ISSN 2007-9737

Adaptive Attention Reasoning Transformer Using Neuro-Fuzzy Modulation

Adrian E. Adame*, Mauricio A. Sanchez, Juan R. Castro

Universidad Autonoma de Baja California,
Mexico

{adrian.adame, mauricio.sanchez, jrcastror}@uabc.edu.mx

Abstract. The transformer model architecture has been shown to be very effective at understanding sequence relations. For tasks like natural language processing (NLP), we can make decisions using dynamic relationships among linguistic elements. While this architecture excels at capturing complex dependencies through self-attention mechanisms, their application to tabular data often creates powerful, complex models. By using neuro-fuzzy systems, we can provide rule-based interpretability, but they typically lack the performance of robust deep learning models on complex datasets. Using the best instrument of each model, we proposed a new hybrid approach using the rule-based knowledge and the self-attention mechanism. This model employs a neuro-fuzzy system that adjusts the contribution of each attention head in an interpretable manner. This creates a fuzzy attention system, focusing not only on pattern recognition and relationships but also providing a high-level justification for its reasoning strategy. Our approach utilizes the self-attention mechanism within a set of learnable fuzzy rules to enhance the interpretation of the relationship between input features and the model's decision-making process. We demonstrate the efficacy of the model on supervised regression and classification tasks, showing that it maintains a competitive performance, creating a new approach for combining state-of-the-art Transformer architectures with neuro-fuzzy models.

Keywords. Fuzzy systems, transformer, attention, interpretability, tabular data.

1 Introduction

Deep learning, particularly the Transformer architecture proposed by Vaswani et al. [40], has become the state-of-the-art model to model complex relations and handle large contextual relationships in various domains. This is achieved by the attention mechanism, which learns and has general context information of all data representations, having the possibility to analyze different projections at the same time in parallel thanks to the multiple attention heads that follow the same process. This model has been the standard for most current applications and a wide area of research, replacing CNNs[10], RNNs[35], and LSTMs[17] due to the simplicity of how it can manage, having in some sense "memory," being capable of correlating past tokens to new ones, and adapting to new changes with minimum loss. Although this model can adapt to and fit most problems, its computational process also increases as the number of tokens grows. It often leads to the curse of deep learning: "black box" models, being incapable of giving a precise answer to how they got to a particular output, are one of the major drawbacks of their use. The calculated attention weights can be viewed, yet they are unable to give an answer to the "thought" process or interpretation of the data [21].

On the other hand, fuzzy systems[41] have been proven to be a "grey-box" approach for modeling complex relations; their rule-based knowledge and inference can be understood in linguistic terms, which gives fuzzy models the advantage of being able to demonstrate a "reasoning" process. Combining neural networks with a fuzzy system provides the model the ability to adjust to complex relations and have understandable rules to show how it relates to every sample[22]. But it has

drawbacks; manually created fuzzy systems often require domain experts to determine the number of rules to showcase those relations, or if it dynamically adjusts, we might encounter the ”curse of rule dimensionality”[9], showcasing how many rules will be necessary given the number of feature inputs we have, and even if we manage to mitigate those issues, ANFIS[22] are shallow networks, making the model prone to not capturing long, complex non-linearities in the data and being incapable of adapting well.

We can see a new opportunity for research. On one hand, we have a deep learning model that is capable of handling complex relations and has proven to excel at pattern recognition, but it lacks interpretability; on the other hand, we have a logic system that provides reasoning through readable rule-based inference, although its weakness lies in managing high-dimensional data. Merging these two different yet similar models, we can leverage the best of each one, creating a hybrid model that is capable of handling high-dimensional data while having a reasoning process through its inference.

In this paper, we propose the Attention Reasoning Transformer using Neuro-Fuzzy Modulation, a novel hybrid architecture meant to work as a first step to fill this gap. This model introduces a Type-1 Fuzzy System modulator into the transformer's self-attention mechanism. The fuzzy rules are used at the attention output, allowing the fuzzy rules to dynamically and interpretably modulate the flow of information. However this approach suffers from a vanishing gradient problem as the transformer model becomes more complex, that's why we added an auxiliar contrastive learning loss to migitate and help stabilize the overall model traning process.

In subsequent sections of this research, we are going to first introduce the current state-of-the-art for fuzzy systems, neuro-fuzzy neural networks, transformers, and an introduction to their self-attention mechanism. Secondly, state our model architecture for all the processes from the initialization of the fuzzy system, the data embeddings, and the preparations for the Transformer until the output of the system, mentioning how the backpropagation updates the learning parameters of the overall system. Lastly, we present the results of our experiments, the performance under regression and classification tasks, and, by the end, a section for discussions and future work.

2 Related Work

This work merges three important yet distinct areas of research. To establish the context of our proposed model we will first review the area of Fuzzy Systems, from its first introduction to classical fuzzy sets, to a more complex yet applicable system, the fuzzy inference systems and its improvement over complex problems and its adjustment, and finally survey the self-attention of the Transformer architecture and the way it changed how we go through complex relations.

2.1 Fuzzy Systems

Zadeh first introduced the logic and methodology to work with fuzzy sets in an article called the same way, ”Fuzzy Sets”[41], this contribution represented a new approach to handling uncertainty and vagueness in data by using linguistic terms and showing its relations by ”if-then” rules. Fuzzy logic works by soft-assigning membership values to all samples in a given set; that is, each sample has a degree of membership in all other sets, compared to crisp logic that assigns a sample to be only in one set and one set alone.

Fuzzy logic used fuzzy sets to create a knowledge base and infer the degree of membership of the input sample; with a set of ”if-then” rules, we can model complex patterns and interpret the output calculation. This idea goes one step towards model interpretability, going from a ”black box” to a ”grey box” model. Let's take for example a deep neural network. We know that a Deep Neural Network can model complex relations, and although we know mathematically how the computer calculates the output of the system, we don't know why it chose that particular set of weights. With a fuzzy set and the rule-based knowledge, we can actually interpret and know how each input contributed to the output and the relations among the data. The rule-based system then gives a human-readable reasoning process,

defining this technique as the main advantage over other ”black-box” models.

Over the years, fuzzy sets have been evolving from their first introduction[6], but T1 FS is the most widely used even to this day, and although it doesn't have an explicit way to handle uncertainty, it's equally capable of handling vagueness just like any other high-order FS.

To this day and for most applications, a T1 FS is more than enough to solve it; such examples are an Hybrid T1 FS Functions for Forecasting[11], a Fuzzy System to support Medical Diagnosis [13] and even applications to mobile networks has been done[42]. This is because of the effectiveness of soft-assigning degrees of membership for the samples.

By itself a FS creates an understandable knowledge base; its inference can be used to reason or process information, creating a Fuzzy Inference System (FIS). This system incorporates three key steps: The translation of the samples to a soft-assign cluster followed by the rule-based knowledge which calculates the inference of the system, and at the end we transform this value via a defuzzification process that takes this inference and outputs a crisp value.

Currently there exist three types of FIS: Mamdani[27], Sugeno/Takagi/San (TSK)[38], and Tsukamoto[39]. Mamdani Inference Systems are based on membership functions, which are easily interpreted; Sugeno Inference Systems could be linear or non-linear functions, which are usually adjusted by external algorithms to achieve a better-fitted and accurate model; Tsukamoto, on the other hand, lies within the other two FIS, that is, defining an interpretable membership function and an overall well-fitted model.

Sugeno Fuzzy Inference Systems are widely used in hybrid models due to the ability to better adjust the consequent polynomial functions and in some way have interpretable reasoning. However, to converge on satisfactory results, we often require domain experts to initialize those rules and the antecedent membership function parameters. This process is time-consuming and may not be optimal.

2.2 Neuro-Fuzzy Neural Networks

To address the manual-tuning limitation of classical FIS, neuro-fuzzy networks were developed [22]. These model hybrids combine fuzzy logic reasoning with the learning capabilities of neural networks. The idea of a neuro-fuzzy system is to adjust both the antecedent membership functions and the consequent, leveraging the back-propagation algorithm via optimization algorithms, commonly using first-order optimization, e.g., via Gradient Descent and its derivations such as Adam[24], AdamW[25], Lion[8], SMORMS3[16], etc., or even second-order algorithms such as the Levenberg-Marquardt algorithm[31] which uses the Hessian to compute the gradients of all parameters.

The most important example is the *Adaptive Neuro-Fuzzy Inference System* (ANFIS) introduced by Jang [22]. ANFIS is a Type-1 TSK-style fuzzy system implemented in a five-layer network architecture, which can learn both the antecedent membership functions and the consequent parameters. This idea was motivated by leveraging the reasoning process of the fuzzy system and the ability to adjust to the input data to model complex relations while maintaining a degree of interpretability. This model has been applied to numerous areas of research, such as Pattern Recognition of fluid flow in a 3D domain[2], an Accurate power forecasting [36], and an ANFIS for controlling fluid levels [20].

The ANFIS model is not the only neuro-fuzzy neural network; rather, it was the starting point for an area of research combining different types of FIS and more robust networks. Examples of this work are NEFCON[32] or hybridizing versions of ANFIS. Although this model presented a new step towards model interpretability, in essence they are still shallow networks, capable of approximate data modeling but lacking the performance and context adjustment of deep learning models. Increasing the number of rules might mitigate the issue, but otherwise it appears that as the problem in hand becomes more complex, the number of rules and membership functions per feature increases, and the main goal of the model is lost[9].

2.3 Transformers and Self-Attention

Since the first appearance of the Transformer architecture by Vaswani et al. [40], its introduction changed the way we handle and confront complex and contextual relations. This model replaced Convolutional Neural Networks, Recurrent Neural Networks, and even Long-Short Term Memory Networks (LSTM) as the primary building block of deep complex models [33]. This now has become the main architecture for most of the research in the area, applied to even numerous fields such as Robotics[5], Medical applications[14], with Natural Language Processing (NLP) being the main area of application[30, 28] and in most recent years even Computer Vision with ViT Transformers (ViT) [15, 34].

The power of the Transformer model then lies in the attention mechanism. capable of learning complex relations and the context of the data representations by using a set of three projections: Query, Value, and Key (Q, K, and V). These projections calculate the relevance of a given token with respect to all other tokens in the input embeddings. This calculation learns to understand the relation across all the tokens, routing information, and computing a score for each token's query against every other token's key. Those scores are scaled by the square root of the key's vector dimension $\sqrt{d_k}$ for numerical stability, passed through a softmax shown in Eq. (1) function to create the attention weights, and then used to compute a weighted sum of the values. The model then is not adjusting a pair of random, unrelated weights; now it trains with a rich contextual embedding that has in some sense "paid" attention to every other token in the sequence:

$$softmax(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}}. \tag{1}$$

This attention process is not done just one time during the process; in fact, it repeats itself given the number of "heads" in parallel, giving the model the flexibility to generate different types of meanings (i.e., one head for syntax order and another for semantic meaning). This MultiHead Attention (MHA) is stacked with a Feed-Forward Network (FFN), which applies a non-linear transformation to the output of the attention mechanism. MHA and FFN is stacked across multiple blocks, making the Transformer a very robust yet complex deep learning model.

Even though the attention and the Transformer might be an excellent model to handle context and relations, it has a major flaw: the number of parameters needed to generate a rich representation of the tokens, and as the number of tokens and number of heads increases, the attention calculation becomes more computationally expensive. This converts the model into a "black-box" model; the token relations are among millions or even billions of parameters. In recent years an active area of research has focused on mitigating those issues like sparse attention [26] and QKV compressions [3]. Separately, to reduce the cost of training and fine-tuning, parameter-efficient methods like Low-Rank Adaptation (LoRA)[18] have been developed.

If we reduce the number of parameters needed to optimize a Transformer model, we are still working with a deep neural network, and for applications where we need to understand how the model choose or give a certain answer it's just not possible now, after the training process of a model, we can view the attention weights and see how this mechanism relates the tokens, but we might not be able to take this set of weights as an accurate representation stated by Jain and Wallace: *"In addition to improving predictive performance, these are often touted as affording transparency: models equipped with attention provide a distribution over attended-to input units, and this is often presented (at least implicitly) as communicating the relative importance of inputs. However, it is unclear what relationship exists between attention weights and model outputs"*[21].

At last we will talk about Transformer for Tabular Data. In the first instance, this model was created to use text as the primary source of data, and all the embedding calculations and positional encoding were designed specifically to handle this type of information. With the idea to leverage the advantages of the attention mechanism, an active area of research is being developed, and different kinds of tabular embeddings are being proposed,

such as a feature embedding using each input feature as a contextual vector[19].

This review reveals a clear and new area of research. On one hand, neuro-fuzzy systems provide better "interpretability," as the inference of these systems uses "if-then" rules and softly assigns the data to a given membership degree, but it lacks the deep and contextual reasoning power of modern architectures. However, the Transformer architecture, while state-of-the-art and capable of handling complex relationships, is fundamentally uninterpretable due to its large number of parameters and "black-box" nature. No model exists that integrates the best of each model; that is the area we are boarding with this paper as a first approach towards interpretability and state-of-the-art models.

3 Proposed Model

3.1 Model Architecture

This models works as a first approach on how it's possible to merge a Neuro-Fuzzy System with Transformers for tabular data, covering one challenge we found during the design and training process, a problem founded commonly on Deep Neural Networks, the vanishing gradient problem and how we mitigate the issue by adding an auxiliary contrastive loss to help stabilize the gradient flow for the fuzzy systems and its rules.

3.1.1 Fuzzy System Initialization

The initial requirement for our proposed model is to obtain the rule configuration for the fuzzy system (i.e., number of rules, number of inputs, type of membership functions, etc.). To obtain these parameters, an initial step using a clustering algorithm is used (in this paper, the Fuzzy C-Means clustering algorithm[4] was chosen, but it would work with any distance-based clustering algorithm). Using the final clusters of the clustering algorithm, we then define the Gaussian membership functions, which require two parameters as shown in Eq. (2):

$$\mu(x; m, \sigma) = e^{-\frac{1}{2}(\frac{x-m}{\sigma})^2}, \tag{2}$$

where the centers of the clusters are the centers of each Gaussian membership function, and σ is the standard deviation of that particular cluster.

Since the Gaussian membership function is differentiable, it is suitable to use it as the MF for the rules, since we will be adapting these parameters to better fit the data during the training process.

After the rules are defined and the Gaussian membership functions initialized, we then proceed with the calculation of the normalized firing strengths of the rules. Let $X \in \mathbb{R}^{p \times q}$ be a batch of p input samples, where each p_i is a D-dimensional feature vector. The fuzzy system contains k rules (each cluster is a rule), each defined by a q-dimensional center vector $m_{i,k}$ and a q-dimensional standard deviation vector $\sigma_{i,k}$.

For a single input sample $x_{p,i}$, the firing strength (or degree of membership) $\alpha_{p,k}$ of the k_{th} rule is calculated by evaluating the Gaussian membership function corresponding to that rule in that particular sample. As shown in Eq. 3, the $\alpha_{p,k}$ firing strength is the distance between the input sample and the center, scaled by the standard deviation of the function. The generalized calculation of $\alpha_{p,k}$ for all inputs X and the rules:

$$\begin{aligned} \alpha_{p,k}(\mathbf{x}) &= T_{i=1}^{n} \mu_{F_i^k}(x_{p,i}) = \prod_{i=1}^{n} \mu_{F_i^k}(x_{p,i}) \\ &= \prod_{i=1}^{n} exp\left\{ -\frac{1}{2} \left(\frac{x_{p,i} - m_{i,k}}{\sigma_{i,k}} \right)^2 \right\}. \end{aligned} \tag{3}$$

Before initializing the Transformer architecture, we need to ensure the firing strengths are in the range of [0-1] to get an equal contribution of each sample in the given rule. To do this, we normalize by dividing each rule's firing strength by the sum of all rule activations, ensuring the resulting weights sum to 1 for any given input:

$$\phi_{p,k}(\mathbf{x}) = \frac{\alpha_{p,k}}{\sum_{l=1}^{r} \alpha_{p,l}(\mathbf{x})} + \epsilon, \tag{4}$$

where ϵ is a small constant to ensure numerical stability and prevent division by 0. This $\phi_{p,k}$ vector is the primary output of the fuzzy modulator. It represents a soft partition of the input data and is used to directly modulate the attention mechanism in the subsequent Transformer blocks.

3.1.2 Tabular Data Embeddings Initialization

Before diving into the Transformer mechanism, we need to first transform the data from the normal distribution to a set of embeddings, since the Transformer and the self-attention work on data representations of a given depth by projecting three sets of matrices (W_Q, W_K, W_V) on the representations. Since the proposed model is focused on tabular data, we cannot use the standard encoding and embedding "word2seq" lookup tables found in natural language processing[29]. Instead, we must create a sequence of embeddings for the continuous input features. This is achieved by treating each feature like a sequence employing a numerical feature tokenizer.

Projecting each of the q input features into the d_{model}-dimensional embedding space, this layer consists of a learnable weight matrix $W_{feat} \in \mathbb{R}^{q \times d_{model}}$ and a bias $b_{feat} \in \mathbb{R}^{1 \times d_{model}}$. Each feature x_q from the input x is individually embedded, transforming the input from $X \in \mathbb{R}^{p \times q}$ to a feature tensor $\mathbf{E}_{feat} \in \mathbb{R}^{p \times q \times d_{model}}$.

This transformation is shown in Eq. (5), where each of the weights is stacked following the order of the input features:

$$\mathbf{E}_{feat} = X^{p \times q \times 1} \odot \mathbf{W}_{feat}^{q \times d_{model}} + \mathbf{b}_{feat}^{q \times d_{model}}, \quad (5)$$

where $\odot$ represents element-wise multiplication with broadcasting. This operation scales the d_{model}-dimensional embedding for the q_{th} feature by that feature's scalar input value x_p.

To reduce the time of inference to compute the output of the Transformer, and following the architectural design of Vision Transformers (Vision Transformers), we prepend a single, learnable weight *Classification CLS Token* $e_{cls} \in \mathbb{R}^{1 \times d_{model}}$ to the sequence of feature tokens. This results in a final embedded input $\mathbf{E}_{feat} \in \mathbb{R}^{p \times (q+1) \times d_{model}}$. This token is a global representation of the entire input embeddings, and its corresponding output value from the final Transformer will be used for the prediction task.

3.1.3 Encoder-Only Transformer

With the normalized firing strengths and the embedding of the input features, we then introduce the third module of the model, the stack of $\mathbf{N}$ Encoder-Only Transformer Blocks. The reason for using an Encoder-Only model is that this architecture generates new representations of our data based on a given set of embeddings; in regression or classification tasks, this can lead to favorable results because the attention mechanism adjusts, allowing the fuzzy system to guide the adjustments of the membership functions' parameters.

Each Encoder-Only Transformer block i performs two main operations: a neuro-fuzzy modulated multi-head attention and a position-wise feed-forward network (FFN), with residual connections and layer normalization applied to each main mechanism, to create a better path for the gradients to flow across all parameters on the model, that is, from the output to the fuzzy system.

The key innovation of our model lies within the Multi-Head Attention (MHA) mechanism. From its original paper, *"the Multi-Head Attention allows the model to jointly attend to information from different representation subspaces at different positions. With a single attention head, averaging inhibits this."*[40]:

$$\begin{aligned} MultiHead(Q,K,V) &= Concat(head_1, ..., head_h)W_O \\ where \quad head_i &= Attention(QW_Q^i, KW_K^i, VW_V^i), \end{aligned} \quad (6)$$

where the projections are weight matrices $W_Q^i \in \mathbb{R}^{d_{model} \times d_k}$, $W_K^i \in \mathbb{R}^{d_{model} \times d_k}$, $W_V^i \in \mathbb{R}^{d_{model} \times d_k}$ respectively.

And the equation of a single-head attention mechanism is shown in Eq. 7:

$$Attention(Q,K,V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \quad (7)$$

where Q, K, V are the token embeddings of our input, or in this case, the embedding$\mathbf{E}_{feat}$ is passed three times, since the self-attention mechanism looks at the same tokens from

different space projections, and d_k is the factor between the number of attention heads and the d_{model}-dimension of the tokens.

3.1.4 Modulation Using the Fuzzy Firing Strengths

Our model modulates the overall output of each attention calculation using the previously calculated normalized firing strength of the fuzzy system $\phi_{p,k}$. Crucially, we match the number of rules to the number of attention heads to perform an element-wise multiplication, that is, $K == H$.

The output of the attention values x (after the calculation of the attention) has a shape of $attention_{probs} \in \mathbb{R}^{p \times attn_{heads} \times q \times d_{model}}$. The firing strengths are then reshaped to $\phi^{p \times k \times 1 \times 1}$ to match the attention probabilities calculated for each attention head; the result is shown in Eq. 8:

$$X_{mod} = MHA(Q, K, V) \cdot \phi = X_{attn} \cdot \phi. \quad (8)$$

This operation allows each fuzzy rule to control or influence its corresponding attention head. If a rule does not fire for a given input (e.g., $\phi_k \approx 0$), its associated attention head is effectively "turned off" for that input, which gives the model a reasoning process in an interpretable manner. The output of this modulation X_{mod} is then passed to the feed forward network, and the process repeats for $\mathbf{N}$ Encoder-Only Transformer blocks.

The last part of the Transformer block is the Position-Wise Feed-Forward Network; the modulated multi-head attention output is passed through this last part to refine and select the best patterns of each of the token's calculated representations independently.

This FFN network consists of two linear transformation layers with a SwiGLU activation[37] in between, which has been shown to improve performance over the standard ReLU[1] and GLU[12] variations.

The FFN computation is as shown in Eq. 9:

$$FFN(X_{mod}) = (SwiGLU(X_{mod}W_1 + b_1))W_2 + b_2, \quad (9)$$

where $W_1 \in \mathbb{R}^{d_{model} \times d_{fnn}}$, $W_2 \in \mathbb{R}^{d_{fnn} \times d_{model}}$, and b_1, b_2 are the corresponding biases. This type of feed-forward network adjusts the Swish activation function for a given batch of tokens, giving the model flexibility to activate neurons based on the weight values and their corresponding relation to other tokens.

The SwiGLU activation function is defined as:

$$SwiGLU(x) = (xW + b) \odot \sigma(xW + b). \quad (10)$$

3.1.5 CLS Token Head Inference

To speed up the inference process of the transformer and using the global token representation CLS Token, the output of the model is calculated using this $1 \times 1 \times d_{model}$ token and using a last linear transformation to get from our global context token to the desired number of outputs. As shown in Eq. (11), the CLS Token is transformed from a $1 \times 1 \times d_{model}$-dimensional vector to the number of outputs:

$$\hat{y}_{p,j} = CLS_{token}W + b, \quad (11)$$

where $W \in \mathbb{R}^{d_{model} \times j}$ and b is a bias vector.

3.2 Auxiliary Contrastive Loss

Since every module of the proposed model is fully differentiable, it is possible to adjust all parameters, considering the overall error signal from the output to the very first module, the fuzzy system. We saw that for a considerable number of fuzzy rules (i.e., 10 or more) and a large depth of d_{model}-dimensional tokens, the training process becomes unstable and the gradient of the fuzzy parameters is a very faint signal. Since the Gaussian membership functions of the fuzzy rules are sensitive to small changes and the gradient that flows to the very start of the model suffers a numerous set of operations from the Deep Transformer architecture, we need to help the fuzzy system to better adjust and improve these faint gradient signals. Taking inspiration from the technique to repel or attract similar pairs for a given output value, we use a variation of the contrastive learning used in self-supervised tasks[7]. Using the cosine similarity to calculate the distance between the activated firing strengths for a given output, the idea is to activate similar firing strengths for similar outputs and deactivate

or separate dissimilar firing strengths when the output isn't the correct one. This auxiliary loss is aggregated to the overall loss of the main task (i.e., regression or classification); Eq. (12) shows the addition and a new hyperparameter λ_{aux} to scale the difference:

$$\mathcal{L}_{total} = \mathcal{L}(\mathcal{W}) + \lambda_{aux} \cdot \mathcal{L}_{aux}(\phi; Y), \tag{12}$$

where $\mathcal{L}(\mathcal{W})$ is the main loss of the model, $\mathcal{L}_{aux}(\phi; Y)$ is the auxiliary loss function which requires the normalized firing strengths ϕ of the calculated batch and Y is the current batch target used to find the similarity of the firing strength for a given output.

This auxiliary loss is calculated over a batch of B samples. For every pair of samples (i, j) in the batch, first we define a similarity flag S_{ij} which is 1 when similar output targets are approximately the same, and 0 if they are dissimilar. We define the distance between their normalized firing strengths ϕ_i and ϕ_j using cosine distance:

$$d_\phi(i, j) = 1 - \frac{\phi_i \cdot \phi_j}{\|\phi_i\| \, \|\phi_j\|}, \tag{13}$$

where the indices i, j indicates pairs of the firing strengths, the similarity calculates the dot-product between the pair and its divided by their magnitudes.

The contrastive loss $Loss_{aux}$ is then defined as the sum of an attraction loss for similar pairs and a repulsion loss for all dissimilar pairs as shown in Eq. (14)

$$\begin{aligned} \mathcal{L}_{aux} = & \frac{1}{N_{\text{sim}}} \sum_{i,j|S_{ij}=1} d_\phi(i, j) \\ & + \frac{1}{N_{\text{dis}}} \sum_{i,j|S_{ij}=0} \max(0, m - d_\phi(i, j)) \end{aligned} \tag{14}$$

Where N_{sim} and N_{dis} are the total number of similar and dissimilar pairs, respectively, $d_\phi(i, j)$ indicates the distance between a pair of firing strengths and m is a similarity threshold used to indicate how dissimilar is a target with respect of the distance. This function attracts the firing strengths representations for similar-target input pairs and repels all the firing strength representations for dissimilar-target inputs.

3.3 An Overview on the Backpropagation Algorithm

The loss function is selected by the type of task we are solving for, to showcase the backpropagation algorithm we will take for example a regression problem, the loss function then will be defined by the sum of the squared differences between the calculated output and the targets, the SSE equation is shown in Eq. (17):

$$e_{p,j} = y_{p,j} - \hat{y}_{p,j}, \tag{15}$$

$$E_p = \sum_{i=1}^{j} e_{p,j}^2, \tag{16}$$

$$SSE = \sum_{p=1}^{q} E_p = \sum_{p=1}^{q} \sum_{i=1}^{j} e_{p,j}^2, \tag{17}$$

where $e_{p,j}$ indicates the j_{th} output of the model by a batch os samples p, $y_{p,j}$ the target for the j_{th} output and $\hat{y_{p,j}}$ the output of the model.

With this, we then redefine the loss function as the squared Frobenius norm:

$$\mathcal{L}(\mathcal{W}) = SSE = \|\mathbf{e}\|_F^2, \tag{18}$$

where **e** is the vectorized batch of errors given by $\vec{e}\vec{e}^T$.

As the last step to calculate the total loss, we need to add the auxiliary function:

$$\mathcal{L}(\mathcal{W}; \phi) = \mathcal{L}(\mathcal{W}) + \lambda_{aux} \cdot \mathcal{L}_{aux}(\phi; Y), \tag{19}$$

where $\mathcal{W}$ is the final model prediction or the estimation, which depends on all N transformer blocks, $\phi \in \mathbb{R}^{p \times k}$ is the normalized firing strengths, Y are the targets and λ_{aux} is a hyperparameter controlling the auxiliary loss weight.

The total gradient for any given set of parameters θ of the model is:

$$\frac{\partial \mathcal{L}(\mathcal{W}; \phi)}{\partial \theta} = \frac{\partial \mathcal{L}(\mathcal{W})}{\partial \theta} + \lambda_{aux} \cdot \frac{\partial \mathcal{L}_{aux}(\phi; Y)}{\partial \theta}. \tag{20}$$

This equation reveals a key architectural property; different parameters groups receive different gradient signals.

3.3.1 Gradient for Transformer and Embedding Parameters W

The transformer parameters W_{trans} (W^K_{attn}, W^Q_{attn}, W^V_{attn}, W^O, FFN weights, feature tokenizer and $\mathbf{e}_{CLS}$ Token) do not affect the fuzzy firing strengths ϕ. Therefore these parameters do not participate in the computation of $\mathcal{L}_{aux}$ and the gradient from the auxiliary loss is zero:

$$\frac{\partial \mathcal{L}_{aux}}{\partial W} = 0. \tag{21}$$

These parameters receive gradients only from the main task loss. For a specific set of weight W^ℓ in the ℓ-th transformer block, the gradient flows backward through all subsequent blocks, that is from the output $\hat{y}$ to the ℓ-th block. Let H^j be the output of block j and E_N be the final transformer output fed to the prediction head. The chain rule applies as follows:

$$\frac{\partial \mathcal{L}(\mathcal{W})}{\partial W^\ell} = \frac{\partial \mathcal{L}(\mathcal{W})}{\partial \hat{y}_{p,j}} \cdot \frac{\partial \hat{y_{p,j}}}{\partial E_N} \cdot \left(\prod_{j=\ell+1}^{N} \frac{\partial H^j}{\partial H^{j-1}} \right) \cdot \frac{\partial H^\ell}{\partial W^\ell}, \tag{22}$$

where we calculate the gradient of the loss with respect to the given j-th output then by the gradient of the j-th output with respect of the E_N output of the transformer (i.e., the CLS token output) and then we reach the calculation of all the subsequent Transformer blocks given by $\prod_{j=\ell+1}^{N} \frac{\partial H^j}{\partial H^{j-1}}$, where this operation is the product of all Jacobians from all the blocks above block ℓ and finally the gradient of the H^ℓ block with respect with the actual parameter W^ℓ. Since the transformer receives an embedding vector as input, we need to calculate the gradient with respect to all elements of the vector and the current block, that is why the need of the Jacobian of each block. One important aspect of the transformer is the residual connections of the form $H^j = H^{j-1} + f(H^{j-1})$. Therefore, the Jacobian includes an identity component:

$$\frac{\partial H^j}{\partial H^{j-1}} = \mathbf{I} + \frac{\partial f}{\partial H^{j-1}}, \tag{23}$$

where $\mathbf{I}$ is the identity Matrix which connects each block and its residual, f represents a series of operations for the givenℓ-th block . This connection helps mitigate vanishing gradients of deep transformer networks, but the signal reaching the fuzzy system is still weak.

3.3.2 Parameters for the Fuzzy System

The fuzzy system parameters (Gaussian centers $m_{k,i}$ and standard deviations $\sigma_{k,i}$) receive gradients from both losses:

$$\frac{\partial \mathcal{L}_{total}}{\partial m_{k,i}} = \frac{\partial L(\mathcal{W})}{\partial m_{k,i}} + \lambda_{aux} \cdot \frac{\partial \mathcal{L}_{aux}}{\partial m_{k,i}}, \tag{24}$$

$$\frac{\partial \mathcal{L}_{total}}{\partial \sigma_{k,i}} = \frac{\partial L(\mathcal{W})}{\partial \sigma_{k,i}} + \lambda_{aux} \cdot \frac{\partial \mathcal{L}_{aux}}{\partial \sigma_{k,i}}, \tag{25}$$

where $\mathcal{L}(\mathcal{W})$ is the main loss of the system which involves the N Transformer blocks and λ_{aux} the weight scale hyperparameter.

As demonstrated in the model architecture, our proposal adds a fuzzy modulation in the attention output. After computing multi-head attention, we apply element-wise multiplication with the normalized firing strengths as stated on Eq. 8. This creates a branch in the gradient flow. By the chain rule for element-wise multiplication:

The gradient with respect to the attention output is:

$$\frac{\partial L(\mathcal{W})}{\partial X_{attn}} = \frac{\partial L(\mathcal{W})}{\partial X_{mod}} \cdot \phi, \tag{26}$$

where X_{mod} is the attention output modulated by the normalized firing strengths phi. The gradient with respect to the firing strengths from the main loss:

$$\frac{\partial L(\mathcal{W})}{\partial \phi} = \sum_{q,d_{model}} \frac{\partial \mathcal{L}}{\partial X_{mod}} \cdot X_{attn}, \tag{27}$$

where X_{attn} is the attention probabilities before modulation and X_{mod} the attention probabilities after modulation, therefore the sum of all elements of the attention logits will calculate the gradient for ϕ.

Equation 27 represents the gradient signal from the main task that flows to the fuzzy system. However as we saw in Eq. (22), this gradient has passed through N transformer blocks, making it weak and noisy, that's the motivation of adding an auxiliary loss.

The gradient $\frac{\partial \mathcal{L}(\mathcal{W})}{\partial \phi}$ needs to propagate to the fuzzy system to update the parameters, flowing from the normalization to the very calculation of the firing strengths. Starting from the calculated normalized firing strengths, using the quotient rule, the gradient with respect to the raw firing strength is:

$$\frac{\partial \mathcal{L}}{\partial \alpha_{p,k}} = \frac{1}{\sum_{l=1}^{r} \alpha_{p,l}} \left[\frac{\partial \mathcal{L}}{\partial \phi_{p,k}} - \phi_{p,k} \sum_{l=1}^{k} \frac{\partial \mathcal{L}}{\partial \phi_{p,j}} \right], \tag{28}$$

where we take into account two sections of the firing strengths, where $\alpha_{p,k}$ is in the sum and when there is not, therefore the gradient of the raw firing strengths accounts the normalized firing strengths ϕ and the path from the main loss $\mathcal{L}$ to ϕ and.

Now that we have the gradient for the raw firing strength, we must flow back to the actual parameters, calculating the gradient for the loss with respect to the membership functions we get:

$$\frac{\partial L}{\partial e^{-1\frac{1}{2}\left(\frac{x_{p,i}-m_{k,i}}{\sigma_{k,i}}\right)}} = -\frac{\partial L}{\partial \alpha_{p,k}} \cdot \alpha_{p,k} \cdot e^{-1\frac{1}{2}\left(\frac{x_{p,i}-m_{k,i}}{\sigma_{k,i}}\right)}, \tag{29}$$

where $e^{-1\frac{1}{2}\left(\frac{x_{p,i}-m_{k,i}}{\sigma_{k,i}}\right)}$ is the Gaussian function for the parameters of the k-th rule and the i-th input.

Finally, we compute the gradients with respect to the Gaussian parameters. For the center we get:

$$\frac{\partial \mathcal{L}}{\partial m_{k,i}} = \sum_{p=1}^{q} \frac{\partial \mathcal{L}}{\partial \alpha_{p,k}} \cdot \alpha_{p,k} \cdot \left(\frac{x_{p,i} - m_{k,i}}{\sigma_{k,i}^2} \right). \tag{30}$$

For the standard deviations:

$$\frac{\partial \mathcal{L}}{\partial \sigma_{k,i}} = \sum_{p=1}^{q} \frac{\partial \mathcal{L}}{\partial \alpha_{p,k}} \cdot \alpha_{p,k} \cdot \left(\frac{(x_{p,i} - m_{k,i})^2}{\sigma_{k,i}^3} \right). \tag{31}$$

Table 1. Description of the benchmark datasets used for experimentation

Name	Task	# In	# Out	#Samples
EB	R	2	2	119
CCS	R	8	1	1030
ASN	R	5	1	1503
EE	R	8	2	768
Iris	C	4	3	150
BCD	C	30	2	569
Derm	C	8	6	366
Seeds	C	7	2	210

Both equations 30 and 31 sums over the batch of samples and takes into account the raw firing strengths $\alpha_{p,k}$ and the derivate of its gaussian function respectively.

The auxiliary loss provides gradients $\frac{\partial L_{aux}}{\partial \phi}$ that encourage similar samples to activate similar fuzzy rules while separating dissimilar samples. This gradient flows directly form the total loss to the fuzzy system, evading the deep and complex operations done during the transformer, this flow goes trough the same chain as the main loss (Eq. (28, 30 and 31).

4 Experiments and Results

To test our proposed Deep Fuzzy Transformer model, 8 benchmark datasets were used, 4 for system identification or regression tasks and 4 for class labeling or classification tasks. The selected datasets for regression are: engine behavior dataset[23], with 2 inputs (fuel rate, speed) and 2 outputs (torque, nitrous oxide emissions) with 1199 samples; airfoil self-noise[23] with 5 inputs (frequency, angle of attack, chord length, free-stream velocity, and suction side displacement thickness) and 1 output (scaled sound pressure level), with 1503 samples; concrete compressive strength[23], with 8 inputs (cement, blast furnace slag, fly ash, water, superplasticizer, coarse aggregate, fine aggregate, and age) and 1 output (concrete compressive strength), with 1030 samples; energy efficiency[23], with 8 inputs (relative compactness,

Table 2. Results for all regression datasets for our proposed Model, using KFold Cross Validation, where R^2, MSE, number of rules (r), number of parameters (k), and AIC are shown

Dataset Name	Performance measurements	Fuzzy rules									
		1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
Engine dataset	R² (output 1)	0.996116	0.996089	**0.997350**	0.997307	0.997177	0.997343	0.996985	0.995243	0.995255	0.994716
	std(R²)	0.000185	0.000367	0.000288	0.000354	0.000018	0.000237	0.000578	0.000277	0.001146	0.000535
	R² (output 2)	0.987303	0.985474	**0.988342**	0.988811	0.986520	0.986138	0.984264	0.982802	0.977034	0.974669
	std(R²)	0.001101	0.004043	0.000819	0.000904	0.000631	0.000740	0.000114	0.001558	0.000425	0.001169
	MSE	**1.077150**	1.771005	2.803565	4.378993	6.462693	9.086970	12.096276	15.650375	19.674759	24.078118
	std(MSE)	0.041942	0.037392	0.020352	0.017030	0.021392	0.022895	0.014844	0.006586	0.023141	0.039975
	k	218	866	1886	3250	5042	7226	9718	12674	16022	19642
	AIC	-**2.049649**	8.792409	25.604826	48.333859	78.391383	114.898030	156.574773	206.035425	262.124295	322.611996
Concrete compressive strength	R² (output 1)	0.843221	0.881204	0.888814	0.906144	0.904255	0.908338	0.908971	0.906003	0.906403	**0.907900**
	std(R²)	0.001503	0.003773	0.002023	0.001840	0.010113	0.004171	0.005999	0.005470	0.006038	0.007485
	MSE	1.005303	**0.889960**	1.206336	1.659960	2.365159	3.203627	4.149817	5.296471	6.594382	7.935352
	std(MSE)	0.000375	0.014806	0.006053	0.015741	0.001916	0.003840	0.007268	0.005765	0.006779	0.006279
	k	127	404	795	1276	1910	2670	3493	4496	5625	6790
	AIC	-**2.553066**	2.520080	10.045368	19.186560	31.530354	46.242955	62.237861	81.740768	103.635751	128.230344
Airfoil Self-Noise	R² (output 1)	0.841018	0.919628	0.928531	0.941835	0.947124	0.947679	0.945374	0.947451	**0.948457**	0.946891
	std(R²)	0.001018	0.002378	0.002153	0.001045	0.002058	0.002671	0.001316	0.001280	0.001675	0.000964
	MSE	11.520686	5.966031	6.269775	**5.761453**	6.504678	7.515245	9.154638	10.952622	13.093875	11.346130
	std(MSE)	0.094290	0.250552	0.149787	0.059399	0.032111	0.104386	0.128338	0.138600	0.058835	0.116290
	k	146	532	1098	1816	2770	3924	5194	6736	8478	10300
	AIC	-0.615151	**3.810078**	11.226369	20.564208	33.178923	48.517293	65.468519	85.948636	109.112625	133.389275
Energy Efficiency	R² (output 1)	0.780950	0.994184	0.995462	0.995401	0.996413	0.996574	0.997141	0.997281	0.997014	**0.997340**
	std(R²)	0.006931	0.000049	0.000494	0.000908	0.000268	0.000703	0.000384	0.000393	0.000689	0.000200
	R² (output 2)	0.791442	0.986656	0.981529	0.989498	0.991309	0.993328	0.993987	0.993564	**0.994959**	0.993847
	std(R²)	0.008423	0.000326	0.001555	0.001478	0.000916	0.001307	0.000725	0.000333	0.000377	0.000659
	MSE	7.365363	1.047408	**1.025423**	1.402556	1.882504	2.595595	3.370338	4.266227	5.320111	6.434367
	std(MSE)	0.248235	0.003820	0.041096	0.022611	0.005025	0.004377	0.000669	0.001891	0.002655	0.001459
	k	116	336	662	1066	1562	2188	2874	3652	4578	5546
	AIC	0.221397	**3.565704**	11.359731	21.565065	34.251763	50.352687	68.117369	88.411851	112.389673	137.725524

ISSN 2007-9737

Table 3. Regression datasets performance comparison between our proposed model, a baseline MLP and a High Order Polynomial Sugeno Inference System (HOP). $R^2 \pm std(R^2)$ is used for comparison.

Dataset	Performance measurements	Model				
		MLP - S	MLP - M	MLP-L	HOPS	DFT
Engine Behavior	R^2 (output 1)	0.121749	0.066771	0.526575	**0.999456**	0.990116
	std(R^2)	0.072594	0.106023	0.074725	0.000598	0.008185
	R^2 (output 2)	-0.817194	-1.046939	-0.151957	0.962426	**0.987303**
	std(R^2)	0.102218	0.228095	0.215789	0.064938	0.004101
	AIC	41.5706	152.185	715.263	9.287285	**-2.049649**
Concrete Compressive Strength	R^2 (output 1)	0.663619	0.696015	0.704437	0.716875	**0.849221**
	std(R^2)	0.013052	0.001181	0.006036	0.128287	0.001503
	AIC	51.6914	187.046	858.078	0.840264	**-2.553066**
Airfoil Self-Noise	R^2 (output 1)	0.103814	0.383701	0.529789	0.677552	**0.919628**
	std(R^2)	0.022633	0.011711	0.007458	0.242973	0.002378
	AIC	34.863	124.822	579.347	3.162669	**3.810078**
Energy Efficiency	R^2 (output 1)	0.439066	0.330556	0.672490	**0.996202**	0.994184
	std(R^2)	0.120999	0.075312	0.017171	0.006229	0.000049
	R^2 (output 2)	0.637542	0.640419	0.659857	0.983413	**0.986656**
	std(R^2)	0.068427	0.017399	0.004713	0.004507	0.000326
	AIC	73.6597	256.286	1156.02	**0.855542**	3.565704

Table 4. Results for all classification datasets for our proposed Model, using Stratified KFold Cross Validation, where Cross Entropy Loss, Accuracy, Precision and F1-Score are shown

Dataset	Performance measurements	Loss (CEL)	Accuracy	Precision	F1-Score
Iris	Mean	3.972975	0.971111	0.975926	0.970819
	Min	3.817929	0.966667	0.972222	0.966330
	Max	4.235186	0.973333	0.977778	0.973064
	Std	0.186438	0.003143	0.002619	0.003174
Breast Cancer Diagnosis	Mean	2.811211	0.975982	0.976838	0.975860
	Min	2.547836	0.973622	0.974957	0.973497
	Max	2.981546	0.977162	0.978018	0.977055
	Std	0.188889	0.001669	0.001344	0.001671
Dermatology	Mean	0.047473	0.970896	0.976158	0.970300
	Min	0.007446	0.961937	0.969477	0.961252
	Max	0.094135	0.978003	0.981051	0.977322
	Std	0.035700	0.006688	0.004891	0.006715
Seeds	Mean	1.157935	0.942857	0.947751	0.941936
	Min	0.738016	0.938095	0.942989	0.936755
	Max	1.806229	0.947619	0.952976	0.946545
	Std	0.465035	0.003888	0.004090	0.004017

Table 5. Classification datasets performance comparison between our proposed model and a baseline MLP

Dataset	Performance Metrics	Model			
		MLP - S	MLP - M	MLP-L	DFT
Iris	Accuracy	0.957778	0.955556	0.951111	**0.971111**
	Precision	0.964392	0.962222	0.958095	**0.975926**
	F1-Score	0.957160	0.955152	0.950471	**0.970819**
Breast Cancer Diagnosis	Accuracy	0.967774	0.973058	0.970729	**0.975982**
	Precision	0.959494	0.973641	0.972076	**0.976838**
	F1-Score	0.967742	0.972986	0.970548	**0.975860**
Dermatology	Accuracy	0.970986	0.970896	0.967217	**0.970896**
	Precision	0.973793	0.974477	0.971327	**0.976158**
	F1-Score	0.970596	0.970489	0.966268	**0.970300**
Seeds	Accuracy	**0.957143**	0.953968	0.946032	0.942857
	Precision	**0.963536**	0.959696	0.951019	0.947751
	F1-Score	**0.956447**	0.955315	0.945007	0.941936

surface area, wall area, roof area, overall height, orientation, glazing area, and glazing area distribution) and 2 outputs (heating load, and cooling load), with 768 samples.

The selected datasets for classification are: iris dataset[23], with 4 inputs (sepal length, sepal width, petal length and petal width) and 3 identified classes (setosa, versicolour, and virginica); breast cancer diagnosis "Winsconsin"[23] with 30 inputs and 2 identified classes (malignant, benign);dermatology dataset with 8 inputs and 6 identified classes (psoriasis, seboreic dermatitis, lichen planus, pityriasis rosea, cronic dermatitis, pityriasis rubra pilaris) and seeds datasets with 7 inputs (area A, perimeter P, compactness C, length of kernel, width of kernel, asymmetry coefficient and length of kernel groove.) and 3 identified classes (Kama, Rosa and Canadian). These datasets characteristics are shown in Table 1. Using abbreviations for constraining purposes, the abbreviations for the type of task are Regression Task (R) and Classification Task (C), and the datasets are abbreviated as Engine Behavior (EB), Concrete Compressive Strength (CCS), Airfoil Self-Noise (ASN), Energy Efficiency (EE), Iris, Breast Cancer Diagnosis (BCD), Dermatology (Derm), and Seeds, respectively.

A series of tests for each dataset were executed, for regression tasks we did an scaling analysis going from r=1,..., 10 rules, the idea behind it is to show how the model performs when incrementing the number of fuzzy rules and the overall parameters of the system. We did a K-fold cross validation with K-fold=10 with 3 repetitions, an overall of over 30 models per rule per dataset. The performance measurements used are: Coefficient of determination R^2 and Mean Square Error (MSE). An important measure we added is the Akaike Information Criterion (AIC), shown in Eq. (32), where k is the number of parameters, q is the amount of data pairs and L the like hood function maximized value, where L=MSE. This criterion gives and estimation of the overall model complexity and the performance, so is measure by the lowest possible value, that is the lower the value, the better the obtainer model performance:

$$AIC = \frac{2k}{q} + log(L). \tag{32}$$

For classification tasks, we follow a similar approach, using a variation of KFold, we select Stratified KFold as the main validation procedure with KFold=10. The performance measurements used are: Accuracy, Precision, and F1 Score. Since the number of rules of the fuzzy system is the number of classes to identify, we did not perform a analysis scaling like for the regression task.

Finally, we did a comparison between 2 models, a baseline MLP model with 3 scales (i.e., small, medium and large) and a High Order Polynomial Sugeno Fuzzy Inference System to make a starting point of this new approach. We did the same number of experiments for every model, we will compare our best model for each dataset with this baselines implementations.

5 Discussions and Future Work

A collection of multiple regression and classification datasets were used to conduct our experiments and verify the performance of our proposed model. Tables 2-4 shows all results for regression task from the obtained R^2, MSE, number of rules (r), number of parameters (k) and AIC. Tables 3-5 shows a comparison between our model, a baseline MLP model and a High Order Polynomial Sugeno FIS. We performed one exhaustive experimentation using KFold cross validation for regression, stratified KFold cross validation for classification, with 10 fold, in regression we performed a scaling analysis from the fuzzy rules from 1 to 10, repeating the 10 Folds per rule, that's a total of 30 models per rule per dataset, as in classification the number of rules were the number of class labels to predict, we did every Fold 3 times, 30 models per dataset were also done.

Our experimental evaluation demonstrates that the proposed model Deep Fuzzy Transformer achieves better performance compared to standard MLP baselines across regression and a small but notable improvement in classification tasks, on the other hand compared with a High Order Polynomial Sugeno, we had a slightly better improvement.

For regression tasks, we will discuss Table 2. For the engine behavior and Concrete Compressive Strength showed that 1 fuzzy rule is sufficient, although the coefficient of determination improved as the number of fuzzy rules increased, the overall number of parameter did too and as we showed before, the AIC metric is a good baseline to compare models due to the fact that relates the overall loss value and the number of parameters required, in those two datasets the selected model had the lower value among all rules. Compared with the baselines MLP's and the HOPS FIS, it achieves a better average performance. For the Airfoil Self-Noise and Energy Efficiency dataset showed that 2 is a good enough number of fuzzy rules to handle the complexity of the data. In the case of the Airfoil data, 1 fuzzy rule is not enough to capture and represent the overall non-linearity of the patterns, and although we outperform the other models, it was possible to achieve an even greater performance with 2 rules, after 5 rules the coefficient plateaus while the AIC metric increased. This indicates that 2 rules is more than enough and adding rules just leads to an unnecessary and complex model. For the Energy Efficiency it shows that 1 fuzzy rule is not enough to really capture information and it might need more parameters to improve the performance, as the results pointed out, after we get to 5 fuzzy rules both outputs plateaus, so given the large gap in parameters between models, 2 fuzzy rules is the best spot to select, as the parameter count is relatively small and the performance is very similar, compared to the other models, we outperform the MLP baseline by a large margin, and with the HOPS we achieve a significantly better performance, which justifies the increase in model complexity and the resulting higher AIC.

For classification tasks, compared with the baselines MLP, we got a slight better results in two datasets, the other ones we have very similar results, but the increment is modest, compared to the number of parameters the transformer deals with, as we said before we over 30 models per dataset, adding the fact that we have to match the number of classes to the number of heads of the attention mechanism, its hard to tell the winner in each case. The iris dataset is a very common benchmark and the relative performance of both model were pretty similar, having the slight advantage our model, in which case it could be to the fact that this dataset has a good distribution between classes. The Breast Cancer Diagnosis is another common benchmark but it has unbalance in the distribution of the predicted classes, but with the stratified KFold we tried to reduce the casualties, the overall performance on the Accuracy, Precision, and F1-Score were good. We select the dermatology dataset due to the fact of the number of classes which are 6, so it supposed a more complex solution, our model handles all classes with a very good estimation, as the F1-Score metrics calculates the precision, false positives, false negatives and calculates the weigthed result.

With the auxiliary loss talked previously, it helped achieve better results, we noted that as the number of parameters and the number of rules increases, the training process became unstable after a long time, giving a vanishing gradient problem or gradient spikes, which affected the initial set of tests we performed. When adding the contrastive loss for the firing strength the fuzzy system adjusted better the fuzzy parameters of the Gaussian membership functions. While the main task loss provides task specific learning signal, the auxiliary loss ensures that fuzzy rules organize inputs by similarity, creating a structured representation of the data and its patterns.

Compared to classical neuro-fuzzy systems like ANFIS, our architecture addresses the scalability limitations by integrating the Transformer architecture for large contextual relationships in the data. Resulting in adding a new approach to better interpret the reasoning process inside the attention mechanism while maintaining competitive performance. The auxiliary function also proved to stabilize the overall training process, making it an important feature to future integrations of fuzzy systems and deep learning models.

Several limitations warrant discussion. Our evaluation is limited to the 8 benchmark datasets we tried in this research, but further experimentation and applications in other domains might strengthen its generalization and adaptability capabilities of the proposed model. We also

compare with a baseline MLP and just one FIS implementation, but for an extensive search for competitive performance might require to try other models or maybe new transformer architectures. Lastly the lambda weight for the auxiliary loss is dataset-dependant, which might increase the number of hyperparameter to search for, another thing that deep learning model struggle with, the search and optimization of these hyper-parameters.

Despite these limitations, and given the extensive series of experiments, our models confirms that using a fuzzy rule based modulation by a neuro-fuzzy system provides a new approach combining the fuzzy's system interpretability while maintain complex relations using the Transformer architecture. The dual gradient for the fuzzy system proves to be a good approach to handle deep architectures and stabilize sensitive parameters (i.e the fuzzy system) while maintaining a pretty good competitive performance.

For future work, we are working on doing more experiments trying with different embeddings, types of data and still comparing with other models. We will work in new ways to improve this hybrid model, from trying different style architectures, to even change to a modern state-of-the-art attention, as this represent a new approach to neuro-fuzzy transformers, there is so much to discover and improve that who knows, perhaps in a future work we improve this first model. As new architectures and techniques arises, we will try it and revel our findings. We are confident that the fuzzy logic and fuzzy system are a safe path towards solving one of the main problems in AI, intrerpretability.

Acknowledgments

We would like to thank the MyDCI program of the Division of Graduate Studies and Research, UABC, and Tijuana Institute of Technology for his consideration during the Workshop *World Health and Artificial Intelligence* (WHAI) and his support throughout the creation of this paper. Lastly thank the financial support provided by SECIHTI with his scholarships for Graduate programs and studies, CVU number: 2015688.

References

1. **Agarap, A. F. (2019).** Deep Learning using Rectified Linear Units (ReLU). DOI: 10.48550/arXiv.1803.08375. ArXiv:1803.08375.
2. **Babanezhad, M., Nakhjiri, A. T., Marjani, A., Shirazian, S. (2020).** Pattern recognition of the fluid flow in a 3D domain by combination of Lattice Boltzmann and ANFIS methods. Scientific Reports, Vol. 10, No. 1, pp. 15908. DOI: 10.1038/s41598-020-72926-3.
3. **Bai, J. (2024).** Sparse Attention Mechanisms in Large Language Models: Applications, Classification, Performance Analysis, and Optimization. Advances in Computer, Signals and Systems, Vol. 8, No. 6, pp. 130–136. DOI: 10.23977/acss.2024.080618.
4. **Bezdek, J. C., Ehrlich, R., Full, W. (1984).** FCM: The fuzzy c-means clustering algorithm. Computers & Geosciences, Vol. 10, No. 2-3, pp. 191–203. DOI: 10.1016/0098-3004(84)90020-7.
5. **Brohan, A., Brown, N., Carbajal, J., Chebotar, Y., Dabis, J., Finn, C., Gopalakrishnan, K., Hausman, K., Herzog, A., Hsu, J., Ibarz, J., Ichter, B., Irpan, A., Jackson, T., Jesmonth, S., Joshi, N. J., Julian, R., Kalashnikov, D., Kuang, Y., Leal, I., Lee, K.-H., Levine, S., Lu, Y., Malla, U., Manjunath, D., Mordatch, I., Nachum, O., Parada, C., Peralta, J., Perez, E., Pertsch, K., Quiambao, J., Rao, K., Ryoo, M., Salazar, G., Sanketi, P., Sayed, K., Singh, J., Sontakke, S., Stone, A., Tan, C., Tran, H., Vanhoucke, V., Vega, S., Vuong, Q., Xia, F., Xiao, T., Xu, P., Xu, S., Yu, T., Zitkovich, B. (2023).** RT-1: Robotics Transformer for Real-World Control at Scale. DOI: 10.48550/arXiv.2212.06817. ArXiv:2212.06817.
6. **Castillo, O., Melin, P., Valdez, F., Gonzalez, C., Garcia, M., Mancilla, A., Cortes-Antonio, P., Soria, J. (2025).** A Review on the Role of Fuzzy Logic in Hybrid Intelligent Systems. Computación y Sistemas, Vol. 29, No. 3. DOI: 10.13053/cys-29-3-5897.

7. **Chen, T., Kornblith, S., Norouzi, M., Hinton, G. (2020).** A Simple Framework for Contrastive Learning of Visual Representations. DOI: 10.48550/arXiv.2002.05709. ArXiv:2002.05709.
8. **Chen, X., Liang, C., Huang, D., Real, E., Wang, K., Liu, Y., Pham, H., Dong, X., Luong, T., Hsieh, C.-J., Lu, Y., Le, Q. V. (2023).** Symbolic Discovery of Optimization Algorithms. DOI: 10.48550/arXiv.2302.06675. ArXiv:2302.06675.
9. **Cui, Y., Wu, D., Xu, Y. (2021).** Curse of Dimensionality for TSK Fuzzy Neural Networks: Explanation and Solutions. DOI: 10.48550/arXiv.2102.04271. ArXiv:2102.04271.
10. **Cun, Y. L., Boser, B., Denker, J. S., Howard, R. E., Habbard, W., Jackel, L. D., Henderson, D. (1990).** Handwritten digit recognition with a back-propagation network, chapter 49. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 396–404.
11. **Dalar, A. Z., Egrioglu, E. (2025).** Blending traditional and novel techniques: Hybrid type-1 fuzzy functions for forecasting. Engineering Applications of Artificial Intelligence, Vol. 148, pp. 110445. DOI: 10.1016/j.engappai.2025.110445.
12. **Dauphin, Y. N., Fan, A., Auli, M., Grangier, D. (2017).** Language Modeling with Gated Convolutional Networks. DOI: 10.48550/arXiv.1612.08083. ArXiv:1612.08083.
13. **De Medeiros, I. B., Soares Machado, M. A., Damasceno, W. J., Caldeira, A. M., Dos Santos, R. C., Da Silva Filho, J. B. (2017).** A Fuzzy Inference System to Support Medical Diagnosis in Real Time. Procedia Computer Science, Vol. 122, pp. 167–173. DOI: 10.1016/j.procs.2017.11.356.
14. **Denecke, K., May, R., Rivera-Romero, O. (2024).** Transformer Models in Healthcare: A Survey and Thematic Analysis of Potentials, Shortcomings and Risks. Journal of Medical Systems, Vol. 48, No. 1, pp. 23. DOI: 10.1007/s10916-024-02043-5.
15. **Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N. (2020).** An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. DOI: 10.48550/ARXIV.2010.11929.
16. **Funk, S., .** RMSprop loses to SMORMS3 - Beware the Epsilon! https://sifter.org/simon/journal/20150420.html.
17. **Hochreiter, S., Schmidhuber, J. (1997).** Long Short-Term Memory. Neural Computation, Vol. 9, No. 8, pp. 1735–1780. DOI: 10.1162/neco.1997.9.8.1735.
18. **Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W. (2021).** LoRA: Low-Rank Adaptation of Large Language Models. DOI: 10.48550/arXiv.2106.09685. ArXiv:2106.09685.
19. **Huang, X., Khetan, A., Cvitkovic, M., Karnin, Z. (2020).** TabTransformer: Tabular Data Modeling Using Contextual Embeddings. DOI: 10.48550/arXiv.2012.06678. ArXiv:2012.06678.
20. **Ismail, A. N., Prajitno, P., Adhitya, K. T. (2021).** Application of the adaptive neuro-fuzzy inference system (anfis) for simulating water fluid level control systems on horizontal separator. AIP Conference Proceedings, Vol. 2374, No. 1, pp. 020015. DOI: 10.1063/5.0059111.
21. **Jain, S., Wallace, B. C. (2019).** Attention is not Explanation. DOI: 10.48550/arXiv.1902.10186. ArXiv:1902.10186.
22. **Jang, J.-S. (1993).** ANFIS: adaptive-network-based fuzzy inference system. IEEE Transactions on Systems, Man, and Cybernetics, Vol. 23, No. 3, pp. 665–685. DOI: 10.1109/21.256541.
23. **Kelly, M., Longjohn, R., Nottingham, K. (2025).** UCI machine learning repository. https://archive.ics.uci.edu.
24. **Kingma, D. P., Ba, J. (2017).** Adam: A Method for Stochastic Optimization. DOI: 10.48550/arXiv.1412.6980. ArXiv:1412.6980.

25. **Loshchilov, I., Hutter, F. (2019).** Decoupled Weight Decay Regularization. DOI: 10.48550/arXiv.1711.05101. ArXiv:1711.05101.

26. **Lou, C., Jia, Z., Zheng, Z., Tu, K. (2024).** Sparser is Faster and Less is More: Efficient Sparse Attention for Long-Range Transformers. DOI: 10.48550/arXiv.2406.16747. ArXiv:2406.16747.

27. **Mamdani, E., Assilian, S. (1975).** An experiment in linguistic synthesis with a fuzzy logic controller. International Journal of Man-Machine Studies, Vol. 7, No. 1, pp. 1–13. DOI: 10.1016/S0020-7373(75)80002-2.

28. **Mandal, L., Harsha Vardhan, T. B., Narasimha Reddy, K. G., Yeswanth Reddy, D. S., Bal, S. (2025).** Edu Vault: An Interactive, Multilingual, and Intelligent Topic-Conscious Video Discovery System for Enhanced Conceptual Learning Using Advanced NLP Techniques. Computación y Sistemas, Vol. 29, No. 3. DOI: 10.13053/cys-29-3-5888.

29. **Mikolov, T., Chen, K., Corrado, G., Dean, J. (2013).** Efficient Estimation of Word Representations in Vector Space. DOI: 10.48550/arXiv.1301.3781. ArXiv:1301.3781.

30. **Minaee, S., Mikolov, T., Nikzad, N., Chenaghlu, M., Socher, R., Amatriain, X., Gao, J. (2025).** Large Language Models: A Survey. DOI: 10.48550/arXiv.2402.06196. ArXiv:2402.06196.

31. **Moré, J. J. (1978).** The Levenberg-Marquardt algorithm: Implementation and theory. **Watson, G. A.**, editor, Numerical Analysis, Springer, Berlin, Heidelberg, pp. 105–116. DOI: 10.1007/BFb0067700.

32. **Nürnberger, A., Nauck, D., Kruse, R. (1999).** Neuro-fuzzy control based on the NEFCON-model: recent developments. Soft Computing - A Fusion of Foundations, Methodologies and Applications, Vol. 2, No. 4, pp. 168–182. DOI: 10.1007/s005000050050.

33. **Pereira, G. A., Hussain, M. (2024).** A Review of Transformer-Based Models for Computer Vision Tasks: Capturing Global Context and Spatial Relationships. DOI: 10.48550/arXiv.2408.15178. ArXiv:2408.15178.

34. **Ramírez Cerna, L., Rodriguez Melquiades, J. A., Escobedo Cárdenas, E. J., Cámara Chávez, G., Garcia Miranda, D. (2025).** Facial Expressions Recognition in Sign Language Based on a Two-Stream Swin Transformer Model Integrating RGB and Texture Map Images. Computación y Sistemas, Vol. 29, No. 2. DOI: 10.13053/cys-29-2-5119.

35. **Rumelhart, D. E., McClelland, J. L., Group, P. R. (1986).** Parallel Distributed Processing, Volume 1: Explorations in the Microstructure of Cognition: Foundations. The MIT Press. DOI: 10.7551/mitpress/5236.001.0001.

36. **Salameh, T., Farag, M. M., Hamid, A.-K., Hussein, M. (2025).** Adaptive neuro-fuzzy inference system for accurate power forecasting for on-grid photovoltaic systems: A case study in Sharjah, UAE. Energy Conversion and Management: X, Vol. 26, pp. 100958. DOI: 10.1016/j.ecmx.2025.100958.

37. **Shazeer, N. (2020).** GLU Variants Improve Transformer. DOI: 10.48550/arXiv.2002.05202. ArXiv:2002.05202.

38. **Takagi, T., Sugeno, M. (1985).** Fuzzy identification of systems and its applications to modeling and control. IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-15, No. 1, pp. 116–132. DOI: 10.1109/TSMC.1985.6313399.

39. **TSUKAMOTO, Y. (1993).** An approach to fuzzy reasoning method. In **Dubois, D., Prade, H., Yager, R. R.**, editors, Readings in Fuzzy Sets for Intelligent Systems. Morgan Kaufmann, pp. 523–529. DOI: https://doi.org/10.1016/B978-1-4832-1450-4.50055-9.

40. **Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., Polosukhin, I. (2023).** Attention Is All You Need. DOI: 10.48550/arXiv.1706.03762. ArXiv:1706.03762.

41. Zadeh, L. (1965). Fuzzy sets. Information and Control, Vol. 8, No. 3, pp. 338–353. DOI: 10.1016/S0019-9958(65)90241-X.

42. Zaldivar-Herrera, J. E., Sánchez-Fernández, L. P., Rodríguez-Méndez, L. M., Zagaceta-Álvarez, M. T. (2024). A Fuzzy Inference Model for Evaluating Data Transfer in LTE Mobile Networks via Crowdsourced Data. Computación y Sistemas, Vol. 28, No. 3. DOI: 10.13053/cys-28-3-4826.

Article received on 31/10/2025; accepted on 15/12/2025.
**Corresponding author is Adrian E. Adame.*