# Hybrid Deep Learning Type-2 Fuzzy Logic Systems For Explainable AI

Ravikiran Chimatapu, Hani Hagras
*School of Computer Science and Electronic Engineering*
*University of Essex*
Colchester, UK

Mathias Kern, Gilbert Owusu
*BT Labs*
*BT plc*
Ipswich, UK

***Abstract*—The recent years have witnessed a rapid rise in the use of Artificial Intelligence (AI) systems, in particular Machine Learning (ML) models. The vast majority of AI systems employ black box models that lack transparency in operation and decision making. This lack of transparency curtails the use of these AI systems in regulated applications (such as medical, financial applications, etc.) where it is important to understand the reasoning behind the predictions of the AI system. In these situations, interpretable models need to be used. However, interpretable models can turn into black-box models for high dimensional inputs. There are a variety of approaches that have been proposed to solve this problem. In this paper, we present a novel hybrid deep learning type-2 fuzzy logic system for explainable AI which addresses these challenges to provide a highly interpretable model that has reasonable performance when compared to the other black box models.**

**Keywords—Explainable Artificial Intelligence, Interval Type-2 Fuzzy Logic System, Deep Learning**

## I. Introduction

The latest decade did witness the wide adoption of AI and ML in critical domains such as finance, healthcare, automotive, education, criminal justice, etc. There are concerns being raised that the complex AI systems (such as deep learning, random forest, support vector machines, etc.) used in such systems could lead to a lack of transparency. The UK Parliament house of lords AI select committee, for example, in their report mention that "*We believe it is not acceptable to deploy any artificial intelligence system which could have a substantial impact on an individual's life, unless it can generate a full and satisfactory explanation for the decisions it will take*" [1]. The European Parliament has also recognized this and the recently adopted General Data Protection Regulation (GDPR) has a clause that reiterates the right of all individuals to obtain "*meaningful explanations of the logic involved when automated decision making takes place*" [2].

To this end, a new line of research has emerged that focuses on developing explainable Artificial Intelligence tools [3]. According to the Defence Advanced Research Projects Agency (DARPA), there are several approaches that are being pursued to realize Explainable AI (XAI) [4]. The first approach is to modify deep learning techniques to be more interpretable. This approach is called deep explanations [4] and saliency mapping is often used for these models. Saliency mapping is achieved by repeatedly testing a network to find out which part of the input influences the output [5]. LRP [6], DeepLIFT [7], CAM [8], etc, are some of the methods that use saliency mapping to provide deep explanations. However, the problem with these approaches is that the models only show the relationship between the inputs and the outputs and the interconnections between the inputs that create the intermediate layers are harder to analyse and generally require the help of experts in these techniques.

The second approach that has been proposed to achieve XAI is to use model induction through model agnostic methods. LIME [9] is one of the approaches which is model agnostic. LIME is used to probe the behaviour of the model by inducing perturbations on the inputs, this data is then used to find out which features contribute to the output and create a linear model which is locally faithful i.e, the model faithfully reproduces the output of the original model for a particular input. The problem with this approach is that we use an external model which means that the explanations provided are not always accurate.

The third approach is to use existing interpretable and causal models such as decision trees, Bayesian rules, hidden markov models, fuzzy logic, etc. However, these models can be less accurate than the corresponding black box models and they can also become opaque for high dimensional inputs [10, 11].

An alternative to the above approaches is to use a combination of an interpretable model and deep learning methods. Deep type-2 fuzzy logic system (D2FLS) is one such approach where the intuition behind autoencoders is combined with interpretable type-2 fuzzy systems. In this system interval type-2 fuzzy logic systems are arranged in a series are trained in the same way stacked autoencoders are trained. i.e, greedy layer wise training [12]. The Fuzzy Logic Systems (FLSs) are trained one layer at a time using unsupervised learning to learn important features or combine features. After all the layers are trained a final layer is added and the whole model is retrained using supervised learning.

There are several advantages to the above approach. The first is that since fuzzy logic systems use linguistic IF-Then rules it makes the system inherently interpretable. The second is that the system can be trained using both labelled and unlabelled data. The third benefit is that since the system is composed of rules and membership functions they can be relatively easily changed if there are any problems with them while its opaque counterparts can't be directly changed.

Combining fuzzy logic with deep learning is not a new concept and it has been done in recent years in systems like Fuzzy Restricted Boltzmann Machine [13] where the connections and biases of a Restricted Boltzmann Machine are fuzzy parameters. In Fuzzy Deep Neural Network [14], the

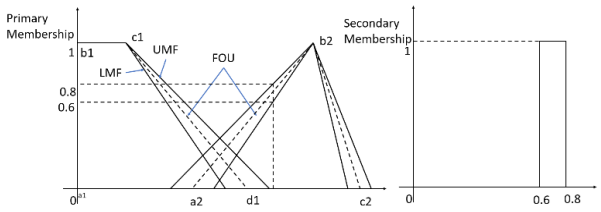


Fig. 1. Interval Type-2 Fuzzy Sets [15]

fuzzy and neural representations are trained at the same time and combined in the final layer. In Fuzzy Deep Belief Network [16], a Deep Belief Network (DBN) is trained on the data and a fuzzy membership is designed for each of the output of the DBN, etc. The problem with these methods is that they have been designed to improve the performance of models or handle uncertainty in the data and did not focus on the interpretability of these systems making them unsuitable for XAI without further improvements. Other Fuzzy Logic and Deep Learning hybrid techniques such as Takagi Sugeno Deep Fuzzy Network (TSDFN) [17] where the authors create a Neural Network with all the hidden nodes as Takagi Sugeno Fuzzy Logic systems and others like Fuzzy Deep Learning [18], etc. can be interpretable but they can also become opaque for high dimensional inputs.

In Section II, we present a brief overview of the Interval type-2 fuzzy logic system. Section III will present a brief overview of Stacked Autoencoders. Section IV will present the proposed Deep Type-2 Fuzzy Logic system. Section V will present the experiments and results. Finally, Section VI will present the conclusions and future work.

## II. A Brief Overview of Interval Type-2 Fuzzy Logic

An Interval Type-2 Fuzzy Logic System (IT2FLS) is characterized by a Membership Function(MF) $\mu(x)$ where each element of this set is a fuzzy set in the interval [0,1] and the third dimension of these fuzzy set is set to 1 (depicted in Fig. 1). A Type-1 fuzzy logic system, on the other hand, is characterized by a type-1 fuzzy set where each element of this set is a number in the interval [0,1] [19-25].

The IT2FLS works in the following way: In the first step the inputs are fuzzified into interval type 2 fuzzy sets, the inference engine then activates the rule base using the input interval type 2 fuzzy sets and produces output interval type 2 fuzzy sets and in the final step the output interval type 2 fuzzy sets are defuzzified into crisp numbers[26-28].

There are two methods for defuzzify interval type 2 fuzzy sets into crisp numbers. The first method is a two-step process where first the output type 2 fuzzy sets are type reduced into type-1 fuzzy sets followed by defuzzification of the type reduced sets. The second method is a single step process called direct defuzzification [19].

D2FLS uses the type reduction method proposed by Nie and Tan [29] and weighted scaled dominance approach [30] for regression and classification datasets respectively.

## III. A Brief Overview of Stacked Autoencoder

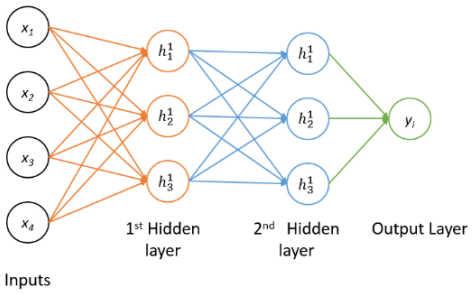


Fig. 2. Stacked Auto Encoder [31]

An Autoencoder tries to reconstruct an approximation of the input at the output i.e, the target output is the input of the model. Given a set of training samples $[x_1, x_2, .., x_j]$, where $x_i \in R$, the autoencoder first encodes the input $x_j$ to a hidden representation $y(x_j)$ based on equation (1), then it decodes the representation $y(x_j)$ back into a reconstruction of the input $z(x_j)$ based on (2).

$$y(x) = f(W_1 x + b_1) \quad (1)$$

$$z(x) = g(W_2 y(x) + b_2) \quad (2)$$

The intuition behind this is to restrict the number of neurons in the hidden layer so that the model is forced to learn important features of the inputs or compress the inputs.

A stacked autoencoder(SAE) is a deep neural network depicted in Fig. 2, and it is created by stacking autoencoders. To explain it more clearly, consider an SAE with $l$ layers, the first layer is trained like an autoencoder. Once this is done the first hidden layer becomes the input of the next Autoencoder. Similarly, after training the $k^{th}$ hidden layer, the output of this layer is used as the input for the $(k+1)^{th}$ hidden layer. Once all the hidden layers are trained the output layer is added to the SAE and then all the layers are retrained using labelled data. This training method is called greedy layer-wise training [12].

## IV. Deep Type-2 Fuzzy Logic System

The Deep Type-2 Fuzzy Logic System (depicted in Fig. 3) consists of two or more FLSs where the output of the first FLS is the input of the second FLS and the output of the second FLS is the input of the third FLS and so on. This structure helps the D2FLS to have a low number of rules to reduce the complexity of the system. This mitigates one of the problems with modelling datasets with a large number of features using FLSs i.e, a FLS might require a large number of rules to model these datasets.

The D2FLS is trained similar to the way stacked autoencoders are training i.e, using greedy layer-wise training [12]. The training is done using Big Bang Big Crunch algorithm (BB-BC) [32] instead of gradient descent algorithms. Each FLS layer is called a Fuzzy Autoencoder (FAE) (Depicted in Fig. 4) and is trained like an autoencoder. The two parts of the FAE are represented as follows:

$$h = f(x) \quad (3)$$

Where $h$ is a vector that represents the output of the encoder and $f(x)$ represents the encoder.

$$\hat{x} = g(h) \quad (4)$$

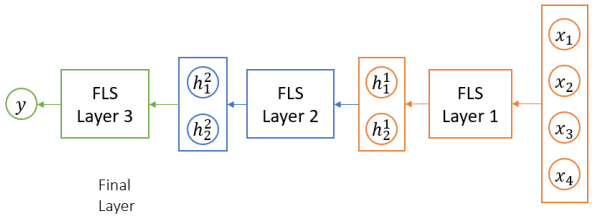


Fig. 3. A Deep Type-2 Fuzzy Logic System Architecture

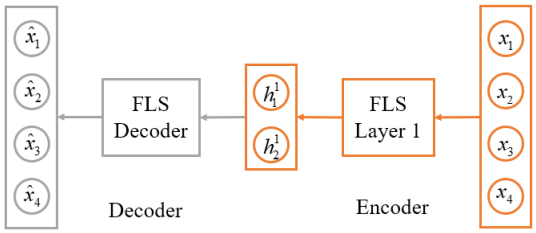


Fig. 4. Fuzzy Autoencoder Architecture

Where $\hat{x}$ is the output of the decoder i.e, the reconstructed input.

To simplify the training process for the FAE the membership functions of the antecedents of the decoder are formed from the MFs which represent the consequents of the encoder. The consequents of the decoder are formed from the MFS that form the antecedents of the encoder

## A. *Fuzzy Autoencoder Training*

The MFs and rule base of the Fuzzy Autoencoder are trained using the BB-BC algorithm and the training is comprised of three steps. In the first step, the MFs of the encoder and the decoder are trained as a type-1 fuzzy logic system. In the second step, the type-1 fuzzy sets of the model created in the previous step are transformed into interval type-2 fuzzy sets and in the final step, the rule bases of both the encoder and decoder are retrained. Once the training is completed the decoder of the FAE is discarded and the next FLS layer is similarly trained with the outputs of the previous layer as the inputs. We do this until the required number of FAEs are trained.

### *1) Optimize Type-1 Fuzzy Autoencoder*

In this step, we train a type-1 FAE using BB-BC and the membership functions and rules are encoded in the following format to create the individuals of the BB-BC algorithm. The MFs for each input or output is encoded in the following format based on the number of fuzzy sets per input or output.

$$M_i = m_1^1, m_2^1, m_1^2, m_2^2, .., m_1^j, m_1^j \tag{5}$$

Where $j$ represents the number of fuzzy sets per input/output. This is depicted in Fig. 5. The first value and the last value of $M_i$ is always set to 0 and 1 respectively and all inputs and outputs are normalized between 0 and 1.

Each rule of the FLS is encoded using the below representation.

$$R_l = r_1^1, r_2^1, .., r_1^a, r_2^a, c_1, .., c_b \tag{6}$$

Where $R_l$ represents the $l^{th}$ rule with each rule having $a$ antecedents and $b$ consequents. Each antecedent is represented by 2 values the first represents the index of the $i$ inputs and the second point represents the $j^{th}$ fuzzy set of the input from Equation (5).

Finally, we use Equations (5) and (6) together to represent the FAE using below.

$$E_n = M_1^e, .., M_i^e, .., M_{i+k}^e, R_1^e, .., R_l^e \tag{7}$$

$$FAE = E_x, R_1^d, .., R_l^d \tag{8}$$

Where $i$ is the number of inputs, $k$ is the number of hidden output, $R_l^e$ are the rules for the encoder and $R_l^d$ are the rules for the decoder.

### *2) Transform T1MFs to IT2MFs*

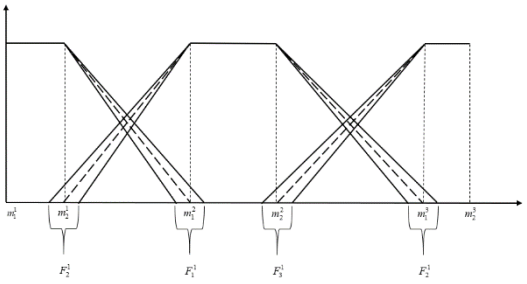

Fig. 5. Representation of a Footprint of uncertainty (FOU)

In this step, we transform the type-1 fuzzy sets of the FAE into interval type-2 fuzzy sets using BB-BC by adding a Footprint Of Uncertainty (FOU) to the start and end of the fuzzy sets of the antecedents and the consequents as depicted in Fig. 5 while retaining the parameters from the previous step. The FOU parameters are encoded in the following format.

$$F = f_1^1, .., f_j^1, .., f_j^i, f^{(i+1)}, .., f^{(i+k)} \tag{9}$$

Where the model has $i$ inputs and $j$ fuzzy sets per inputs and $k$ fuzzy sets for the output.

### *3) Optimizing the Rule Base of the IT2 FAE*

In this step, we retrain the rules of the FAE using the BB-BC algorithm. The rules of the encoder and decoder are encoded in the following format while retaining all the other parameters from the previous step.

$$RL_n = R_1^e, .., R_l^e, R_1^d, .., R_l^d \tag{10}$$

Where $R_l^e$ are the rules for the encoder and $R_l^d$ are the rules for the decoder.

## B. *Optimization method for the Final layer*

To train the full D2FLS, we first train a set of FAEs and then add the final layer and retrain the whole system using labeled data using. Hence, this part of the training is supervised. This step is similar to the way FAEs are trained i.e, we train the D2FLS using the same three steps.

### *1) Optimize the Type 1 Final Layer*

In this step, we train the final layer as a Type-1 FLS while we retrain the MFs and rules of the set of encoders trained in the previous step using BB-BC algorithm. The MFs and rules are encoded in the following format.

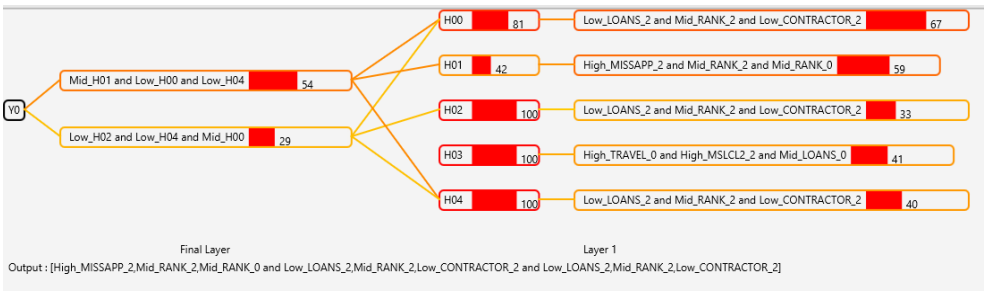


Fig. 6. Detailed explanation

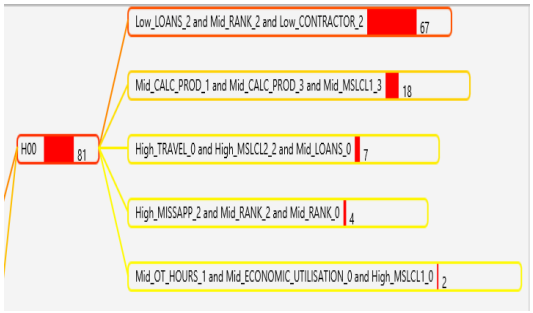


Fig. 7. Detailed explanation for one hidden output

$$D2FLS = E_1,..,E_n,M_1^f,..,M_1^f,...M_{(o+p)}^f,R_1^f,..,R_x^f \quad (11)$$

Where $E_n$ represents the MFs and rules of the encoder parts of the FAEs created using Equation (7), $o$ represents the number of inputs to the final layer, $p$ represents the number of fuzzy sets for the outputs and $x$ represents the number rules in the final layer.

*2) Transform T1MFs of the final layer into IT2MFs*

In this step, we transform the type-1 MFs of the final layer into IT2MFs and retrain the IT2MFs of the encoders using BB-BC algorithm. This step is similar to the way we the FOUs were added to the MFs of the FAEs (depicted in Fig. [FOU]). The parameters are encoded in the following format.

$$FOU_{fl} = F_1,..,F_n,,f_1^1,..,f_j^o,F^{(o+1)},..,F^{(o+p)} \quad (12)$$

Where $F_n$ represents the FOUs for the $n$ encoders, $o$ represents the number of inputs of the final layer with each input having $j$ fuzzy sets and $p$ represents the number of outputs of the final layer.

*3) Optimizing the IT2 rule base of the D2FLS*

In this step, we retrain the rules of the final layer along with the rules of $n$ encoders using the BB-BC algorithm. The parameters for this step are encoded in the following format.

$$Rules = RL_1,..,RL_n,..,R_1^f,..,R_x^f \quad (13)$$

Where $RL_n$ represents the rules of the encoders, $x$ represents the number of rules for the final layer.

## C. Method for Extracting Explanations from the D2FLS

In this section, we provide details of the method used to extract simple interpretable explanations, i.e., provide a qualitative understanding of the relationship between the input features and the output [9]. The algorithm used to create these explanations is suitable for a two-layer D2FLS but they can be extended for multiple layers.

For generating the explanation (depicted in Fig 6) a compound input for the final layer based on the rules that contribute the most to the outputs of the first layer is created. For example, in Fig 6 the output H00 of the first layer is named Low loans_2, Mid Rank_2 and Low Contractor_2 because this rule contributes the most (67%) to this output, there are five other rules that contribute to this output (depicted in Fig 7) but their contribution is smaller. We can then use these compound inputs to find out the relationship between the output and the input features. For example, in Fig 6, the rule which contributes 54% to the output is composed of 2 compound inputs (High Missapp_2, Mid Rank_2 and Mid Rank_0) and (Low loans_2, Mid Rank_2 and Low Contractor_2). We can then use the same method for the second rule (29% contribution) which is composed of one compound input (Low loans_2, Mid Rank_2 and Low Contractor_2). We can of course drill down further but this should provide an easy to understand the interpretation of the D2FLS. These values are computed as follows:

$$\bar{R}_n^g = \bar{F}_n * c_n^g \quad (14)$$

$$\underline{R}_n^g = \underline{F}_n * c_n^g \quad (15)$$

Where $\bar{F}_n$ and $\underline{F}_n$ represent the upper and lower firing levels of the $n^{th}$ rule and $c_n^g$ represents the $g^{th}$ consequent of the $n^{th}$ rule of the final layer.

$$R_{navg}^g = (\bar{R}_n^g + \underline{R}_n^g)/2 \quad (16)$$

$$R_{nval}^g = R_{navg}^g / \sum_{n=1}^{n} R_{navg}^g * 100 \quad (17)$$

We use equation (17) to calculate the values for each of the rule and consequent combinations for the final layer.

$$\bar{R}_l^h = \bar{F}_l * c_l^h \quad (18)$$

$$\underline{R}_l^h = \underline{F}_l * c_l^h \quad (19)$$

Where $\bar{F}_l$ and $\underline{F}_l$ represent the upper and lower firing levels of the $l^{th}$ rule and $c_l^h$ represents the $h^{th}$ consequent of the $l^{th}$ rule of the first layer.

$$R_{lavg}^h = (\bar{R}_l^h + \underline{R}_l^h)/2 \quad (20)$$

$$R_{lval}^h = R_{lavg}^h / \sum_{l=1}^{l} R_{lavg}^h * 100 \quad (21)$$

We use the Equation (21) to calculate the scores for the first layer and an example of these values are depicted in Fig 6.

## V. Experiments and Results

In this paper, the D2FLS model is compared to a Sparse stacked autoencoder (SSA) and a multi-layer perceptron (MLP). We use 4 datasets for these experiments. The first two are regression datasets and their details are below.

- BT PWA (BTP): The sixth dataset is collected from British Telecom in the UK. The data consists of around 30000 records and 44 attributes and the data was used in the following papers [33]
- Swiss Premium (SP): This dataset is from health insurance premium prediction. This dataset consists of around 200,000 records with 199 inputs. We used 100000 records for unsupervised training and the other 100000 records for supervised training. For more details about this dataset please refer to the following [34].

We use Mean Absolute Error (MAE) as the fitness function for the above two regression datasets.

$$MAE = \frac{\sum_{i=1}^{n} | y_i - \hat{y}_i |}{n} \tag{22}$$

The other two dataset are binary classification datasets and their details are given below.

- Santander CTP (SCTP): This dataset is from Santander. It is a binary classification problem where we try to identify which customer will make a transaction, it consists of around 400,000 records with 200 inputs [35]. We used 200,000 records for unsupervised training and 200,000 records for supervised training.
- BT Customer Service (BTCS): The dataset is supplied by BT and the data is about predicting whether a customer service call is about their broadband connection or not. The data consists of about 100,000 records with 500 attributes. We used 50000 records for unsupervised training and 50,000 records for supervised training.

We use Average Recall as the fitness function for the above 2 classification datasets.

$$Recall_{positive} = \frac{tp}{tp + fn} \tag{23}$$

Where True positive $tp$ is the number of correct positive predictions and False negative $fn$ is the number of incorrect negative predictions.

$$Recall_{negative} = \frac{tn}{tn + fp} \tag{24}$$

Where True Negative $tn$ is the number of correct negative predictions and False Positive $fp$ is the number of incorrect positive predictions.

$$Recall_{avg} = \frac{Recall_{positive} + Recall_{negative}}{2} \tag{25}$$

Where $Recall_{avg}$ represents the Average recall for a binary classification problem.

All the inputs and outputs of these datasets were normalized between 0 and 1. We then randomly divided the training part of the datasets into three parts: 70% of the data is used for training, 15% of the data is used for validation and the last 15% of the data is used for testing.

The sparse stacked autoencoder was trained using greedy layer-wise training[12]. We used 2 hidden layers with 100 and 15 neurons each. Adam Algorithm [36] was used for training the SSA and we set the learning rate as 0.001, beta 1 as 0.9 and beta 2 as 0.999 and trained it for 200 epochs.

The Multilayer perceptron we used had 1 hidden layer with 15 neurons in the hidden layer. And we used Adam Algorithm for training the MLP with the same parameters as the SSA.

The Deep Type 2 Fuzzy Logic System we used had 2 layers with 100 rules and 3 antecedents per rules for each layer and 3 MFs per input. For the regression datasets, the first layer had 50 outputs and for the categorical datasets, the first layer had 30 outputs. For the BB-BC algorithm, we used 30 individuals with 500 iterations as parameters.

We then trained the three models on the two categorical datasets 5 times and the mean and standard deviation of the MAE of these runs is calculated and shown in Table I. Similarly, for the two regression datasets we trained the trained the models 5 times and the mean and standard deviation of these 5 runs were calculated and shown in Table II.

From Table I, we can see that the D2FLS is within 6% of the SSA and MLP in the BT customer service dataset and performs within 2 % in the other classification dataset. From Table II, we can see that the D2FLS model performs reasonably well when compared to the SSA and MLP models on the regression datasets. This shows that the D2FLS provides good performance when compared to its opaque deep learning counterparts while providing us with good interpretability as seen from Fig 6 and 7.

## VI. Conclusions and Future Work

The widespread use of Artificial Intelligence models has necessitated the emergence of a new line of research focused on developing explainable AI tools. A majority of this research has been focused on existing interpretable models, model induction or deep explanation methods to solve this problem.

We argued that there is an alternative method available which combines the power of Deep learning method with the interpretability and flexibility of interpretable models. We outlined the proposed Deep Type-2 Fuzzy Logic system which aims to solve this problem for high dimensional inputs. We showed that this model performs well and provides comparable results to those achieved by Stacked autoencoder and Multi-layer Perceptron. This is especially true in categorical datasets where it performs within 5 percent of the other two models.

TABLE II. Comparison of the Models Over The Categorical Data

| Dataset | Mean MAE D2FLS | Std of MAE D2FLS | Mean MAE SSA | Std of MAE SSA | Mean MAE MLP | Std of MAE MLP |
|---|---|---|---|---|---|---|
| CTP | 63.16% | 1.57 | 64.77% | 2.24 | 65.5% | 0.67 |
| BTCS | 69.37% | 0.05 | 75.84% | 0.01 | 74.63% | 0.03 |

TABLE I. Comparison of the Models Over The Regression Data

| Dataset | Mean MAE D2FLS | Std of MAE D2FLS | Mean MAE SSA | Std of MAE SSA | Mean MAE MLP | Std of MAE MLP |
|---|---|---|---|---|---|---|
| BTP | 0.057 | 0.0052 | 0.038 | 0.0003 | 0.038 | 0.0002 |
| SP | 0.0474 | 0.0051 | 0.0268 | 0.0014 | 0.026 | 0.0003 |

The proposed method the D2FLS is a good approach to solving high dimensional problem where there is a need to train from labelled and unlabeled data and where the interpretability of the model is especially important. Hence, we can conclude that hybrid deep learning and fuzzy logic approaches are a key component in making Artificial intelligence models trustworthy and ultimately more useful. For future work we propose to extend the D2FLS model to cater to image classification and natural language processing.

## VII. REFERENCES


[1] "AI in the UK: ready, willing and able?," UK Parliament (House of Lords) Aritificial Intelligence Committee,, 16 April 2017. [Online]. Available: https://publications.parliament.uk/pa/ld201719/ldselect/ldai/100/100.pdf

[2] B. F. Goodman, Seth, "European Union regulations on algorithmic decision-making and a "right to explanation"," *2016 ICML Workshop on Human Interpretability in Machine Learning (WHI 2016), New York, NY,* 2016.

[3] F. Poursabzi-Sangdeh, D. G. Goldstein, J. M. Hofman, J. W. Vaughan, and H. Wallach, "Manipulating and measuring model interpretability," *arXiv preprint arXiv:1802.07810,* 2018.

[4] D. Gunning, "Explainable artificial intelligence (xai)," *Defense Advanced Research Projects Agency (DARPA), nd Web,* 2017.

[5] L. H. Gilpin, D. Bau, B. Z. Yuan, A. Bajwa, M. Specter, and L. Kagal, "Explaining explanations: An approach to evaluating interpretability of machine learning," *arXiv preprint arXiv:1806.00069,* 2018.

[6] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek, "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation," *PloS one,* vol. 10, no. 7, p. e0130140, 2015.

[7] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 2017: JMLR. org, pp. 3145-3153.

[8] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2921-2929.

[9] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should i trust you?: Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016: ACM, pp. 1135-1144.

[10] Z. C. Lipton, "The Mythos of Model Interpretability," *Queue,* vol. 16, no. 3, p. 30, 2018.

[11] R. Chimatapu, H. Hagras, A. Starkey, and G. Owusu, "Explainable AI and Fuzzy Logic Systems," in *International Conference on Theory and Practice of Natural Computing*, 2018: Springer, pp. 3-20.

[12] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *Advances in neural information processing systems*, 2007, pp. 153-160.

[13] P. Chen, C. Zhang, L. Chen, and M. Gan, "Fuzzy restricted Boltzmann machine for the enhancement of deep learning," *IEEE Transactions on Fuzzy Systems,* vol. 23, no. 6, pp. 2163-2173, 2015.

[14] Y. Deng, Z. Ren, Y. Kong, F. Bao, and Q. Dai, "A hierarchical fused fuzzy deep neural network for data classification," *IEEE Transactions on Fuzzy Systems,* vol. 25, no. 4, pp. 1006-1012, 2017.

[15] R. Chimatapu, H. Hagras, A. Starkey, and G. Owusu, "Interval Type-2 Fuzzy Logic Based Stacked Autoencoder Deep Neural Network For Generating Explainable AI Models in Workforce Optimization," presented at the 2018 IEEE International Conference on Fuzzy Systems (FUZZ), in press.

[16] S. Zhou, Q. Chen, and X. Wang, "Fuzzy deep belief networks for semi-supervised sentiment classification," *Neurocomputing,* vol. 131, pp. 312-322, 2014.

[17] S. Rajurkar and N. K. Verma, "Developing deep fuzzy network with Takagi Sugeno fuzzy inference system," in *2017 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 2017: IEEE, pp. 1-6.

[18] S. Park, S. J. Lee, E. Weiss, and Y. Motai, "Intra-and inter-fractional variation prediction of lung tumors using fuzzy deep learning," *IEEE journal of translational engineering in health and medicine,* vol. 4, pp. 1-12, 2016.

[19] J. Mendel, *Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Directions*. Springer, 2017.

[20] J. M. Mendel and X. Liu, "Simplified interval type-2 fuzzy logic systems," *IEEE Transactions on Fuzzy Systems,* vol. 21, no. 6, pp. 1056-1069, 2013.

[21] A. Starkey, H. Hagras, S. Shakya, and G. Owusu, "A Multi-Objective Genetic Type-2 Fuzzy Logic Based System for Mobile Field Workforce Area Optimization," *Journal of Information Sciences,* vol. 333, pp. 390-411, 2016.

[22] H. Hagras and C. Wagner, "Introduction to interval type-2 fuzzy logic controllers—Towards better uncertainty handling in real world applications," *IEEE Systems, Man and Cybernetics eNewsletter,* vol. 27, 2009.

[23] C. Lynch, H. Hagras, and V. Callaghan, "Embedded interval type-2 neuro-fuzzy speed controller for marine diesel engines," in *International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU 2006)*, Paris, France, 2006, pp. 1340-1347.

[24] H. Hagras, V. Callaghan, M. Colley, and M. Carr-West, "A fuzzy-genetic based embedded-agent approach to learning and control in agricultural autonomous vehicles," in *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C)*, 1999, vol. 2: IEEE, pp. 1005-1010.

[25] H. Hagras, M. Colley, V. Callaghan, and M. Carr-West, "Online learning and adaptation of autonomous mobile robots for sustainable agriculture," *Autonomous Robots,* vol. 13, no. 1, pp. 37-52, 2002.

[26] M. Antonelli, D. Bernardo, H. Hagras, and F. Marcelloni, "Multiobjective evolutionary optimization of Type-2 fuzzy rule-based systems for financial data classification," *IEEE Transactions on Fuzzy Systems,* vol. 25, no. 2, pp. 249-264, 2017.

[27] A. Sakalli, T. Kumbasar, E. Yesil, and H. Hagras, "Analysis of the performances of type-1, self-tuning type-1 and interval type-2 fuzzy PID controllers on the magnetic levitation system," in *2014 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 2014: IEEE, pp. 1859-1866.

[28] J. Andreu-Perez, F. Cao, H. Hagras, and G.-Z. Yang, "A self-adaptive online brain–machine interface of a humanoid robot through a general type-2 fuzzy inference system," *IEEE Transactions on Fuzzy Systems,* vol. 26, no. 1, pp. 101-116, 2016.

[29] M. Nie and W. W. Tan, "Towards an efficient type-reduction method for interval type-2 fuzzy logic systems," in *2008 IEEE International Conference on Fuzzy Systems (IEEE World Congress on Computational Intelligence)*, 2008: IEEE, pp. 1425-1432.

[30] D. Bernardo, H. Hagras, and E. Tsang, "A genetic type-2 fuzzy logic based system for the generation of summarised linguistic predictive models for financial applications," *Soft Computing,* vol. 17, no. 12, pp. 2185-2201, 2013.

[31] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: a deep learning approach," *IEEE Transactions on Intelligent Transportation Systems,* vol. 16, no. 2, pp. 865-873, 2015.

[32] O. Erol and I. Eksin, "A new optimization method: big bang–big crunch," *Advances in Engineering Software,* vol. 37, no. 2, pp. 106-111, 2006.

[33] R. Chimatapu, H. Hagras, A. Starkey, and G. Owusu, "A Big-Bang Big-Crunch Type-2 Fuzzy Logic System for Generating Interpretable Models in Workforce Optimization," in *2018 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 2018: IEEE, pp. 1-8.

[34] *Swiss healthcare premium prediction ( https://www.kaggle.com/comparisdata/premium-prediction)*,

[35] *Santander Customer Transaction Prediction ( https://www.kaggle.com/c/santander-customer-transaction-prediction)*, Banco Santander,

[36] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980,* 2014.

[37] V. Callaghan, M. Colley, H. Hagras, J. Chin, F. Doctor, and G. Clarke, "Programming iSpaces—A tale of two paradigms," in *Intelligent Spaces*: Springer, 2006, pp. 389-421.