric load forecasting in a region of Germany. The data were provided by one of local electricity suppliers, and describe electric load every 15 minutes. We used the data from the year of 1999 for training, and from the year of 2000 for validation and testing. The forecast was 24 hours ahead. The input variables were the load at the same hour one, two, three days and a week ago, type of these days ('normal' day or holiday), type of the predicted day, type of the day following the predicted day, day of week for the predicted day, the number of the predicted hour, and the number of the predicted quarter of that hour (13 inputs altogether). Fig. 5 shows the forecast for the last two weeks of April, 2000. This period includes Easter, which can be distinguished by lower consumption for four days in succession. Mean absolute percentage error (MAPE) of the forecast for this period equals 2.6044%, which is quite good and acceptable for practical use.

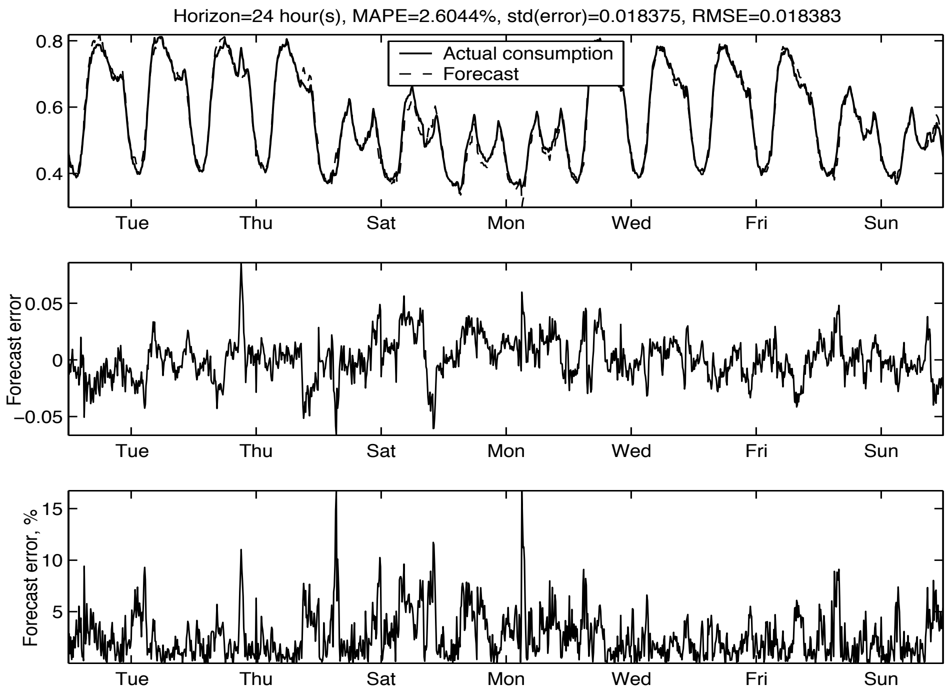

Fig. 5: One day ahead forecast of electric load for a region in Germany for the period of 17.04.2000 – 30.04.2000 (load data are scaled to [0, 1] according to the supplier's requirement)

We also applied the NFKN to classification problems using real-world data from the well-known UCI repository [UCI]. The parameters of the data sets are listed in Table 1. Note that two data sets, 'Australian credit' and 'German credit', have several categorical inputs. All the data sets have two classes.

Table 1: Data sets used in experiments

Data set	**Number of samples**	**Numerical attributes**	**Categorical attributes**	**Classes**
Wisconsin breast cancer	683	9	0	*benign* (65.5%), *malignant* (34.5%)
Australian credit	690	6	8	*positive* (44.5%), *negative* (55.5%)
German credit	1000	7	13	*good customer* (70%), *bad customer* (30%)

The results are summarized in Table 2, and are averages of 10-fold cross-validation. The column 'neurons' describes the NFKN architectures: the numbers separated by '+' indicate the number of the hidden and output neurons respectively. The column 'weights' shows the number of tunable parameters. The next column shows average number of epochs required for the learning algorithm to converge. The last two columns show the classification accuracy.

All the results are at the level of accuracy achieved with the best classification techniques, e.g. the support vector machines [Duc].

Table 2: Results of classification experiments

Data set	**Neurons**	**Weights**	**Epochs**	**Training set accuracy**	**Checking set accuracy**
Wisconsin breast cancer	1+1	44	4	98.03%	97.51%
Australian credit	2+1	116	11.7	89.58%	85.8%
German credit	3+1	240	29	85.6778	75.2%

5 Conclusion

In the paper, a modification of the FKN approach was proposed. The advantages of the new neuro-fuzzy model (NFKN) were demonstrated in experiments with real-world data.

Although the NFKN demonstrated quite good results in the experiments described above, we expect that its performance can be further improved via the tuning of the centers of MFs, and the use of MFs different from triangular. For the tuning of the MFs, the gradient-based approach may be used. Another option is the use of clustering methods. The clustering can be performed independently for each synapse, so it can be reduced to the one-dimensional case.

For the processing of very large data sets when the storage of matrices for the complete data set is impossible because of memory limitations, sequential algorithms can be easily derived. For this purpose, the recursive least squares method for the estimation of the output layer weights and iterative scheme for the calculation of the gradient for the hidden layer weights can be employed.

Bibliography

[Duc] Duch, W.: Datasets used for classification: comparison of results. Computational Intelligence Laboratory, Nicolaus Copernicus University, Torun, Poland. http://www.phys.uni.torun.pl/kmk/projects/datasets.html

[Hec87] Hecht-Nielsen, R: Kolmogorov's mapping neural network existence theorem. Proc. IEEE Int. Conf. on Neural Networks, San Diego, CA, Vol. 3 (1987) 11-14

[Jan92] Jang, J.-S. R.: Neuro-Fuzzy Modeling: Architectures, Analyses and Applications. PhD Thesis. Department of Electrical Engineering and Computer Science, University of California, Berkeley (1992)

[KB04] Kolodyazhniy, V., and Bodyanskiy, Ye.: Fuzzy Kolmogorov's Network. Proc. 8th Int. Conf. on Knowledge-Based Intelligent Information and Engineering Systems (KES 2004), Wellington, New Zealand, September 20-25, Part II (2004) 764-771

[KBO04] Kolodyazhniy, V., Bodyanskiy, Ye., and Otto, P.: Universal Approximator Employing Neo-Fuzzy Neurons. Proc. 8th Fuzzy Days, Dortmund, Germany, Sep. 29 – Oct. 1 (2004) CD-ROM

[Kol57] Kolmogorov, A.N.: On the representation of continuous functions of many variables by superposition of continuous functions of one variable and addition. Dokl. Akad. Nauk SSSR **114** (1957) 953-956

[Kos92] Kosko, B.: Fuzzy systems as universal approximators. Proc. 1st IEEE Int. Conf. on Fuzzy Systems, San Diego, CA (1992) 1153-1162

[Kur91] Kůrková, V.: Kolmogorov's theorem is relevant. Neural Computation 3 (1991) 617-622

[LYH02] Lopez-Gomez, A., Yoshida, S., and Hirota, K.: Fuzzy functional link network and its application to the representation of the extended Kolmogorov theorem. International Journal of Fuzzy Systems **4** (2002) 690-695

[Spr96] Sprecher, D.A.: A numerical implementation of Kolmogorov's superpositions. Neural Networks **9** (1996) 765-772

[Spr97] Sprecher, D.A.: A numerical implementation of Kolmogorov's superpositions II. Neural Networks **10** (1997) 447–457

[UCI] UCI Machine Learning Repository, http://www.ics.uci.edu/~mlearn/MLRepository.html

[YNK99] Yam, Y., Nguyen, H. T., and Kreinovich, V.: Multi-resolution techniques in the rules-based intelligent control systems: a universal approximation result. Proc. 14th IEEE Int. Symp. on Intelligent Control/Intelligent Systems and Semiotics ISIC/ISAS'99, Cambridge, Massachusetts, September 15-17 (1999) 213-218

[Yam92] Yamakawa, T., Uchino, E., Miki, T., and Kusanagi, H.: A neo fuzzy neuron and its applications to system identification and prediction of the system behavior. Proc. 2nd Int. Conf. on Fuzzy Logic and Neural Networks "IIZUKA-92", Iizuka, Japan (1992) 477-483