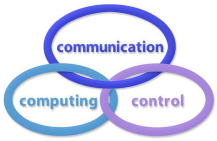
INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
Online ISSN 1841-9844, ISSN-L 1841-9836, Volume: 20, Issue: 4, Month: August, Year: 2025
Article Number: 7127, https://doi.org/10.15837/ijccc.2025.4.7127



CCC Publications

On equivalence between Takagi-Sugeno-Kang fuzzy systems with triangular membership functions and Neural Networks with ReLU activation in two or more dimensions

B. Bede, V. Kreinovich, P. Toth

Barnabas Bede*
Department of Mathematics
DigiPen Institute of Technology
9931 Willows Rd. NE, Redmond, WA, 98052, USA
*Corresponding author: bbede@digipen.edu

Vladik Kreinovich
Department of Computer Science
University of Texas at El Paso
500 W University Ave, El Paso, Tx 79968, USA
email: vladik@utep.edu

Peter Toth
Department of Computer Science
DigiPen Institute of Technology
9931 Willows Rd. NE, Redmond, WA, 98052, USA
email: petert@digipen.edu

Abstract

We prove the equivalence between Takagi-Sugeno-Kang (TSK) fuzzy systems and neural networks with ReLU activation function in two or more dimensions. The TSK fuzzy systems considered will have tetrahedral membership functions for their antecedents and singleton outputs. We show an example of a fuzzy system that is locally equivalent to a neural network based on the proposed method, and we discuss the potential to provide a local analysis to explain the decision process of neural networks.

Keywords: fuzzy sets, fuzzy systems, neural networks, ReLU activation, Takagi-Sugeno-Kang fuzzy systems

1 Introduction

Neural networks [4] and deep learning [6] are at the forefront of current AI development. Especially popular are Generative AI models with applications that range from text [16] to image generation [5]. Neural networks are largely regarded as black box methods, and their explainability and transparency decrease with their size and complexity [11].

Explainable AI [13] discusses the explainability of machine learning models. There are two main directions for the consideration of the explainability of ML models. The first approach relies on the construction of explainable models for various applications. One of the options for this approach is explainable fuzzy AI [9]. The second approach deals with deciphering neural networks and understanding their decision making process [10]. Both approaches are essential for safety-critical applications, where explainability is mandated either by law or by users.

Fuzzy sets were introduced by L. Zadeh [14] as mathematical models for uncertain quantities and are widely utilized in control applications [8]. Takagi-Sugeno (TS) fuzzy systems [12] are fuzzy models that combine the learning ability of neural networks, with the explainability of fuzzy systems [7]. As a consequence, TS fuzzy systems are promising candidates for explainable machine learning applications.

The equivalence between Takagi-Sugeno fuzzy systems with triangular membership functions and neural networks with ReLU activation was proved in the one-dimensional case in [2] and further generalized in [3]. The extension of these results to the case of multiple dimensions and multiple layers of neural networks was discussed as an open question, and a potential approach was proposed.

In the present paper, we continue this train of ideas and we prove the equivalence between Takagi-Sugeno fuzzy systems with tetrahedron membership functions as antecedents and neural networks with ReLU activation, in the two-dimensional case. These results provide a new local result to analyze the input-output connections for a neural network, improving their explainability.

2 Preliminaries

2.1 Fuzzy Systems

A fuzzy set [14] is a set with a continuum of membership degrees over a universe of discourse X, modeled by a function $A : X \to [0, 1]$, where we interpret $A(x)$ as the membership degree of the element x in the fuzzy set A [1].

Fuzzy systems of Mamdani type [8], describe the input output relationship in terms of fuzzy if-then rules of the form:

$$\text{If } \mathbf{x} \text{ is } A_i \text{ then } y \text{ is } B_i, i = 1, ..., n.$$

Given the above rule base and a crisp (scalar or vector) input $\mathbf{x} \in X$, the fuzzy output can be calculated as

$$B'(y) = \bigvee_{i=1}^{n} A_i(\mathbf{x}) \wedge B_i(y),$$

where $x \wedge y = \min\{x, y\}$ and $x \vee y = \max\{x, y\}$ The defuzzified output can be obtained as

$$COG(B') = \frac{\int_W B'(y) \cdot y \cdot dy}{\int_W B'(y) \cdot dy}.$$

where W denotes the support of the fuzzy system $B'(y)$. Mamdani fuzzy systems have a very intuitive interpretation, however, they are less easy to train due to the nonlinear input-output relationship.

To resolve the issues presented by training of Mamdani fuzzy systems, Takagi-Sugeno (TS) fuzzy system are considered [12], [7]. The rule base of a TS fuzzy system is

$$\text{if } \mathbf{x} \text{ is } A_i \text{ then } y = y_i, \ i = 1, \ldots, n$$

TS fuzzy systems have a fuzzy input, however they have a crisp output. This makes them more suitable for applications. The Output of a TS fuzzy system is

$$TS(\mathbf{x}) = \frac{\sum_{i=1}^{n} A_i(\mathbf{x}) y_i}{\sum_{i=1}^{n} A_i(\mathbf{x})}.$$

In the one-dimensional case, we can consider triangular membership functions of the form $A = (a, b, c)$ and a TS fuzzy system determined by a partition on an interval $\underline{x} = b_0 = b_1 \leq b_2 \leq ... \leq b_{0n-1} = b_n = \bar{x}$. In this case the triangular membership functions determine a Ruspini partition, i.e,

$$A_i = (b_{i-1}, b_i, b_{i+1}), i = 1, ..., n - 1,$$

we can illustrate its structure by a computational graph as in Figure 1.

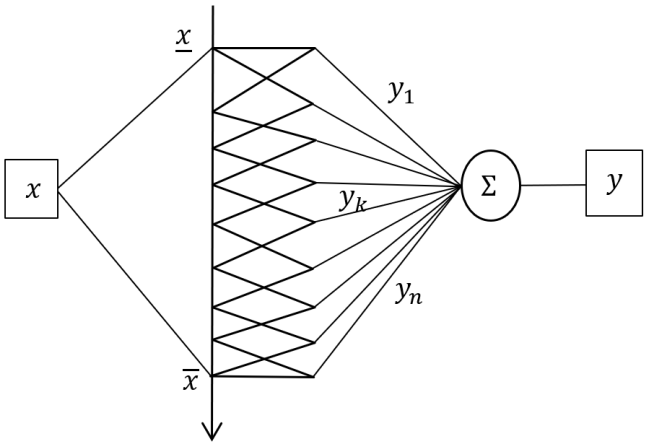


Figure 1: The structure of a TS-system

2.2 Neural networks

A *ReLU-based neuron* with n inputs $x_1, \ldots, x_n$, is defined as the function

$$y = \varphi\left(\sum_{j=1}^{n} a_j x_j + b\right)$$

with weights a_j, bias b, and activation function $\varphi(x) \stackrel{\text{def}}{=} \max(0, x)$.

A layer of a neural network is a vector formed by m neurons organized in a parallel fashion as its components

$$y_i = \varphi\left(\sum_{j=1}^{n} a_{ij} x_j + b_i\right), i = 1, ..., m.$$

A neural network is a function defined as the successive application of several neural network layers to a vector of inputs, such that the input of the next layer is the output of the current layer. The output of layer L can be computed iteratively

$$y_i^L = \varphi\left(\sum_{j=1}^{n} a_{ij}^{L-1} y_j^{L-1} + b_i^{L-1}\right), i = 1, ..., m^{L-1}$$

for each layer $L = 1, ..., N-1$, with the output of the last layer being

$$y = \sum_{j=1}^{n} w_j^{N-1} y_j^{N-1} + b^{N-1}.$$

In the present paper, we consider a regression network, but the results can easily be extended to a situation where there is an additional function applied to the last layer, such as another ReLU, sigmoid or softmax, making it a classification network.

3 Main results

3.1 Equivalence between TS fuzzy systems with triangular membership and Neural Networks with ReLU activation in one dimension

In [2], the authors proved the following equivalence theorem in the one dimensional case:

Theorem 1. *The connection between ReLU based neural networks and TS fuzzy systems defined on a closed interval in one dimension is given by the followings:*

(i) For every TS system with triangular membership functions, there exists a 1-hidden-layer ReLU-based neural network that produces the same output function $F(x)$.

(ii) For every ReLU-based neural network, there exists a TS system with triangular membership functions that produces the same output $F(x)$.

The ideas that prove the theorem are based on the connection between triangular membership and ReLU functions. If $A = (a, b, c)$, $a < b < c$ is a triangular membership function then we can write it in terms of ReLU functions as

$$A(x) = \frac{\varphi(x-a) - \varphi(x-b)}{b-a} - \frac{\varphi(x-b) - \varphi(x-c)}{c-b}.$$

If $a = b$ then we have

$$A(x) = -\frac{\varphi(x-b) - \varphi(x-c)}{c-b},$$

while for $b = c$ we consider

$$A(x) = \frac{\varphi(x-a) - \varphi(x-b)}{b-a}.$$

By direct calculation one can rewrite the TS system in Figure 1 as a neural network. To rewrite a ReLU-based neural network as a TS layer, one observes that the output of a ReLU layer of a neural network is piecewise linear just as the output of a TS layer. By matching the two functions with each other at the connection points of two linear pieces of the functions, we can define the two systems to produce the same output. We have included a review of the proof ideas, since we want to extend these to multiple dimensions.

We will adapt an incremental approach and prove our result in two dimensions.

3.2 Equivalence between TS fuzzy systems with and Neural Networks with ReLU activation in two dimensions

In the two-dimensional case, we will start looking at the problem of equivalence between a ReLU-based neural network and fuzzy system on a local scale. Let x, y denote the inputs of a ReLU-based neural network layer, w_i be the output weights, d be an output bias, a_i, b_i represent the input weights and c_i are biases. The network can be expresses as

$$NN(x, y) = \sum_{i=1}^{n} w_i \varphi(a_i x + b_i y + c_i) + d.$$

Since the network output is piecewise linear, it locally consists of plane regions, so we can find a region bounded by lines $a_i x + b_i y + c_i = 0$, $i \in S$ with S being a subset of indices in $1, ..., n$. Next, assume that the smallest region that contains the point (x, y) is a triangle. This does not restrict the generality, since in the case when this would not be triangular, it could be further subdivided into triangles. Let us consider then that the region that contains the point (x, y) is bounded by the lines

$$a_i x + b_i y + c_i = 0, i = 1, 2, 3.$$

Let us denote $A_i(x_i, y_i)$, $i = 1, 2, 3$ the coordinates of the vertices of the triangle. We will consider fuzzy sets with tetrahedral membership functions that are 0 outside the interior of the triangle $A_1A_2A_3$ and given by the expressions below inside the triangle

$$M_1(x, y) = 1 + \frac{\begin{vmatrix} x - x_1 & y - y_1 \\ x_2 - x_3 & y_2 - y_3 \end{vmatrix}}{\begin{vmatrix} x_2 - x_1 & y_2 - y_1 \\ x_3 - x_1 & y_3 - y_1 \end{vmatrix}},$$

$$M_2(x, y) = 1 + \frac{\begin{vmatrix} x - x_2 & y - y_2 \\ x_3 - x_1 & y_3 - y_1 \end{vmatrix}}{\begin{vmatrix} x_3 - x_2 & y_3 - y_2 \\ x_1 - x_2 & y_1 - y_2 \end{vmatrix}},$$

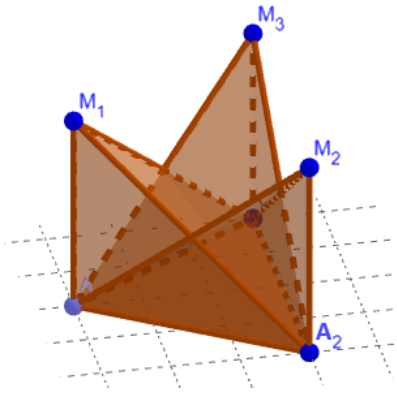


Figure 2: The antecedents for the given TSK-system

and

$$M_3(x,y) = 1 + \frac{\begin{vmatrix} x-x_3 & y-y_3 \\ x_1-x_2 & y_1-y_2 \end{vmatrix}}{\begin{vmatrix} x_1-x_3 & y_1-y_3 \\ x_2-x_3 & y_2-y_3 \end{vmatrix}}.$$

The expression for the function $M_1(x,y)$ is obtained by linear interpolation between the points $(x_1,y_1,1)$, $(x_2,y_2,0)$ and $(x_3,y_3,0)$, for $M_2(x,y)$ as interpolation between $(x_1,y_1,0)$, $(x_2,y_2,1)$ and $(x_3,y_3,0)$ and finally for $M_3(x,y)$ as interpolation between $(x_1,y_1,0)$, $(x_2,y_2,0)$ and $(x_3,y_3,1)$

Now we consider the fuzzy rule base that consist of the three rules that have antecedents M_1, M_2, M_3 respectively and consequences given by the output of the Neural network at the vertices of the triangle A_1, A_2, A_3 respectively. If we denote $z_i = NN(x_i, y_i)$ we can write

$$\text{If } (x,y) \text{ is } M_i(x,y) \text{ then } z = z_i, i = 1,2,3.$$

The TSK fuzzy system determined by the above rule base is given in mathematical form by the expression

$$TSK(x,y) = \sum_{i=1}^{3} M_i(x,y) z_i$$

and it will perform linear interpolation between the points (x_1,y_1,z_1), (x_2,y_2,z_2) and (x_3,y_3,z_3). Figure 2 illustrates the antecedents for our TSK fuzzy system. As the Neural Network $NN(x,y)$ performs linear interpolation between the same points, we obtain the local equivalence between the neural network $NN(x,y)$ and the TSK fuzzy system $TSK(x,y)$. As a conclusion, we can formulate the following theorem.

Theorem 2. *The Neural Network $NN(x,y)$ is equivalent locally (in the interior of the triangle $\Delta A_1A_2A_3$ to the TSK fuzzy system $TSK(x,y)$ given as above, i.e., we have*

$$NN(x,y) = TSK(x,y), \text{ for any } (x,y) \in int(\Delta A_1A_2A_3)$$

This local equivalence within a triangle can be naturally extended to the equivalence on each polygonal-shaped bounded domain – e.g., on a box $[-X,X] \times [-Y,Y]$. Indeed, as we have mentioned earlier, this box can be triangulated so that on each triangle, the NN-computed function is linear. The above construction produces, for each point p of a triangulation and for each triangle that has this point as a vertex, a linear membership function defined on this triangle that takes value 1 on this vertex. By combining all these linear fragments, we get a piece-wise linear membership function (see, e.g., [15]) – just like a triangular membership function is obtained by combining two linear pieces on two intervals with a common endpoint.

Then, instead of several TSK rules with conclusion $f(p)$ coming from different triangles and corresponding linear membership functions, we can have a single TSK rule with this combined membership function.

The graph of this combined membership functions is a pyramid whose base in a polygon formed by all the triangles from our triangulation that have p as a vertex. Such pyramids are known as *polygonal pyramids*. For the 1-D case, a union of two intervals with a common endpoint is an interval, so such a pyramid would be a triangle. In this sense, polygonal pyramids are a natural generalization of triangles. So, each combined membership function $M(x, y)$ of two variables is a natural 2-D analog of triangular membership functions $M(x)$.

3.3 Multi-Dimensional case is similar

In the previous section, we showed that in the 2-D case, the results of a ReLU-based neural network can be described by a fuzzy system with a natural generalization of triangular membership function – namely, membership function whose graphs form a polygonal pyramid – a 3-D analog of a triangle.

In the multi-dimensional case, the same arguments can prove that the function $f(x)$ computed by any ReLU-based neural network with any number of layers can be described a fuzzy system with membership functions whose graphs form a polyhedral pyramid – a natural multi-dimensional generalization of a polygonal pyramid.

Let us show how to prove this result. Any function $f(x)$ computed by a ReLU-based neural network is a composition of piece-wise linear functions $\max(0, a_1 \cdot x_1 + \ldots + a_n \cdot x_n + a_0)$ computed by individual ReLU neurons. It is well known that the composition of linear functions is also linear. Thus, the composition of piece-wise linear functions is also piece-wise linear. In other words, there are finitely many domains on each of which the resulting function is linear.

Since the overall result is continuous, the common boundary of two neighboring domains corresponds to the case when the two linear expressions (that describe the desired function of each domain) are equal. Thus, each such boundary is described by a linear equation – and is, therefore, part of some hyperplane. So, the boundary of each domain consists of pieces of hyperplanes – multi-D analogies of lines in 2-D spaces and planes in 3D space. In other words, each domain is a polyhedron – a natural multi-D generalization of a polygon in 2-D spaces and 3D polyhedra in 3D space.

It is known that every polyhedron can be triangulated – i.e., represented as a finite union of simplexes. So, to represent the desired function, it is sufficient to represent a linear function on a simplex by TSK fuzzy rules.

It is known that each point $x = (x_1, \ldots, x_n)$ inside a d-dimensional simplex with vertices $v_1, \ldots, v_{p+1}$ is a convex combination of the vertices, i.e., has the form $x = c_1 \cdot v_1 + \ldots + c_{p+1} \cdot v_{p+1}$, where $c_i \geq 0$ and $c_1 + \ldots + c_{p+1} = 1$.

It is also known that the coefficients c_i of this representation linearly depend on the coordinates x_i of the point. This can be shown easily: by an appropriate linear (affine) transformation T from x_i to some other coordinates y_j, we can transform each simplex into a standard simplex with vertices $v_1 = (1, 0, \ldots, 0)$, $v_2 = (0, 1, 0, \ldots, 0)$, $\ldots$, $v_{p-1} = (0, \ldots, 0, 1, 0)$, $v_p = (0, \ldots, 0, 1)$, and $v_{p+1} = (0, \ldots, 0)$.

In the standard simplex, every point $y = (y_1, \ldots, y_n)$ can be represented as $y = c_1 \cdot v_1 + \ldots + c_{p+1} \cdot v_{p+1}$ with $c_1 = y_1$, $\ldots$, $c_p = y_p$, and $c_{p+1} = 1 - y_1 - \ldots - y_p$. By combining these linear expressions with the linear transformation T, we get linear functions $c_i(T(x))$ describing c_i as linear functions of coordinates in the original simplex.

Thus, each linear function $L(x)$ on a simplex can be described by $p + 1$ TSK rules, with $i = 1, \ldots, p + 1$:

$$\text{If } c_i(x) \text{ then } z = L(v_i)$$

Now, similar to the 2-D case, we can combine, for each vertex p of the triangulation, all the linear functions $c_i(x)$ that attain the value 1 on this vertex into a single piece-wise linear membership function $M_p(x)$.

The graph of this function has the shape of a polyhedral pyramid – a natural multi-D generalization of a triangle. We can also, similar to the 2-D case, combine all the rules of the above type – corresponding to this vertex p – into a single rule:

$$\text{If } M_p(x) \text{ then } z = f(p),$$

where $f(x)$ is a NN-computed function that we are trying to describe. On each bounded domain, these rules will describe exactly the desired NN-computed function $f(x)$.

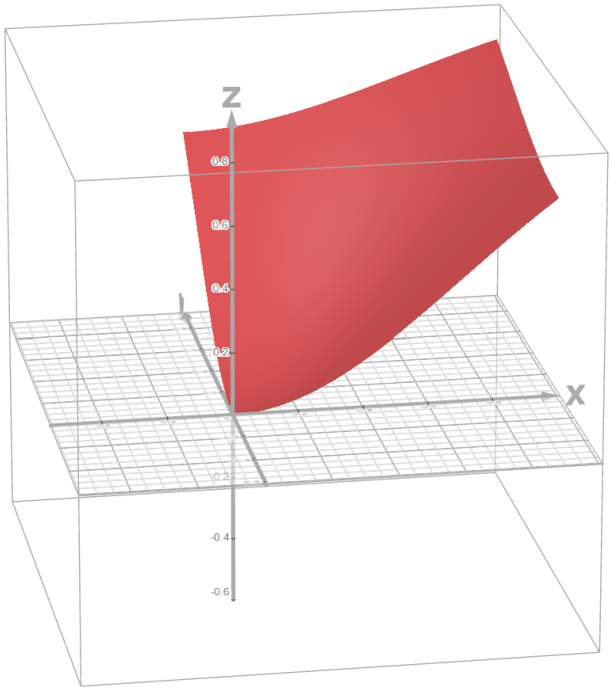


Figure 3: $f(x,y) = 1 - e^{-x^2-y^2}$ on [0,1]x[0,1]

3.4 Experimental Results

We are going to demonstrate the connection between Neural Networks and the 2-D Takagi-Sugeno system through experimentation.

We are using *keras* and *tensorflow* libraries to train a neural network to approximate a specific function. Visualize the specific function and its neural network approximation. The parameters of the trained network and the results in Theorem 2 are used to determine the parameters of the 2-D TS fuzzy system and then test the fuzzy system that we obtain and compare to the Neural Network outputs and visualize comparing to actual expected function values.

Let us choose $f(x,y) = 1 - e^{-x^2-y^2}$ as the function to approximate and limit its domain to the unit square. Figure 3

Train a neural network with one hidden layer of 3 nodes, set acceptable minimal loss, and achieve high accuracy. The training will use metrics of Mean Squared Error for loss and Root Mean Squared Error for accuracy. ReLU activation for the single hidden layer and identity function for the single output. By tuning the parameters, we will find that the Adam optimizer and 0.01 learning rate can yield with a very good approximation: 0.0001 loss and 0.01 accuracy. Figure 4

Now we use the parameters of the trained neutral network and identify the $\Delta A_1A_2A_3$, its points A_1, A_2 and A_3 corresponding to M_1, M_2 and M_3 antecedents, along with z_1, z_2 and z_3 parameters for the TSK system. Figure 5

Now we use these parameters and the TSK function to calculate and compare the TS system generated values to the Neural Network outputs. Observe that the TS system approximates the function in the very same way as the Neural Network does. The fuzzy system that we obtain is of the form

$$\text{If } (x,y) \text{ is } M_i(x,y) \text{ then } z = z_i, i = 1,2,3.$$

with $M_i(x,y)$ having tetrahedron memberships with the base being $\Delta A_1A_2A_3$ and consequences being $z_1 = 0.63449395$, $z_2 = 0.17345192$ and $z_3 = 0.64203346$.

We perform an additional check and choose 5000 points inside $\Delta A_1A_2A_3$, apply the TSK function on all points, and aggregate the absolute difference between the results of NN and TSK, finding it less than 10^{-5}, accounting for the precision of the computer calculations only. Figure 6 shows TSK values based on inside points of $\Delta A_1A_2A_3$ approximating function $f(x,y) = 1-e^{-x^2-y^2}$. Furthermore, checked and the values do not match when points are chosen from outside $\Delta A_1A_2A_3$.

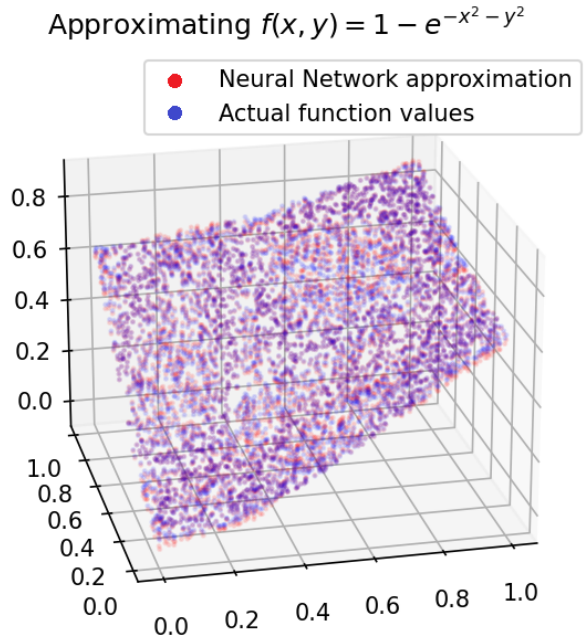


Figure 4: Neural Network approximation and actual values

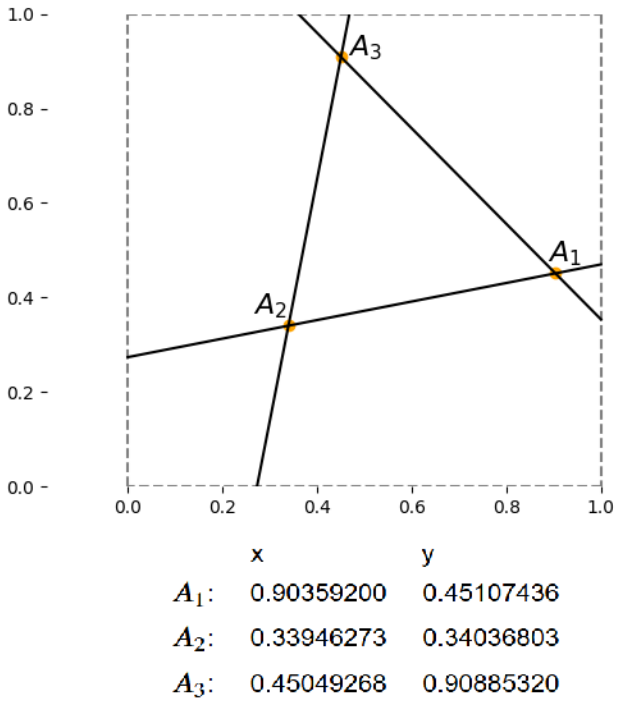


Figure 5: $\Delta A_1 A_2 A_3$ is the support of the membership functions M_1, M_2, M_3

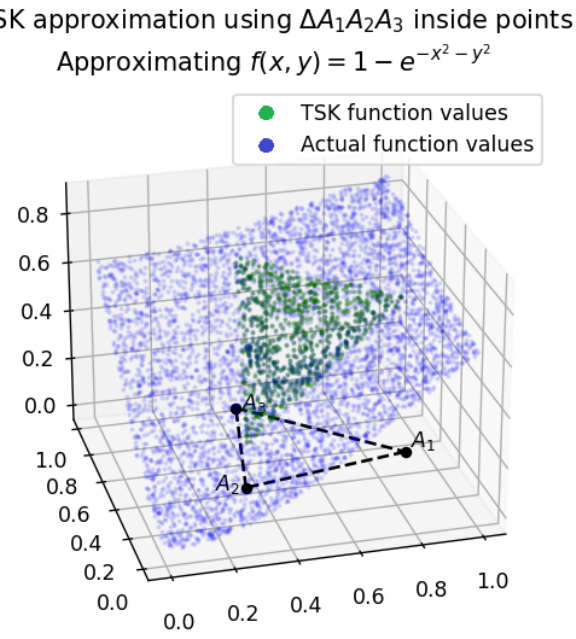


Figure 6: TSK approximation

This method allows us to extract a fuzzy rule base from a neural network built using existing efficient libraries such as keras and tensorflow, allowing both efficient learning algorithms to be used in conjunction with fuzzy systems and improves explainability of neural networks.

4 Conclusions and Further Research

We have extended previous results on the equivalence between TSK fuzzy systems and neural networks to the two-dimensional case. We found that locally, a neural network layer with ReLU activation can be equivalently described as a TSK fuzzy system with tetrahedron memberships. We extended the result to the multi-dimensional case as well. For future research we plan to extend the results from local to global results and to deep learning algorithms.

Acknowledgements

The first author is thankful to late Prof. Dr. Ioan Dzitac for his friendship and support over the years, as well as to his family for continuing his legacy.

Author contributions

The authors contributed equally to this work.

Conflict of interest

The authors declare no conflict of interest.

References

[1] Bede, B.: Mathematics of Fuzzy Sets and Fuzzy Logic. Springer (2013)

[2] Bede, B., Kreinovich, V., Toth, P.: Equivalence between 1-d takagi-sugeno fuzzy systems with triangular membership functions and neural networks with relu activation. In: North American Fuzzy Information Processing Society Annual Conference. pp. 44–56. Springer (2023)

[3] Bede, B., Kreinovich, V., Toth, P.: Equivalence between tsk fuzzy systems with triangular membership functions and neural networks with relu activation on the real line. In: North American Fuzzy Information Processing Society Annual Conference. Springer (2024)

[4] Bishop, C.M.: Neural networks and their applications. Review of scientific instruments **65**(6), 1803–1832 (1994)

[5] Cao, H., Tan, C., Gao, Z., Xu, Y., Chen, G., Heng, P.A., Li, S.Z.: A survey on generative diffusion models. IEEE Transactions on Knowledge and Data Engineering (2024)

[6] Goodfellow, I., Bengio, Y., Courville, A.: Deep learning. MIT press (2016)

[7] Jang, J.S.: ANFIS: adaptive-network-based fuzzy inference system. IEEE Transactions on Systems, Man, and Cybernetics **23**(3), 665–685 (1993). https://doi.org/10.1109/21.256541

[8] Mamdani, E.H., Assilian, S.: An experiment in linguistic synthesis with a fuzzy logic controller. International journal of man-machine studies **7**(1), 1–13 (1975)

[9] Mencar, C., Alonso, J.M.: Paving the way to explainable artificial intelligence with fuzzy modeling: tutorial. In: Fuzzy Logic and Applications: 12th International Workshop, WILF 2018, Genoa, Italy, September 6–7, 2018, Revised Selected Papers. pp. 215–227. Springer (2019)

[10] Mundhenk, T.N., Chen, B.Y., Friedland, G.: Efficient saliency maps for explainable ai. arXiv preprint arXiv:1911.11293 (2019)

[11] Oh, S.J., Schiele, B., Fritz, M.: Towards reverse-engineering black-box neural networks. Explainable AI: interpreting, explaining and visualizing deep learning pp. 121–144 (2019)

[12] Sugeno, M., Kang, G.: Structure identification of fuzzy model. Fuzzy sets and systems **28**(1), 15–33 (1988)

[13] Xu, F., Uszkoreit, H., Du, Y., Fan, W., Zhao, D., Zhu, J.: Explainable ai: A brief survey on history, research areas, approaches and challenges. In: Natural Language Processing and Chinese Computing: 8th CCF International Conference, NLPCC 2019, Dunhuang, China, October 9–14, 2019, Proceedings, Part II 8. pp. 563–574. Springer (2019)

[14] Zadeh, L.A.: Fuzzy sets. Information and control **8**(3), 338–353 (1965)

[15] Zámečiková, H., Perfilieva, I., Kosheleva, O., Kreinovich, V.: A natural extension of f-transform to triangular and triangulated domains necessitates the use of triangular membership functions. Proceedings of the 13th Conference of the European Society for Fuzzy Logic and Technology EUSFLAT, Riga, Latvia, July 21–25 (2025)

[16] Zhao, W.X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., Min, Y., Zhang, B., Zhang, J., Dong, Z., et al.: A survey of large language models. arXiv preprint arXiv:2303.18223 (2023)

Journal's webpage: http://univagora.ro/jour/index.php/ijccc/

This journal is a member of, and subscribes to the principles of,
the Committee on Publication Ethics (COPE).
https://publicationethics.org/members/international-journal-computers-communications-and-control

Cite this paper as:

Bede, B.; Kreinovich, V.; Toth, P. (2025). On equivalence between Takagi-Sugeno-Kang fuzzy systems with triangular membership functions and Neural Networks with ReLU activation in two or more dimensions, *International Journal of Computers Communications & Control*, 20(4), 7127, 2025.
https://doi.org/10.15837/ijccc.2025.4.7127