Turkish Journal of Electrical Engineering and Computer Sciences

Volume 30 | Number 3

Article 30

3-1-2022

The analysis and optimization of CNN Hyperparameters with fuzzy tree modelfor image classification

KÜBRA UYAR

ŞAKİR TAŞDEMİR

İLKER ALİ ÖZKAN

Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2022) 30: 961 – 977
© TÜBİTAK
doi:10.55730/1300-0632.3821

The analysis and optimization of CNN Hyperparameters with fuzzy tree model for image classification

Kübra UYAR[*], **Şakir TAŞDEMİR**, **İlker Ali ÖZKAN**
Department of Computer Engineering, Technology Faculty, Selçuk University, Konya, Turkey

Received: 28.07.2021 • **Accepted/Published Online:** 13.12.2021 • **Final Version:** 21.03.2022

Abstract: The meaningful performance of convolutional neural network (CNN) has enabled the solution of various state-of-the-art problems. Although CNNs achieve satisfactory results in computer-vision problems, they still have some difficulties. As the designed CNN models are deepened to achieve much better accuracy, computational cost and complexity increase. It is significant to train CNNs with suitable topology and training hyperparameters that include initial learning rate, minibatch size, epoch number, filter size, number of filters, etc. because the initialization of hyperparameters affects classification results. On the other hand, it is not possible to make a definite inference for the hyperparameter initialization and there is uncertainty. This study is carried out to model uncertainty using fuzzy inference system (FIS). The designed fuzzy model provides estimation of classification result depending on CNN topology and training hyperparameters. GoogleNet and Inceptionv3 that contain inception-modules, ShuffleNet that contains shuffle-blocks, DenseNet201 that contains dense-blocks, EfficientNet, ResNet18, ResNet50, ResNet101, and MobileNetv2 that contain residual-blocks, and InceptionResNetv2 that includes both inception-modules and residual-blocks were evaluated as CNN models. Test sample dataset was obtained by training CNN models with various training hyperparameter combinations. CNN models were trained on Animal Diagnostics Lab (ADL) which is a histopathological dataset includes healthy and inflamed kidney, lung, and spleen images. A new FIS tree model that is more computationally efficient and easier to understand than a single FIS was designed and classification accuracy prediction of CNN models depending on hyperparameter combinations was performed. The best, the worst, and the average classification accuracies obtained with CNN models that use best training hyperparameter set are 97.70%, 93.60%, and 96.30%, respectively. Moreover, Cifar10 and Cifar100 benchmark datasets were experimented to reveal true capability and limitations of the proposed approach. Experimental results indicate that the designed FIS tree model provides a successful hyperparameter evaluation mechanism with an average RMSE value of 1.2652.

Key words: Classification, convolutional neural network, fuzzy logic, histopathological data, hyperparameter.

1. Introduction

Hyperparameters in CNN control the behaviors of the training process and have a significant effect on classification performance; however, determining the hyperparameters is a challenge to researchers. This study introduces a new FIS tree model to analyze the effects of topological and training hyperparameters on classification success and to make accuracy prediction of various CNN models depending on a set of training hyperparameters.

Some studies about CNN hyperparameter analysis in the literature have been reviewed. Bochinski et al. [1] proposed an evolutionary algorithm-based framework to automatically optimize the CNN structure by

[*]Correspondence: kubrauyar@selcuk.edu.tr

means of hyperparameters using MNIST dataset. Mukhtar and Kong [2] studied one of the reinforcement learning approaches, which is Q-Learning paradigm developing two reward functions that uses side-channel metrics to tune the CNN hyperparameters. Soon et al. [3] studied to minimize the user variability in training process of CNN by automatically searching and optimizing the CNN architecture on vehicle logo recognition system. They chose the architecture and hyperparameters of CNN with the stochastic method of particle swarm optimization. Yoo [4] performed a univariate dynamic encoding algorithm in order to optimize hyperparameters of the CNN validating the proposed method with two neural network models with MNIST dataset. They claimed the idea that the proposed algorithm provides fast convergence speed and less computational complexity to optimize hyperparameters of the network. Zhang et al. [5] introduced Bayesian optimization for automatic searching of optimal configurations of hyperparameters in CNNs by applying probabilistic surrogate models. They approximated the validation error function of hyperparameter configurations, such as Gaussian processes. Cabada et al. [6] presented the use of a genetic algorithm for hyperparameter tuning in a CNN for the recognition of learning-centered emotions. They suggested the idea that automatic hyperparameter definition of CNN using genetic algorithms helps to improve the accuracy. Pan et al. [7] suggested an orthogonal array tuning method for hyperparameter optimization and CNN model with optimized hyperparameters and nonhyperparameters to classify the Titan multispectral LiDAR data into the six land-cover types which are building, tree, road, grass, soil, and water. Andoine et al. [8] introduced and proved the efficiency of weighted random search method, a combination of random search and probabilistic greedy heuristic for CNN hyperparameter optimization. Wang et al. [9] proposed the strategy of CNN hyperparameter optimization depending on the perceptual hash algorithm using its training characteristics. Yilmaz et al. [10] proposed a novel deep learning-based human activity recognition system. They applied a metaheuristic approach to enhance the extracted features for better recognition performance. AlexNet, Vgg16, GoogleNet, and ResNet152 models were combined to produce a hybrid model. Soylu et al. [11] developed a software named Pykopt which tries to find the best combination from the given set of hyperparameters by using the genetic algorithms. They optimized hyperparameters of the Vgg16 model on the dataset containing 87,000 pictures of diseased and healthy plant leaves using the designed software obtaining 98.67% accuracy. Singh et al. [12] proposed a multilevel particle swarm optimization (MPSO) algorithm to explore the architecture and hyperparameters of the CNN and to prevent human factors to determine these parameters manually. They discovered an approach to suggest the suitable CNN architecture and its hyperparameters using MPSO in a specified search space. Lastly, Yeh et al. [13] presented a new algorithm based on simplified swarm optimization to optimize the hyperparameters of LeNet. They claimed that the accuracy of the proposed algorithm is higher than the original LeNet model and PSO-LeNet and this algorithm can be applied to more complex CNN models such as AlexNet.

In some of the studies, CNN topology hyperparameters such as filter size and number of filters were used, while in others, analyses were carried out using both topology and training hyperparameters. There are some studies about machine learning-based and evolutionary algorithm-based hyperparameter optimization. In this paper, we define the CNN model and training hyperparameters as the configuration of the network structure and thus pose the problem of finding the predicted classification accuracy result using the proposed FIS tree model. The proposed study is carried out to model uncertainty using FIS tree model.

The main novelty and contribution of this paper is the implementation of the designed FIS tree model to predict the classification accuracy of the various CNN models depending on a set of training hyperparameters. Model type, minibatch size, initial learning rate, and epoch number were taken as inputs of the proposed FIS tree model and the classification result of the CNN model was estimated using FIS tree model. More clearly, analysis of training hyperparameters using different CNN topologies can be performed using the proposed FIS tree model, which determines the fuzzy model parameters by performing automatic learning and tuning and eliminates the human factor. The proposed model is more computationally efficient and easier to understand than a single FIS with the same number of inputs.

This paper is structured as follows: Section 2 explains the proposed FIS tree model. The details of the experimental setup used to deploy the system is presented in Section 3. Experimental results and discussion are explained in Section 4. Finally, conclusions of the system is summarized in Section 5.

2. Proposed FIS tree model

In this section, the basis of fuzzy logic, various FIS tree models, and the details of the designed FIS tree model were presented.

2.1. Fuzzy logic

Fuzzy logic is an alternative approach to solve many linear and nonlinear problems that contain uncertainty. The characteristics of the system are represented by rules, membership functions, and concluding processes. The general architecture of FIS is illustrated in Figure 1.

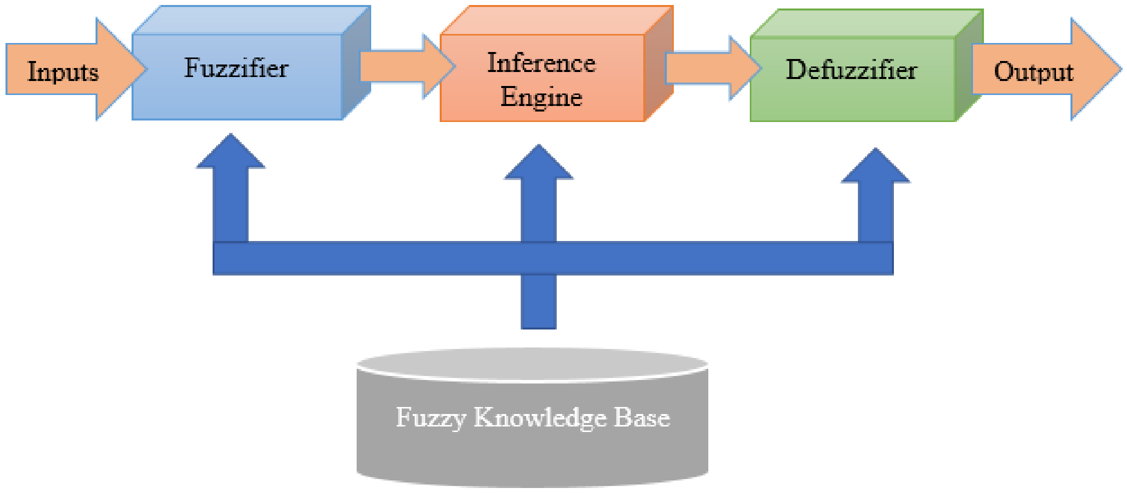


Figure 1. The general architecture of FIS.

The input values taken into the system are converted into fuzzy values by using membership functions in the fuzzifier unit. In the fuzzy inference unit, using the knowledge base, results are obtained from the fuzzy values that come to it. Knowledge of what and how to draw these results is kept in the knowledge base. The defuzzifier allows the incoming values to be scaled to the desired range.

2.2. FIS tree models

In this study, a new FIS tree model that is a collection of connected FIS models was designed for the classification accuracy prediction. Fuzzy trees are more computationally efficient and easier to understand than a single FIS with the same number of inputs. Figure 2 illustrates commonly used FIS tree structures that are incremental, aggregated, or cascaded (combined).

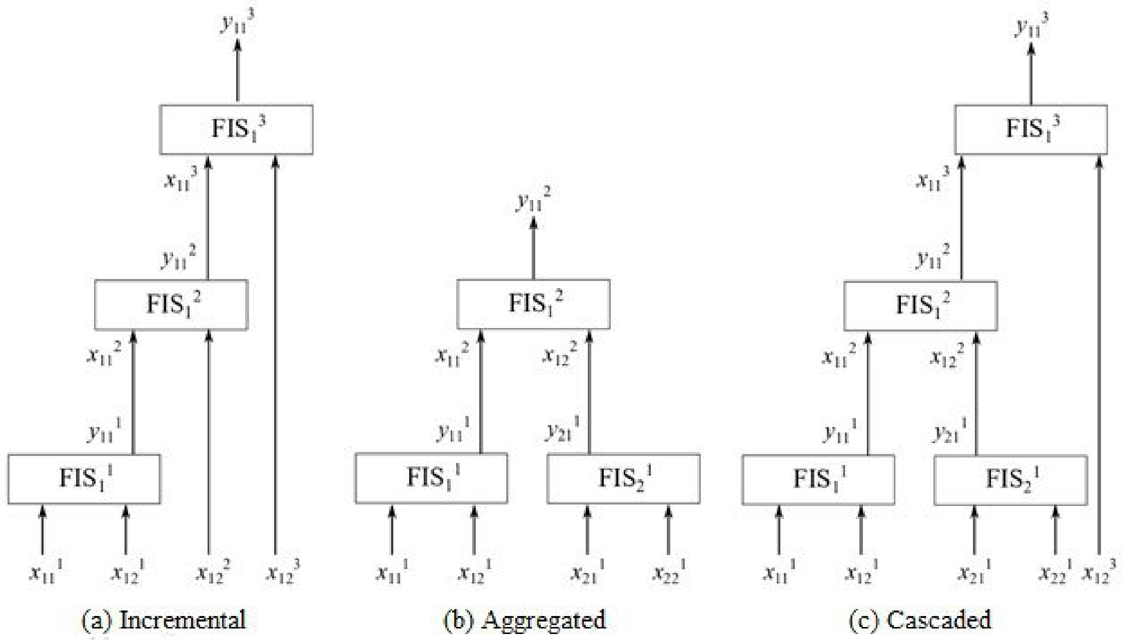


Figure 2. Various FIS tree structures.[1]

[1]MathWorks Inc.(2021). Fuzzy Trees [online]. Website https://www.mathworks.com/help/fuzzy/fuzzy-trees.html [accessed 19/10/2021].

- In the incremental model, each input value usually contributes to the inference process to a certain extent. Therefore, one input is not significantly correlated with the other inputs.
- In the aggregated model, one input is significantly correlated with the other inputs in the system.
- In the cascaded model, incremental and aggregated models are combined. This structure is suitable for a system that includes both correlated and uncorrelated inputs.

It is important to determine the hyperparameters used during network training in CNN because hyperparameters are correlated with each other and hyperparameter combinations affect classification success. Therefore, aggregated model was preferred in the experimental studies.

2.3. Design of the proposed FIS tree model

We have proposed an effective FIS tree model for classification accuracy prediction of various CNN models considering training hyperparameters. The proposed FIS tree model was obtained using interconnected FIS objects rather than as a single monolithic FIS object. In the experiments, four input attributes which are model type represents CNN models with different topologies, initial learning rate, minibatch size, and epoch number were used to predict the output data attribute (classification accuracy). The input attributes of the FIS models in FIS tree were determined by considering the correlation coefficient of the experimental test samples explained in Section 3.4. Moreover, learning and tuning phases were applied to determine designed FIS tree parameters automatically.

Correlation coefficient is used to measure how strong a relationship is between variables. The mathematical definition of this coefficient can be explained in Equation (1).

$$Correlation - coefficient = \frac{\sum X_i Y_i - n\bar{X}\bar{Y}}{\sqrt{(\sum X_i^2 - n\bar{X}^2)(\sum Y_i^2 - n\bar{Y}^2)}} \quad (1)$$

where $\bar{X}$ corresponds to the mean of the X variable, $\bar{Y}$ corresponds to the mean of the Y variable, and n corresponds to the number of observations.

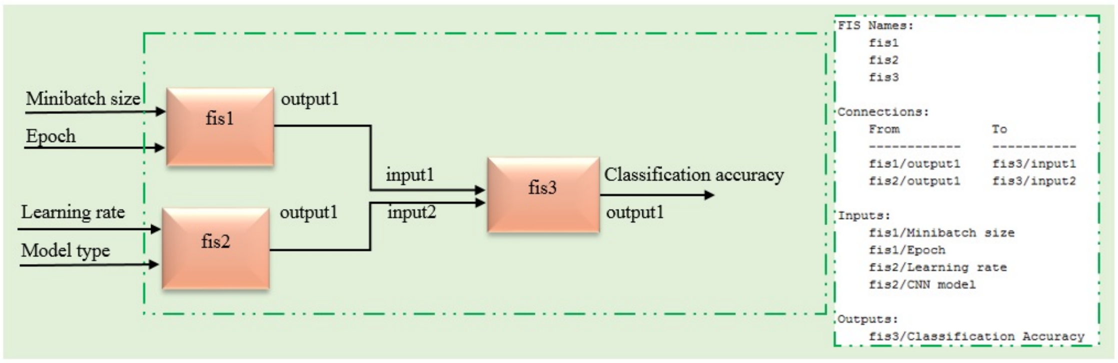
For the construction of the proposed FIS tree, the input attributes were ranked based on their correlation coefficients. Different correlation results are obtained for different test samples. They depend on CNN models, hyperparameter combinations, and validation dataset used.

After obtaining test sample dataset that contains experimental result records of the CNN models with various hyperparameter combinations, a numerical matrix with the dimension of 100x5 was created because there are 100 experimental results that give the classification accuracy depending on CNN model with the hyperparameter combination of learning rate, minibatch size, and epoch number. In order to determine inputs of the FIS tree model, input attributes (CNN model, learning rate, minibatch size, and epoch number) were ranked based on their correlations with the output attribute (classification accuracy). The correlation coefficients of the attributes are listed in Table 1.

According to Table 1, the first three input attributes have negative values and the last attribute has a positive value. The input attributes that have negative correlations were ranked in descending order by the absolute value of their correlation coefficients. This order is minibatch size, initial learning rate, and model type. The last attribute, epoch, that has a positive correlation does not need to be ordered. These rankings indicate that minibatch size and epoch have the highest negative and positive correlations with classification accuracy. Input attributes with negative and positive correlation values were paired up to combine both positive and negative effects on the output for prediction. As a result, the inputs were grouped according to their ranks as follows: minibatch size and epoch and initial learning rate and model type. Considering these analyses, the proposed FIS tree model was designed as illustrated in Figure 3. In addition, Table 2 lists the details of the proposed FIS tree model.

Table 1. Correlation coefficients of the attributes.

Parameter	Model type	Learning rate	Minibatch size	Epoch	Classification accuracy
Correlation coefficient	–0.0947	–0.1729	–0.3367	0.0938	1.0000



```
FIS Names:
    fis1
    fis2
    fis3

Connections:
    From            To
    ------------    -----------
    fis1/output1    fis3/input1
    fis2/output1    fis3/input2

Inputs:
    fis1/Minibatch size
    fis1/Epoch
    fis2/Learning rate
    fis2/CNN model

Outputs:
    fis3/Classification Accuracy
```

Figure 3. Proposed FIS tree model.

Table 2. Details of the fis1, fis2, and fis3.

Fis name	Connection	Input name	Data range	The number of membership function
fis1	input	Minibatch size	32–256	4
	input	Epoch	5–15	3
	output1	Accuracy	92.7900–99.3267	25
fis2	input	Learning rate	0.001–0.1	5
	input	Model type	0–9	10
	output1	Accuracy	92.7900–99.3267	25
fis3	input1	Accuracy	92.7900–99.3267	25
	input2	Accuracy	92.7900–99.3267	25
	output1	Accuracy	92.7900–99.3267	25

Parameters of the designed FIS tree model were tuned in a two-step process: *learning process* which means learning and tuning the rules of the FISs in the FIS tree and *tuning process* which means learning the membership function parameters of the FISs in the FIS tree. In the designed fuzzy system, Takagi-Sugeno-Kang [14] was used as the inference mechanism and centroid was used as the defuzzification method.

2.4. Steps of the proposed FIS tree model

The flow chart illustrated in Figure 4 shows the overall architecture of the proposed approach. The proposed study is carried out in six steps.

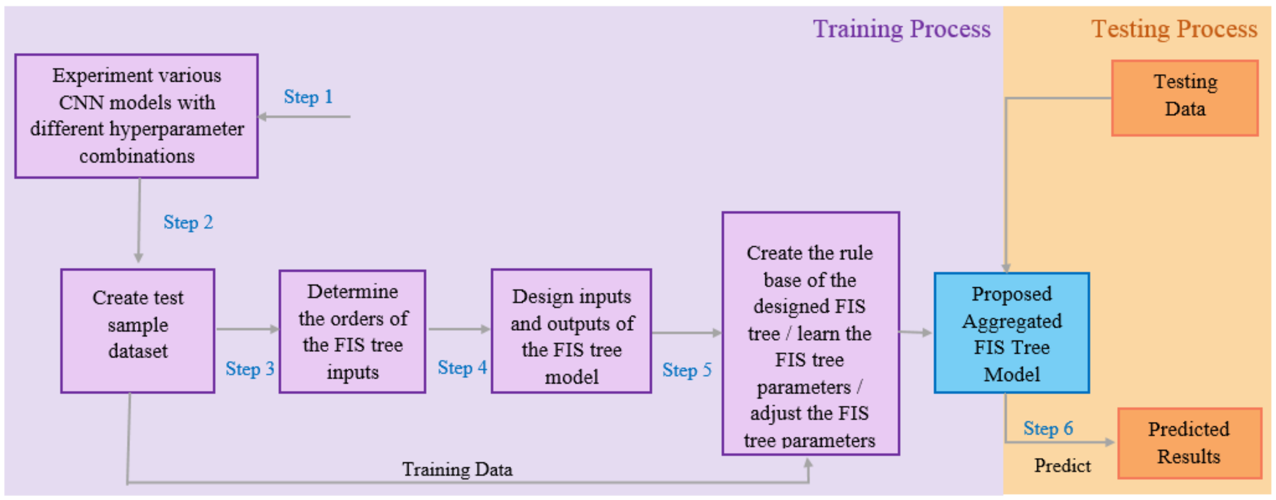

Training Process
Testing Process
Experiment various CNN models with different hyperparameter combinations
Step 1
Testing Data
Step 2
Create test sample dataset
Step 3
Determine the orders of the FIS tree inputs
Step 4
Design inputs and outputs of the FIS tree model
Step 5
Create the rule base of the designed FIS tree / learn the FIS tree parameters / adjust the FIS tree parameters
Proposed Aggregated FIS Tree Model
Step 6
Predict
Predicted Results
Training Data

Figure 4. The flow chart of proposed aggregated FIS tree model.

- **STEP 1:** Training and testing were carried out by various CNN models such as GoogleNet and Inceptionv3 that contain inception-modules, ShuffleNet that contains shuffle-blocks, DenseNet201 that contains dense-blocks, EfficientNet, and ResNet18, ResNet50, ResNet101, and MobileNetv2 that contain residual-blocks and, InceptionResNetv2 that includes both inception-modules and residual-blocks. CNN models were trained on ADL histopathological dataset.

- **STEP 2:** Each test sample consists of model type, initial learning rate, minibatch size, number of the epoch, and classification result. To generate each test sample, a hyperparameter set was taken as the training parameters and tested on a CNN model, which was determined as the model type. The classification result was also added to the test sample. Similarly, other test samples were obtained in the same way. As a result, the experimental test samples were obtained with the experimented 100 hyperparameter combination with the dimension of a 100x5 numerical matrix.
- **STEP 3:** To determine the orders of the inputs of the FISs in the proposed FIS tree model, the correlation coefficients of the experimental test samples were calculated.
- **STEP 4:** The inputs of the FISs in the proposed FIS tree model were grouped according to their ranks as follows: minibatch size and epoch and initial learning rate and model type. Besides, the classification accuracy result was determined as the output of the proposed fuzzy system.
- **STEP 5:** The rule base of the proposed model was learned while keeping the input and output membership functions constant using the particle swarm optimization method. After learning the new rules, the input and output of the membership function parameters were tuned with the parameters of the learned rules using the pattern search optimization method. The generation of the rule base and adjustment of FIS tree parameters were obtained automatically by eliminating the human factor.
- **STEP 6:** Using 5-fold cross-validation, classification accuracy prediction of the proposed FIS tree model was tested depending on the used CNN model and hyperparameter set. As a result, the selection of the best CNN model and hyperparameter set were obtained depending on the classification accuracies.

3. Experimental setup

The details of the experimental setup used to deploy the system are described in this section.

3.1. CNN

In this study, a comprehensive analysis has been carried out using 10 CNN models that have different topologies. GoogleNet and Inceptionv3 that contain inception-modules, ShuffleNet that contains shuffle-blocks, DenseNet201 that contains dense-blocks, EfficientNet, and ResNet18, ResNet50, ResNet101, and MobileNetv2 that contain residual-blocks, and InceptionResNetv2 that includes both inception-modules and residual-blocks were used in experimental studies. Model type, which is one of the input attributes of the proposed FIS tree model, represents these CNN models with different topologies. Here, the effects of 10 various CNN models on classification result was also analyzed.

3.2. Dataset

The classification success of CNN models was measured on a histopathological dataset that contains mammalian kidney, lung, and spleen images provided by ADL [15]. It is aimed to analyze the effects of hyperparameters on deep learning architects by trying to achieve the highest score with a CNN on this dataset. ADL contains 335, 308, and 320 images of healthy and inflamed kidney, lung, and spleen images, respectively. Sample histopathological images are illustrated in Figure 5 and the class distributions of the datasets are given in Table 3.

To train deep learning models, large datasets are required; however, it is not possible to obtain large datasets manually. Data augmentation methods that have a positive effect on the performance of deep learning networks [16–18] can be used to overcome this problem. Therefore, rotation (90^0, 180^0, 270^0) and reflection (vertical) were applied to increase the size of the dataset. Totally, the dataset for experiments consists of 3852 images labeled in 2 categories. After the data was separated into training and test, both the training and test data are shuffled in themselves. Table 4 explains the dataset summary during the training and test process.

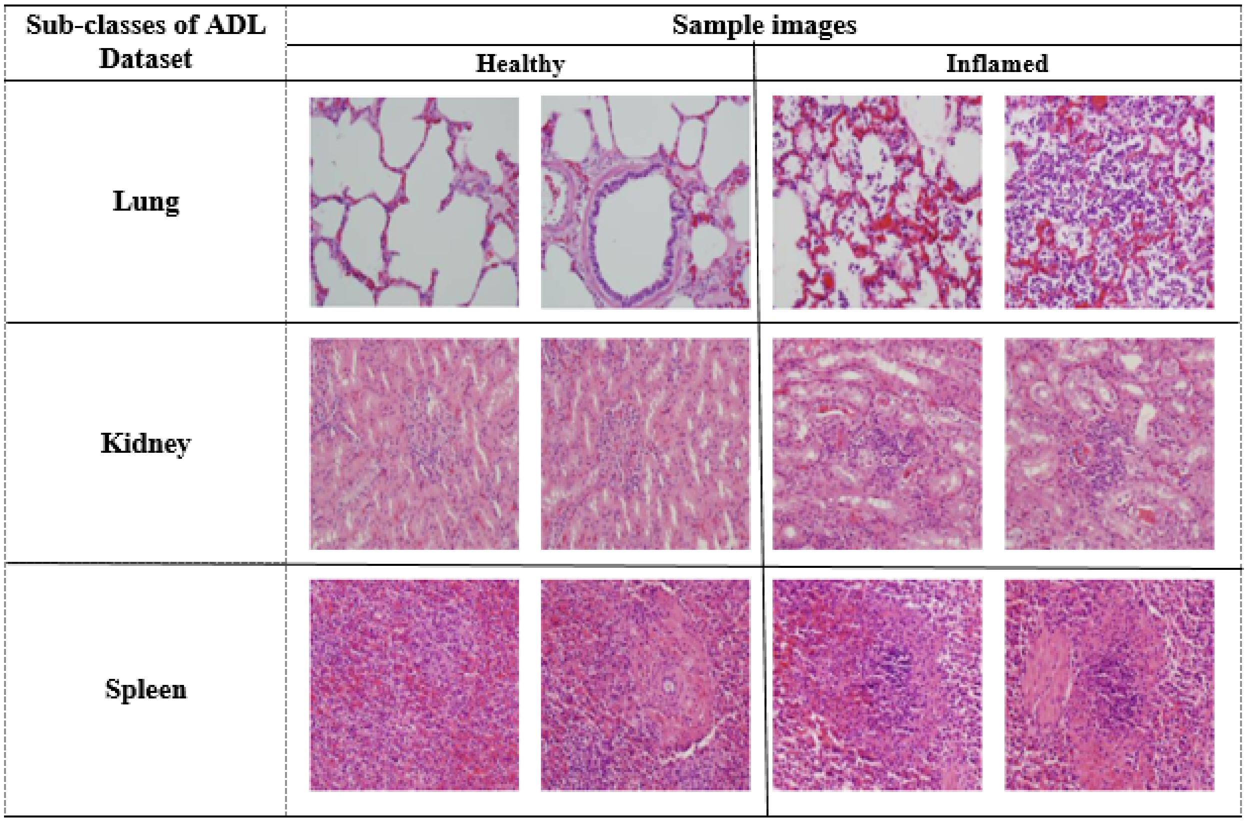


Figure 5. Sample histopathological images.

Table 3. Class distributions of the datasets.

Dataset name	# of classes	# of normal	# of inflamed	# of images in total
ADL kidney	2	178	157	335
ADL lung	2	155	153	308
ADL spleen	2	159	161	320

Table 4. Dataset summary during training and test.

Dataset	# of training samples	# of test samples	# of training samples	# of test samples
	Original		With data augmentation	
ADL kidney	268	67	1072	268
ADL lung	247	61	985	247
ADL spleen	256	64	1024	256

3.3. Determination of hyperparameter value intervals

The determination of the best or the optimum set of hyperparameters in CNN training depends on CNN model, dataset, the size of the dataset, and problem at hand. Value intervals of the training hyperparameters and the number of different values for each hyperparameter are given in Table 5. For all experiments, the drop factor is 0.9, the drop period is 5, and the weight decay is 0.0001. Stochastic gradient descent with momentum (SGDM) optimization algorithm, an iterative method that minimizes or maximizes the objective function, provides the acceleration of the gradient vectors in the right directions, and tries to find minimum or maximum with iterations [19] was used in the training process.

Table 5. Parameter value intervals and the number of different values for each hyperparameter.

Parameter	Value interval	The number of different values
Model type	0–9	10
Learning rate	0.001–0.1	5
Minibatch size	32–256	4
Number of epoch	5–15	3

Based on the literature review, the following part makes some inferences about training hyperparameters to guide for our experimental studies. The most appropriate hyperparameter ranges used in this study were determined considering the dataset, CNN architecture, and the size of dataset.

3.3.1. Learning rate

For the classification problem, it is important to choose the optimal learning rate to minimize loss fuction [20]. The higher learning rate speeds up the learning process that causes an increase in loss function and low learning rate makes the loss function decrease slowly. We trained CNN models with an initial learning rate of 0.001, 0.005, 0.05, 0.01, and 0.1. During the network training, the learning rate was updated by taking 0.9 times its current value with the intervals of 5 epochs. For this reason, CNN training was carried out with various learning rates.

3.3.2. Minibatch size

The larger minibatch size causes running of the model for a long period of time with constant weights that causes overall performance loses and increases the memory requirements [20]. Carrying out the experiments with small minibatch sizes can be more beneficial [21]. Therefore, CNN models were trained with a minibatch size of 32, 64, 128, and 256.

3.3.3. Number of epochs

The number of epochs can be determined with respect to classification accuracy values of CNN models during training process [20]. It is important to determine an ideal epoch number to prevent overfitting. Classification results were obtained for 5, 10, and 15 epochs.

3.4. Experimental test samples

By taking different CNN architectures and training hyperparameter combinations, experimental test samples were generated. To generate each test sample, the hyperparameter array was taken as the training parameters and tested on the CNN model, which was determined as the model type. The classification result was also added to this test sample. Similarly, other test samples were obtained in the same way. As a result, experimental test samples were obtained with the experimented 100 hyperparameter combination with the dimension of a 100x5 numerical matrix. The data in this matrix were used as input and output attributes of the proposed FIS tree model.

3.5. Performance measures

To determine whether the proposed model is good enough, a confusion matrix was used and the performance measures obtained from the confusion matrix which are accuracy, precision, recall, and f-measure were calculated. Moreover, 5-fold cross-validation was used for the classification task and fuzzy logic implementation. In addition to these metrics, root mean square error (RMSE) was also used to measure FIS tree parameter learning and tuning performances. RMSE values have been calculated based on the actual value in the test sample and predicted value that has been calculated from designed FIS tree model. Mathematical expressions of the performance metrics are defined in Equations (2)–(6).

$$Accuracy = (TP + TN)/(TP + TN + FP + FN) \tag{2}$$

$$Precision = TP/(TP + FP) \tag{3}$$

$$Recall = TP/(TP + FN) \tag{4}$$

$$F - measure = (2 * (Recall * Precision))/(Recall + Precision) \tag{5}$$

$$RMSE = \sqrt{(mean((actual_{value} - predicted_{value})^2))} \tag{6}$$

where TP is true positive, TN is true negative, FP is false positive, and FN is false negative.

3.6. Execution environment

In this study, training and testing were carried out on a computer with the following technical specifications: Intel Core i9-10920X 3.50GHz processor, 128 GB of RAM, and four GeForce RTX 2080Ti graphics cards. The CNN models were implemented on Matlab2020b using Deep Learning Toolbox and Fuzzy Logic Toolbox.

4. Experimental results and discussion

In this section, the results of the proposed system using the experimental setup detailed in Section 3 are presented.

4.1. Learning and tuning the parameters of the proposed FIS tree model

The proposed FIS tree model uses multiple two-input-one-output FIS objects to reduce the total number of rules used in the inference process. According to the proposed fuzzy structure, *fis1* and *fis2* directly take the input values and generate intermediate accuracy values, which are further combined using *fis3* (Figure 3). The generation of the rule base and adjustment of designed FIS tree parameters were obtained automatically using some optimization methods by eliminating the human factor.

4.1.1. Parameter learning of the proposed FIS tree model

The rule base of the proposed FIS tree model was learned while keeping the input and output membership functions constant using the particle swarm optimization method. The maximum number of rules to be learned in each FIS is 55 and learning was terminated after 75 iterations. The total number of learned rules for each fold at the end of the parameter learning process is listed in Table 6.

Table 6. The total number of learned rules for each fold at the end of the parameter learning process.

Fold No	Fold1	Fold2	Fold3	Fold4	Fold5
Total number of learned rules	108	102	103	104	102

4.1.2. Parameter tuning of the proposed FIS tree model

After learning the new rules, the input and output of the membership function parameters of the designed FIS tree model were tuned with the parameters of the learned rules using the pattern search optimization method. The parameter tuning was terminated after 50 iterations.

4.2. Results

In this section, the results of the proposed FIS tree mechanism for the classification accuracy prediction are presented. The performance of the proposed FIS tree model was validated using ADL dataset. In the experimental studies, RMSE was used to measure learning and tuning performances. Figures 6a and 6b illustrate the best RMSE values during the learning and tuning of the proposed FIS tree model, respectively.

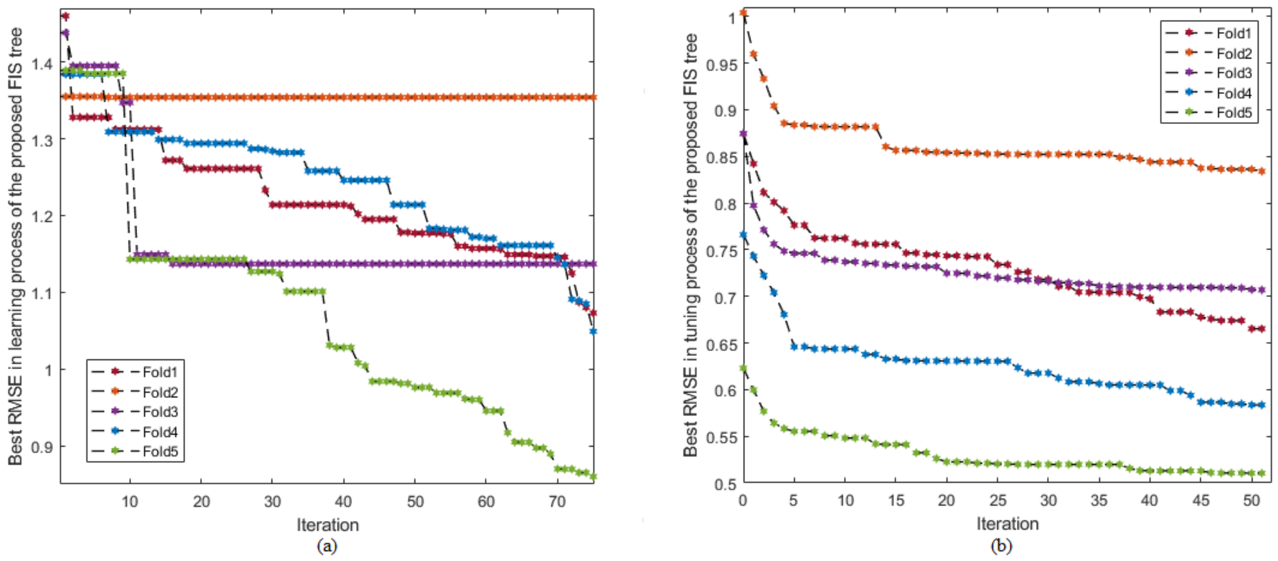


Figure 6. Best RMSE values during (a) learning and (b) tuning of the proposed FIS tree model.

Looking at Table 5, it is seen that 600 (10x5x4x3) different hyperparameter combinations (number of different combinations of hyperparameters) should be created. Instead of trying 600 different combinations, successful classification predictions were obtained with less experimental work using only 100 experimental test samples and an average of 104 learned rules.

Comparing with the validation results for prediction of accuracy, the proposed system shows high accuracy for accuracy prediction. It was found that the system has good training ability and could be trusted. The RMSE values during learning, tuning, and test processes are given in Table 7 for each fold.

Table 7. The RMSE values during learning, tuning, and test processes using validation set.

Fold no	Total number of learned rules	Learning RMSE	Tuning RMSE	Test RMSE
1	108	1.3931	1.5147	1.5147
2	102	1.2261	1.0848	1.0848
3	103	1.5968	1.4293	1.4293
4	104	1.0822	0.9610	0.9610
5	102	1.2899	1.3366	1.3366

Successful classification accuracy predictions for different parameter combinations which are model type, initial learning rate, minibatch size, and epoch number can be made with an average RMSE value of 1.2652. While accuracy prediction with the smallest RMSE was obtained in fold4, the highest RMSE value was obtained in fold1. Average actual and predicted accuracy values and RMSE values during the test process are illustrated in Figure 7. In this figure, the actual accuracy, the expected accuracy, and the difference between these two values are presented visually.

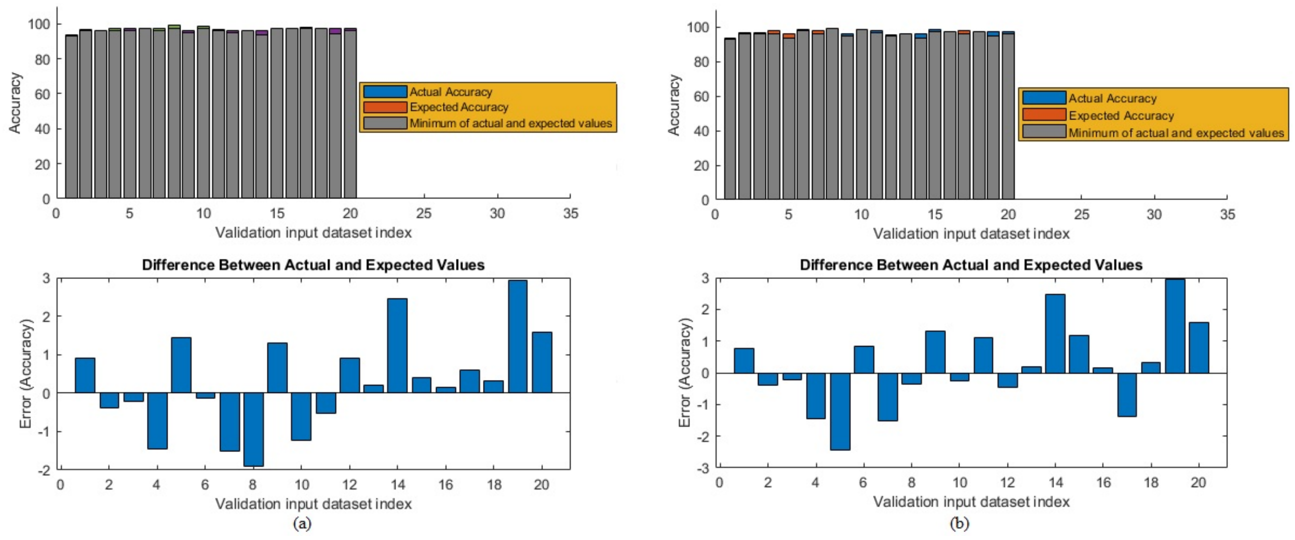


Figure 7. (a) Average actual and predicted accuracy during learning process (b) RMSE error values of test data during test process.

The hyperparameter optimization process was employed to find the best model and optimum hyperparameters for the histopathological data classification task. According to experimental results, the best hyperparameters are as follows: initial learning rate is 0.001, minibatch size is 128, and the number of epochs is 10. Table 8 lists and Figure 8 illustrates the performance measures of the various CNN models based on the best training hyperparameters for the histopathological data classification task, respectively.

According to experimental results as given in Table 8, all CNN models have classification success with a rate of over 93.60%. GoogleNet has outperformed other CNN models with an accuracy of 97.70%. Moreover,

Table 8. Performance measures of the various CNN models based on the best training hyperparameters.

Model type	Model name	Accuracy (%)	Precision (%)	Recall (%)	F-measure (%)
0	DenseNet201	97.56	97.67	97.55	97.61
1	EfficientNet	93.60	93.92	93.60	93.76
2	GoogleNet	97.70	97.79	97.69	97.74
3	ResNet18	96.45	96.61	96.44	96.53
4	ResNet50	96.54	96.68	96.55	96.61
5	ResNet101	96.92	97.11	96.88	96.99
6	ShuffleNet	96.42	96.55	96.40	96.47
7	MobileNetv2	96.28	96.68	96.26	96.47
8	Inceptionv3	96.43	96.56	96.41	96.49
9	InceptionResNetv2	95.14	95.31	95.14	95.22

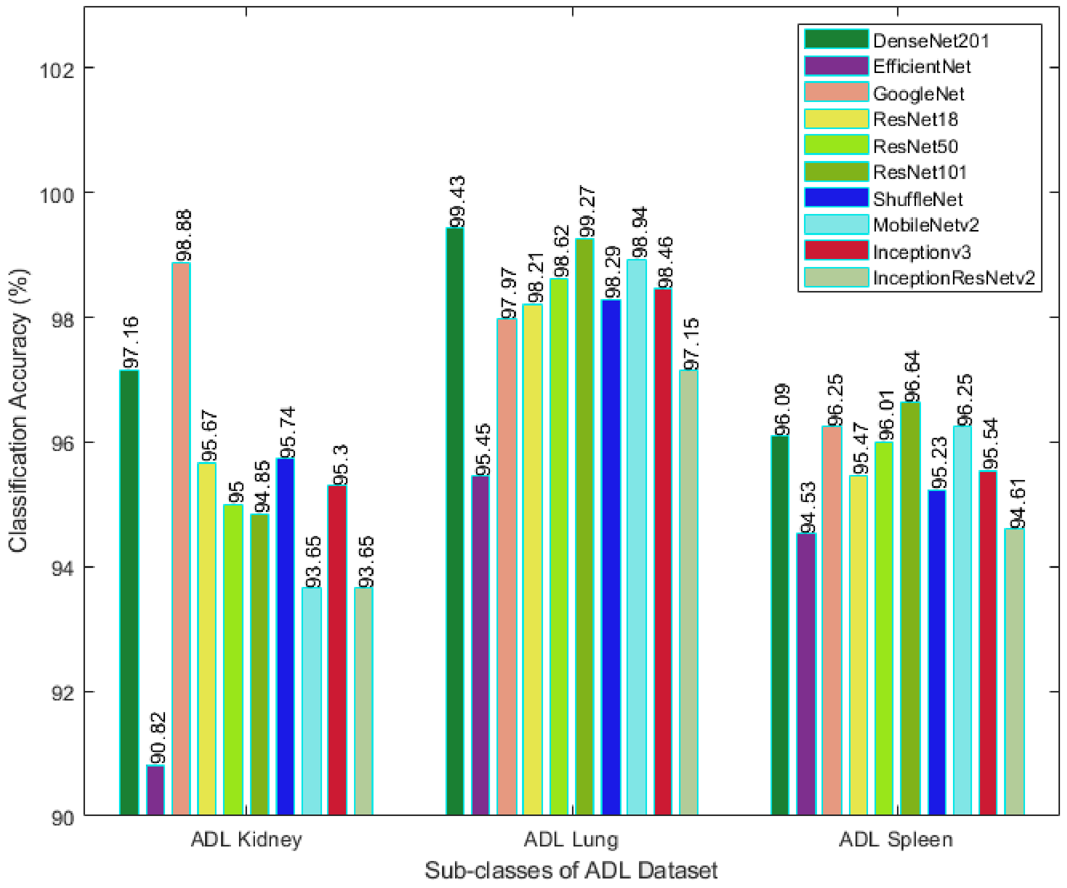


Figure 8. Classification accuracy results of CNN models with the best hyperparameter values.

GoogleNet obtained the best values for all calculated performance metrics. DenseNet201 and ResNet101 follow this model with a classification accuracy of 97.56% and 96.92%, respectively. Finally, all precision and recall values are very close to each other due to the low FP and FN rates. It supports the idea that the proposed method is a successful model.

Figure 8 illustrates the classification performances of the CNN models for histopathological data classes. GoogleNet increased the accuracy value by up to 97.70% and outperformed all other CNN models because GoogleNet has a good feature extraction ability due to inception blocks and it gives successful classification performances for small datasets. As it can be seen from Figure 8, although images with lungs can be classified more easily, it is much more difficult to distinguish between kidney and spleen histopathological images.

In this study, classification accuracy prediction based on CNN topology and training hyperparameters taken as input attributes of the proposed FIS tree has been discussed comprehensively. Experimental test samples were generated by applying various hyperparameter combinations to CNN architectures and a new application was presented to the literature with the proposed fuzzy tree model.

Additionally, Cifar10 and Cifar100 benchmark datasets were experimented to reveal the true capability and limitations of the proposed approach. The accuracy, precision, recall, and f-measure performance metrics of each CNN model for the classification task of Cifar10 and Cifar100 datasets are listed in Table 9.

Table 9. Performance measures of the various CNN models based on the best training hyperparameters for the classification of Cifar10 and Cifar100 datasets.

Dataset	CNN model	Accuracy (%)	Precision (%)	Recall (%)	F-measure (%)
Cifar10	DenseNet201	91.66 ± 0.0028	91.65 ± 0.0031	91.66 ± 0.0028	91.65 ± 0.0030
	EfficientNet	88.77 ± 0.0014	88.76 ± 0.0015	88.77 ± 0.0014	88.76 ± 0.0015
	GoogleNet	88.63 ± 0.0042	88.81 ± 0.0036	88.63 ± 0.0042	88.72 ± 0.0038
	ResNet18	88.02 ± 0.0040	88.04 ± 0.0041	88.02 ± 0.0040	88.03 ± 0.0040
	ResNet50	89.48 ± 0.0031	89.47 ± 0.0032	89.48 ± 0.0031	89.48 ± 0.0032
	ResNet101	90.21 ± 0.0017	90.22 ± 0.0016	90.21 ± 0.0017	90.22 ± 0.0017
	ShuffleNet	87.06 ± 0.0024	87.08 ± 0.0025	87.06 ± 0.0023	87.07 ± 0.0024
	MobileNetv2	86.57 ± 0.0023	86.58 ± 0.0028	86.57 ± 0.0023	86.57 ± 0.0025
	Inceptionv3	90.35 ± 0.0018	90.37 ± 0.0018	90.35 ± 0.0018	90.36 ± 0.0018
	InceptionResNetv2	90.16 ± 0.0019	90.17 ± 0.0020	90.16 ± 0.0019	90.16 ± 0.0020
Cifar100	DenseNet201	94.69 ± 0.0102	94.76 ± 0.0098	94.69 ± 0.0102	94.73 ± 0.0100
	EfficientNet	91.06 ± 0.0141	91.22 ± 0.0138	91.06 ± 0.0141	91.14 ± 0.0140
	GoogleNet	91.53 ± 0.0116	91.85 ±0.0121	91.53 ± 0.0116	91.69 ± 0.0118
	ResNet18	90.78 ± 0.0069	90.93 ± 0.0064	90.78 ± 0.0069	90.86 ± 0.0067
	ResNet50	92.89 ± 0.0114	93.00 ± 0.0109	92.89 ± 0.0115	92.94 ± 0.0112
	ResNet101	93.23 ± 0.0127	93.36 ± 0.0119	93.23 ± 0.0126	93.29 ± 0.0123
	ShuffleNet	88.93 ± 0.0143	89.06 ± 0.0137	88.93 ± 0.0143	88.99 ± 0.0140
	MobileNetv2	89.72 ± 0.0197	89.85 ± 0.0194	89.72 ± 0.0197	89.79 ± 0.0196
	Inceptionv3	93.27 ± 0.0102	93.37 ± 0.0093	93.27 ± 0.0102	93.32 ± 0.0102
	InceptionResNetv2	94.41 ± 0.0082	94.51 ± 0.0083	94.41 ± 0.0082	94.46 ± 0.0083

Table 9 shows that all CNN models have classification success with a rate of over 86.57%. DenseNet201 outperformed other CNN models with an accuracy of 91.66% and 94.69% for Cifar10 and Cifar100 classification tasks, respectively. Moreover, DenseNet201 obtained the best values for all calculated performance metrics for the classification of both Cifar10 and Cifar100 datasets. Because DenseNet201 has the benefit of good feature extraction especially for large datasets due to dense blocks that each layer obtains additional inputs from all previous layers and transfers its feature maps to all subsequent layers. Additionally, this model encourages the reuse of features and reduces the number of parameters. Moreover, DenseNet201 gives satisfactory results for the classification task of large datasets.

In order to show the effectiveness of the proposed FIS tree model, hyperparameter optimization with the proposed FIS tree model and other methods of classification and different feature extractors were compared. Table 10 lists the results of the comparison of different techniques for the classification task. The first five rows correspond to the different methods previously tested [6] and the sixth row corresponds to the result of this work.

Table 10. Comparison of the proposed FIS tree model with other methods.

Classifier/feature extractor	Local binary patterns	Geometric-based (%)	Convolution filters (%)
K-nearest neighbors	70	61	65
Artificial neural network	74	51	61
Support vector machine	69	61	63
CNN	-	-	74
CNN+Genetic algorithms	-	-	82
Proposed FIS tree model	-	-	96.30

The classification process which was carried out without the need for feature extraction and using fuzzy logic approach gave more successful results. This improvement shows that we can obtain favorable results for the hyperparameter optimization of various CNN topologies using the proposed FIS tree models.

4.3. Discussion

In this study, a new FIS tree model was designed for the classification accuracy prediction depending on various CNN models and training hyperparameter sets. Histopathological images of normal and inflamed kidney, lung, and spleen were used to perform experimental studies. By taking different CNN architectures and different training hyperparameter combinations, experimental test samples dataset was created. To generate each test sample, the hyperparameter array was taken as the training parameter and tested on the CNN model, which was determined as the model type. When different CNN architectures and training hyperparameters are given, the effects of these parameters on classification were examined and classification success was estimated with the proposed FIS tree model. Successful classification accuracy predictions for different parameter combinations which are model type, initial learning rate, minibatch size, and the number of epoch can be made with an average RMSE value of 1.2652. When the CNN model and training hyperparameter set are given, almost 99.00% accuracy is obtained with the proposed FIS tree model in calculating the classification accuracy. The best, the worst, and average classification accuracies obtained with CNN models that use optimal hyperparameter set are 97.70%, 93.60%, and 96.30%, respectively. Additionally, the proposed method also obtained satisfactory results for the classification task of large datasets such as Cifar10 and Cifar100. Experimental results conduct that the proposed fuzzy approach gives satisfactory results for the classification task of various object types.

5. Conclusions

In this study, a new FIS tree model that is more computationally efficient and easier to understand than a single FIS was presented to automatically predict the classification accuracy of the CNN models in various structures based on training hyperparameter set. In order to analyze different CNN architectures and interpret the results, ten well-known CNN models have been considered. Due to the designed FIS tree model, an inference mechanism that calculates the success of the classification has been developed by taking the selected CNN model and the determined set of training hyperparameters as input. With the proposed FIS tree model, high accuracy predictions of classification success can be made in classification problems using various CNN topologies and training hyperparameters. Instead of trying all combinations of hyperparameter values, an experimental set can be created with simpler experimental test samples and classification success of the designed CNN model can be estimated according to the given training hyperparameter values. More broadly, analysis of training hyperparameters using different CNN topologies can be performed using the proposed FIS tree model which determines the designed fuzzy model parameters by performing automatic learning and tuning and eliminating human factor.

Conflict of interest statement

On the behalf of the corresponding author, all authors declare that they have no conflicts of interest.

Authors' contributions

All authors contributed to the manuscript and approved it.

Acknowledgments

This study was supported by Selçuk University and Coordinatorship of Faculty Member Training Program with Project No: 2017-OYP-047.

References

[1] Bochinski E, Senst T, Sikora T. Hyper-parameter optimization for convolutional neural network committees based on evolutionary algorithms. In: IEEE 2017 ICIP International Conference on Image Processing; 2017. pp. 3924-3928.

[2] Mukhtar N, Kong Y. Hyper-parameter optimization for machine-learning based electromagnetic side-channel analysis. In: 2018 ICSEng 26th International Conference on Systems Engineering; 2018. pp. 1-7.

[3] Soon FC, Khaw HY, Chuah JH, Kanesan J. Hyper-parameters optimisation of deep CNN architecture for vehicle logo recognition. Intelligent Transport Systems 2018; 12 (8): 939-946. doi: 10.1049/iet-its.2018.5127

[4] Yoo Y. Hyperparameter optimization of deep neural network using univariate dynamic encoding algorithm for searches. Knowledge-Based Systems 2019; 178: 74-83. doi: 10.1016/j.knosys.2019.04.019

[5] Zhang M, Li H, Lyu J, Ling SH, Su S. Multi-level CNN for lung nodule classification with gaussian process assisted hyperparameter optimization 2019. arXiv preprint arXiv:1901.00276.

[6] Cabada RZ, Rangel HR, Estrada MLB, López HMC. Hyperparameter optimization in CNN for learning-centered emotion recognition for intelligent tutoring systems. Soft Computing 2020; 24 (10): 7593-7602. doi: 10.1007/s00500-019-04387-4

[7] Pan S, Guan H, Chen Y, Yu Y, Gonçalves WN et al. Land-cover classification of multispectral LiDAR data using CNN with optimized hyper-parameters. ISPRS Journal of Photogrammetry and Remote Sensing 2020; 166: 241-254. doi: 10.1016/j.isprsjprs.2020.05.022

[8] Andonie R, Florea AC. Weighted random search for CNN hyperparameter optimization. International Journal of Computers Communications & Control 2020; 15 (2): doi: 10.15837/ijccc.2020.2.3868.

[9] Wang Y, Wang Y, Li H, Cai Z, Tang X et al. CNN Hyperparameter optimization based on CNN visualization and perception hash algorithm. In: 2020 DCABES 19th International Symposium on Distributed Computing and Applications for Business Engineering and Science; 2020. pp. 78-82.

[10] Yilmaz AA, Guzel MS, Bostanci E, Askerzade I. A novel action recognition framework based on deep-learning and genetic algorithms, in IEEE Access 2020; 8: 100631-100644. doi:10.1109/ACCESS.2020.2997962.

[11] Soylu K, Güzel MS, Soylu BE, Bostanci GE. Genetic hyperparameter optimization library development and its application on plant disease detection problem. In: 2020 SIU 28th Signal Processing and Communications Applications Conference; 2020. pp. 1-4. doi: 10.1109/SIU49456.2020.9302246

[12] Singh P, Chaudhury S, Panigrahi BK. Hybrid MPSO-CNN: Multi-level particle swarm optimized hyper-parameters of convolutional neural network. Swarm and Evolutionary Computation 2021; 63:100863. doi: 10.1016/j.swevo.2021.100863

[13] Yeh WC, Lin YP, Liang YC, Lai CM. Convolution neural network hyperparameter optimization using simplified swarm optimization 2021. arXiv preprint arXiv:2103.03995.

[14] Sugeno M. Industrial Applications of Fuzzy Control. USA: Elsevier Science Inc., 1985.

[15] Vu TH, Mousavi HS, Monga V, Rao G, Rao UKA. Histopathological image classification using discriminative feature-oriented dictionary learning. IEEE 2016 Transactions on Medical Imaging 2016; 35 (3): 738-751. doi : 10.1109/TMI.2015.2493530

[16] Perez L, Wang J.The effectiveness of data augmentation in image classification using deep learning 2017. arXiv preprint arXiv:1712.04621.

[17] Xu Y, Jia R, Mou L, Li G, Chen Y et al. Improved relation classification by deep recurrent neural networks with data augmentation 2016. arXiv preprint arXiv:1601.03651.

[18] Wong SC, Gatt A, Stamatescu V, McDonnell MD. Understanding data augmentation for classification: when to warp?. In: IEEE 2016 DICTA International Conference on Digital Image Computing: Techniques and Applications; 2016. pp. 1-6.

[19] İnik Ö, Uyar K, Ülker E. Gender classification with a novel Convolutional Neural Network (CNN) model and comparison with other machine learning and deep learning CNN models. Journal Of Industrial Engineering Research 2018; 4 (4): 57-63.

[20] Thenmozhi K, Reddy US. Crop pest classification based on deep convolutional neural network and transfer learning. Computers and Electronics in Agriculture 2019;164: 104906. doi: 10.1016/j.compag.2019.104906

[21] Breuel TM. The Effects of Hyperparameters on SGD Training of Neural Networks 2015. arXiv preprint arXiv:1508.02788.