IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 4 May 2023, accepted 3 July 2023, date of publication 12 July 2023, date of current version 19 July 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3294566

RESEARCH ARTICLE

Handwritten Pattern Recognition Using Birds-Flocking Inspired Data Augmentation Technique

YIHAN XU[1] **AND AAMIR WALI**[2]
[1]Desheng Technology Company Ltd., Guangzhou 510630, China
[2]FAST School of Computing, National University of Computer and Emerging Sciences, Lahore 54770, Pakistan
Corresponding author: Aamir Wali (aamir.wali@nu.edu.pk)

ABSTRACT Recently, a cellular automata learning and prediction (CALP) model was proposed that considered images as living cells and used cellular automata to evolve and synthetically augment handwritten data. In this paper, another data augmentation method is proposed inspired by the flocking pattern of birds. It is proposed that any image sample in a handwritten data set can be represented as an assembly of birds. Each bird is specifically located at a point or pixel to collectively form an image. Using a well-defined flocking mechanism, the new positions for these birds can be calculated. Each snapshot of the new position can be considered as a new version of the original image, thus generating more data. The best flocking pattern is determined using biologically-inspired genetic algorithms. The quality of the synthetic data is demonstrated by using it in a handwritten pattern recognition system. For this purpose, the CALP framework is used with few modifications. It is shown that using the new data along with the original data to train classifiers improves the accuracy of classifiers than when they are trained using only the original data. Following the same test bed as CALP, 25 different data sets and classifier pairs were used for experimentation. The experimental results show that by also using the data set generated through flocking, the performance of the base classifiers is significantly improved even when smaller training data is used. We also compare the performance of the proposed technique with other state-of-the-art oversampling methods.

INDEX TERMS Data augmentation, flocking, genetic algorithm, handwritten pattern recognition, oversampling techniques, CALP, ensemble methods.

I. INTRODUCTION

Nature has always been a great source of inspiration for technological advancements. Researchers have found ways to use nature-inspired techniques in various domains, such as optimization, machine learning, robotics, and computer vision. This paper presents a nature-inspired augmentation technique to improve the performance of machine learning models for handwritten pattern recognition.

The success of machine learning models depends on the quality and quantity of the data used for training. The availability of large amounts of labeled data allows the models to generalize better. However, collecting and labeling data can be expensive and time-consuming. To address this issue, researchers have explored ways to augment the existing data to improve the performance of machine learning models. One promising approach is to use nature-inspired augmentation techniques. In this study, we focus on one such technique that uses the flocking pattern of birds.

In everyday use, flocking means to move together in the form of a crowd. This can apply to birds, fishes, and even bacteria. However, in a more technical sense, it refers to the process in which independent multi-agents arrange themselves into an ordered motion based on some rules and environmental variables. [1]

Flocking is now one of the basic collective responses of multi-agent systems. It has shown its effectiveness in different areas of computer science, engineering, finance, and

The associate editor coordinating the review of this manuscript and approving it for publication was Huiyu Zhou.

biology. It has been used to develop simulations of land combat for marine corps [2], to control the formation and movement of robots [3], space flight and unmanned aerial vehicles (UAV) [4] and to stabilize relative equilibria in moving steered particles [5]. It has also been used in the dynamic functional model of the human brain [6] and to model the volatility of financial markets and assets [7].

In this paper, we propose a nature-inspired data augmentation technique. First, we consider that in any image sample taken from a handwritten data set, the black pixels represent an assembly of birds and it shows their initial position. Then using a well-defined flocking mechanism, the new positions for these birds can be calculated. Each snapshot of the new position can be considered as a new augmented version of the original image. These new versions of the data set are referred to as *flocked* versions. Synthetic data set generation is used firstly to balance imbalanced data sets and secondly, to increase the size of the training data [8], [9], [10], [11]. Data generation is primarily of interest in situations where there is a need for a large amount of training data, such as in deep learning. Recent studies in image prediction and classification use rotation, zoom range, width shift, height shift, shear range, and horizontal flip, etc., for the augmentation of the datasets. These techniques are too simple, and there is a need to explore techniques that are nature-inspired.

Finally, this synthetic data generation technique is compared with other state-of-the-art over-sampling methods such as cellular automata learning and prediction model (CALP) [12], synthetic minority oversampling technique (SMOTE) [13] and its variations, ADASYN [14] and bootstrapping [15]. The comparison is done by also including the over-sampled data to train classifiers and then using unseen data to record the performance of the classifiers. To generate, train and test, the CALP [12] framework is used. CALP is basically a meta-ensemble that trains each independent classifier on different versions of the data and then combines them to form an ensemble.

Ensembles have been used extensively in classification and recognition systems. Some examples include handwritten digits recognition [16], pattern recognition [17], brain state recognition using fmri [18], semantic segmentation [19], early-stage diabetes prediction [20] and face recognition [21] and have application ranging from medical data [22] to crude oil price prediction [23]. CALP originally generates data by evolution through cellular automata. For experimentation, this data generation module is replaced by the proposed technique that uses flocking instead. Birds can flock in numerous ways depending on, for example, their initial direction. By changing the initial direction, different flock sequences can be generated. These sequences represent a flocked version of the same data set.

In this paper, we propose a data augmentation technique using flocking. The aim of this paper is four folds. First, to show that flocking data sets add more variations to the data set leading to improved training. Second, to show how various flocked versions of the data can be plugged into the CALP framework to form an ensemble. Third, to demonstrate that generating data using flocking can result in better classifiers than data generated using other over-sampling methods such as SMOTE, ADASYN etc. Finally, it is illustrated that when the training data is small and it is over-sampled using the proposed technique, the performance of classifiers improves significantly.

The main contributions of this paper are as follows:

- The only study that uses the flocking patterns of birds to generate or augment data.
- Validate by experiments that this technique is better than other state-of-the-art synthetic data generation/over-sampling techniques
- Use of genetic algorithms to generate the direction set
- Demonstrate how the previously proposed, biologically-inspired CALP model can be adapted for other nature-inspired phenomena (such as flocking patterns of birds).
- Finally, propose another model for handwritten pattern recognition (with CALP as a base study)

The rest of the paper is organized as follows. Section II provides a brief background on flocking. Section III presents the details and design of our model. Section IV highlights the results and discussion. The conclusion follows in section V.

II. BACKGROUND

In this paper we use flocking in the everyday sense, i.e., to move together in the form of a crowd. This can apply to birds, fishes and even bacteria. So, in this section, we examine various flocking models that are derived from this sense.

The earliest work and probably the first on modeling the movement of birds i.e. bird flocking was presented by [24]. The model takes in two input parameters: flock count and a constant for speed. The paper highlights that any movement of birds in the form of a group generally contains three aspects:

- Alignment: bird changing direction
- Separation: avoiding collision
- Cohesion: moving towards or joining other birds

The block diagram for the bird flocking mechanism is given in Figure 1. Interestingly, this paper forms the basis for the rest of the papers on flocking. The models presented afterward mainly cover one or more of these 4 aspects namely, alignment, separation, cohesion and speed. For example, the idea of dynamic speed instead of being a constant for all birds is given in [25]. The speed depends on the density of birds in the neighborhood.

Similarly, [1] presents a new model for alignment, i.e., changing the direction of birds. In this paper, different weights such as distance-dependent communication, spatial, heading-angle diameters and communication are used to update the direction of a bird.

Degond and Motsch [26] present yet another model for fish flocking. They propose a different approach for collision and alignment. They present a conservation equation to determine the density of fish and then use it to update alignment.

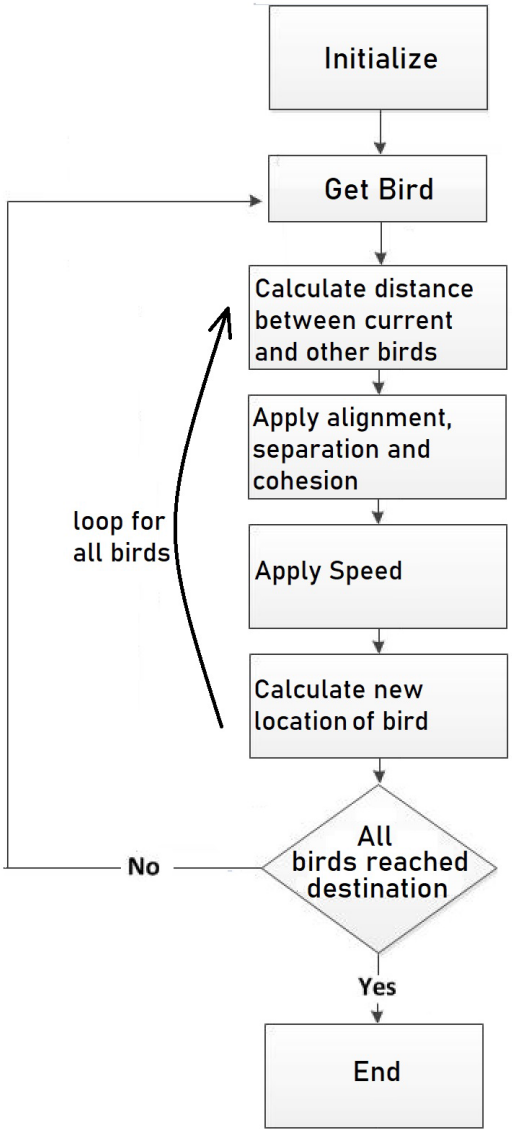

FIGURE 1. Block diagram for bird flocking algorithm.

They also propose a dynamic velocity-update mechanism. Similarly, [27] also discusses and presents a new algorithm for all three aspects but it focuses on individual displacement and movement of a fish or a bird instead of a flock.

In this paper, we use the flocking model given by [24] because of its simplicity and originality. As mentioned earlier every bird in this model always moves at a constant speed. We made two modifications to the flocking algorithm. Firstly, in the original algorithm, the birds are spawned at random locations. This was modified such that the number of birds and their spawning would be according to the handwritten image. Secondly, every bird is assigned a direction randomly. The algorithm was modified to accept a direction parameter given by coordinate (r, c). The direction for all birds was initialized in such a way that they would all point towards the coordinate (r, c).

Figure 2 shows the flocking algorithm proposed by [24] in action with the two modification mention above. The demonstration is given on the image of the handwritten Roman digit '3'. In the figure, the original image and its next three generations are shown for two different input directions.

A. FLOCKING AND SYNTHETIC DATA GENERATION

No significant work has been done that combines flocking with either over-sampling or even pattern recognition. However, related areas involving pattern recognition include evolutionary algorithms [28], particle swarm optimization [29], [30], and biologically inspired models [12], [31]. Having said that, the way how flocking has been used in this paper is novel.

B. CALP FRAMEWORK

In this paper, we propose a data augmentation technique that is based on the bird flocking. To evaluate this technique, we use the cellular automata learning and prediction (CALP) framework. This model is briefly discussed next.

CALP [12] is also a data oversampling and ensemble technique that is based on cellular automata (CA). Cellular automata consists of an array of alive and dead cells, and a set of rules. These rules decides the fate of each cells in the array in subsequent generations resulting in modified arrays that can be considered as augmented data. So, in short, CALP augments data by evolution through cellular automata that is based on a set of rules called *ruleset*. Integration of CA into classical machine and deep learning models is fairly new with limited studies such as [31] and [32].

CALP contains three main modules [12]. Firstly, a trainer that contains an ensemble of n classifiers. Each classifier has a rule r_i associated with it. The training data is considered an array of alive or dead cells, where the black pixels of the images are considered alive. The training data is 'evolved' using the rule r_i to create an evolved version of the data that is then fed to the ith classifier that produces a trained model m_i. Since there are n $(r_1 \ldots r_n)$ rules, they are collectively referred to as a *ruleset*.

Secondly, CALP has a predictor that also has n sub-predictors each of which has the exact same rule r_i associated with them as there were in the training phase. The unseen sample is separately evolved using the rule r_i and passed on to the ith predictor that contains the corresponding trained model m_i. Each predictor determines the class of the unseen data and uses the majority vote to decide the final label.

Thirdly, CALP has a rule set selection module that determines the *ruleset* consisting of specifically n rules, one for each classifier/sub-predictor. The rules determine how the data would evolve. These three modules collectively enable CALP to not only generate or augment data in the best possible way using biologically-inspired process of evolution, but also provides an ensemble framework with training and prediction components.

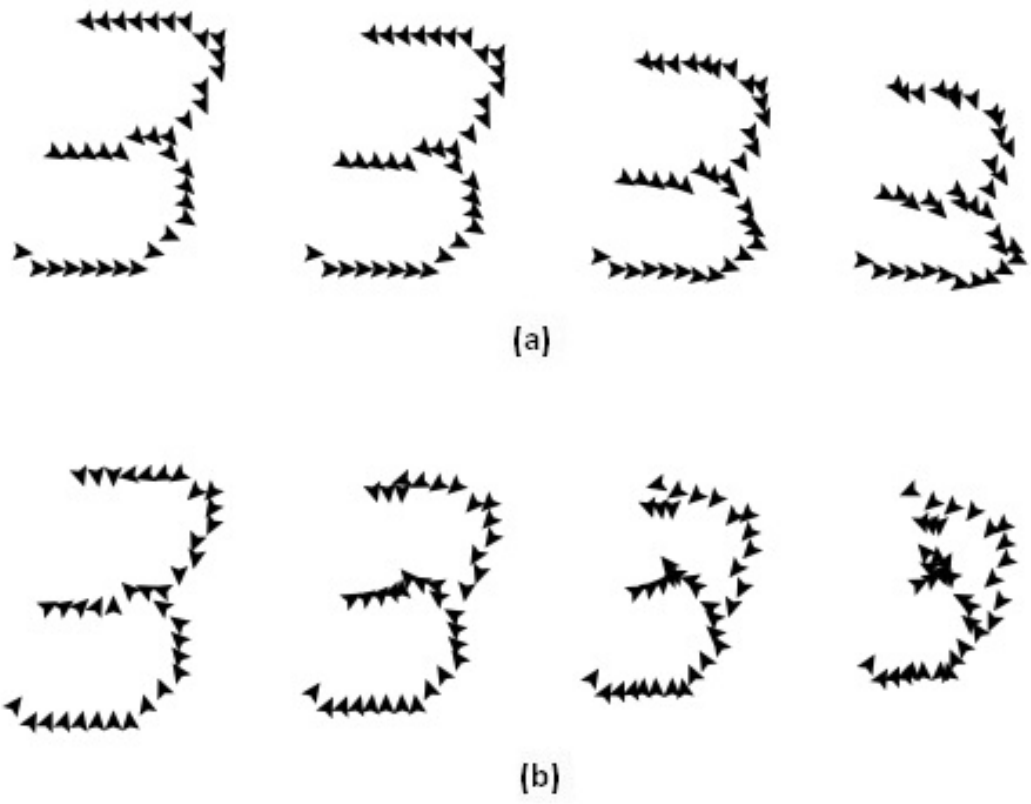

FIGURE 2. Flocking in Action: the original images (20 × 20) are on the left, followed by their next three generations. In (a), the input direction coordinate is (20,20) so the movement of all the birds is towards the bottom right whereas in (b) the movement is towards the center as the direction coordinate is set to (10,10).

In this paper, we augment data using bird flocking and update the CALP model accordingly. This and any other modification made to CALP are discussed in the next section.

III. FLOCKING FOR DATA GENERATION

Handwritten text is not consistent. Even if we ignore the digitization and scanning phase, it is different right from the generation process. So to develop recognition system for handwritten text, machine learning tools and techniques become necessary. It has been shown that variation in handwriting can be modeled using various biologically inspired phenomenon such as [12] and [31]. We therefore use bird flocking patterns and movements to generate different variations of the data.

A. HANDWRITTEN DATA, BIRDS AND FLOCKING

Every handwritten pattern is considered as an arrangement of birds. A bird is a pixel within an image that is 'on' or black. The on-pixel indicates the presence, while the off-pixel indicates the absence of a bird. Flocking in this context is the process of moving all the birds in a handwritten image to a new position thus generating a new version of the image. The direction in which the birds can move or flock is controlled by a parameter (r, c) representing a coordinate. Thus, the 'flock' function takes in three inputs namely, the data set D, r and c, and outputs the next immediate state which is the synthetic or flocked version D^f of the dataset D as shown in Figure 3.

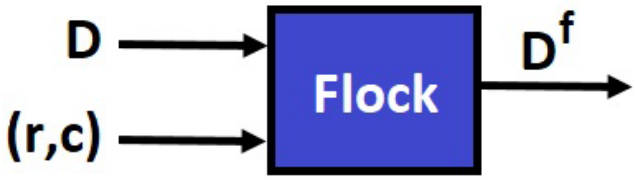

FIGURE 3. The flock function that takes in a data set and the coordinates (r,c) that represent the flock direction and outputs the immediate next state of the birds.

The value of (-1,-1) can also be passed this function. This value forces the 'flock' function to return the original, unaltered image. An image can be flocked in $row \times col$ number of ways where row and col are the dimensions of the binary image.

B. MODIFICATIONS IN CALP FRAMEWORK

Some modifications to the CALP framework have been proposed in this section. Once the flock function is in place, the flock function replaces the evolve function. This modification

FIGURE 4. The evolve function in CALP is replaced with the Flock function.

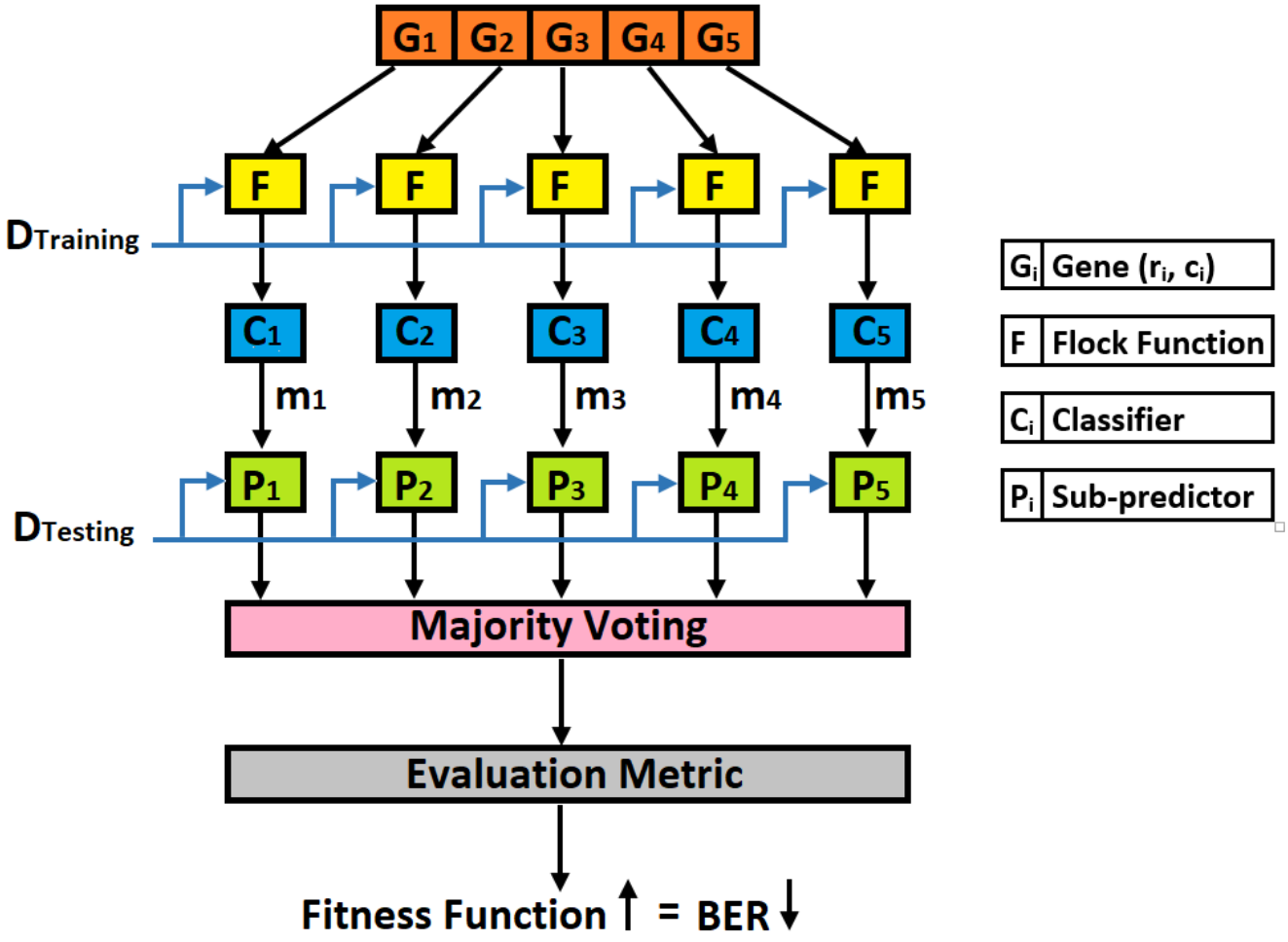

FIGURE 5. The evolve function in CALP is replaced with the Flock function.

is depicted in Figure 4. Instead of a rule r, the flock function takes the coordinates of the flocking direction. The resulting framework is called FLOCK-CALP.

The evolve function takes in a CALP rule whereas the flock function takes in flock direction. This leads to the second modification. CALP has a set of rules $\{r_1, r_2, \ldots r_n\}$ called *ruleset* where r_i specifies how the data is evolved for classifier i in the ensemble. Instead of the *ruleset*, we require a flock direction-set $\delta = \{(r_1, c_1), (r_2, c_2) \ldots (r_n, c_n)\}$. Here (r_i, c_i) specifies the direction in which the data will flock for classifier i in the ensemble. For clarity, if $\delta = \{(r_1, c_1), (r_2, c_2), (r_3, c_3), (r_4, c_5), (r_5, c_5)\}$ then an ensemble of 5 classifiers would be built. The data set D is flocked using parameters (r_1, c_1) and then given as input to the first classifier and so on.

The flock direction-set δ or the *ruleset* used originally in CALP, specifies the configuration of the ensemble in terms of how many classifiers would an ensemble encompass. In CALP, this is calculated by the '*ruleset* selection module'. The *ruleset* is calculated using a systematic approach where all possible combinations of rules are considered, and the ones that contribute towards high classification accuracy are selected. So, thirdly, a new flock direction-set selection algorithm is proposed.

TABLE 1. The derived rule set for each data set using 5 different classifiers.

Dataset	Classifier	Derived δ
Music Clefs	CNN	{(19,13),(13,22),(25,2),(11,20),(1,5)}
	NN	{(17,6),(24,9),(24,11),(18,30),(26,18)}
	LR	{(17,22),(19,20),(18,4),(18,13),(17,20)}
	Zarbi	{(26,8),(19,30),(21,18),(4,27),(1,20)}
	DT	{(3,25),(24,21),(3,5),(1,9),(15,20)}
Roman Digits	CNN	{(10,6),(13,7),(20,5),(18,11),(15,8)}
	NN	{(2,10),(8,2),(19,11),(19,12),(13,14)}
	LR	{(13,7),(13,2),(13,5),(14,5),(15,1)}
	Zarbi	{(9,4),(13,2),(4,11),(6,19),(16,17)}
	DT	{(15,5),(14,5),(19,16),(20,13),(17,1)}
Devanagari Digits	CNN	{(1,1),(1,24),(28,3),(9,25),(1,26)}
	NN	{(19,13),(16,16),(9,18),(28,3),(1,6)}
	LR	{(28,2),(27,7),(27,15),(28,10),(28,3)}
	Zarbi	{(24,11),(23,2),(22,6),(28,10),(28,3)}
	DT	{(22,30),(28,3),(11,8),(28,28),(21,1)}
Bangla Digits	CNN	{(1,25),(9,6),(13,7),(9,5),(17,17)}
	NN	{(19,20),(1,18),(18,17),(20,4),(19,22)}
	LR	{(28,14),(29,9),(30,27),(29,8),(29,14)}
	Zarbi	{(17,14),(4,26),(28,12),(3,16),(24,7)}
	DT	{(17,13),(20,8),(11,29),(4,26),(4,25)}
Music Accidentals	CNN	{(22,14),(11,18),(6,26),(25,17),(15,6)}
	NN	{(26,28),(18,28),(1,13),(24,26),(13,15)}
	LR	{(11,4),(11,26),(11,20),(11,1),(11,12)}
	Zarbi	{(29,12),(20,4),(18,6),(9,27),(21,14)}
	DT	{(1,19),(1,8),(2,28),(4,5),(2,9)}

The direction-set δ is generated using genetic algorithms. A chromosome represents a potential δ. We set the default size of the chromosome to 5 genes and each gene represents a flock direction. Thus the chromosome represent a set of flock directions and one of them will eventually be selected to form δ. The population size is set to 10. Note that the default size of the ensemble will also 5 due to 5 genes in a chromosome, but the value can be changed. There is a different δ for every data set D and classifier C pair.

The block diagram for calculating the fitness function of a chromosome is given in Figure 5. When calculating the fitness, only 5% of the data set is used, which is further split into train (*D_Training*) and test (*D_Test*) set. The training data is first flocked separately using the flock function for each gene in the chromosome. Each flocked version is then fed to its classifier C_i for training, resulting in trained model m_i. Then (*D_Test*) is passed to its sub-predictor P_i, to determine the class of the test samples. The class of all 5 sub-predictor is passed to the majority function that decides the final label for all samples in (*D_Test*). Finally, the performance of the complete ensemble is measured. The performance of the model is obtained in terms of balanced error rate or *BER* using k-fold cross validation. The chromosomes that produce a low *BER* are considered as fit and are selected for the next generation.

These fitter chromosomes go through the usual genetic algorithm operators, crossover and mutation, to generate more samples which are then checked for fitness. This process continues as long as fitter members become part of the populations. When this is not the case, iterations stop and the fittest chromosome is selected as δ.

IV. RESULTS AND DISCUSSION

FLOCK-CALP is a model to which any classifier can be plugged-in. So to test the model we used 5 different classifiers - convolutional neural networks (CNN), neural network (NN), logistic regression (LR), Zarbi [33] and decision trees (DT). The CNN had 2 convolution-pooling layer pairs. The layers had 20 and 50 features maps of size 5×5. The pool size was 2×2 each. MATLAB's patternet was used as NN classifier while MATLAB's fitctree package was used for decision trees. The exact same experimental setup of [12] is replicated - with the same classifiers with the exact same settings. Also, the exact same data sets are used except for the synthetic data set against which we used the music accidentals data set.

Five data sets were used to evaluate the performance of FLOCK-CALP based on Flocking. Each data set is discussed briefly.

1) Music Clef Symbols: Clef symbols are used to indicate the pitch. This data set has 2128 samples. All samples were normalized to 900 features. The class size is 3. The data set was taken from CVC Music symbols database [34].
2) Handwritten Roman Digits: The Roman digits data set is taken from MNIST [35]. This data set has 60000 samples. All samples were normalized to 400 features. The class size is 10.
3) Bangla Handwritten Digits: Bangla data set has 19392 samples. All samples were normalized to 900 features. The class size is 10. The data set was taken from ISI character databases [36] and [37].
4) Devanagari Handwritten Digits: Devanagari data set has 18780 samples. All samples were normalized to 900 features. The class size is 10. The data set was taken from ISI character databases [36] and [37].
5) Music Accidentals Symbols: Music accidentals are notes of a pitch. This data set has 1964 samples. All samples were normalized to 900 features. The class size is 4. The data set was taken from CVC Music symbols database [34].

A. RULE SET

The direction set δ calculated using genetic algorithm is given in table 1. The δ is calculated for all data set-classifier pairs. The δ is a very important parameter for the flock learning and prediction model.

From table 1 it can be seen that all the direction sets are different. This is expected since the working of each classifier and the structure of each handwritten data set are different. However, some interesting similarities exist. In the case of the Devanagari data set, the flock direction (28,3) is selected for all classifiers. The effect of this particular flock direction is that it tends to make the image more compact and dense.

From the table 1 it can also been seen that the direction $(-1, -1)$ is not selected. This particular directional value is used to prevent any flocking and to retain original images. The purpose of these directional values is to allow the possibility of including un-flocked data sets in the training phase

TABLE 2. Comparison of various classifiers with FLOCK-CALP. The recorded values represent the mean and standard deviation of *BER*.

Data set	Classifier	Base classifier	Ensemble based FLP
Music Clefs	CNN	$0.017_{0.02}$	$\mathbf{0.007}_{0.00}$
	NN	$0.042_{0.01}$	$\mathbf{0.024}_{0.00}$
	LR	$0.046_{0.11}$	$\mathbf{0.029}_{0.00}$
	Zarbi	$0.165_{0.01}$	$\mathbf{0.095}_{0.02}$
	DT	$0.123_{0.02}$	$\mathbf{0.045}_{0.00}$
Roman Digits	CNN	$0.054_{0.00}$	$\mathbf{0.042}_{0.00}$
	NN	$0.140_{0.02}$	$\mathbf{0.096}_{0.00}$
	LR	$0.232_{0.04}$	$\mathbf{0.181}_{0.00}$
	Zarbi	$0.339_{0.04}$	$\mathbf{0.300}_{0.02}$
	DT	$0.252_{0.01}$	$\mathbf{0.097}_{0.00}$
Devanagari Digits	CNN	$0.029_{0.02}$	$\mathbf{0.009}_{0.00}$
	NN	$0.138_{0.01}$	$\mathbf{0.092}_{0.00}$
	LR	$0.141_{0.02}$	$\mathbf{0.102}_{0.00}$
	Zarbi	$0.240_{0.00}$	$\mathbf{0.171}_{0.02}$
	DT	$0.340_{0.01}$	$\mathbf{0.099}_{0.00}$
Bangla Digits	CNN	$0.028_{0.00}$	$\mathbf{0.014}_{0.00}$
	NN	$0.204_{0.04}$	$\mathbf{0.103}_{0.00}$
	LR	$0.183_{0.04}$	$\mathbf{0.110}_{0.00}$
	Zarbi	$0.369_{0.00}$	$\mathbf{0.289}_{0.02}$
	DT	$0.362_{0.00}$	$\mathbf{0.131}_{0.00}$
Music Accidentals	CNN	$0.015_{0.00}$	$\mathbf{0.009}_{0.00}$
	NN	$0.080_{0.01}$	$\mathbf{0.061}_{0.00}$
	LR	$0.080_{0.01}$	$\mathbf{0.059}_{0.00}$
	Zarbi	$0.459_{0.01}$	$\mathbf{0.138}_{0.02}$
	DT	$0.270_{0.02}$	$\mathbf{0.095}_{0.00}$

TABLE 3. Mean and standard deviation (shown in subscript form) of *BER* using varying sizes of training data. The best results are in bold.

Data set	Classifier	75% of data set	25% of data set	25% original + over-sampled				CALP on 25% of data set	FLOCK-CALP on 25% of data set
				SMOTE	AdaSyn	SMOTE bor-derline	Safe SMOTE		
Music Clefs	CNN	$0.017_{0.02}$	$0.052_{0.03}$	$0.020_{0.01}$	$0.018_{0.01}$	$0.021_{0.01}$	$0.019_{0.01}$	$\mathbf{0.015}_{0.00}$	$\mathbf{0.015}_{0.00}$
	NN	$0.042_{0.01}$	$0.068_{0.01}$	$0.057_{0.01}$	$0.060_{0.01}$	$0.066_{0.01}$	$0.053_{0.00}$	$0.038_{0.01}$	$\mathbf{0.033}_{0.00}$
	LR	$0.046_{0.11}$	$0.066_{0.01}$	$0.063_{0.01}$	$0.066_{0.01}$	$0.061_{0.01}$	$0.067_{0.00}$	$0.047_{0.01}$	$\mathbf{0.04}_{0.00}$
	Zarbi	$0.165_{0.01}$	$0.166_{0.01}$	$0.156_{0.00}$	$0.257_{0.06}$	$0.147_{0.00}$	$0.174_{0.04}$	$0.113_{0.01}$	$\mathbf{0.108}_{0.00}$
	DT	$0.123_{0.02}$	$0.178_{0.04}$	$0.183_{0.08}$	$0.209_{0.04}$	$0.201_{0.05}$	$0.176_{0.05}$	$0.113_{0.01}$	$\mathbf{0.059}_{0.00}$
Roman Digits	CNN	$0.054_{0.00}$	$0.065_{0.00}$	$0.066_{0.00}$	$0.065_{0.00}$	$0.068_{0.00}$	$0.061_{0.00}$	$0.060_{0.00}$	$\mathbf{0.052}_{0.00}$
	NN	$0.140_{0.02}$	$0.173_{0.16}$	$0.157_{0.01}$	$0.269_{0.13}$	$0.184_{0.01}$	$0.158_{0.00}$	$0.123_{0.01}$	$\mathbf{0.107}_{0.00}$
	LR	$0.232_{0.04}$	$0.229_{0.04}$	$0.251_{0.00}$	$0.276_{0.09}$	$0.250_{0.04}$	$\mathbf{0.158}_{0.01}$	$0.188_{0.04}$	$0.192_{0.00}$
	Zarbi	$0.339_{0.04}$	$0.341_{0.01}$	$0.344_{0.00}$	$0.403_{0.01}$	$0.418_{0.02}$	$0.352_{0.01}$	$\mathbf{0.301}_{0.00}$	$0.312_{0.00}$
	DT	$0.252_{0.01}$	$0.317_{0.00}$	$0.315_{0.00}$	$0.321_{0.01}$	$0.327_{0.01}$	$0.319_{0.00}$	$0.130_{0.00}$	$\mathbf{0.105}_{0.00}$
Deva Digits	CNN	$0.029_{0.02}$	$0.039_{0.01}$	$0.030_{0.00}$	$0.026_{0.00}$	$0.028_{0.00}$	$0.027_{0.00}$	$0.026_{0.00}$	$\mathbf{0.014}_{0.00}$
	NN	$0.138_{0.01}$	$0.218_{0.06}$	$0.225_{0.02}$	$0.226_{0.01}$	$0.318_{0.13}$	$0.214_{0.03}$	$0.107_{0.00}$	$\mathbf{0.102}_{0.00}$
	LR	$0.141_{0.02}$	$0.130_{0.00}$	$0.134_{0.01}$	$0.129_{0.00}$	$0.140_{0.01}$	$0.142_{0.02}$	$0.147_{0.02}$	$\mathbf{0.113}_{0.00}$
	Zarbi	$0.240_{0.00}$	$0.242_{0.00}$	$0.249_{0.00}$	$0.267_{0.00}$	$0.286_{0.01}$	$0.254_{0.00}$	$0.235_{0.00}$	$\mathbf{0.184}_{0.00}$
	DT	$0.340_{0.01}$	$0.416_{0.01}$	$0.417_{0.01}$	$0.435_{0.01}$	$0.452_{0.01}$	$0.419_{0.00}$	$0.168_{0.00}$	$\mathbf{0.110}_{0.00}$
Bangla Digits	CNN	$0.028_{0.00}$	$0.043_{0.00}$	$0.055_{0.02}$	$0.035_{0.00}$	$0.034_{0.00}$	$0.038_{0.00}$	$0.037_{0.00}$	$\mathbf{0.024}_{0.00}$
	NN	$0.204_{0.04}$	$0.247_{0.06}$	$0.259_{0.00}$	$0.287_{0.06}$	$0.299_{0.00}$	$0.327_{0.01}$	$0.128_{0.00}$	$\mathbf{0.113}_{0.00}$
	LR	$0.183_{0.04}$	$0.180_{0.02}$	$0.175_{0.01}$	$0.183_{0.01}$	$0.173_{0.00}$	$0.166_{0.01}$	$0.144_{0.00}$	$\mathbf{0.125}_{0.00}$
	Zarbi	$0.369_{0.00}$	$0.371_{0.01}$	$0.371_{0.00}$	$0.405_{0.01}$	$0.417_{0.01}$	$0.380_{0.00}$	$0.324_{0.01}$	$\mathbf{0.301}_{0.00}$
	DT	$0.362_{0.00}$	$0.456_{0.01}$	$0.459_{0.01}$	$0.472_{0.01}$	$0.452_{0.02}$	$0.465_{0.00}$	$0.208_{0.00}$	$\mathbf{0.143}_{0.00}$
Music Acc.	CNN	$\mathbf{0.015}_{0.00}$	$0.034_{0.01}$	$0.015_{0.00}$	$0.011_{0.00}$	$0.012_{0.00}$	$0.013_{0.00}$	$0.030_{0.01}$	$0.026_{0.00}$
	NN	$0.080_{0.01}$	$0.169_{0.04}$	$0.125_{0.01}$	$0.124_{0.01}$	$0.118_{0.00}$	$0.124_{0.00}$	$\mathbf{0.068}_{0.01}$	$0.072_{0.00}$
	LR	$0.080_{0.01}$	$0.109_{0.04}$	$0.115_{0.00}$	$0.114_{0.00}$	$0.120_{0.01}$	$0.120_{0.00}$	$0.085_{0.00}$	$\mathbf{0.072}_{0.00}$
	Zarbi	$0.459_{0.01}$	$0.458_{0.04}$	$0.453_{0.01}$	$0.505_{0.02}$	$0.549_{0.01}$	$0.447_{0.00}$	$0.304_{0.02}$	$\mathbf{0.151}_{0.00}$
	DT	$0.270_{0.01}$	$0.347_{0.04}$	$0.379_{0.00}$	$0.378_{0.00}$	$0.395_{0.03}$	$0.341_{0.01}$	$0.167_{0.01}$	$\mathbf{0.108}_{0.00}$

since the flock function with direction value $(-1, -1)$ does not modify or 'flock' the data set. This also illustrates the significance of the 'flocking' process, i.e., training on flocked data produces better results than the one that is not.

B. RESULTS OF FLOCK-CALP USING RULE SET δ

Table 2 compares the performance of some well-known classifiers with FLOCK-CALP. As mentioned earlier, five classifiers and five handwritten data sets are used. The performance

TABLE 4. Performance of FLOCK-CALP compared with other ensemble techniques. The performance is recorded in terms of the mean and standard deviation of the balanced error rate.

Data set	Adaboost	Bagging	Random forest	CALP	FLOCK-CALP
Music Clefs	$0.205_{0.01}$	$0.038_{0.01}$	$\mathbf{0.034}_{0.00}$	$0.055_{0.02}$	$0.045_{0.00}$
Roman Digits	$0.733_{0.01}$	$0.098_{0.01}$	$0.098_{0.01}$	$0.106_{0.00}$	$\mathbf{0.097}_{0.00}$
Deva Digits	$0.739_{0.00}$	$0.139_{0.00}$	$0.130_{0.00}$	$0.126_{0.00}$	$\mathbf{0.099}_{0.00}$
Bangla Digits	$0.789_{0.01}$	$0.160_{0.01}$	$0.160_{0.00}$	$0.147_{0.01}$	$\mathbf{0.131}_{0.00}$
Music Accidentals	$0.259_{0.05}$	$0.154_{0.07}$	$0.103_{0.2}$	$0.1_{0.00}$	$\mathbf{0.095}_{0.00}$

is calculated using 4-folds cross validation method and recorded in terms of *BER*. From the table 2 it can be seen that FLOCK-CALP has significantly improved the classification accuracy in *all* cases. This clearly illustrates that flocking the data sets in different directions, and then using them in an ensemble enables the ensemble to train on various alterations of the data which in turn leads to significantly better accuracy.

C. PERFORMANCE OF FLOCK-CALP ON SMALLER TRAINING DATA

FLOCK-CALP flocks data in different directions to generate more data. Then this data is used in parallel, in the form of an ensemble for training. This raises an interesting research direction: can FLOCK-CALP perform well even when trained using fewer training examples? Therefore, in this subsection we compare different classifiers trained on smaller and larger training data with FLOCK-CALP trained on smaller training data. We also use oversampling techniques to generate more data and compare the performance of base classifiers trained on this data with FLOCK-CALP. Table 3 compares the performance of various classifiers with FLOCK-CALP on different sizes of training data. Again, the performance is calculated using 4-folds cross validation method and recorded in terms of *BER*.

In table 3, column 3 and 4 show the *BER* of single classifiers trained on 75% and 25% of the data set respectively. Column 5-8 specify the *BER* when 25% of the original data plus the same size of data generated using other oversampling techniques was used for training. Column 9 and 10 records the performance of CALP and our proposed FLOCK-CALP model trained on 25% of the data set. CALP is considered here because our model is very similar to it. One major difference between the two is that CALP uses cellular automata to evolve data whereas we use the notion of flocking.

From the table, it can be seen that FLOCK-CALP trained on only 25% of the data set performs better than a single classifier trained on 25%, 75%, and oversampled data sets in almost all cases (21 out of 25). FLOCK-CALP also performs better than CALP on smaller training data. This result is significant, especially in the case of classifiers such as convolutional neural networks that require huge amounts of training data. Using FLOCK-CALP such a classifier can be made to perform better on fewer training examples. This can also be useful in situations where data generation is expensive.

D. PERFORMANCE OF OTHER ENSEMBLE METHODS WITH FLOCK-CALP

In this subsection, the FLOCK-CALP which is also an ensemble is compared to other state-of-the-art ensembles such as CALP, bagging, Adaboost and random forest. Bagging or bootstrap aggregation involves generating a subset of the dataset for training by random sampling of data with replacement. Adaboost employs boosting where the mis-classified samples in the training data are again selected in the next iterations. Random forest is also a bagging technique but use the decision trees as base classifiers.

The purpose of comparing ensembles is to illustrate that the proposed flocking-based, augmentation-driven ensemble also performs better than other ensembles. The classifier used within all the ensemble techniques is decision trees. Table 4 shows the classification performance of all data sets in terms of mean and standard deviation of the *BER* using various ensemble techniques such as CALP, bagging, random forest, etc.

From the table, it can be seen that besides the music clefs data set, FLP performs better than all state-of-the-art ensemble techniques on all other data sets.

V. CONCLUSION AND FUTURE DIRECTION

In this paper, we propose a nature-inspired data augmentation technique. First, we consider that in any image sample taken from a handwritten data set the black pixels represent an assembly of birds and it shows their initial position. Then using a well-defined flocking mechanism, the new positions for these birds can be calculated. Each snapshot of the new position can be considered as a new augmented version of the original image. These new versions of the data set are referred to as *flocked* versions. We also show how the fascinating flocking movements of birds to generate more data can be fused with classification algorithms to develop another pattern recognition technique. We demonstrate our proposed method using 5 handwritten data sets and 5 different classifiers. The following conclusions are drawn from the experiments:

- FLOCK-CALP based on Flocking performs better than the conventional classifiers.
- FLOCK-CALP based on Flocking requires a small-sized training data to perform better than a single classifier trained on larger data,

- FLOCK-CALP based on Flocking also performs better than the other ensemble methods such as CALP, bagging and random forest.

For future work it would be worth looking into other direction set δ generation methods besides genetic algorithms.

REFERENCES

[1] S.-Y. Ha, E. Jeong, and M.-J. Kang, "Emergent behaviour of a generalized Viscek-type flocking model," *Nonlinearity*, vol. 23, no. 12, pp. 3139–3156, Dec. 2010.
[2] A. Ilachinski, "Irreducible semi-autonomous adaptive combat (ISAAC): An artificial-life approach to land warfare," Center Naval Analyses Alexandria VA, Arlington, VA, USA, Tech. Rep., 1997.
[3] N. E. Leonard, D. A. Paley, F. Lekien, R. Sepulchre, D. M. Fratantoni, and R. E. Davis, "Collective motion, sensor networks, and ocean sampling," *Proc. IEEE*, vol. 95, no. 1, pp. 48–74, Jan. 2007.
[4] L. Perea, G. Gómez, and P. Elosegui, "Extension of the Cucker–Smale control law to space flight formations," *J. Guid., Control, Dyn.*, vol. 32, no. 2, pp. 527–537, Mar. 2009.
[5] D. A. Paley, N. E. Leonard, R. Sepulchre, D. Grunbaum, and J. K. Parrish, "Oscillator models and collective motion," *IEEE Control Syst.*, vol. 27, no. 4, pp. 89–105, Aug. 2007.
[6] K. E. Joyce, S. Hayasaka, and P. J. Laurienti, "A genetic algorithm for controlling an agent-based model of the functional human brain," *Biomed. Sci. Instrum.*, vol. 48, p. 210, Sep. 2012.
[7] S. Ahn, H.-O. Bae, S.-Y. Ha, Y. Kim, and H. Lim, "Application of flocking mechanism to the modeling of stochastic volatility," *Math. Models Methods Appl. Sci.*, vol. 23, no. 9, pp. 1603–1628, Aug. 2013.
[8] M. Bayer, M.-A. Kaufhold, and C. Reuter, "A survey on data augmentation for text classification," *ACM Comput. Surv.*, vol. 55, no. 7, pp. 1–39, Jul. 2023.
[9] K. Maharana, S. Mondal, and B. Nemade, "A review: Data pre-processing and data augmentation techniques," *Global Transitions Proc.*, vol. 3, no. 1, pp. 91–99, Jun. 2022.
[10] F. Charte, A. J. Rivera, M. J. del Jesus, and F. Herrera, "MLSMOTE: Approaching imbalanced multilabel learning through synthetic instance generation," *Knowl.-Based Syst.*, vol. 89, pp. 385–397, Nov. 2015.
[11] A.-A. Semenoglou, E. Spiliotis, and V. Assimakopoulos, "Data augmentation for univariate time series forecasting with neural networks," *Pattern Recognit.*, vol. 134, Feb. 2023, Art. no. 109132.
[12] A. Wali and M. Saeed, "Biologically inspired cellular automata learning and prediction model for handwritten pattern recognition," *Biologically Inspired Cognit. Archit.*, vol. 24, pp. 77–86, Apr. 2018.
[13] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, Jun. 2002.
[14] H. He, Y. Bai, E. A. Garcia, and S. Li, "ADASYN: Adaptive synthetic sampling approach for imbalanced learning," in *Proc. IEEE Int. Joint Conf. Neural Netw. (IEEE World Congr. Comput. Intell.)*, Jun. 2008, pp. 1322–1328.
[15] C. Z. Mooney, R. D. Duval, and R. Duvall, *Bootstrapping: A Nonparametric Approach to Statistical Inference*, no. 94–95. Newbury Park, CA, USA: Sage, 1993.
[16] P. Zhang, T. D. Bui, and C. Y. Suen, "A novel cascade ensemble classifier system with a high recognition performance on handwritten digits," *Pattern Recognit.*, vol. 40, no. 12, pp. 3415–3429, Dec. 2007.
[17] E. Tasci and A. Ugur, "A novel pattern recognition framework based on ensemble of handcrafted features on images," *Multimedia Tools Appl.*, vol. 81, no. 21, pp. 30195–30218, Sep. 2022.
[18] C. Cabral, M. Silveira, and P. Figueiredo, "Decoding visual brain states from fMRI using an ensemble of classifiers," *Pattern Recognit.*, vol. 45, no. 6, pp. 2064–2074, Jun. 2012.
[19] Q. Zhou, X. Wu, S. Zhang, B. Kang, Z. Ge, and L. J. Latecki, "Contextual ensemble network for semantic segmentation," *Pattern Recognit.*, vol. 122, Feb. 2022, Art. no. 108290.
[20] U. E. Laila, K. Mahboob, A. W. Khan, F. Khan, and W. Taekeun, "An ensemble approach to predict early-stage diabetes risk using machine learning: An empirical study," *Sensors*, vol. 22, no. 14, p. 5247, Jul. 2022.
[21] H. Li, F. Shen, C. Shen, Y. Yang, and Y. Gao, "Face recognition using linear representation ensembles," *Pattern Recognit.*, vol. 59, pp. 72–87, Nov. 2016.
[22] N. Saeed, R. Al Majzoub, I. Sobirov, and M. Yaqub, "An ensemble approach for patient prognosis of head and neck tumor using multimodal data," in *3D Head and Neck Tumor Segmentation in PET/CT Challenge*. Strasbourg, France: Springer, Sep. 2021, pp. 278–286.
[23] J. Sun, P. Zhao, and S. Sun, "A new secondary decomposition-reconstruction-ensemble approach for crude oil price forecasting," *Resour. Policy*, vol. 77, Aug. 2022, Art. no. 102762.
[24] U. Wilensky, "Netlogo flocking model," Northwestern Univ., Evanston, IL, USA, Tech. Rep., 1998. [Online]. Available: https://ccl.northwestern.edu/netlogo/models/Flocking
[25] R. Duan, M. Fornasier, and G. Toscani, "A kinetic flocking model with diffusion," *Commun. Math. Phys.*, vol. 300, no. 1, pp. 95–145, Nov. 2010.
[26] P. Degond and S. Motsch, "Continuum limit of self-driven particles with orientation interaction," *Math. Models Methods Appl. Sci.*, vol. 18, no. 1, pp. 1193–1215, Aug. 2008.
[27] P. Degond and S. Motsch, "Large scale dynamics of the persistent turning Walker model of fish behavior," *J. Stat. Phys.*, vol. 131, no. 6, pp. 989–1021, Jun. 2008.
[28] L. Yuelong, L. Jinping, and M. Li, "Character recognition based on hierarchical RBF neural networks," in *Proc. 6th Int. Conf. Intell. Syst. Design Appl.*, vol. 1, Oct. 2006, pp. 127–132.
[29] M. Keyarsalan, G. A. Montazer, and K. Kazemi, "Font-based Persian character recognition using simplified fuzzy ARTMAP neural network improved by fuzzy sets and particle swarm optimization," in *Proc. IEEE Congr. Evol. Comput.*, May 2009, pp. 3003–3009.
[30] N. O. S. Ba-Karait and S. M. Shamsuddin, "Handwritten digits recognition using particle swarm optimization," in *Proc. 2nd Asia Int. Conf. Modeling Simulation (AMS)*, May 2008, pp. 615–619.
[31] A. Wali and M. Saeed, "M-CALP—Yet another way of generating handwritten data through evolution for pattern recognition," *Biosystems*, vol. 175, pp. 24–29, Jan. 2019.
[32] A. Wali, "CA-NN: A cellular automata neural network for handwritten pattern recognition," *Natural Comput.*, pp. 1–8, Dec. 2022.
[33] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander, "Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring," *Science*, vol. 286, no. 5439, pp. 531–537, Oct. 1999.
[34] A. Fornés, J. Lladós, and G. Sánchez, "Old handwritten musical symbol classification by a dynamic time warping based method," in *Proc. Int. Workshop Graph. Recognit.* Cham, Switzerland: Springer, 2007, pp. 51–60.
[35] Y. LeCun, C. Cortes, and C. J. Burges, "The MNIST database of handwritten digits," 1998. [Online]. Available: http://yann.lecun.com/exdb/mnist/
[36] U. Bhattacharya and B. B. Chaudhuri, "Databases for research on recognition of handwritten characters of Indian scripts," in *Proc. 8th Int. Conf. Document Anal. Recognit. (ICDAR)*, 2005, pp. 789–793.
[37] U. Bhattacharya and B. B. Chaudhuri, "Handwritten numeral databases of Indian scripts and multistage recognition of mixed numerals," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 3, pp. 444–457, Mar. 2009.

YIHAN XU is currently with Desheng Technology Company Ltd., Guangzhou, China. Her research interests include automation, AI, and ML.

AAMIR WALI received the Ph.D. degree in computer science from the FAST-National University of Computer and Emerging Sciences. He has been teaching with the Department of Computer Science, FAST-National University of Computer and Emerging Sciences, since 2004. His research interests include font development, writing systems, machine learning, image processing, human–computer interaction, and virtual/augmented reality.