CHAPTER 4

FUZZY MONARCH BUTTERFLY OPTIMIZATION ALGORITHM BASED FEATURE SELECTION AND FUZZY CONVOLUTION BIDIRECTIONAL-LSTM CLASSIFIER

4.1 MONARCH BUTTERFLY OPTIMIZATION ALGORITHM (MBOA)

Metaheuristic algorithm MBO mimics the flight of butterflies. Every monarch butterfly individual in MBO is positioned and optimised in just two areas: the northern United States America (USA) (Land 1), southern Canada, and Mexico (Land 2). Accordingly, two ways are used to update the monarch butterflies' locations. The migration ratio can be used to change the migration operator, which initially creates the offspring's (position updating). Later, a butterfly adjusting operator will fine-tune the placements of other butterflies.

In other words, the migration operators and butterfly adjustment operators of the MBO algorithm fundamentally define the searching directions of certain monarch butterfly-species. Furthermore, the migration operators and the butterfly adjusting operators may be used together. The MBO approach may therefore define a balance between intensification and diversity, making it particularly suited for concurrent processing and a significant concept in the meta-heuristics domain (Wang et al. 2019, Alweshah et al. 2020).

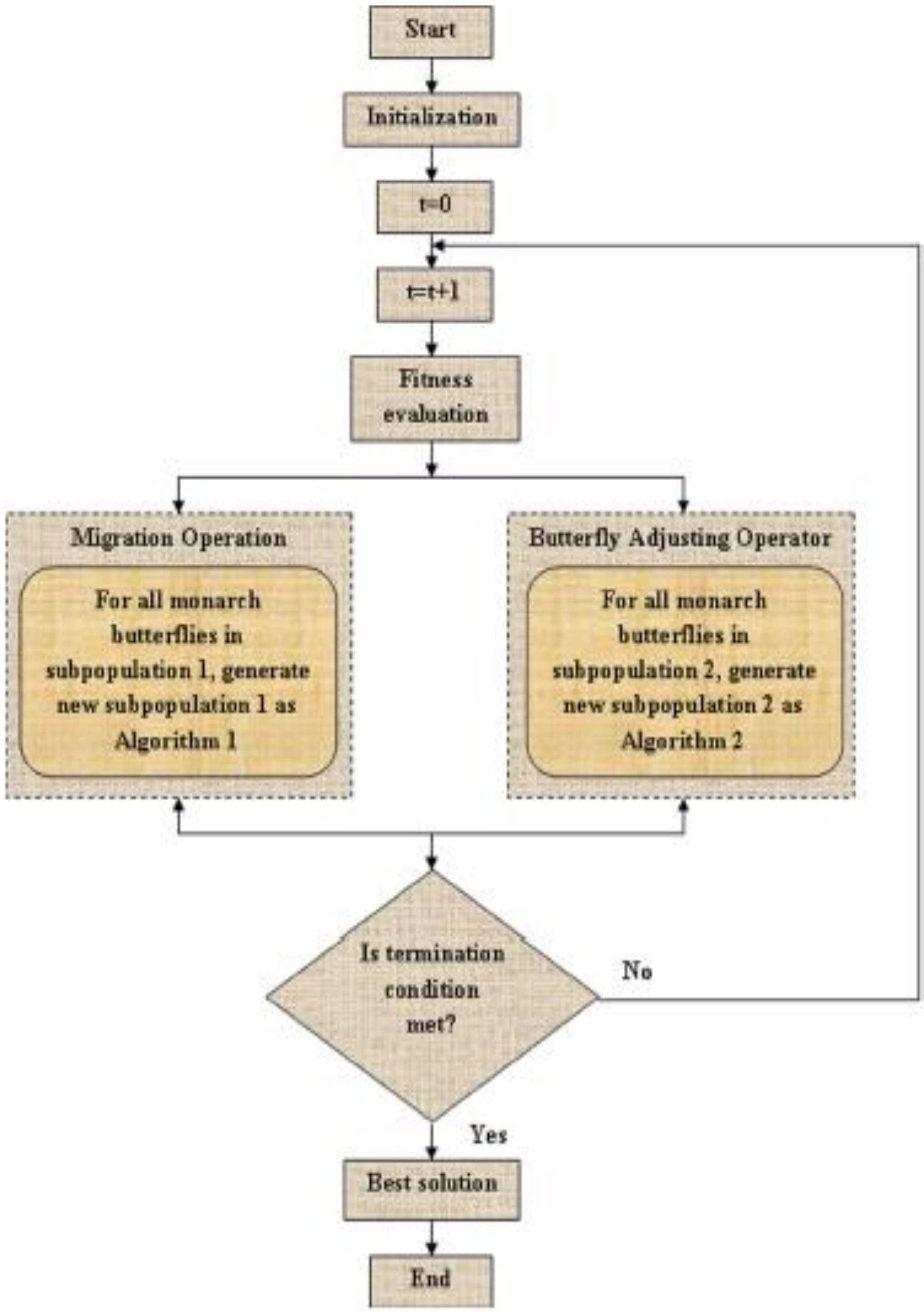

Figure 4.1. Flowchart for Monarch Butterfly Optimization Technique

The guidelines listed below serve as a summary of the monarch butterflies migratory behaviour (See figure 4.1).

1. Only Lands 1 or 2 are the only locations where monarch butterflies can be found. It suggests that the monarch butterfly population is entirely contained in Lands 1 and 2.

2. Each baby monarch butterfly is created using a migration operator from a monarch butterfly that is already present in Land 1 or Land 2.

3. An elderly monarch butterfly will disperse its young in order to keep the population unaltered. This is accomplished using MBO approach where parents are replaced with fit offspring. On the other hand, if newly generated offspring does not exhibit comparably greater fitness than its parent, it may be removed. In this case, the parent is kept and not thrown away.

4. Monarch butterflies with best fitness move to next generations; no operator can change them. This is done to make sure that as the number of monarch butterfly generations rises, there will be a decline in the population effectiveness or quality.

The goal of MBO is to obtain the new, probably superior solutions in place of the comparatively subpar ones and keep them for the next generation. Since just the migration must be adjusted and the operators must be changed, MBO installation is relatively straightforward. Additionally, MBO is appropriate for parallel processing due to the simultaneous execution of the migration and adjustment operators. As a result, the algorithm is able to balance intensity and diversification. In most cases, MBO has a huge ability to get global optimum, but it is usually capable of repeatedly pushing into many local optimums. The levy flight can totally identify the search procedure during the adjusting operator.

4.2 PROPOSED METHODOLOGY

Investigating the voice signals, extracting characteristics, and attempting to map these elements onto the answer are the goals of this effort (three categories of PD and healthy individuals). The disease detection model, which is shown in Figure 4.2, contains several stages, from voice recording to the final result. The initial phases include feature extraction, dimensionality reduction, selection, and classification. The training and testing phases make up the categorization process. The classifier performance metrics show whether or not the detection system was successful.

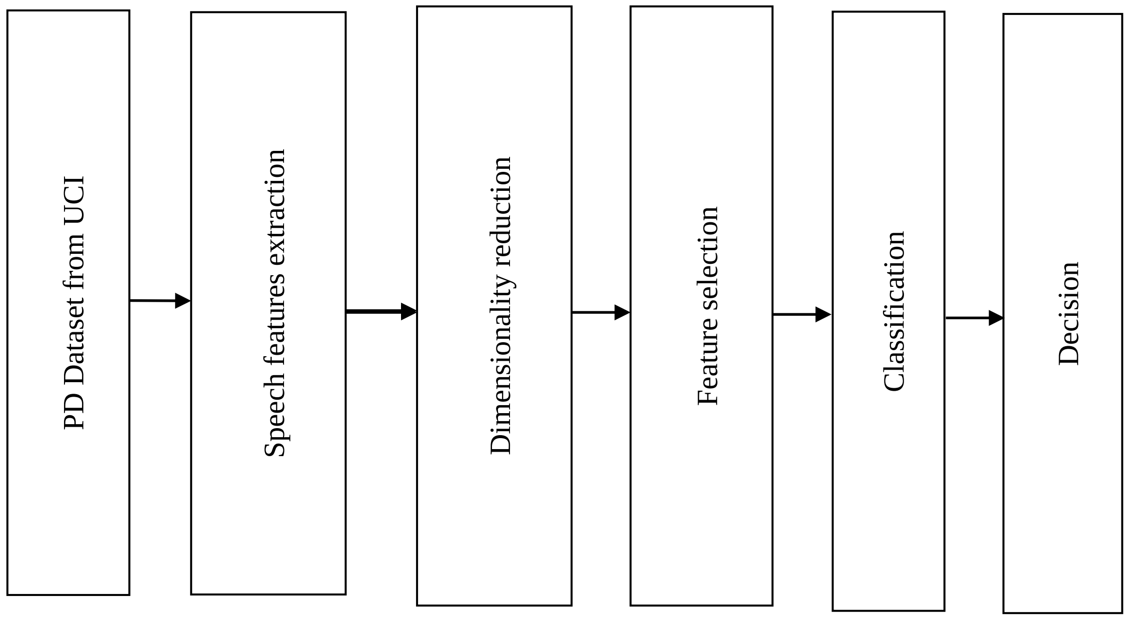

Figure 4.2. Steps of the Proposed Parkinson Detection Model

In this chapter, novel Feature Selection and classification system is presented for the classification of Parkinson's disease. The proposed technique consists of four key steps: Extracting Vocal (Speech) features from the dataset based on speech types; using KPCA to reduce dimensionality; executing FS with FMBOA; using FCBi-LSTM classifiers on subsets; and finally evaluating these classification results in terms of MCC, F-Measure, and Accuracy. To differentiate between patients with PD and healthy individuals, the FCBi-LSTM classifier is made to mix different voice feature types at the feature level. In Figure 4.3, the suggested system flowchart is displayed.

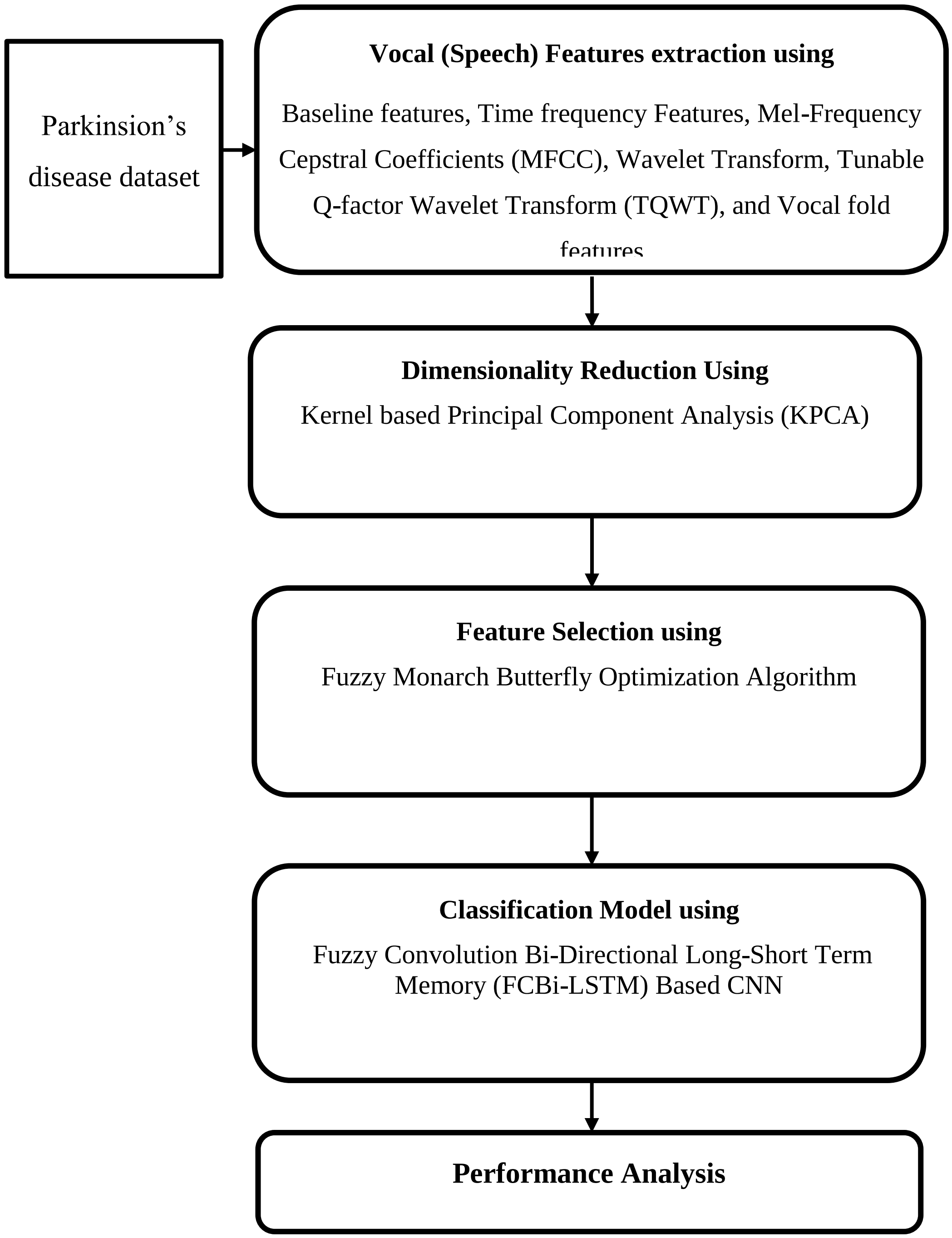

Figure 4.3. Work of Flow of the Proposed Method

For PD diagnosis, a Parkinson dataset was developed that comprises the characteristics extracted from speech samples. The UCI machine-learning repository provided the data for this work, which was only recently examined (Sakar et al. 2019). Features that have been obtained from the dataset include Vocal fold properties, MFCC, TQWT, baseline features, and time frequency features. An approach that is widely used to reduce dimensionality is KPCA. KPCA is given a smaller dimensionality linear subspace, while the new sound recording characteristics of PD receive the largest variance in the overall sound recording feature space. Feature selection uses a dimension-reduced feature vector obtained using FMBOA.

4.2.1 Feature Selection using FMBOA

An innovative wrapper-based FS algorithm (FMBOA) is presented for feature set selections. The features chosen out of every sample (m indicates the number of voice samples) are now passed onto a classifier. Every classifier estimates its individual class label, and the evaluation metric makes the ultimate decision. Feature weights are assigned to the actual features which indicate significances to classifications and features having maximum weights are selected. The MBO algorithm depends on the migratory behavior. A subset of features with relevance from data is graded based on their fitness and significance. The FMBOA method has been shown to produce better classification accuracy results when used generally without alteration, illustrating the fact that this algorithm makes a trade-off between global and local search. In this work, MBOA global search attribute algorithm was modified for generating more accurate results and increase the algorithm effectiveness in obtaining the correct features at the outset and before switching to local search. The individual butterflies comprehend the characteristics and communicate with one another locally to share knowledge throughout the swarm, demonstrating the system ability for evolution (Alweshah 2020). It is done using two procedures, the Butterfly operator adjustment and the Migration operator.

Migration operator: Equation provides an explanation of the Butterfly migration technique. (4.1) (Wang et al. 2015),

$$\mathrm{X}_{\mathrm{i,k}}^{\mathrm{t+1}} = \mathrm{X}_{\mathrm{r_1,k}}^{\mathrm{t}} \tag{4.1}$$

where $\mathrm{X}_{\mathrm{i,k}}^{\mathrm{t+1}}$specifies the $\mathrm{k^{th}}$ elements of $\mathrm{X_i}$ (features of PD) at generation t + 1, defining the location of the butterfly i and $\mathrm{X}_{\mathrm{r_1,k}}^{\mathrm{t}}$represents the $\mathrm{k^{th}}$ elements from newly chosen attributes from PD. Here, r denotes an arbitrary number specified using the Equation (4.2) below,

$$r = rand * peri \tag{4.2}$$

where, peri refers to the time duration for migration. In another scenario, if r> p, and k^{th} elements of the location of the new features are then computed using the Equation (4.3),

$$X_{i,k}^{t+1} = X_{r_2,k}^{t} \tag{4.3}$$

where $X_{r_2,k}^{t+1}$ specifies the k^{th} elements of X_{r_2} (PD attributes) at generations t in r_2, p represents the ratio between the monarch and attributes (Feng et al. 2016). An outline of a butterfly migration operator is provided in Algorithm 4.1.

ALGORITHM 4.1 PESUDOCODE FOR FMBOA MIGRATION OPERATOR

1. Begin
2. for i=1 toNP1 (for each butterfly in subpopulation no.1 (samples))
3. for $k = 1$ toD (for each butterfly in i^{th} monarch butterfly(features)) do
4. For best feature selection, compute r using Equation (4.2), then compute rand using the uniform distribution function.
5. if $r \leq p$ select monarch butterflies (features) in sub-populations (samples) 1 randomly findr_1
6. Create k^{th} elements of $X_i$using Equation (4.1)
7. else
8. Select monarch butterflies (features) in sub-populations (samples) 2 randomly r_2
9. Create k^{th} elements of X_i by Equation (4.3)
10. end if
11. end for k
12. end for i
13. end

Butterfly operator adjustment: By measuring the p value ratio between the migration directions between feature 1 and feature 2, this method achieves equilibrium. A large value of p indicates that the selection of butterflies from feature 1 exhibits superior fitness than those from feature 2, and vice versa. When the random number created is less than or equal to the p value, the location of the butterflies is altered. The following Equation displays the changed Butterfly location using Equation (4.4),

$$X_{j,k}^{t+1} = X_{best,k}^{t} \tag{4.4}$$

where $X_{j,k}^{t+1}$stands for the k^{th} elements of X_j at t + 1+- generation, representing the butterfly position j, and $X_{best,k}^{t}$specifies the k^{th} components of X_{best} in both features-1 and features-2 at the current generation t. Consequently, the following Equation (4.5) updates it if rand> p.

$$X_{j,k}^{t+1} = X_{r_3,k}^{t} \tag{4.5}$$

Next, if the rand is more than fitness, the below Equation (4.6) will revise the current position,

$$X_{j,k}^{t+1} = X_{j,k}^{t+1} + \alpha * (dx_k - 0.5) * g_w \tag{4.6}$$

Where the fitness shows the change factor for the butterfly and the dx refers to the walking steps for the j butterfly(features) decided by flying the Lévy as given by Equation (4.7),

$$dx = \mathrm{Levy}\left(X_j^t\right) \tag{4.7}$$

And α in Equation (4.7) is the weighted variables computed with Equation (4.8),

$$\alpha = S_{max}/t^2 \tag{4.8}$$

And g_w in Equation (4.6) refers to a Gaussian weighted variable computed with Equation

$$g_w = \mathrm{gaussmf}(X, \mathrm{params}) = f(X, \sigma, c) = e^{\frac{-(X-c)^2}{2\sigma^2}} \tag{4.9}$$

where *σ indicates* the features standard deviations and *c* refers to the means of the features. S_{max}indicatesthe maximum walking butterfly lengths in single-step, and t denotes the current generation. The definition of the butterfly-adjusting system is depicted in algorithm 4.2.

ALGORITM 4.2: PESUDOCODE FOR ADJUSTING OPERATOR

1. Begin
2. for $\mathrm{j} = 1$ to NP2 (for butterflies in sub-populations no.2 (samples)) do
3. Find walking steps (dx)for j butterflies(features) using equation (4.7)
4. Obtain weighted variables (α)& Gaussian weighted variables (g_w) for j butterflies (features) using equations (4.8,4.9)
5. for k=1 toD (for each butterfly in ith monarch butterflies(features)) do
6. Calculate r using equation (4.2) for optimal feature choices and generated randomly with uniform distributions
7. if $\mathrm{rand} \leq \mathrm{p}$ then retrieve the kth elements of the $X_{j,k}^{t+1}$using equation (4.4)
8. else select a monarch butterflies (features) FMBOA in sub-population (samples) 2 in random say r_3
9. Generate kth elements of $X_i$using equation (4.5)
10. if rand>fitness then
11. Select new feature position of PD using the equation (4.6)
12. end if
13. end if
14. end for k
15. end for i & end

The common behavior of the MBO algorithm is specified in pseudocode 4.3 once the behavior of the butterflies in pseudocode 4.1 and pseudocode 4.2 (Shi et al. 2015) are studied.

ALGORITHM 4.3. PSEUDOCODE OF FMBOA

1. Begin
2. While $\left(\mathrm{G} < \mathrm{Max}_{\mathrm{gen}}\right)$
3. Create sub-population no.1 for butterflies (features) with NP1optimal features
4. sub-population no.2 for each butterfly (features) with NP2 on the rest

5. Generate offspring solutions for sub-populations 1
6. Generate offspring solutions for sub-populations 2
7. Choose new feature positions of PD using Equation (4.6)
8. end while
9. Obtain global best features from classified PD instances
10. end

4.2.2 PD classification using FCBi-LSTM classifier

The FCBi-LSTM classifier is used in this work to accomplish PD classification. The fuzzy membership function is used to compute the weights in the FCBi-LSTM classifier. For stepwise Parkinson classification, FCBi-LSTM classifiers use PD features and hidden statuses. They also retain contextual data in internal memory states to determine correlations between features of PD.

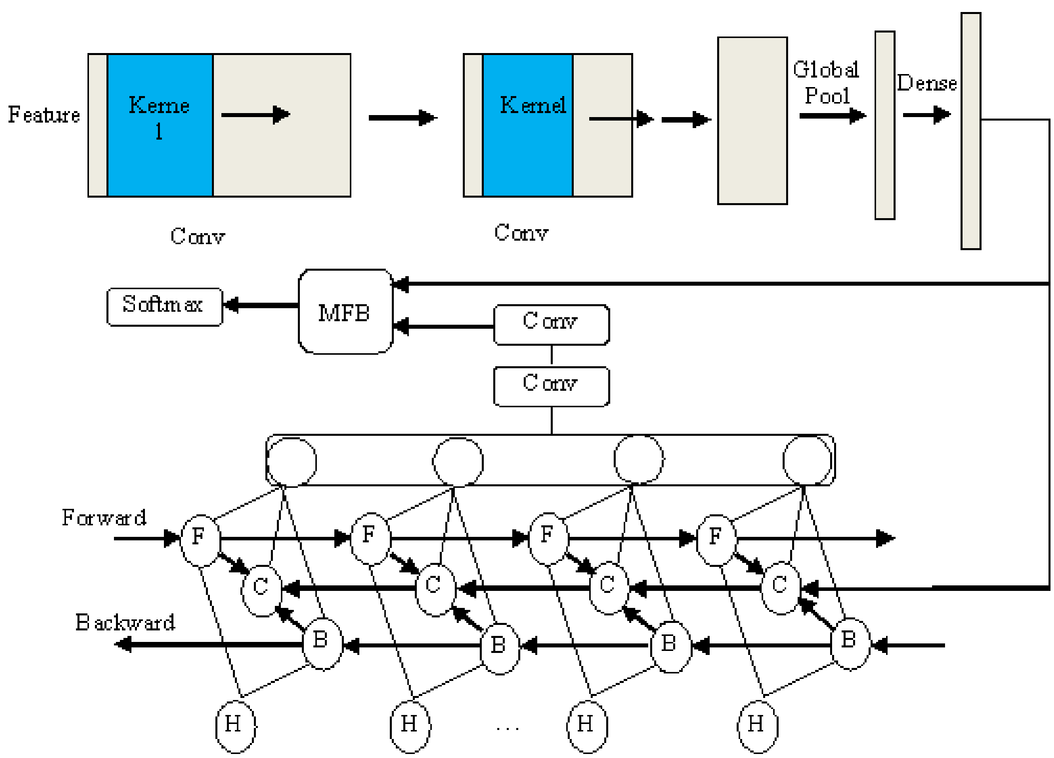

Figure 4.4. Process Chart of the Proposed FCBI-LSTM Classifier

The FCBi-LSTM network structure is depicted in Figure 4.8. CNN are employed in the FCBi-LSTM model to analyze the features selected from the PD dataset. According to Shi et al. (2015), CNN is a DLT used to discover a solution to classification difficulties. Convolution and max pooling layers make up the network. Convolution is done, then pooling, and the end output is passed to the next Convolution layer, and so on. The CNN applies partial filters for convolution based on the information stored in the biological visual cells' local perception. The inner product is produced by the interaction between the regional sub-matrix of the input terms and the local filters. The filters produce many output matrices from the Convolution layer using Equation (4.10), the enhanced representation of the potential features from the PD database. The size of each resulting matrix is determined by (N_{m+1}), and the process is finished.

$$x_i^{l,j} = f\left(b_j + \sum_{a=1}^{m}\sum_{b=1}^{n} fw_{ab}^{j} x_{i+a}^{l-1,j+b}\right) \tag{4.10}$$

A convolution layer is included between x_i^(l,j),l, where j is the numbers of the matching outcome matrix for PD classifications and I is the values of the ith convolution outcome matrix. The left and right indices have values ranging from 0 to N, where N is the numbers of convolutions in the outcome matrix and f is the sigmoid functions. The CNN pooling layer reduces the matrix dimension while maintaining the intrinsic correlations between the best features. The outcome of the convolution layers become the input of the average pooling layers, which in turn becomes the input of the following layer. The process is described by Equation (4.11).

$$x_i^{l,j} = \frac{1}{a.b} \times \sum_{i=1}^{a}\sum_{j=1}^{b} x_{i,j} \tag{4.11}$$

Where $x_i^{l,j}$specifies the outcome term for regional pairs once pooling is done. The finalized Convolution layers’ output is used as intermediate variables in the Bi-LSTM process (Shi et al. 2015). The Bi-LSTM is an RNN variation. Bi-LSTM may make use of the data, which is necessary for the analysis of PD samples. The selected feature vector, h0, is combined with the 2nd PD sample features' vectors by Bi-LSTM to get the new vectors, h1. The next PD sample feature vector is then combined with h1 to form h2, and so on, until hN is created. The previous condition and the present input determine the current output. Typically, the supplied input selected feature is indicated by the phrase x = {x_1,x_2,•••,x_t,•••,x_T }, which depict the tth PD sample and the total of PD specimens is given by T. Thus, using Equation (4.12), the following can be obtained:

$$h_t = \sigma_h(FW_{xh}x_t + FW_{hh}h_{t-1} + b_h) \tag{4.12}$$

where FW_xh represents the fuzzy weighting matrix of the hidden layer's inputs, FW_hh represents the hidden layer's fuzzy weight matrices, b_h signifies the hidden layer's bias, and σ_h represents the activation functions. Additionally, h_t shows the outcome of the hidden layers at time t. By utilizing Equation (4.13), it is obtained.

$$y_t = \sigma_y(FW_{ho}h_t + b_o) \tag{4.13}$$

The sample's anticipated label is shown by y_t, the result's bias is indicated by b_o, the activation function is indicated by σ_y, and Which provides the weight matrix between the outcome and the hidden layers. The internal "LSTM cell" cycle, which takes the shape of a self-loop, is also a part of the external RNN cycle. A non-linear element is not produced by LSTM for input transformation and loop cells. Nevertheless, disregard the fact that the sigmoid weight is fixed at 0 and 1 using Equation (4.14), and that the gate f_i^((t)) controls the ring weights.

$$f_i^{(t)} = \sigma\left(b_i^f + \sum_j FU_{i,j}^f x_j^{(t)} + \sum_j FW_{i,j}^f h_j^{(t-1)}\right) \tag{4.14}$$

here x_t indicates the current input vectors, h_t is the hidden layer's current vectors, and h_t represents the integrated outcome of the LSTM cells. The terms bias fuzzy cyclic weights, input fuzzy weights, and forgetting gates, respectively, are represented by the symbols b^f , FU^f and FW^f. To compute fuzzy weights, one uses the Gaussian Membership function. As a result, using Equation (4.15), the internal configuration of the LSTM cell is changed as follows, where the conditional self-ring weight $f_i^{(t)}$exists.

$$s_i^{(t)} = f_i^{(t)} s_i^{(t-1)} + g_i^{(t)} \sigma\left(b_i + \sum_j FU_{i,j} x_j^{(t)} + \sum_j FW_{i,j} x_j^{(t-1)}\right) \tag{4.15}$$

Here, b, FU, and FW represent, respectively, the bias, input fuzzy weight, and cyclic fuzzy weight of the forgetting gates in LSTM cells. $g_i^{(t)}$external input gates unit is similar to the oblivion gates in the form of its individual parameters. $g_i^{(t)}$external input gate unit is given by Equation (4.16),

$$g_i^{(t)} = \sigma\left(b_i^g + \sum_j FU_{i,j}^g x_j^{(t)} + \sum_j FW_{i,j}^g h_j^{(t-1)}\right) \tag{4.16}$$

$h_j^{(t)}$ LSTM cell outcome is closed by $q_j^{(t)}$, then the outcome gate is defined using Equation (4.17),

$$q_j^{(t)} = \sigma\left(b_i^o + \sum_j FU_{i,j}^o x_j^{(t)} + \sum_j FW_{i,j}^o h_j^{(t-1)}\right) \tag{4.17}$$

where b^o, FU^o and FW^orefer to the forgetting gate, the input fuzzy weight, and the cyclic weights of the bias, respectively. During these transitions, the state of s(t), an additional input with fuzzy weights i, and 3-gates of i is selected. Here, three more criteria are required. LSTM only considers one direction, even though it gathers the long-term sequence information. This suggests that the current frame is influenced by the LSTM's present state. In order to establish the link, the current PD sample is used in conjunction with the subsequent PD sample. Bi-LSTM is the best option for the circumstances at hand. The first layer is made up of forward LSTM, followed by reverse LSTM. Expression is used to compute the outcome (4.18-4.19),

$$h_t = \alpha h_t^f + \beta h_t^b \tag{4.18}$$

$$y_t = \sigma(h_t) \tag{4.19}$$

where h_t^f indicates the forward LSTM result, which take PD samples between x_1 and x_T to be the input, h_t^b specifies the output of backward LSTM using samples of PD between x_T and x_1, α and β refer to the (α+β=1) significance of forwards and backwards LSTM, h_t specifies the 2-LSTM elements-wise, y_t stands for the classification outcomes of the classifiers for diagnosing PD. Bi-LSTMs gets the details of the structure, so that its execution is better compared to single directional LSTMs. The final outcome of Bi-LSTM is given by the Equations (4.20-4.22),

$$h_t^c = h_{t-1}^c + \tanh\left(FW_1 h_{t-1}^c + FW_2\left(h_{t-1}^f + h_{t-1}^b\right) + b_h\right) \tag{4.20}$$

$$h_t = \alpha h_t^f + \beta h_t^b + \gamma h_t^c \tag{4.21}$$

$$y_t = \sigma(h_t) \tag{4.22}$$

here $h^c{}_t$ signifies the chosen features taking part in Bi-LSTM to diagnose PD including $\alpha + \beta + \gamma = 1$. The ultimate output of the Bi-LSTM is evaluated by two Convolution layers to achieve the diagnosis of PD. Multi-modal factorized bilinear pooling are used for reintegrating the features that CNN and Bi-LSTM evaluated features process. Equation (4.23) of Bi-LSTM is described as,

$$y_i = 1^T\left(FU_i^T x \circ FV_i^T z\right) + b \tag{4.23}$$

where FU_i and FV_i refer to the weights; b indicates the bias weight; x, z stands for the features to be fused by 2-classifiers, y_i refers to the h values of the fused features along with the outcomes of classification.

4.3 Experimental Results

The proposed FCBi-LSTM classifier performance in tests is discussed in this part, and it is compared to that of other classifiers such as FCLSTM-CNN, CNN, and SVM. The UCI Machine Learning repository provided the data that was used in this research, and it has recently been used in other analyses. The Parkinson dataset database was used for the tests on the PD recognition system, which were carried out using matrix laboratory R2016 (MATLAB R2016a). The machine that the development is performed on has the following specifications:

Intel(R) CoreTMi3-4160T CPU@3.10 GHz 3.09 GHz processor, 4.00 GB RAM, Windows 8.1 pro, 64-bit OS, and 1 TB hard disc.

For the butterfly optimisation algorithm, the objective function used in Ackley and the elitism parameter is set to be 2. This describes how many of the best habitats to keep from one generation to the next. The maximum step size is 1.0 and the period is 1.2 i.e. 12 months in a year. The number of hidden layers used in Bi-LSTM is 100.

4.3.1. Performance Metrics

The classifier's attained prediction performance is assessed using evaluation measures. These metrics like F-Measure, accuracy, error, and MCC help to verify how well the classifiers predict outcomes.

4.3.2. Results Comparisons

The proposed FCBi-LSTM classifier as well as more modern classifiers like CNN, SVM, and FCLSTM-CNN is used in experiments with three different feature types. While TQWT+MFCC + Concat and TQWT+ Wavelet + Concat had respective accuracy of 95.7455% and 94.2252% (refer to Table 4.1), the combined accuracy of TQWT+MFCC + Wavelet attributes was 94.3965% (f-measure of 95.3965%). The ensemble (MFCC+Wavelet+Concat) performs significantly worse than alternative methods, as demonstrated by metrics like as accuracy, F-Measure, and MCC (See Table 4.1).

Table 4.1. Accuracy Comparison Results of Classifiers with Triple Feature with KPCA+ FMBOA

Feature Extraction Combination	FCBi-LSTM	FCLSTM	CNN
TQWT+MFCC+Wavelet	94.3965	92.1457	86.6697
TQWT+MFCC+Concat	95.7455	92.1854	91.0315
TQWT+ Wavelet + Concat	94.2252	92.2252	87.1695
MFCC + Wavelet + Concat	96.2865	94.2557	93.2752

Table 4.1 compares accuracy results of classifiers based on KPCA+ FMBOA with triple features, which including FCBi-LSTM, FCLSTM Classifier, and CNN Classifier, with MFCC + Wavelet + Concat having the highest accuracy of 96.2865% of accuracy in FCBi-LSTM, 94.2557% of accuracy for FCLSTM and 93.2752 of accuracy for CNN classifier.

Table 4.2 F-measure Comparison Results of Classifiers with Triple Feature with KPCA+ FMBOA

Feature Extraction Combination	FCBi-LSTM	FCLSTM	CNN
TQWT+MFCC+Wavelet	95.3965	93.0257	84.6697
TQWT+MFCC+Concat	93.3252	90.3252	89.0315
TQWT+ Wavelet + Concat	94.4252	92.2252	85.1695
MFCC + Wavelet + Concat	95.4530	90.4729	91.2752

The table 4.2 compares the F-measure results of classifiers based on KPCA+ FMBOA with triple features, including, FCBi-LSTM, FCLSTM Classifier, and CNN Classifier, with results 95.4530% of F-measure for FCBi-LSTM in the combination of MFCC+ Wavelet+ Concat. For FCLSTM 93.0257% of F-measure in the combination of TQWT+MFCC+Concat and 91.2752% of MFCC + Wavelet + Concat.

Table 4.3 Mathews Correlation Coefficient Comparison Results of Classifiers with Triple Feature with KPCA+ FMBOA

Feature Extraction Combination	FCBi-LSTM	FCLSTM	CNN
TQWT+MFCC+Wavelet	71.3513	66.3669	55.9007
TQWT+MFCC+Concat	69.4017	66.4060	60.1656
TQWT+ Wavelet + Concat	65.9483	64.1457	62.0993
MFCC + Wavelet + Concat	67.9917	65.9960	63.2384

The table 4.3 compares the MCC results of classifiers based on KPCA+ FMBOA with triple features, including, FCBi-LSTM, FCLSTM Classifier, and CNN Classifier, with results 65.9483% of MCC for FCBi-LSTM in the combination of TQWT+ Wavelet + Concat. 64.1457% of MCC for FCLSTM and 62.2384% of MCC for CNN classifier.

4.4 SUMMARY

In this work, the deep FCBi-LSTM classifier which uses voice characteristics to classify Parkinson's disease is introduced. Several characteristics, including as TQWT, Wavelet, MFCC, and Concat, can be extracted from the datasets starting with the input dataset in order to categorize Parkinson's disease detections. Based on the recovered attributes, Kernel Principle Component Analysis based dimensionality reductions are applied to the input datasets. The FMBOA-based Feature Selection approach is used to identify the important features of the dataset once the recurring features have been eliminated. Lastly, the objective of the FCBi-LSTM classifier is revealed to be sequential feature correlation estimation. The classifier is trained with data from the UCI dataset. The outcomes reveal that the suggested classifier yields the best accuracy results when compared to the CNN, SVM, and FCLSTM-CNN classifiers.