# ADFA: Abstraction and Decision Fusion Architecture for Resource-Aware Intelligent Computing

***Abstract*—Resource-aware intelligent computing refers to the trade-off between the restrictions of computational resources and the inference power of the systems. ADFA addresses this challenge through a three-layer architecture; data abstraction, computation, and decision fusion. In the first layer, a set of abstract views of the input data are provided to reduce the computations required for processing them and supply enough information for the next layer. Then, an array of lightweight models in the computation layer process different views and make independent decisions. Finally, the output is produced using a decision fusion tool in the last layer. To evaluate the capability of the proposed architecture, we developed seven ADFA-based models for the classification of handwriting digits. In this regard, we first defined two data abstractions. Then, we trained a Support Vector Machine (SVM) and a three-layer Fully Connected Neural Network (FCN) according to the data abstractions, which led to four independent basic models. Finally, an Adaptive Neuro-Fuzzy Inference System (ANFIS) is employed as their hub to increase the accuracy by performing the decision fusion. Our experiments on the MNIST dataset verify the efficiency of the proposed architecture with an accuracy of 99% in recognizing handwritten digits, where the model size and the number of MAC operations are significantly smaller than state-of-the-art ensemble learning models, also deep neural networks trained on the same dataset.**

***Index Terms*—Data Abstraction, Decision Fusion, ANFIS, Classification.**

## I. Introduction

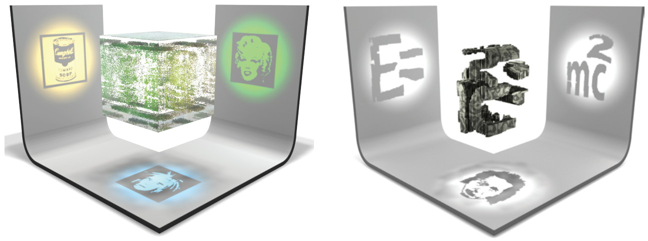
Intelligent computing comprises the development of advanced computational techniques and algorithms to solve problems and make decisions in the context of Artificial Intelligence (AI) and Machine Learning (ML) [1]–[5]. This concept has been followed by many applications in fields such as automation, predictive modeling, and natural language processing. It empowers the systems to learn from data, reason and make decisions, interact with humans, and continuously improve. However, high intelligence often embraces the processing of large amounts of information and execution of computationally intensive algorithms to solve problems, make decisions, and learn from experience [6]. In other words, there is usually a trade-off between the level of intelligence and the limitations of computation resources.

There are different approaches to enhancing the intelligence of systems. As a case in point, boosting productivity through the diversity of processes increases the possibility of achieving positive results. Therefore, when individual models that process the data from different approaches to training come together and share their information, it augments the intelligence of the system. Following this approach, Ensemble Learning [7]–[10] is a well-established machine learning technique that combines multiple models to improve the performance of a predictive model. The idea behind it is that by combining various models, each with its strengths and weaknesses, the resulting combined model can be more accurate and robust than any of the combined models. However, in ensemble learning, the main focus is on increasing accuracy as well as supporting diversity in the analysis of miscellaneous data, while resource-aware computing is not a top point of interest.

Reducing the volume of data to be processed [11] is an efficient approach in resource-aware computing. We have employed data abstraction as a high-potential approach to reducing the amount of data to be processed. It consists of focusing on data from defined points of view and representing them simplified to mitigate their complicatedness. Data abstraction plays a crucial role in the abbreviation of data, and can be achieved through various techniques such as data filtering [12], summarization [13], compression [14], quantization [15], and transformation and projection [16].

Although data abstraction can reduce the volume of data to be processed, it has several potential disadvantages, including information loss, over-simplification, the introduction of inaccuracies into the data, and the lack of data transparency, making it challenging to understand [17]. The multi-abstraction approach addresses these problems by providing multiple perspectives on data from diverse points of view [12], [18]. As an intuitive example, Fig.1 illustrates two sculptural arts where the $2D$ shadows are cast by two $3D$ sculptures [19]. In the first one, the object is a glass cube and the light source inside it has caused each part of the data to be projected with a special shape on a dimension (Fig.1a). In the second one, the object is opaque where some light sources at different angles, each one projects the data as a special shape on a dimension (Fig.1b). Here, by removing any of these projections, our understanding of the sculpture (data) becomes deficient. Therefore, keeping multiple perspectives can improve the model's understanding of the data and support intelligence.

Preparing multiple abstractions from data and processing them by different models may lead to distinctive decisions, where integrating them into a single result is called decision fusion [20], [21]. It is an important aspect of a collaborative system that can provide a way to combine the predictions of its subsystems into a single, more robust, and accurate decision.




(a) Sculpturing by glass cube. (b) Sculpturing by opaque object.

Fig. 1: Two shadow sculptures representing the importance of having variant points of view of objects [19].

In this paper, we present a three-layer architecture named ADFA with the aim of resource-aware intelligent computing. ADFA is based on data abstraction, processing by lightweight models, and decision fusion. Accordingly, to demonstrate the capability of the proposed architecture, we have developed seven ADFA-based models for the classification of handwriting digits. Also, we have utilized the power of ANFIS in the representation and manipulation of uncertainty in order to fuse multiple decisions.

The rest of the paper is organized as follows. Section II presents the basic concepts of SVM, FCN, and ANFIS. Then, the proposed architecture is demonstrated in Section III. Section IV explains our methodology for designing the experiment on the dataset, and experimental results are discussed in Section V. Finally, the paper is concluded in Section VI.

## II. Background and Preliminaries

*1) Support Vector Machine:* SVM is a supervised ML tool capable of solving various classification and regression problems. The primary concept underlying SVM is identifying a set of hyperplanes as decision boundaries in the feature space that achieves the maximum possible separation between the classes of data.

The kernel trick [22] enables SVM to handle problems that are not linearly separable. It maps the input data to a higher-dimensional space, where nonlinear hyperplanes can be used to classify input data. Suppose that $\{(x_1, y_1), \ldots, (x_m, y_m)\}$ is a training dataset, where each $x_i \in \mathcal{R}^n$ is a data point and $y_i \in \{-1, 1\}$ is its label. Eq.1 presents a nonlinear hyperplane that classifies the input data $x_s$ binarily, where $\alpha_i$ and $b$ are constant values, and $K(x_s, x_i)$ is a kernel function. The common kernels used in SVM are linear, polynomial, and radial basis functions (RBF). Here, Eq.2 is an RBF kernel function that $x_s$ and $x_i$ are its inputs and $\gamma$ is a constant value.

According to Eq.1 and Eq.2, the size of an SVM model, as well as the number of operations it performs to classify an input data, depends on the size of the training dataset. These features are crucial for making decisions about models in the context of resource-aware computing.

$$y_s = sgn\left(\sum_{i}^{n} \left(\alpha_i \; y_i \; K\left(x_s, x_i\right)\right) + b\right) \tag{1}$$

$$K(x_s, x_i) = exp\left(-\gamma \;\; \|x_s - x_i\|^2\right) \tag{2}$$

*2) Fully Connected Neural Network:* An FCN is a supervised ML tool highly applicable to solving classification and regression problems. It is a multi-layer network of artificial neurons such that each neuron receives all of the outputs in the previous layer as its input. Fig.2 illustrates the model of a basic artificial neuron that generates an output $y_j$ according to the input vector $x$.

The training of a neuron refers to the process of finding weights $w$ and bias $b$ for Eq.3 using a training dataset. Here, $\sigma$ is called the activation function.

$$y = \sigma(w.x + b) \tag{3}$$

Nonlinear activation functions play a crucial role in neural networks by introducing nonlinearities into the network. Eq.4 present the ReLU, which is a low-cost nonlinear activation function commonly used in different types of neural networks.

$$\sigma(z) = max(0, z) \tag{4}$$

In order to normalize the output of FCN, a Softmax activation function is applied on vector $y$ of the last layer. Eq.5 presents the Softmax activation function applied on a single output $y_s$.

$$\sigma(y_s) = \frac{exp(y_s)}{\sum_{j=1}^{j=n} exp(y_j)} \tag{5}$$

The size of an FCN model, as well as the number of operations it performs to classify input data, depends on the number of layers and the number of neurons in each layer. On the other hand, the number of neurons in the first layer of FCN is equal to the size of the input data and the number of neurons in the last layer is proportional to the number of possible outputs (the number of classes in a classification task). Also, the number of neurons in hidden layers can vary depending on the complexity of the problem, the size of the input data, and the specific architecture of the neural network. Therefore, reducing the complexity and size of the input data has a high impact on the size of the FCN model and the number of operations required for the classification.

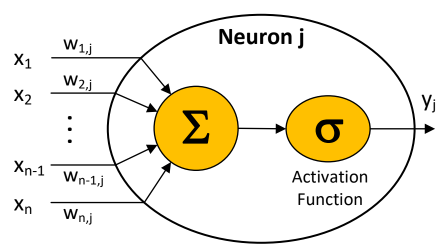



Fig. 2: The model of an artificial neuron.

*3) Adaptive Neuro-Fuzzy Inference System:* The ANFIS is a hybrid ML model that compounds the benefits of artificial neural networks and fuzzy logic to provide accurate and reliable predictions and classifications. It is a type of fuzzy inference system that uses neural networks to adjust and optimize its parameters for modeling complex relationships between input variables and output variables.

A fuzzy inference system can be implemented in different ways depending on the specific problem and data being used. The Sugeno model [23] uses a set of IF-THEN rules with linear consequences, while the Mamdani model [24] uses a set of IF-THEN rules with fuzzy consequences.

The Sugeno model is the basis of the ANFIS model, which consists of five layers [25]; the Fuzzification layer, Rule layer, Normalization layer, Consequent layer, and Defuzzification layer. The Fuzzification layer consists of a set of fuzzy sets that map the inputs to fuzzy membership degrees. Each fuzzy set is characterized by a membership function. Eq.6 presents a Gaussian membership function, where $\sigma_i$ and $c_i$ are the parameters obtained for each membership function $\mu_i$ during the AFIS training phase.

$$\mu_i(x, \sigma_i, c_i) = exp\left(-\frac{(x - c_i)^2}{2\sigma_i^2}\right). \tag{6}$$

The Rule layer contains a set of IF-THEN rules that generate the vector of weights $w$ called the firing strength of the rules. The rules combine all fuzzy linguistic values of different input variables, while each linguistic value is represented by a fuzzy membership function in the Fuzzification layer. Eq.7 presents the firing strength of the $r^{th}$ rule of an ANFIS model, where $R_r$ indicates the set of indices of its inputs.

$$w_r = \prod_{j \in R_r} \mu_j. \tag{7}$$

The Normalization layer normalizes vector $w$ of firing strengths by Eq.8.

$$\bar{w}_r = \frac{w_r}{\sum_i w_i}. \tag{8}$$

The Consequent layer consists of a set of adaptive nodes. The output of $i^{th}$ node in this layer is computed by Eq.9, which is the product of the normalized weights and a linear combination of the inputs. Here, each parameter $\alpha_{i,k}$ is obtained during the training of the ANFIS model.

$$y_i = \bar{w}_i \sum_k (\alpha_{i,k} x_k). \tag{9}$$

The Defuzzification layer consists of one single fixed node that performs the summation of all incoming signals by Eq10 to generate the output.

$$z = \sum_i y_i. \tag{10}$$

According to Eq.6 and Eq.9, the set of trainable parameters of an ANFIS model includes $c_i$ and $\sigma_i$ of each membership function $\mu_i$ as well as consequent parameters $\alpha_{i,k}$.

## III. Proposed Architecture

Our proposed three-layer architecture is presented in Fig.3. The abstraction layer consists of a set of transformations for organizing and presenting the input data from several views. Each view reflects a particular perspective or interpretation of the input data which is useful for processing, decision-making, and problem-solving by a lightweight computation model. In the Computation layer, a set of lightweight models process their corresponding views. According to each model's perspective on the input data, they provide their own decision. Finally, in the third layer of the architecture, the Fusion layer, the final decision is made. The detail of each layer is explained below.

*1) Abstraction Layer:* An abstract view is a well-defined transformation that leads to an abbreviation of the entire data and may include a combination of several features. In other words, when defining an abstract view, there is no need to focus on extracting specific features from data; rather, the focus is on a perspective that makes data simple by removing unnecessary information according to a specific point of view.

*2) Computation Layer:* The decision-making models differ in at least one of these two cases; the type of model and perspective they are trained with. For example, an SVM and an ANN after training will be two different processing models even if they are trained with the same data abstraction. Also, two NNs with the same structure will be two different processing models if they are trained with two variant abstractions.

*3) Fusion Layer:* The decision fusion tool combines multiple predictions from multiple models to make a final decision. It improves the accuracy and robustness of a decision-making system by combining the strengths of multiple models and reducing the impact of individual model limitations. The main characteristics of this tool are the capability of integrating multiple sources of information, robustness in individual model failures, and accuracy improvement by combining the strengths of multiple models.

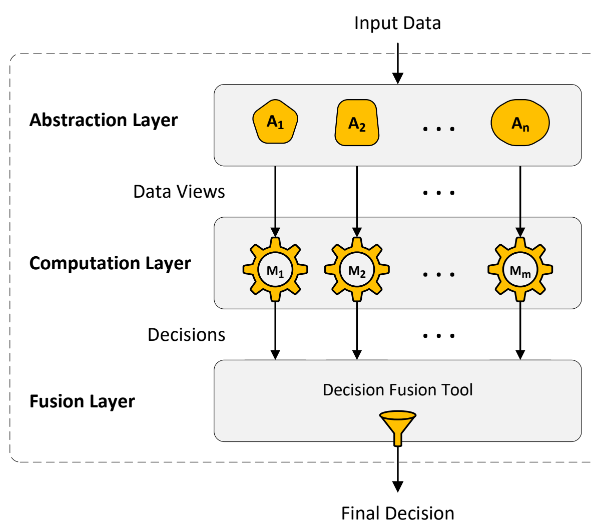



Fig. 3: The proposed 3-layer architecture.

It should be mentioned that ADFA has a different focus of attention compared with parallel ensemble learning [10]. The main focus of attention in ensemble learning is to find an optimal combination of individual models that results in improved accuracy, reduced variance, and increased generalization. On the other hand, the aim of ADFA is to remove the complexities from the data to be suitable for processing with a combination of lightweight models which leads to reducing the amount of computation required to make decisions.

## IV. Methodology

In order to show the efficiency of ADFA, we have developed seven models based on this architecture and classified MNIST handwriting digits by them. This section first introduces the MNIST dataset and then presents the characteristics and the design method of the developed ADFA-based models.

### *A. Data*

We have used the MNIST handwritten digits dataset in the experiments. MNIST is a collection of $60,000$ grayscale images, each representing one handwritten digit. Each digit is a $28 \times 28$ pixel image where the value of each element is an integer in the range $[0, 255]$. In our work, we divided MNIST dataset into four categories. The first category consists of $10000$ images and is used to train the FCN and SVM models. The second category consists of $3000$ images and is used to test the FCN and SVM models. The third category consists of $10000$ images and is used to train the ANFIS in the ADFA-based models. The fourth category consists of $3000$ images for testing the ADFA-based models.

### *B. Experimental Design*

In this section, the experimental design is described in four steps. The first three steps demonstrate the basic components and the last step characterizes the structure of the ADFA-based models that are produced using the basic components.

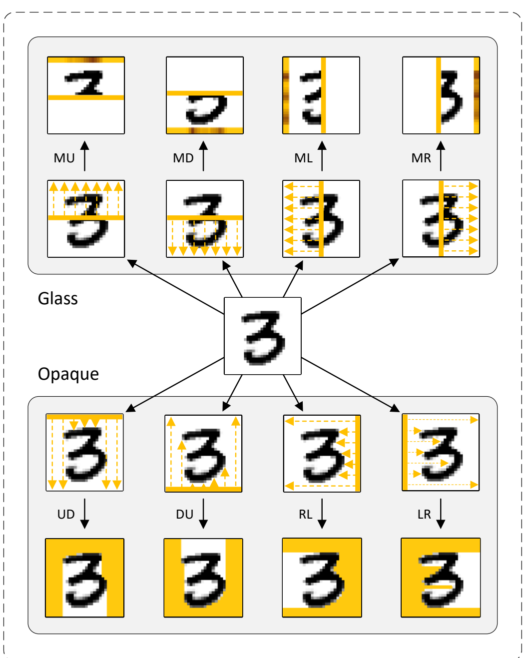



Fig. 4: The Glass / Opaque views of a handwritten digit.

*1) Step 1:* In the first layer of the ADFA-based models, two data abstractions called Glass and Opaque are defined based on different views inspired by two sculpting techniques in Shadow Art [19] (See Fig. 1). In Glass abstraction, the image is assumed as an object through which light passes, where the transparency of its pixels is proportional to their value. In Opaque abstraction, the image is assumed as an opaque object that light cannot pass through non-zero pixels, and a shadow is formed behind them. The shape of this shadow depends on the angle of light radiation and the shape of the outer border of the desired object. Fig. 4 presents eight views that form Glass and Opaque abstractions. The solid yellow line inside the figures indicates the light sources that shine either horizontally or vertically on the object. Abbreviations used for views indicate the origin and destination of light rays and include the letters M (Middle), T (Top), B (Bottom), L (Left), and R (Right). For example, MU means middle-to-top and LR means left-to-right. In Glass abstraction, when the light passes through the figure, a gray-scale shadow is formed on the line on the opposite side; so that the intensity of darkness of each shadow pixel accords with the amount of blurriness of the path it passed. In Opaque abstraction, the shape is assumed to be opaque and light does not pass through it. Therefore, a shadow is formed behind it, which we have specified in the figure as its complement; the light lines from each light source pixel to the first opaque pixel are aligned with it.

Both the abstractions have the same representation and map a $28 \times 28$ input image to a vector of length 112. Each element of the vector corresponds to a light source pixel, and the range of its values is the Real interval $[0, 1]$. In Glass abstraction, the value associated with any light source is proportional to the intensity of the darkness of the corresponding shadow. Also, in Opaque abstraction, the value associated with each light source is the normalized value of the length of the light line that starts from the desired source.

*2) Step 2:* As lightweight computational models of ADFA, two SVMs and two three-layer FCNs are trained with either Glass or Opaque abstractions on the same training dataset. Table I presents a brief description of these models associated with the abbreviation that we use in the rest of the paper.

TABLE I: The basic lightweight computational models.

| **Model Name** | **Description** |
|---|---|
| FG | A 3-layer FCN trained by the Glass abstraction |
| FO | A 3-layer FCN trained by the Opaque abstraction |
| SG | An SVM trained by the Glass abstraction |
| SO | An SVM trained by the Opaque abstraction |

Table II presents the specification of FG and FO models used in the experiments. The information shows that these models are simple and lightweight.

*3) Step 3:* We have employed an ANFIS in the third layer to take advantage of fuzzy logic in the fusion of decisions. The training of the ANFIS models is done according to the decision result of its input models on the same training data set. Fig.

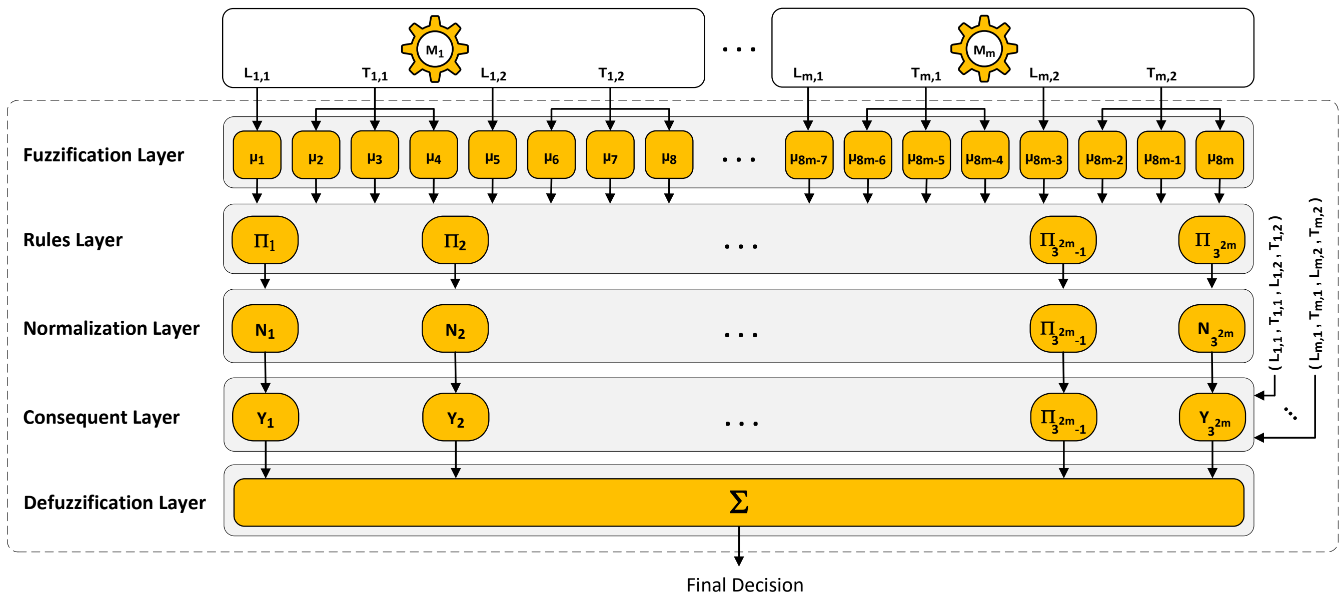



Fig. 5: The ANFIS model used for fusing the decisions of $m$ models.

TABLE II: The specification of FG and FO models.

| Layer | # of Neurons | Activation Function |
|---|---|---|
| Input Layer | 112 | — |
| Hidden Layer | 128 | Relu |
| Output Layer | 10 | Softmax |

5 illustrates the ANFIS model for fusing the decisions made by $m$ models. In fact, the output decision of each basic model $M_i$ in the Computation layer that is sent to ANFIS is the four tuples $(L_{i1}, T_{i1}, L_{i2}, T_{i2})$, which are the two output values with the highest degree of possibility, including their degree of trust. Therefore, ANFIS receives $4m$ inputs from $m$ models; $2m$ output values, and $2m$ trust levels of the output values. Also, the first layer of ANFIS includes $8m$ fuzzification functions, each $L_{ij}$ value, and each $T_{ij}$ value are mapped respectively by one and three Gaussian membership functions (Eq.6) to fuzzy membership degrees.

The number of inference rules in the second layer of ANFIS is $3^{2m}$, where each rule uses the minimization operator to combine the fuzzy degrees obtained by applying the membership functions on the disjoint inputs and producing a fuzzy output. In the Normalization layer, the $i^{th}$ node calculates the ratio of the $i^{th}$ rules' strength to the sum of all rule's strengths.

*4) Step 4:* According to Fig.5, the difference between the ADFA-based models developed in this research is the set of basic models in the second layer (Table I), which ANFIS fuses their decisions. Therefore, to refer to these models, we specify their names as ANFIS inputs. For example, ANFIS(FO,SG) represents an ADFA-based model in which two glass and opaque abstractions are in the first layer, AO and SG models are in the second layer, and an ANFIS model is in the third layer. In this regard, we first developed six models based on the combination of two models selected from Table I and ANFIS, and tested them. Then, two of the best ones, namely ANFIS(FO,FG) and ANFIS(FO,SO), were aggregated and the model ANFIS(FO,FG,SO) was created.

In the next section, the experimental result of evaluating these seven models on the MNIST dataset is discussed according to the four measurements; accuracy, number of MAC operations, model size, and number of trainable parameters.

## V. Experimental Results

The ADFA-based models that were developed have been assessed based on their model size, which determines the memory usage, and their number of MAC (multiply-accumulate) operations needed to achieve a similar close error rate as compared to other contemporary models, which measures the energy efficiency. In this section, our experiments and evaluations on 3000 MNIST test images are reported and discussed.

### A. Analysis of Basic Models

Table III reports the experimental results of the basic models. The models are sorted based on the $\%Err$, where the highest accuracy belongs to model FO and the lowest accuracy to model SG. The information shows that for both the *Opaque* and *Glass* abstractions on the data, the FCN has a higher accuracy. As well, the *Opaque* abstraction has resulted in higher accuracy than *Glass* abstraction, for FCN and SVM. Referring to

From the comparison of Table III and Table IV it is shown that when we combine lightweight computational models with ANFIS based on ADFA architecture, we have a decrease in error rate compared to each lightweight model individually. This is the power of our ensemble architecture with ANFIS

TABLE III: Performance of each model.

| Model Name | %Err | MAC ($10^6$) | Size (MB) | T-P($10^3$) |
|---|---|---|---|---|
| FO | **3.9** | **0.03** | 0.76 | **15.17** |
| FG | 5.2 | **0.03** | **0.27** | **15.17** |
| SO | 6.4 | 1.25 | 2.9 | 10153 |
| SG | 8.8 | 1.25 | 1.7 | 10.15 |

decision fusion layer. However, we have a penalty in model size and the number of MAC operations which are discussed in the following sections.

### *B. Model size*

As shown in Table III, We assessed the size of several lightweight computational models, including FO, FG, SG, and SG, on an individual basis. Additionally, we measured the model size of ADFA with different architectures to examine how placing lightweight models can affect model size. To compare the results with other contemporary models and demonstrate the superiority of ADFA, we presented them in Table V. As shown in Table V, Although the ADFA has a slightly higher error rate than other models, it compensates for this by having a smaller model size and fewer trainable parameters (T-P) than the others. Specifically, the ADFA's model size is significantly smaller than that of other ensemble models (Hetero. Ensemble [26], EnsNet [8]). Despite the small increase in error rate, averaging only $0.7\% - 1\%4$ compared to other models, the reduction in model size has several advantages. For instance, deploying it on edge devices is more feasible due to its smaller memory and storage requirements. Additionally, its smaller model size results in improved processing speed and reduced computational complexity, which are crucial for real-time applications on edge devices.

TABLE IV: Performance of the ADFA-based models with ANFIS decision fusion.T-P is the number of Trainable Parameters.

| Model Name | %Err | MAC ($10^6$) | Size (MB) | T-P($10^3$) |
|---|---|---|---|---|
| ANFIS(FO,FG,SO) | **0.97** | 1.318 | 5.33 | 41.10 |
| ANFIS(FO,FG) | 1.13 | **0.068** | **2.43** | 30.71 |
| ANFIS(FO,SO) | 1.19 | 1.281 | 5.06 | 25.72 |
| ANFIS(FO,SG) | 2.71 | 1.281 | 3.86 | 25.72 |
| ANFIS(FG,SO) | 4.10 | 1.281 | 4.57 | 25.72 |
| ANFIS(FG,SG) | 4.90 | 1.281 | 3.37 | 25.72 |
| ANFIS(SG,SO) | 5.31 | 2.506 | 6.15 | **20.10** |

### *C. Number of MAC Operations*

The number of MAC operations required also measured for each lightweight computational model with different abstraction methods is shown in TABLE III individually. In addition, the number of MAC operations of our ADFA compared to other state-of-the-art models is shown in TABLE V. This comparison shows that the ADFA requires a significantly lower number of MAC operations compared to other models. For example, EnsNet [8] has an ensemble architecture the same to ADFA but it has about $3X$ more MAC operation than ANFIS(FO,FG,SO) . This is despite the fact that its error is only $0.8\%$ less. On average, the proposed model requires $4X$ fewer MAC operations than the LeNet300 which has the same error rate as ANFIS(FO,FG). The reduction in the number of MAC operations required by the ADFA results in a corresponding reduction in energy consumption, leading to improved energy efficiency which is important for deploying on the Edge devices.

TABLE V: Performance of the ADFA-based models with ANFIS decision fusion, T-P is the number of Trainable Parameters

| Model Name | %Err | MAC ($10^6$) | Size (MB) | T-P($10^3$) |
|---|---|---|---|---|
| Hetero. Ensemble [26] | **0.09** | 2.867 | 72.67 | 1128.57 |
| EnsNet [8] | 0.16 | 3.452 | 94.5 | 1587.10 |
| CapsNet [27] | 0.25 | 24.397 | 58.1 | 949.76 |
| VGG-5 [28] | 0.28 | 599.763 | 401.3 | 6263.41 |
| ResNet-9 [29] | 0.32 | 58.177 | 332.1 | 4088.19 |
| MobileNet [30] | 0.33 | 129.224 | 145.18 | 2259.26 |
| ANFIS(FO,FG,SO) | 0.97 | 1.318 | 5.33 | 41.10 |
| ANFIS(FO,SO) | 1.19 | 1.281 | 5.06 | **25.72** |
| ANFIS(FO,FG) | 1.13 | **0.068** | **2.43** | 30.71 |
| LeNet300 [31] | 1.261 | 0.124 | 29.5 | 328.81 |

## VI. Conclusion and Future Work

In the era of data-driven decision making, machine learning models have become ubiquitous in various applications such as image and speech recognition, natural language processing, and autonomous driving. However, as the size and complexity of the data grow, the performance of the models heavily relies on the computational resources, including memory, storage, and processing power. Therefore, resource-aware intelligent computing, which refers to the trade-off between the restrictions of computational resources and the inference power of the systems, has become a crucial research topic in the field of artificial intelligence. The proposed architecture of ADFA addresses this challenge through a three-layer approach; data abstraction, computation, and decision fusion. In the first layer, a set of abstract views of the input data are provided to reduce the computations required for processing them while providing sufficient information for the next layer. Then, an array of lightweight models in the computation layer process different views and make independent decisions. Finally, the output is produced using a decision fusion tool in the last layer. This architecture enables the reduction of computational resources while maintaining high inference power, making it suitable for deployment on edge devices with limited computational resources.Our experiments on the MNIST dataset verify the efficiency of the proposed architecture with an accuracy more than $90\%$ in recognizing handwritten digits, where the model size and the number of Multiply-Accumulate (MAC) operations are significantly smaller than state-of-the-art ensemble learning models and deep neural networks trained on the same dataset. Additionally, the proposed ADFA-based models achieved comparable accuracy with ensemble models while having much smaller model size and fewer MAC operations, demonstrating the effectiveness of the proposed architecture.Therefore ADFA architecture has the potential for real-time applications on edge devices with limited computational resources, enabling the deployment of efficient and accurate intelligent systems in various applications.

## References


[1] A. H. Bukhari, M. A. Z. Raja, N. Rafiq, M. Shoaib, A. K. Kiani, and C.-M. Shu, "Design of intelligent computing networks for nonlinear chaotic fractional rossler system," *Chaos, Solitons & Fractals*, vol. 157, p. 111985, 2022.

[2] W. Doghri, A. Saddoud, and L. Chaari Fourati, "Cyber-physical systems for structural health monitoring: sensing technologies and intelligent computing," *The Journal of Supercomputing*, vol. 78, no. 1, pp. 766–809, 2022.

[3] S. Mao, L. Liu, N. Zhang, M. Dong, J. Zhao, J. Wu, and V. C. Leung, "Reconfigurable intelligent surface-assisted secure mobile edge computing networks," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 6, pp. 6647–6660, 2022.

[4] Z. Tong, F. Ye, M. Yan, H. Liu, and S. Basodi, "A survey on algorithms for intelligent computing and smart city applications," *Big Data Mining and Analytics*, vol. 4, no. 3, pp. 155–172, 2021.

[5] S. Zhu, T. Yu, T. Xu, H. Chen, S. Dustdar, S. Gigan, D. Gunduz, E. Hossain, Y. Jin, F. Lin *et al.*, "Intelligent computing: The latest advances, challenges, and future," *Intelligent Computing*, vol. 2, p. 0006, 2023.

[6] S. J. Russell, *Artificial intelligence a modern approach*. Pearson Education, Inc., 2010.

[7] Y. Guo, X. Wang, P. Xiao, and X. Xu, "An ensemble learning framework for convolutional neural network based on multiple classifiers," *Soft Computing*, vol. 24, pp. 3727–3735, 2020.

[8] D. Hirata and N. Takahashi, "Ensemble learning in cnn augmented with fully connected subnetworks," *arXiv preprint arXiv:2003.08562*, 2020.

[9] R. Shaw, E. Howley, and E. Barrett, "An intelligent ensemble learning approach for energy efficient and interference aware dynamic virtual machine consolidation," *Simulation Modelling Practice and Theory*, vol. 102, p. 101992, 2020.

[10] Y. Yang, H. Lv, and N. Chen, "A survey on ensemble learning under the era of deep learning," *Artificial Intelligence Review*, pp. 1–45, 2022.

[11] I. Lahmar, A. Zaier, M. Yahia, and R. Boaullegue, "A multiple fuzzy c-means ensemble cluster forest for big data," in *Hybrid Intelligent Systems: 21st International Conference on Hybrid Intelligent Systems (HIS 2021), December 14–16, 2021*. Springer, 2022, pp. 442–451.

[12] P. Garcia, "Effectiveness of multi-abstraction computing tools on promoting exploratory self-learning in engineering: a case study using a custom real-time operating system for remote learning," in *2021 IEEE 8th International Conference on e-Learning in Industrial Electronics (ICELIE)*. IEEE, 2021, pp. 1–6.

[13] M. Ahmed, "Data summarization: a survey," *Knowledge and Information Systems*, vol. 58, no. 2, pp. 249–273, 2019.

[14] S. Kim, B. Jang, J. Lee, H. Bae, H. Jang, and I.-C. Park, "A cnn inference accelerator on fpga with compression and layer-chaining techniques for style transfer applications," *IEEE Transactions on Circuits and Systems I: Regular Papers*, 2023.

[15] S. Cong and Y. Zhou, "A review of convolutional neural network architectures and their optimizations," *Artificial Intelligence Review*, vol. 56, no. 3, pp. 1905–1969, 2023.

[16] M. Shaeri and A. M. Sodagar, "Data transformation in the processing of neuronal signals: A powerful tool to illuminate informative contents," *IEEE Reviews in Biomedical Engineering*, 2022.

[17] J. Chen, H. Huang, J. Peng, J. Zhu, L. Chen, C. Tao, and H. Li, "Contextual information-preserved architecture learning for remote-sensing scene classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–14, 2021.

[18] R. Chakwate, A. Subramaniam, and A. Mittal, "Marnet: Multi-abstraction refinement network for 3d point cloud analysis," *arXiv preprint arXiv:2011.00923*, 2020.

[19] N. J. Mitra and M. Pauly, "Shadow art," *ACM Transactions on Graphics*, vol. 28, no. CONF, pp. 156–1, 2009.

[20] K. Tang, Y. Ma, D. Miao, P. Song, Z. Gu, Z. Tian, and W. Wang, "Decision fusion networks for image classification," *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[21] C. Yin, S. Zhang, and Q. Zeng, "Hybrid representation and decision fusion towards visual-textual sentiment," *ACM Transactions on Intelligent Systems and Technology*, 2023.

[22] K. Roushangar, R. Ghasempour, and S. Shahnazi, "Kernel-based modeling," in *Handbook of Hydroinformatics*. Elsevier, 2023, pp. 267–281.

[23] A. Lohani, N. K. Goel, and K. Bhatia, "Takagi–sugeno fuzzy inference system for modeling stage–discharge relationship," *Journal of Hydrology*, vol. 331, no. 1-2, pp. 146–160, 2006.

[24] Y. Chai, L. Jia, and Z. Zhang, "Mamdani model based adaptive neural fuzzy inference system and its application," *International Journal of Computer and Information Engineering*, vol. 3, no. 3, pp. 663–670, 2009.

[25] A. Al-Hmouz, J. Shen, R. Al-Hmouz, and J. Yan, "Modeling and simulation of an adaptive neuro-fuzzy inference system (anfis) for mobile learning," *IEEE Transactions on Learning Technologies*, vol. 5, no. 3, pp. 226–237, 2011.

[26] S. An, M. J. Lee, S. Park, H. S. Yang, and J. So, "An ensemble of simple convolutional neural network models for mnist digit recognition," *ArXiv*, vol. abs/2008.10400, 2020.

[27] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," *Advances in neural information processing systems*, vol. 30, 2017.

[28] H. D. Kabir, M. Abdar, A. Khosravi, S. M. J. Jalali, A. F. Atiya, S. Nahavandi, and D. Srinivasan, "Spinalnet: Deep neural network with gradual input," *IEEE Transactions on Artificial Intelligence*, 2022.

[29] P. Gavrikov and J. Keuper, "Cnn filter db: An empirical investigation of trained convolutional filters," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 19 066–19 076.

[30] J. Sun, A. P. Fard, and M. H. Mahoor, "Xnodr and xnidr: Two accurate and fast fully connected layers for convolutional neural networks," *arXiv preprint arXiv:2111.10854*, 2021.

[31] T. Dettmers and L. Zettlemoyer, "Sparse networks from scratch: Faster training without losing performance," *arXiv preprint arXiv:1907.04840*, 2019.