An Intelligent Handwritten Arabic Characters Recognition System

Asmaa El-Maghraby[1], Mohamed Tarek[1], Reham Galal[1], Toqa Hamdy[1], Youssef AbdelAzim[1] and Fady Anis[1]

Contributing authors: a.elmaghraby@nu.edu.eg; M.TarekSaad@nu.edu.eg; R.Galal@nu.edu.eg; T.Hamdy@nu.edu.eg; Y.AbdelAzim@nu.edu.eg; F.Anis@nu.edu.eg;

Abstract

Handwritten character recognition is the process of teaching machines to recognize human handwritten characters. Handwritten letters are imperfect, differ from person to person, and can be flavored in a variety of ways. As a result, it is not a straightforward task for the machine. A machine learning approach was developed in this research to recognize handwritten numerals and characters with excellent accuracy in comparison to previous models. The dataset used for the Arabic letter dataset is Kaggle's Arabic Handwritten Characters Recognition. It has 16,800 images of Arabic letters in total. As a deep learning method, the convolutional neural network (CNN) methodology was used. The proposed CNN model is based on the Keras model, which uses an Adam optimizer to classify handwritten digit images. The proposed CNN model achieves 98.80% accuracy during testing and 99.06 % accuracy during training. The average macro accuracy was 0.99. It has an average macro recall of 0.98. The global average F1 score was 0.99.

Keywords: Machine Learning, Classification, Supervised Learning, CNN

1 Introduction

Handwritten Arabic character recognition has attracted considerable attention in recent days and become a popular field of research with the rapid development of deep learning and advances in technology[1]. Several studies used

many methods, such as support vector machines (SVMs), K-nearest neighbors (KNNs), neural networks (NNs), Multi-Layer Perceptron (MLP), Hidden Model Markov (HMM), Recurrent Neural Networks (RNN), and convolutional neural networks (CNNs).

Moreover, Arabic is written from right to left and contains 28 letters, each of which has 2–4 different forms depending on the letter's position within a word[2]. The Arabic language is a Semitic language, and more than 360 million people use Arabic as their mother tongue in the countries in the Middle East[3]. It is also, one of the six official languages of the United Nations and the basis for many other languages spoken in the Muslim world, including Persian, Pashto, Sindhi, and Urdu. It would be great to convert many documents into a digital form that can be viewed electronically using Arabic handwritten character recognition. Applications include assisting the blind in reading, reading postal addresses off envelopes, automating offices, archiving, reading customer-filled forms, and retrieving text, as well as enhancing human-computer interfaces[4].

Deep Learning is an application of machine learning for learning the representation of data. It has a very high number of connections and trainable parameters since they have an input layer, numerous nonlinear hidden layers, and an output layer[5].

Convolutional Neural Network (CNN) is a type of Deep Neural Network with a comparatively smaller set of parameters and is easier to train[6].It is a multi-layer feed-forward neural network that extracts features and properties from the input data (images). The input data in a CNN are processed in a grid-like topology[7]. For instance, an image can be viewed as a grid of input values with a size of D $\times$ D $\times$C, where D$\times$D is the number of pixels in the image and C is the number of channels per pixel (1 for grayscale or 3 for RGB)[2]. Convolutional layers, pooling layers, and fully linked layers are fed into this input grid. The following subsections contain a brief description of each layer.

There are several frameworks for Deep Learning as Keras [8] which is a higher-level built on top of TensorFlow and uses Python for programming. Therefore, this paper presents an intelligent approach to handwritten Arabic character recognition using CNN model on the Arabic handwritten characters dataset (AHCR)[9] dataset .It has 16,800 images of Arabic letters in total characters by using TensorFlow/Keras, which uses an adam optimizer to classify handwritten digit images.

The rest of the paper is organized as follows; Section 2 discusses related work in the field of Arabic handwritten characters recognition and its background. Section 3 describes the methodology, section 4 gives an overview of the results, and Section 5 discusses conclusion derived from the results and presents plans for future work.

2 Related Work

Several methods have been proposed, with high recognition rates reported for Chinese[10][11],Urdu[12] ,and English[13][14]. In recent times, Many

researchers have focused on Arabic Handwritten character recognition. Younis[4] applied a CNN model based on AIA9k dataset that contains 8738 images and AHDC dataset that contains 16800 images. Using the CNN with regularization parameters such as batch normalization for handwritten Arabic character recognition. CNN consisted of three convolutional layers followed by a fully connected layer. Results showed that the CNN was able to achieve accuracy of 94.8% and 97.6% on the AIA9k that contain 8738 images and AHDC datasets, respectively.

Al-Taani[15] presented a Residual Neural Network (ResNets)to recognize Arabic Handwritten characters including Arabic digits. The proposed approach achieved accuracies of 99.8 %, 99.05 %, and 99.55% on three datasetson three datasets MADbase(70000 images of Arabic digits), AIA9K(8738 images), and AHCD(16800 images) datasets, respectively. It also achieved a validation accuracy of 98.9% on the constructed dataset based on the three datasets. Alaasam et al[16]. used a CNN to recognize historical Arabic handwritten text. Experiments were conducted on three datasets: historical handwritten text dataset has 74071 images, synthesized text dataset has 76002 images, and printed text dataset has 96289 images. The best accuracy, 85 %, was obtained for the combination of the handwritten text image dataset and the synthesized text image dataset.

Altwaijry and Al-Turaiki[2] evaluated the offline model for recognizing Arabic letters based on the CNN model using two datasets: Hijja which contains 47,434 images of single Arabic characters written by children aged 7–12 years and AHCD containing 16,800 images written by 591 participants. The performance of the model shows accuracies of 97% and 88% on the AHCD dataset and the Hijja dataset. Alabodi and Li [17] proposed a new recognition system based on the geometrical features of Arabic characters. The IFN/ENIT database that has 2200 binary images of handwriting sample forms from 411 writer used in their results. 60% of the words were chosen randomly as a training data set, and the remaining words were used as the testing data set. The average result of the recognition rate is 93.3% for 596 words.

El-Sawy et [1]. used convolutional neural networks (CNN) for Arabic character recognition. The purpose to use deep learning was present the advantages of the power of CNN which are to manage large dimensions of input. The Results showed accuracy with 94.9% on testing images on the AHCD(16800 images) dataset. Balaha et al[18]. selected the best 14 native CNN architectures that differ in a hierarchy of constructing the architecture layers, the best result in a testing accuracy of 91.96 handwritten HMBD dataset that contains 54,115 characters. Furthermore, they improve the optimization methods by transfer learning (TF) and genetic algorithm (GA), leading to a testing accuracy of 92.88 percent.

Ahmed et al[19]. presented a CNN model that extracts optimal features and applied batch normalization and dropout regularization layers for enhancing the overall performance. The model has shown excellent classification results on a diverse set of six benchmark databases, including MADBase

(Digits), CMATERDB (Digits), HACDB (Characters), SUST-ALT (Digits), SUST-ALT (Characters), and SUSTALT (Names). Although, the accuracy of this model is above 99 %. Ullah et al[20]. proposed using the CNN model to recognize handwritten Arabic letters. The model is regularized using batch normalization and dropout operations. Moreover, the model was tested with and without dropout, resulting in a significant difference in the performance. Used AHCD dataset with 16,800 letters and The experimental three types of layers make a model in terms of higher accuracy results that reached 96.78 %.

According to previous studies, No researchers have applied an Intelligence approach to Arabic character recognition. Therefore, this paper presents a graphical character interface (GUI) that can draw a letter using a camera on a 2D-like board in the real-world view, and instantaneously view a digital replica of it along with an accuracy percentage. The CNN model is regularized using batch normalization and dropout training, which uses an Adam optimizer to classify handwritten digit images.

3 Methodology

3.1 Data collection

This paper used the Arabic handwritten characters dataset (AHCD) from Kaggle[9]. It has 16,800 images of Arabic letters in total characters by 60 participants, the age range is between 19 to 40 years, and 90% of the participants are right-hand. The dataset was divided into four files(test images, testing labels, train images, and training labels). The total number of Arabic class labels used in AHCD is 28. Moreover, the 16,800 characters were split into two sets in that 80% of the letters were in training and 20% were in the test set. The training set consisted of 13,440 letters divided into 480 images per class and the test set of 3,360 letters divided into 120 images.

3.2 Data cleaning and processing

Data cleaning is an important step in improving data quality by removing garbage[21] and preparing it for the best-fitting model. The images were reshaped to 32×32 pixels. To overcome such challenges, the input image is reshaped and fed to the network. To improve performance, flipping and rotations were performed, and the entire array was reshaped to a size of 32×32×1. The 32 ×32×1 size image is a greyscale image. Furthermore, the images were normalized. The categorical class labels were also exceeding in the sense that integer values representing different categories were transformed into a binary value matrix.

3.3 Convolutional Neural Network

Convolutional neural networks can convert the input structure seamlessly through each layer of the network and extract the image features

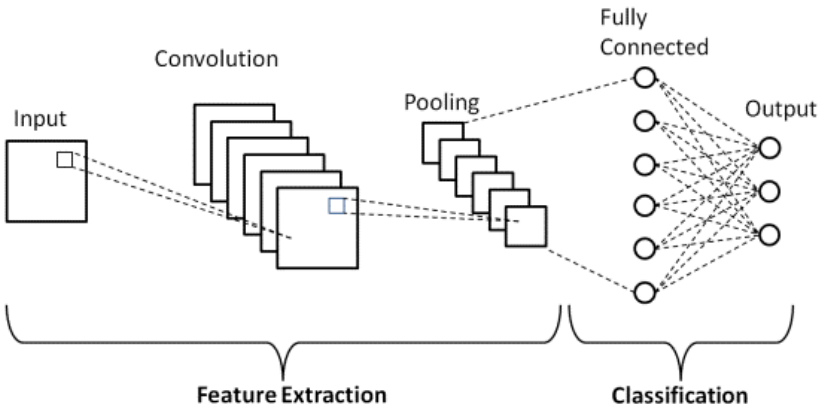

Fig. 1 CNN Architecture

automatically[4]. CNN is based on a mathematical operation called convolution. A convolution is a multiplication operation that multiplies each pixel in the image by each value in the kernel, which is another matrix, and then adds the results. The main advantage of using the convolution operation is that it generates many images from the original image that enhance different features extracted from the original image, resulting in a more powerful classification process[22]. In CNN, three types of layers make up CNN which are convolutional layers, pooling layers, and fully-connected layers as shown in fig 1.

3.3.1 convolutional layer

Convolution is applied to at least one layer of the CNN. A convolutional layer consists of several filters K. Convolutional layer acts as a feature extractor that extracts salient features of the inputs such as corners, edges, and endpoints. A filter is a grid of size N × N × R, where N is the height and width of the filter, and R is the number of image channels C. Each filter slides over all locations in the input grid, multiplying each pixel by the corresponding value in the filter. Then multiplications are all added to produce a single number. Each convolution produces a feature map[23], as shown in Fig 2. The result of applying K convolutional filters is a set of K feature maps, each of size D-N + 1. Feature maps are passed to another mathematical operation called the activation function. This is done to allow for the classification of linearly separable classes[2].

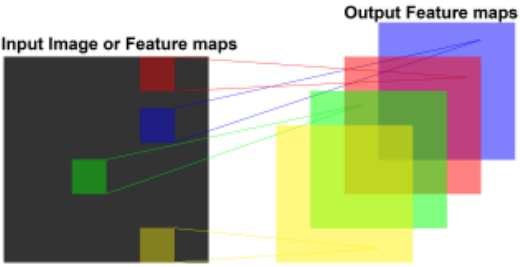

Fig. 2 Illustration of a single convolutional layer that produce k-feature maps

3.3.2 Pooling layer

The convolutional layer is followed by the pooling layer. The pooling layer takes small rectangular blocks from the convolutional layer and subsamples them to produce a single output from that block. The pooling layer reduces the resolution of the image which reduces the precision of the translation (shift and distortion) effect. There are several ways to do this pooling, such as taking the average or the maximum, or a learned linear combination of the neurons in the block. Our pooling layers will always be max-pooling layers; which takes the maximum of the block that they are pooling. All the max-pooling is carried out over a 2 × 2 pixels window[1], as shown in fig 3.

Fig. 3 Illustration of applying max pooling with a window of size 2 × 2

3.3.3 Fully connected layer

After several convolutional and pooling layers, the high-level reasoning in the neural network is done via fully connected layers. Fully connected layers are the basic building blocks of traditional neural networks. They treat the input as one vector instead of two-dimensional arrays. Full connection implies that every neuron in the previous layer is connected to every neuron in the next layer. The output from convolutional and pooling layers represents high-level features and fully connected layers used to classify material (input images) into the appropriate class based on the training of the dataset[24].

3.4 CNN Optimization

In this model, we tested two optimizers: Stochastic Gradient Descent and Adam, and selected Adam as it was found to work better. The loss function

used is the categorical cross-entropy loss, which is widely used to calculate the probability that the input belongs to a particular class. It is usually used as the default function for multi-classification. The training set was divided into several batches, where each batch size consisted of an equal number of samples. Moreover, the model was tested using different number epochs.

3.4.1 Activation Function

The activation function that is used in our approach is non- Linear Units (ReLU) :

$$RELU(X) = max(x, 0) \tag{1}$$

We apply the ReLU non-linearity as an activation function as it has faster convergence than conventional sigmoidal functions, to the output of every convolutional layer and fully connected layer[25]. The ReLU increases the non-linear properties of the decision function and the overall network without affecting the receptive fields of the convolution layer. The use of ReLUs maps more plausibly to biological neurons makes the training of deep neural networks significantly faster and improves its generalization ability[1].

3.4.2 Stochastic Gradient Descent

The gradient descent algorithm updates the parameters (weights and biases) to minimize the error function by taking small steps in the direction of the negative gradient of the loss function[1]:

$$\pi + 1 = \pi - \alpha \nabla E(\pi) \tag{2}$$

where 1 stand for the iteration number, $\alpha > 0$ is the learning rate, P is the parameter vector, and E(π) is the loss function. The gradient of the loss function, ∇E(π), is evaluated using the entire training set, and the standard gradient descent algorithm uses the entire data set at once.

3.4.3 Mini-batch and momentum

The stochastic gradient descent algorithm evaluates the gradient, hence updating the parameters, using a subset of the training set. This subset is called a mini-batch. Mini-batch optimization [26] is to divide the dataset into small batches of examples, compute the gradient using a single batch, make an update, then move to the next batch. Each evaluation of the gradient using the mini-batch is an iteration. At each iteration, the algorithm takes one step towards minimizing the loss function. The full pass of the training algorithm over the entire training set using mini-batches is an epoch. We specify the mini-batch size as 256 and the maximum number of epochs is 30. The gradient descent algorithm might oscillate along the steepest descent path to the optimum. Adding a momentum term[27] to the parameter update is one way to prevent this oscillation. The SGD update with momentum is:

$$\pi + 1 = \pi - \alpha \nabla E(\pi) + \gamma(\pi - \pi - 1) \tag{3}$$

where γ determines the contribution of the previous gradient step to the current iteration.

3.4.4 Regularization

One problem with machine learning estimation is that it can result in overfitting. Adding a regularization term for the weights to the loss function E(Pi) is a way to reduce overfitting. The loss function with the regularization term takes the form[1]:

$$ER(\pi) = E(\pi) + \lambda\Omega(w) \tag{4}$$

where w is the weight vector, λ is the regularization factor (coefficient), and the regularization function, Ω(w) is:

$$\Omega(w) = \frac{1}{2}\, w^2 w \tag{5}$$

3.4.5 Softmax function

The softmax classifier used in the output layer ensures a set of conditional probabilities, for each instance class, whose sum is equal to unity [28]:

$$softmax(x) = \frac{e^x}{\sum_{K=1}^{N} e^x} \tag{6}$$

4 Results

We describe the results obtained after testing the proposed Convolutional Neural Network on the AHCD dataset. We divided the data into three training, and testing with 80% and 20% ratios for each set. The model was formed with Adam, the loss was calculated with categorical cross-entropy, and the metric was set to accuracy. Then we trained for 50 epochs with 20 batch sizes. The proposed CNN model achieves 98.80% accuracy during testing and 99.06 % accuracy during training as shown in fig 4. fig 5 shows training and validation losses.

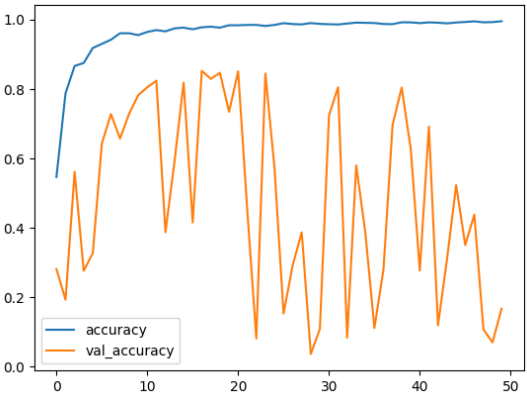

Fig. 4 Training and Validation accuracy

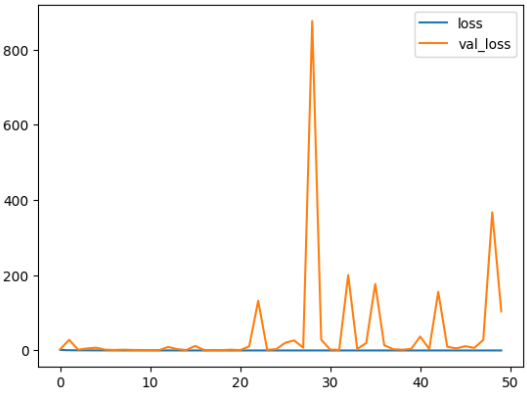

Fig. 5 loss Training and Validation accuracy

The proposed model's performance was evaluated using the following equations for accuracy, precision, recall, and f-measure.

Recall (R): is the fraction of correctly classified images over the total number of images that belong to class x:

$$R = \frac{TP}{TP + FN} \tag{7}$$

Precision (P): the fraction of images that are correctly classified and the total number of images classified:

$$p = \frac{TP}{TP + Fp} \tag{8}$$

F1 measure: which is a measure that combines Recall and Precision:

$$F1 = \frac{2 * precision * recall}{precision + recall} \tag{9}$$

5 Conclusion

This paper presents an intelligence approach for Arabic handwritten recognition using the Convolutional Neural Network(CNN) model. t. The CNN model uses dropout training to prevent overfitting and batch normalization from normalizing the outputs and allowing independent learning of each layer in the network. ReLu was used as an activation function in all layers except the softmax layer and the model was created using the Adam optimizer to classify handwritten digit images. The output is a graphical character interface (GUI) that can draw a letter using a camera on a 2D-like board in the real-world view, and instantaneously view a digital replica of it along with an accuracy percentage. our model achieves excellent results in comparison to previous models using Kaggle's Arabic Handwritten Characters Recognition (AHCD) dataset. The model achieves 98.80% accuracy during testing and 99.06% accuracy during training using epoch 10. The average macro accuracy was 0.99.

In future work, we can use a dataset that will generate by a person writing letters in mid-air continuously for every character by using Visual Reality (VR) For Arabic Handwritten characters recognition by applying this model. We also can apply the model to Arabic digits, and handwritten alpha-numerical items occurring in similarly patterned languages, such as Urdu, Persian, and Pashto.

References

[1] El-Sawy, A., Loey, M., El-Bakry, H.: Arabic handwritten characters recognition using convolutional neural network. WSEAS Transactions on Computer Research **5**(1), 11–19 (2017)

[2] Altwaijry, A.-T. N.: Arabic Handwriting Recognition System Using Convolutional Neural Network. Neural Comput Applic 33, 2249–2261. https://doi.org/10.1007/s00521-020-05070-8

[3] J, O.: the oxford handbook of arabic linguistics. oxford university press, oxford. (2013)

[4] Khaled, Y.: Arabic handwritten character recognition based on deep convolutional neural networks. Jordanian Journal of Computers and Information Technology (JJCIT) **3** (2018)

[5] Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Pereira, F.,

Burges, C.J., Bottou, L., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems, vol. 25. Curran Associates, Inc., ??? (2012). https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

[6] Liang, Y., Wang, J., Zhou, S., Gong, Y., Zheng, N.: Incorporating image priors with deep convolutional neural networks for image super-resolution. Neurocomputing **194**, 340–347 (2016)

[7] Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press. http://www.deeplearningbook.org

[8] Chollet, F., Keras GitHub Repository, [Online]. https://github.com/fchollet/kera

[9] Elsawy, A., Loay, M., El-Bakry, H.: Arabic handwritten characters dataset. https://www.kaggle.com/mloey1/ahcd1

[10] Xiao, X., Jin, L., Yang, Y., Yang, W., Sun, J., Chang, T.: Building fast and compact convolutional neural networks for offline handwritten chinese character recognition. Pattern Recognition **72**, 72–81 (2017)

[11] Zhong, Z., Jin, L., Feng, Z.: Multi-font printed chinese character recognition using multi-pooling convolutional neural network. 2015 13th International Conference on Document Analysis and Recognition (ICDAR), 96–100 (2015)

[12] Ali, H., Ullah, A., Iqbal, T., Khattak, S.: Pioneer dataset and automatic recognition of urdu handwritten characters using a deep autoencoder and convolutional neural network. SN Applied Sciences **2**(2), 1–12 (2020)

[13] Yuan, A., Bai, G., Jiao, L., Liu, Y.: Offline handwritten english character recognition based on convolutional neural network. 2012 10th IAPR International Workshop on Document Analysis Systems, 125–129 (2012)

[14] Bai, J., Chen, Z., Feng, B., Xu, B.: Image character recognition using deep convolutional neural network learned from different languages. In: 2014 IEEE International Conference on Image Processing (ICIP), pp. 2560–2564 (2014). IEEE

[15] Al-Taani, A.T.: Recognition of online arabic handwritten characters using structural features. TTERN RECOGNITION RESEARCH (2010)

[16] Alaasam, R., Kurar, B., Kassis, M., El-Sana, J.: Experiment study on utilizing convolutional neural networks to recognize historical arabic handwritten text. In: 2017 1st International Workshop on Arabic Script Analysis and Recognition (ASAR), pp. 124–128 (2017). IEEE

[17] Al Abodi, J., Li, X.: An effective approach to offline arabic handwriting recognition. Computers & Electrical Engineering **40**(6), 1883–1901 (2014)

[18] Balaha, H.M., Ali, H.A., Youssef, E.K., Elsayed, A.E., Samak, R.A., Abdelhaleem, M.S., Tolba, M.M., Shehata, M.R., Mahmoud, M.R., Abdelhameed, M.M., *et al.*: Recognizing arabic handwritten characters using deep learning and genetic algorithms. Multimedia Tools and Applications **80**(21), 32473–32509 (2021)

[19] Ahmed, R., Gogate, M., Tahir, A., Dashtipour, K., Al-Tamimi, B., Hawalah, A., El-Affendi, M.A., Hussain, A.: Novel deep convolutional neural network-based contextual recognition of arabic handwritten scripts. Entropy **23**(3), 340 (2021)

[20] Ullah, Z., Jamjoom, M.: Arabic handwritten letter recognition using convolutional neural network. PeerJ Computer Science **8**, 995 (2022)

[21] Ullah, Z., Al-Mudimigh, A.S.: Integration and communication to prevent dirty data: the role of madar project. International Information Institute (Tokyo). Information **15**(8), 3459 (2012)

[22] LeCun, Y., Kavukcuoglu, K., Farabet, C.: Convolutional networks and applications in vision. In: Proceedings of 2010 IEEE International Symposium on Circuits and Systems, pp. 253–256 (2010). IEEE

[23] Yamashita, R., Nishio, M., Do, R.K.G., Togashi, K.: Convolutional neural networks: an overview and application in radiology. Insights into imaging **9**(4), 611–629 (2018)

[24] Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Communications of the ACM **60**(6), 84–90 (2017)

[25] Yan, C., Coenen, F., Zhang, B.: Driving posture recognition by convolutional neural networks. IET Computer Vision **10**(2), 103–114 (2016)

[26] Li, M., Zhang, T., Chen, Y., Smola, A.J.: Efficient mini-batch training for stochastic optimization. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 661–670 (2014)

[27] Sutskever, I., Martens, J., Dahl, G., Hinton, G.: On the importance of initialization and momentum in deep learning. In: International Conference on Machine Learning, pp. 1139–1147 (2013). PMLR

[28] Fontes, C.H.: Refinement of the feedforward network in multi-class classification problems using a hybrid approach combining supervised clustering

and a fuzzy classifier. Engineering Applications of Artificial Intelligence **115**, 105242 (2022)