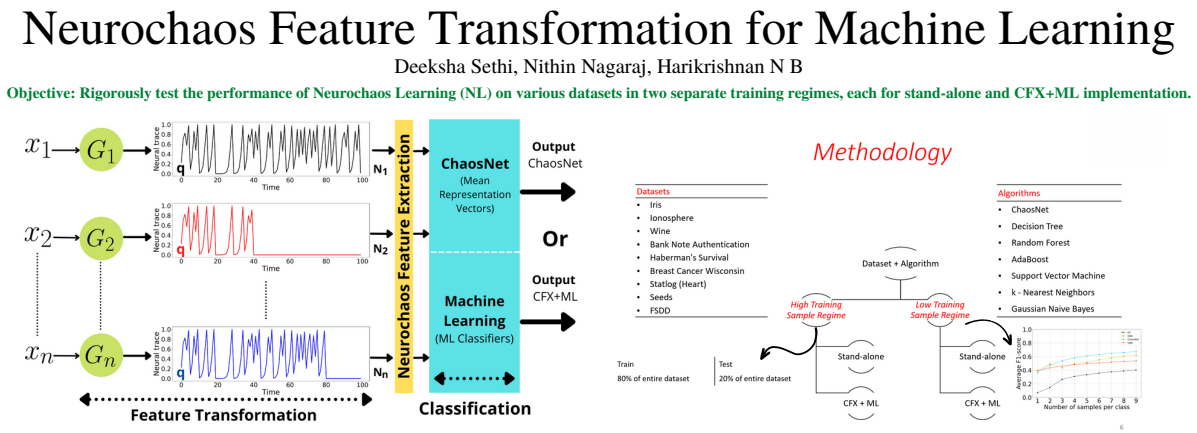
Graphical Abstract

Supplementary Information for "Neurochaos Feature Transformation for Machine Learning"

Deeksha Sethi[1], Nithin Nagaraj[2], Harikrishnan N B[3]



- *High Training Sample Regime*: Highest performance boost obtained is **25.97%** for **Statlog (Heart) dataset using CFX+Decision Tree**.
- *Low Training Sample Regime*: Highest performance boost obtained is **144.38%** for **Haberman's Survival dataset using CFX+Random Forest**.

[1]deeksha.sethi03@gmail.com
[2]nithin@nias.res.in
[3]Corresponding author: harikrishnannb07@gmail.com, +91 73395 34649

Highlights

Supplementary Information for "Neurochaos Feature Transformation for Machine Learning"

Deeksha Sethi[4], Nithin Nagaraj[5], Harikrishnan N B[6]

- A recently proposed brain-inspired learning algorithm, *Neurochaos Learning* (NL) is rigorously tested in this study for the first time.
- We explore a unique combination of neurochaos-based feature transformation and extraction with traditional ML algorithms (CFX+ML).
- Experiments are performed in two diagonals for nine publicly available datasets: High training sample regime (HTSR: $80\% - 20\%$ Test-Train split) and Low training sample regime (LTSR: ≤ 9 samples/class for training with 150 independent trials).
- HTSR: Highest performance boost in macro F1 score obtained is **25.97%** for *Statlog (Heart)* dataset using CFX+Decision Tree. LTSR: Highest performance boost obtained is **144.38%** is obtained for *Haberman's Survival* dataset using CFX+Random Forest.
- NL offers enormous flexibility in combining CFX with any ML/DL classifier to boost its stand-alone performance.

[4]deeksha.sethi03@gmail.com
[5]nithin@nias.res.in
[6]Corresponding author: harikrishnannb07@gmail.com, +91 73395 34649

Supplementary Information for "Neurochaos Feature Transformation for Machine Learning"

Deeksha Sethi[1a], Nithin Nagaraj[2b], Harikrishnan N B[3b,c]

[a] *Department of Electronics and Communication Engineering, BMS Institute of Technology and Management, Bengaluru, 560064, Karnataka, India*
[b] *Consciousness Studies Programme, National Institute of Advanced Studies, Indian Institute of Science Campus, Bengaluru, 560012, Karnataka, India*
[c] *Department of Computer Science & Information Systems and APPCAIR, BITS Pilani, K K Birla Goa Campus, Zuarinagar, 403726, Goa, India*

Abstract

A novel brain-inspired algorithm, *Neurochaos Learning* (NL), was recently proposed [1, 2]. NL is a unique combination of ideas drawn from Chaos Theory, Stochastic Resonance, Neuroscience, and Machine Learning (ML) for classification. This study compares NL chaos-based-hybrid ML architectures with stand-alone ML algorithms. The highest performance boost obtained in the high training sample regime is **25.97%** for *Statlog (Heart)* dataset using CFX+Decision Tree. The highest performance boost obtained in the low training sample regime is **144.38%** for *Haberman's Survival* dataset using CFX+Random Forest. NL offers enormous flexibility in combining CFX with any ML/DL classifier to boost its stand-alone performance.

Keywords: Neurochaos Learning, `ChaosNet`, Feature Transformation, Feature Extraction, Brain Inspired Learning

1. Appendix

This is the appendix pertaining to the main manuscript. It contains the following –

[1] deeksha.sethi03@gmail.com
[2] nithin@nias.res.in
[3] Corresponding author: harikrishnannb07@gmail.com, +91 73395 34649

1. Description of datasets used in our study, including the coding rule for the labels of different classes.
2. Hyperparameter tuning details for each dataset and for each ML algorithm (Decision Tree, Random Forest, AdaBoost, SVM, k-NN) and NL algorithm (`ChaosNet`) used in the study.
3. Testdata macro F1-scores for each algorithm in the high training sample regime for each dataset.

Appendix A. Dataset Description

The ChaosFEX (CFX) feature extraction algorithm requires normalization of the dataset and numeric codes for labels. Therefore, to maintain uniformity, all datasets are normalized[4] for both stand-alone algorithms and their integration with CFX. The labels are renamed to begin from zero in each dataset to ensure compatibility with CFX feature extraction.

Appendix A.1. Iris

Iris [3, 4] aids the classification of three iris plant variants: Iris Setosa, Iris Versicolour, and Iris Virginica. There are 150 data instances in this dataset with four attributes in each data instance: sepal length, sepal width, petal length, and petal width. All attributes are in *cms*. The specified class distribution provided in Table A.10 is in the following order: (Setosa, Versicolour, Virginica).

Table A.1: ***Iris***: Rule followed for renaming of the class labels.

Class Label	**Numeric Code**	**Number of Total Data Instances**
Iris-Setosa	0	50
Iris-Versicolour	1	50
Iris-Virginica	2	50

[4] $X\_norm = \frac{X - \min(X)}{\max(X) - \min(X)}$.

Appendix A.2. Ionosphere

The *Ionosphere* [5, 6] dataset enables a binary classification problem. The classes represent the status of returning a radar signal from the Ionosphere. Label 'g' (Good) denotes the return of the radar signal, and label 'b' (Bad) indicates no trace of return of the radar signal. The goal of this experiment is to identify the structure of the Ionosphere using radar signals. This dataset has 351 data instances and 34 attributes. The specified class distribution provided in Table A.10 is as follows: (Bad, Good).

Table A.2: ***Ionosphere***: Rule followed for renaming of the class labels.

Class Label	**Numeric Code**	**Number of Total Data Instances**
b (Bad)	0	126
g (Good)	1	225

Appendix A.3. Wine

Wine [5, 7] dataset aims to identify the origin of different wines using chemical analysis. The classes are labeled '1', '2', and '3'. It has 178 data instances and 13 attributes ranging from alcohol, and malic acid to hue and proline for the collected samples. The specified class distribution provided in Table A.10 is as follows: $(1, 2, 3)$.

Table A.3: ***Wine***: Rule followed for renaming of the class labels.

Class Label	**Numeric Code**	**Number of Total Data Instances**
1	0	59
2	1	71
3	2	48

Appendix A.4. Bank Note Authentication

Bank Note Authentication [5, 8] is a binary classification dataset. The classes involve Genuine and Forgery. A Genuine class refers to an authentic banknote denoted by '0', while a Forgery class refers to a forged banknote denoted by '1'. The obtained dataset is from images of banknotes belonging to both classes, taken from an industrial camera. It contains 1372 total data instances. The dataset has four attributes retrieved from the images using wavelet transformation. The specified class distribution provided in Table A.10 is as follows: (Genuine, Forgery).

Table A.4: ***Bank Note Authentication***: Rule followed for renaming of the class labels.

Class Label	**Numeric Code**	**Number of Total Data Instances**
0 (Genuine)	0	762
1 (Forgery)	1	610

Appendix A.5. Haberman's Survival

Haberman's Survival [5, 9] is a compilation of sections of a study investigating the lifespan of a patient after undergoing a breast cancer surgery. This is a binary classification problem and the dataset provides information for the prediction of the survival of patients beyond five years. Class '1' denotes survival of the patient for five years or longer after the surgery. Class '2' denotes the death of a patient within five years of the surgery. It contains 306 total data nstances and three attributes. The specified class distribution provided in Table A.10 is as follows: $(1, 2)$.

Table A.5: ***Haberman's Survival***: Rule followed for renaming of the class labels.

Class Label	**Numeric Code**	**Number of Total Data Instances**
1 ($<$ 5yrs)	0	225
2 ($\geq$ 5yrs)	1	81

Appendix A.6. Breast Cancer Wisconsin

Breast Cancer Wisconsin [5, 10] dataset deals with the classification of the intensity of the breast cancer. Class 'M' refers to a malignant level of infection and class 'B' refers to a benign level of infection. It contains a total of 569 data instances and 31 attributes such as radius, perimeter, texture, smoothness, etc. for each cell nucleus. The specified class distribution provided in Table A.10 is as follows: (Malignant - M, Benign - B).

Table A.6: ***Breast Cancer Wisconsin***: Rule followed for renaming of the class labels.

Class Label	**Numeric Code**	**Number of Total Data Instances**
M (Malignant)	0	212
B (Benign)	1	357

Appendix A.7. Statlog (Heart)

Statlog (Heart) [5] enables differentiation between presence and absence of a heart disease in a patient. Class '1' denotes the absence while class '2' denotes the presence of a heart disease. It contains 270 total data instances and 13 attributes including resting blood pressure, chest pain type, exercise induced angina and so on. The specified class distribution provided in Table A.10 is as follows: (Absence , Presence).

Table A.7: ***Statlog (Heart)***: Rule followed for renaming of the class labels.

Class Label	**Numeric Code**	**Number of Total Data Instances**
1 (Absence)	0	150
2 (Presence)	1	120

Appendix A.8. Seeds

Seeds [5] dataset examines three classes of wheat: Kama, Rosa and Canadian using soft X-ray on the wheat kernels to retrieve relevant properties. It contains 210 total data instances and seven attributes namely compactness, length, width etc. of each wheat kernel. The specified class distribution provided in Table A.10 is as follows: (Kama, Rosa, Canadian).

Table A.8: ***Seeds***: Rule followed for renaming of the class labels.

Class Label	**Numeric Code**	**Number of Total Data Instances**
1.0 (Kama)	0	70
2.0 (Rosa)	1	70
3.0 (Canadian)	2	70

Appendix A.9. Free Spoken Digit Dataset

The *Free Spoken Digit Dataset* [11] is a time-series dataset comprising recordings of six speakers. Each speaker recites numbers from one to nine. For each number, every speaker makes 50 recordings. The speaker chosen for all experiments is Jackson. The dataset undergoes preprocessing using a Fast Fourier Transform (FFT) technique. The dataset for speaker Jackson has 500 data instances, and only instances above a threshold of 3000 samples are considered to tackle the varying data length through the dataset. In these data instances, only the first 3005 data samples are examined. Finally, 480 data instances are filtered to feed into the algorithm. The specified class distribution provided in Table A.10 is as follows: $(0, 1, 2, 3, 4, 5, 6, 7, 8, 9)$.

Table A.9: ***FSDD***: Rule followed for renaming of the class labels.

Class Label	**Numeric Code**	**Number of Total Data Instances**
0	0	50
1	1	50
2	2	50
3	3	50
4	4	46
5	5	41
6	6	50
7	7	50
8	8	43
9	9	50

Table A.10: Train-Test split in experiments. High training sample regime corresponds to an 80% and 20% split in training and testing respectively.

Dataset	**Classes**	**Features**	**Training samples per class**	**Testing samples per class**
Iris	3	4	(40, 41, 39)	(10, 9, 11)
Ionosphere	2	34	(98, 182)	(28, 43)
Wine	3	13	(45, 57, 40)	(14, 14, 8)
Bank Note Authentication	2	4	(614, 483)	(148, 127)
Haberman's Survival	2	3	(181, 63)	(44, 18)
Breast Cancer Wisconsin	2	31	(169, 286)	(43, 71)
Statlog (Heart)	2	13	(117, 99)	(33, 21)
Seeds	3	7	(59, 56, 53)	(11, 14, 17)
FSDD	10	3005	(40, 35, 44, 42, 38, 34, 37, 44, 33, 37)	(10, 15, 6, 8, 8, 7, 13, 6, 10, 13)

Appendix B. Comparative Performance Evaluation

In this section, the authors represent the results in two formats (a) bar graph and (b) line graph. The bar graph depicts the comparative results for `ChaosNet`, CFX+ML and stand-alone ML in the high training sample regime. All values plotted in the bar graphs for each dataset are provided in Appendix C from Table C.28 - C.33. On the other hand, the line graph depicts the comparative performance of `ChaosNet`, CFX+ML and stand-alone ML in the low training sample regime.

Appendix B.0.1. Results for Iris

The tuned hyperparameters used and all experimental results for the *Iris* dataset are available in Table B.11 and Figure B.1 respectively.

Table B.11: Hyperparameters used for *Iris* dataset for high and low training sample regime experiments.

Hyperparameter	**Tuned Value**
q	0.141
b	0.499
ϵ	0.147

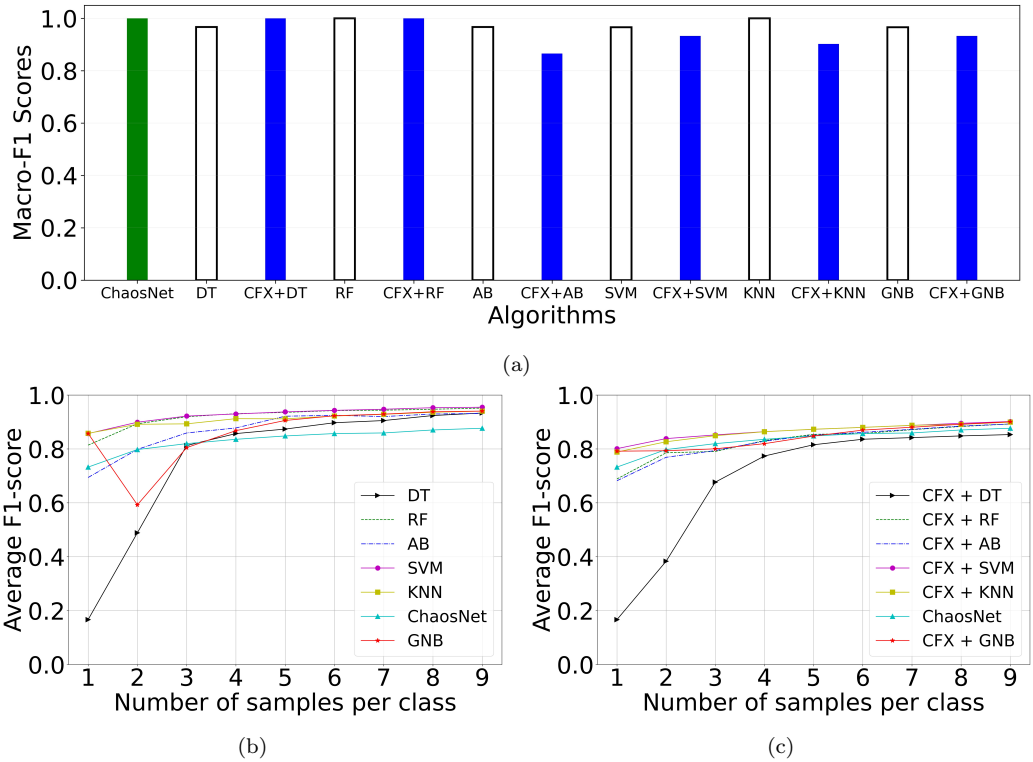


Figure B.1: *Iris*. (a) High training sample regime. (b) Comparative performance of stand-alone algorithms in the low training sample regime. (c) Comparative performance of CFX+ML algorithms in the low training sample regime.

Appendix B.0.2. Results for Ionosphere

The tuned hyperparameters used and all experimental results for the *Ionosphere* dataset are available in Table B.12 and Figure B.2 respectively.

Table B.12: Hyperparameters used for *Ionosphere* dataset for high and low training sample regime experiments.

Hyperparameter	**Tuned Value**
q	0.680
b	0.969
ϵ	0.164

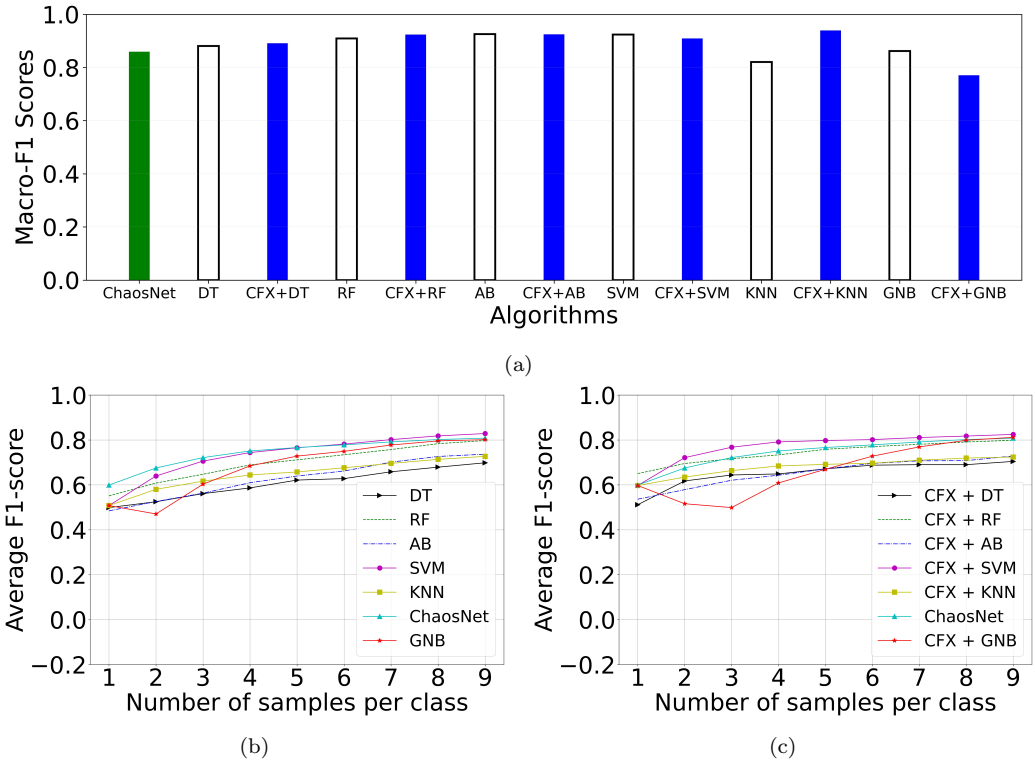


Figure B.2: *Ionosphere.* (a) High training sample regime. (b) Comparative performance of stand-alone algorithms in the low training sample regime. (c) Comparative performance of CFX+ML algorithms in the low training sample regime.

Appendix B.0.3. Results for Wine

The tuned hyperparameters used and all experimental results for the *Wine* dataset are available in Table B.13 and Figure B.3 respectively.

Table B.13: Hyperparameters used for *Wine* dataset for high and low training sample regime experiments.

Hyperparameter	**Tuned Value**
q	0.790
b	0.499
ϵ	0.262

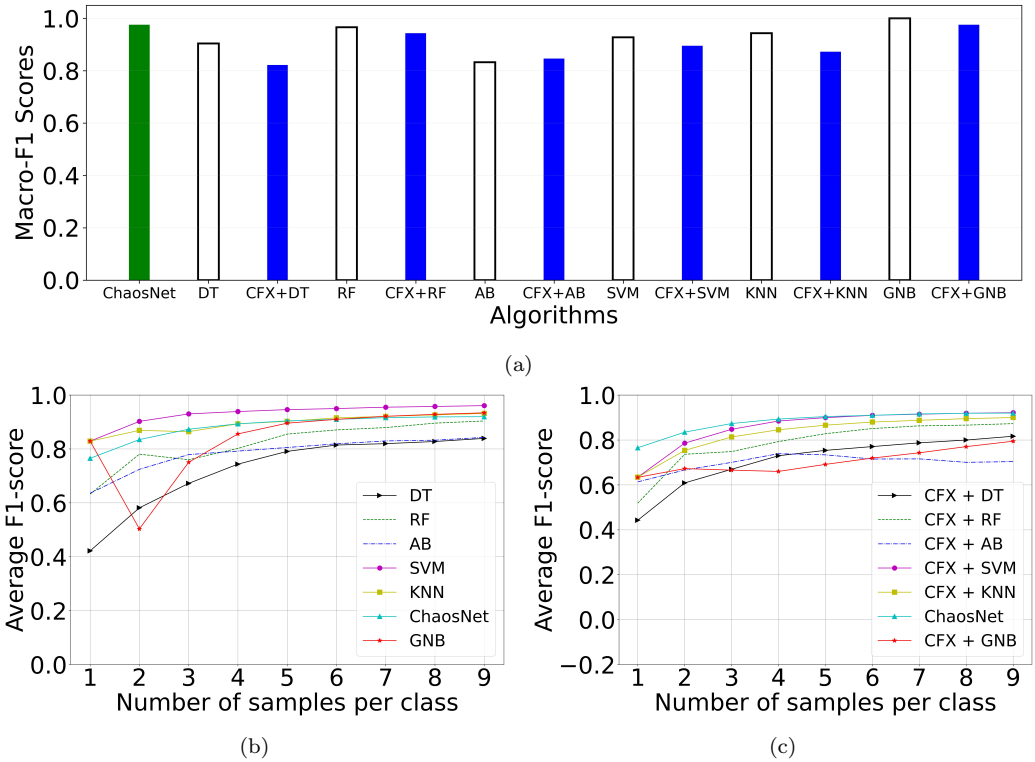


Figure B.3: *Wine*. (a) High training sample regime. (b) Comparative performance of stand-alone algorithms in the low training sample regime. (c) Comparative performance of CFX+ML algorithms in the low training sample regime.

Appendix B.0.4. Results for Bank Note Authentication

The tuned hyperparameters used and all experimental results for the *Bank Note Authentication* dataset are available in Table B.14 and Figure B.4 respectively.

Table B.14: Hyperparameters used for *Bank Note Authentication* dataset for high and low training sample regime experiments.

Hyperparameter	**Tuned Value**
q	0.080
b	0.250
ϵ	0.233

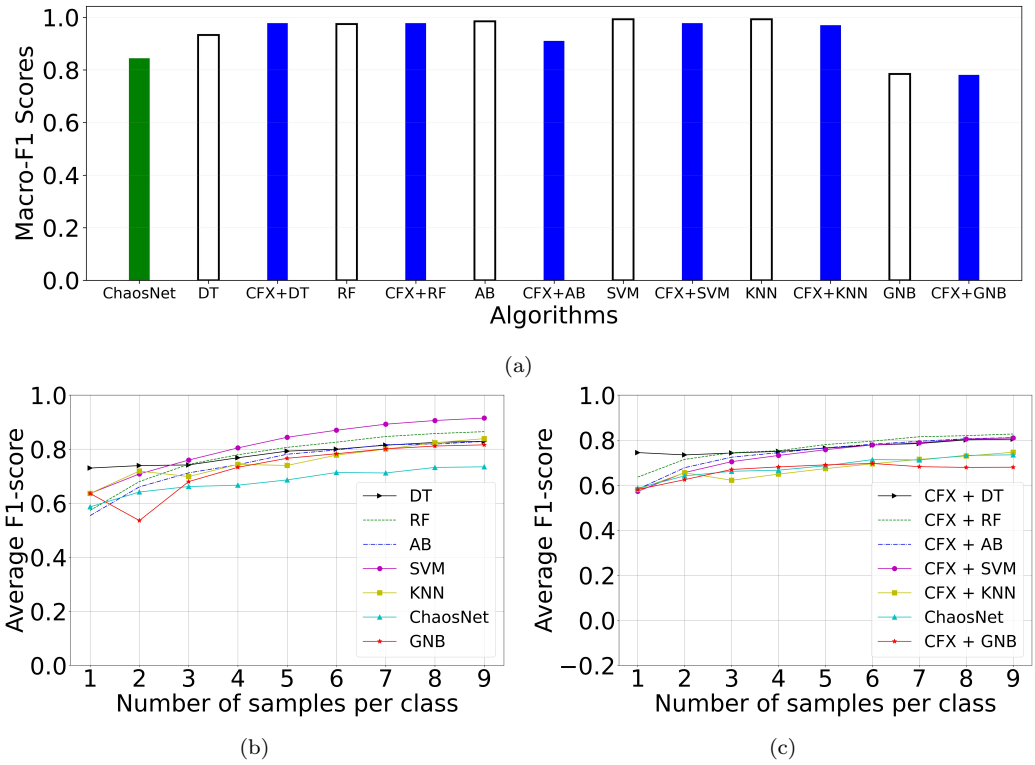


Figure B.4: *Bank Note Authentication.* (a) High training sample regime. (b) Comparative performance of stand-alone algorithms in the low training sample regime. (c) Comparative performance of CFX+ML algorithms in the low training sample regime.

Appendix B.0.5. Results for Haberman's Survival

The tuned hyperparameters used and all experimental results for the *Haberman's Survival* dataset are available in Table B.15 and Figure B.5 respectively.

Table B.15: Hyperparameters used for *Haberman's Survival* dataset for high and low training sample regime experiments.

Hyperparameter	**Tuned Value**
q	0.810
b	0.140
ϵ	0.003

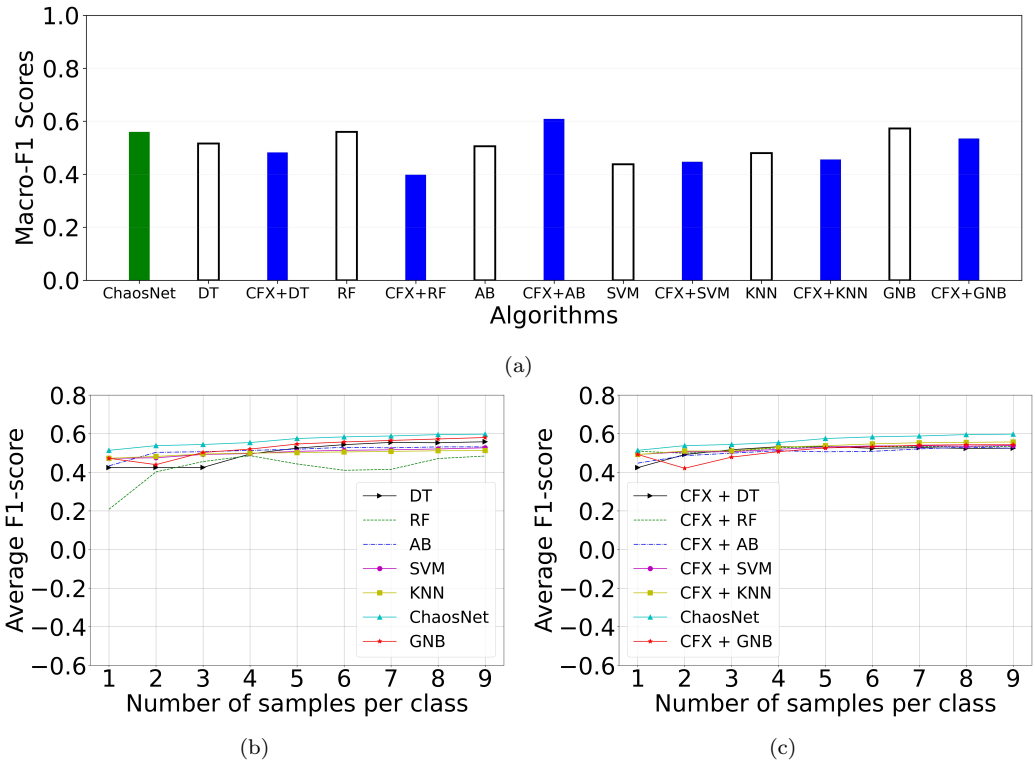


Figure B.5: *Haberman's Survival.* (a) High training sample regime. (b) Comparative performance of stand-alone algorithms in the low training sample regime. (c) Comparative performance of CFX+ML algorithms in the low training sample regime.

Appendix B.0.6. Results for Breast Cancer Wisconsin

The tuned hyperparameters used and all experimental results for the *Breast Cancer Wisconsin* dataset are available in Table B.16 and Figure B.6 respectively.

Table B.16: Hyperparameters used for *Breast Cancer Wisconsin* dataset for high and low training sample regime experiments.

Hyperparameter	**Tuned Value**
q	0.930
b	0.490
ϵ	0.159

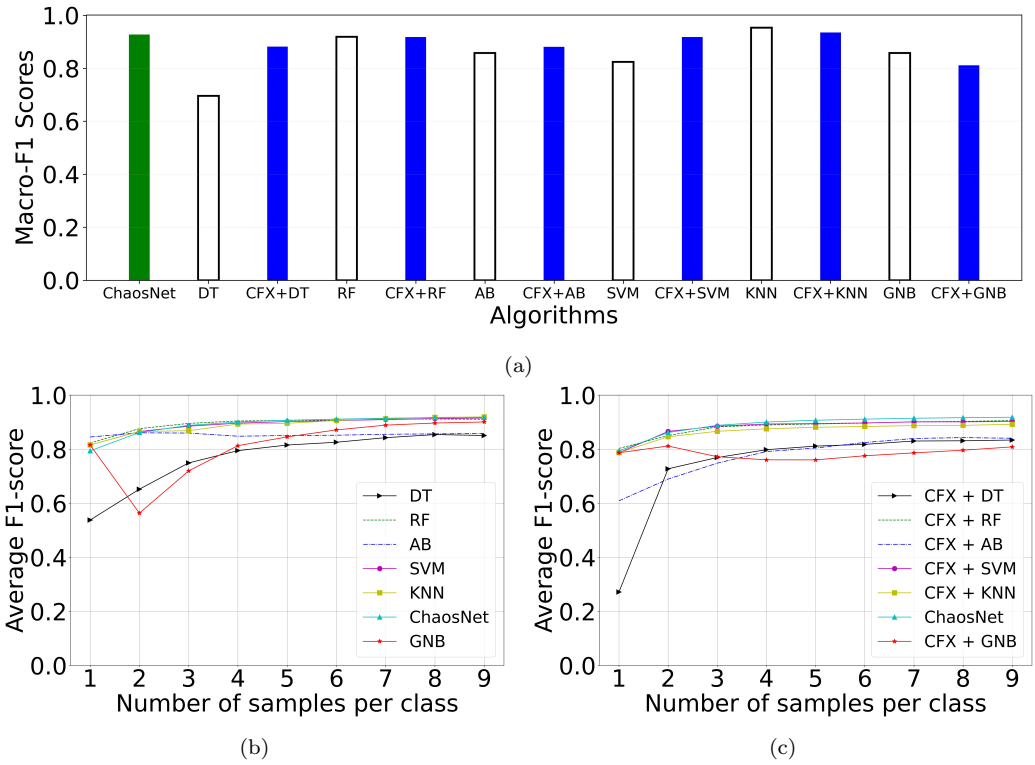


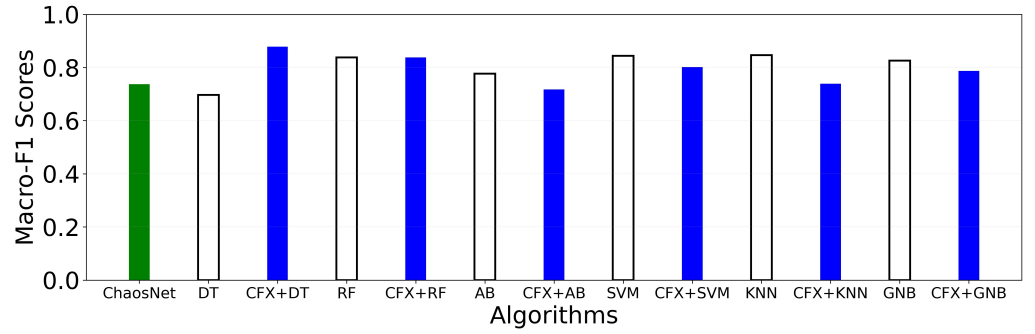
Figure B.6: *Breast Cancer Wisconsin.* (a) High training sample regime. (b) Comparative performance of stand-alone algorithms in the low training sample regime. (c) Comparative performance of CFX+ML algorithms in the low training sample regime.

Appendix B.0.7. Results for Statlog (Heart)

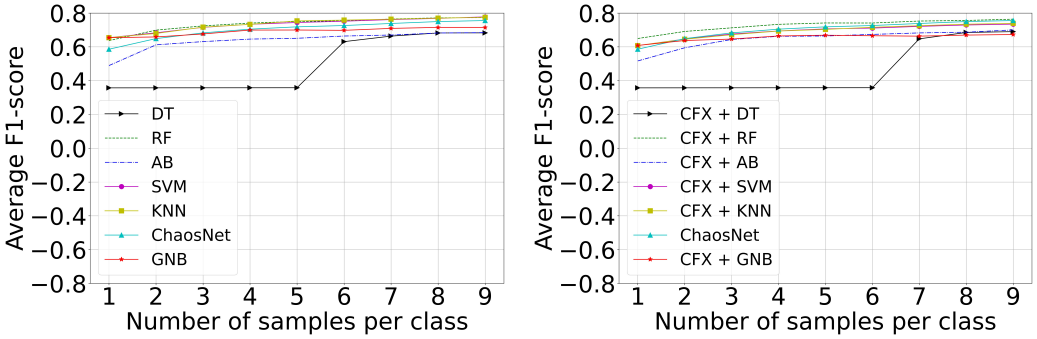
The tuned hyperparameters used and all experimental results for the *Statlog (Heart)* dataset are available in Table B.17 and Figure B.7 respectively.

Table B.17: Hyperparameters used for *Statlog (Heart)* dataset for high and low training sample regime experiments.

Hyperparameter	**Tuned Value**
q	0.080
b	0.060
ϵ	0.170



(a)



(b) (c)

Figure B.7: *Statlog (Heart).* (a) High training sample regime. (b) Comparative performance of stand-alone algorithms in the low training sample regime. (c) Comparative performance of CFX+ML algorithms in the low training sample regime.

Appendix B.0.8. Results for Seeds

The tuned hyperparameters used and all experimental results for the *Seeds* dataset are available in Table B.18 and Figure B.8 respectively.

Table B.18: Hyperparameters used for *Seeds* dataset for high and low training sample regime experiments.

Hyperparameter	**Tuned Value**
q	0.020
b	0.070
ϵ	0.238

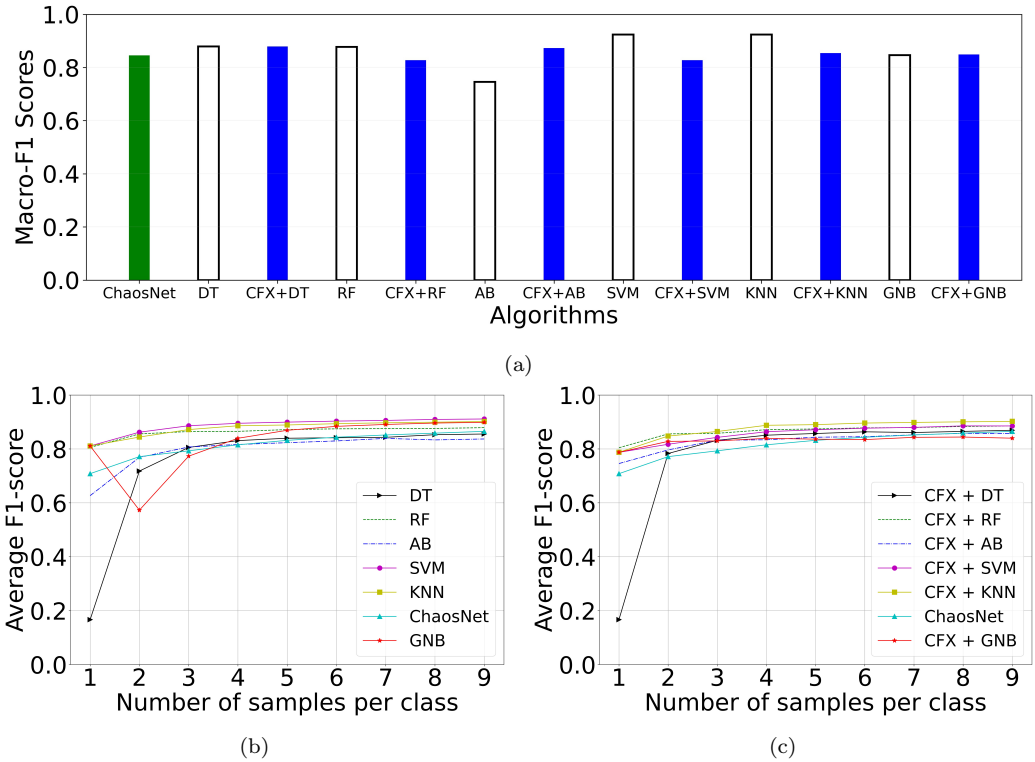


Figure B.8: *Seeds*. (a) High training sample regime. (b) Comparative performance of stand-alone algorithms in the low training sample regime. (c) Comparative performance of CFX+ML algorithms in the low training sample regime.

Appendix B.0.9. Results for Free Spoken Digit Dataset (FSDD)

The tuned hyperparameters used and all experimental results for the *FSDD* dataset are available in Table B.19 and Figure B.9 respectively.

Table B.19: Hyperparameters used for *FSDD* dataset for high and low training sample regime experiments [12]).

Hyperparameter	**Tuned Value**
q	0.340
b	0.499
ϵ	0.178

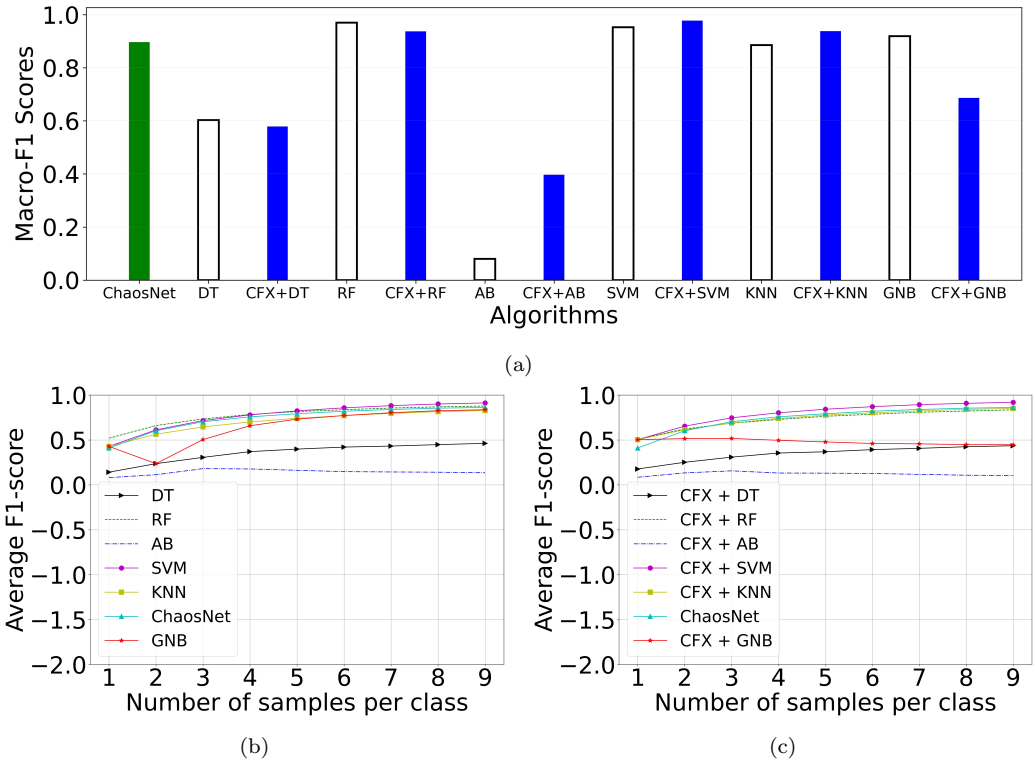


Figure B.9: *Free Spoken Digit Dataset.* (a) High training sample regime. (b) Comparative performance of stand-alone algorithms in the low training sample regime. (c) Comparative performance of CFX+ML algorithms in the low training sample regime.

Appendix B.1. Hyperparameter Tuning

Every ML algorithm has a set of optimal hyperparameters to be found by hyperparameter tuning. The hyperparameters for all the algorithms with respect to each dataset were tuned using five-fold cross-validation. Table B.20 provides the set of hyperparameters tuned for all algorithms in this research.

Table B.20: References for the tuned hyperparameters for all algorithms.

Algorithm	**Acronyms**	**Hyperparameters Tuned**	**Reference**
ChaosNet	-	q, b, ϵ	Tables B.11, B.12, B.13, B.14, B.15, B.16, B.17, B.18 and B.19
Decision Tree	DT	$min\_samples\_leaf$, max_depth, ccp_alpha	Tables C.21 and C.22 in Appendix C
Random Forest	RF	$n\_estimators$, max_depth	Tables C.23 and C.24 in Appendix C
AdaBoost	AB	$n\_estimators$	Table C.25 in Appendix C
Support Vector Machine	SVM	C, $kernel$	Table C.26 in Appendix C
k-Nearest Neighbors	k-NN, KNN	k	Table C.27 in Appendix C
Gaussian Naive Bayes	GNB	-	-

In the case of CFX+ML for *Iris, Ionosphere* and *Wine*, both CFX (q, b, ϵ) and ML hyperparameters were tuned. For the remaining datasets, the hyperparameters tuned for `ChaosNet` (q, b, ϵ) were retained and only the ML hyperparameters were tuned.

Appendix C. Hyperparameter Tuning

The hyperparameter tuning for all algorithms for the respective datasets are provided below:

Appendix C.0.1. Decision Tree

Following are the hyperparameters tuned for Decision Tree:

1. **min_samples_leaf**: Defines the minimum number of samples required for a leaf node in the decision tree. It is tuned from 1 to 10 with a step-size of 1.
2. **max_depth**: Declares the maximum depth to which a decision tree can be grown. It is tuned from 1 to 10 with a step-size of 1.
3. **ccp_alpha**: A numpy array of alpha values obtained by devising Cost Complexity Pruning on the original decision tree. This array is obtained using the *cost_complexity_pruning_path*.

All remaining hyperparameters offered by scikit-learn are retained in their default forms. The results of hyperparameter tuning for Decision Tree are available in Table C.21 and C.22.

Table C.21: ***Decision Tree***: Tuned hyperparameters for all nine datasets (Part I). The performance metric used for the provided results is macro F1-score.

Dataset	Implementation	Tuned Hyperparameters	F1 Score
Iris	Stand-Alone Decision Tree	$min\_samples\_leaf = 3$	0.931
		$max\_depth = 3$	
		$ccp\_alpha = 0.0$	
	ChaosFEX + Decision Tree	$min\_samples\_leaf = 3$	0.955
		$max\_depth = 4$	
		$ccp\_alpha = 0.0$	
		$q = 0.21$	
		$b = 0.969$	
		$\epsilon = 0.13$	
Ionosphere	Stand-Alone Decision Tree	$min\_samples\_leaf = 1$	0.882
		$max\_depth = 2$	
		$ccp\_alpha = 0.0$	
	ChaosFEX + Decision Tree	$min\_samples\_leaf = 1$	0.921
		$max\_depth = 7$	
		$ccp\_alpha = 0.0$	
		$q = 0.21$	
		$b = 0.969$	
		$\epsilon = 0.22$	
Wine	Stand-Alone Decision Tree	$min\_samples\_leaf = 1$	0.916
		$max\_depth = 3$	
		$ccp\_alpha = 0.0$	
	ChaosFEX + Decision Tree	$min\_samples\_leaf = 1$	0.949
		$max\_depth = 6$	
		$ccp\_alpha = 0.0$	
		$q = 0.21$	
		$b = 0.969$	
		$\epsilon = 0.07$	
Bank Note Authentication	Stand-Alone Decision Tree	$min\_samples\_leaf = 1$	0.977
		$max\_depth = 8$	
		$ccp\_alpha = 0.0$	
	ChaosFEX + Decision Tree	$min\_samples\_leaf = 1$	0.961
		$max\_depth = 6$	
		$ccp\_alpha = 0.000836$	
		$q = 0.080$	
		$b = 0.250$	
		$\epsilon = 0.233$	
Haberman's Survival	Stand-Alone Decision Tree	$min\_samples\_leaf = 4$	0.614
		$max\_depth = 6$	
		$ccp\_alpha = 0.005389$	
	ChaosFEX + Decision Tree	$min\_samples\_leaf = 2$	0.648
		$max\_depth = 8$	
		$ccp\_alpha = 0.005389$	
		$q = 0.81$	
		$b = 0.14$	
		$\epsilon = 0.003$	

Appendix C.0.2. Random Forest

Following are the hyperparameters tuned for Random Forest:

Table C.22: ***Decision Tree***: Tuned hyperparameters for all nine datasets (Part II). The performance metric used for the provided results is macro F1-score.

Dataset	Implementation	Tuned Hyperparameters	F1 Score
Breast Cancer Wisconsin (Diagnostic)	Stand-Alone Decision Tree	$min\_samples\_leaf = 1$	0.920
		$max\_depth = 4$	
		$ccp\_alpha = 0.0$	
	ChaosFEX + Decision Tree	$min\_samples\_leaf = 2$	0.945
		$max\_depth = 8$	
		$ccp\_alpha = 0.005595$	
		$q = 0.930$	
		$b = 0.490$	
		$\epsilon = 0.159$	
Statlog (Heart)	Stand-Alone Decision Tree	$min\_samples\_leaf = 6$	0.779
		$max\_depth = 3$	
		$ccp\_alpha = 0.006818$	
	ChaosFEX + Decision Tree	$min\_samples\_leaf = 7$	0.786
		$max\_depth = 4$	
		$ccp\_alpha = 0.0$	
		$q = 0.08$	
		$b = 0.06$	
		$\epsilon = 0.17$	
Seeds	Stand-Alone Decision Tree	$min\_samples\_leaf = 2$	0.918
		$max\_depth = 4$	
		$ccp\_alpha = 0.005844$	
	ChaosFEX + Decision Tree	$min\_samples\_leaf = 2$	0.949
		$max\_depth = 5$	
		$ccp\_alpha = 0.0$	
		$q = 0.02$	
		$b = 0.07$	
		$\epsilon = 0.238$	
FSDD	Stand-Alone Decision Tree	$min\_samples\_leaf = 1$	0.536
		$max\_depth = 9$	
		$ccp\_alpha = 0.0099206$	
	ChaosFEX + Decision Tree	$min\_samples\_leaf = 1$	0.537
		$max\_depth = 10$	
		$ccp\_alpha = 0.0061384$	
		$q = 0.34$	
		$b = 0.499$	
		$\epsilon = 0.178$	

1. **n_estimators**: Defines the number of trees in the forest being grown. The values are tuned from the array $[1, 10, 100, 1000, 10000]$.
2. **max_depth**: Declares the maximum depth each tree the the forest is allowed to have. It is tuned from 1 to 10 with a step-size of 1.

All remaining hyperparameters offered by scikit-learn are retained in their default forms. The results of hyperparameter tuning for Random Forest are available in Table C.23 and C.24

Table C.23: ***Random Forest***: Tuned hyperparameters for all nine datasets (Part I). The performance metric used for the provided results is macro F1-score.

Dataset	Implementation	Tuned Hyperparameters	F1 Score
Iris	Stand-Alone Random Forest	$n\_estimators = 100$	0.944
		$max\_depth = 3$	
	ChaosFEX + Random Forest	$n\_estimators = 10$	0.944
		$max\_depth = 5$	
		$q = 0.21$	
		$b = 0.969$	
		$\epsilon = 0.11$	
Ionosphere	Stand-Alone Random Forest	$n\_estimators = 10000$	0.923
		$max\_depth = 4$	
	ChaosFEX + Random Forest	$n\_estimators = 100$	0.928
		$max\_depth = 10$	
		$q = 0.21$	
		$b = 0.969$	
		$\epsilon = 0.23$	
Wine	Stand-Alone Random Forest	$n\_estimators = 10$	0.980
		$max\_depth = 4$	
	ChaosFEX + Random Forest	$n\_estimators = 10$	0.987
		$max\_depth = 5$	
		$q = 0.21$	
		$b = 0.969$	
		$\epsilon = 0.05$	
Bank Note Authentication	Stand-Alone Random Forest	$n\_estimators = 100$	0.991
		$max\_depth = 8$	
	ChaosFEX + Random Forest	$n\_estimators = 100$	0.967
		$max\_depth = 7$	
		$q = 0.080$	
		$b = 0.250$	
		$\epsilon = 0.233$	
Haberman's Survival	Stand-Alone Random Forest	$n\_estimators = 1$	0.621
		$max\_depth = 3$	
	ChaosFEX + Random Forest	$n\_estimators = 1000$	0.651
		$max\_depth = 9$	
		$q = 0.810$	
		$b = 0.140$	
		$\epsilon = 0.003$	
Breast Cancer Wisconsin (Diagnostic)	Stand-Alone Random Forest	$n\_estimators = 1000$	0.956
		$max\_depth = 9$	
	ChaosFEX + Random Forest	$n\_estimators = 100$	0.955
		$max\_depth = 7$	
		$q = 0.930$	
		$b = 0.490$	
		$\epsilon = 0.159$	

Table C.24: ***Random Forest***: Tuned hyperparameters for all nine datasets (Part II). The performance metric used for the provided results is macro F1-score.

Dataset	Implementation	Tuned Hyperparameters	F1 Score
Statlog (Heart)	Stand-Alone Random Forest	$n\_estimators = 1000$	0.839
		$max\_depth = 2$	
	ChaosFEX + Random Forest	$n\_estimators = 10000$	0.830
		$max\_depth = 4$	
		$q = 0.08$	
		$b = 0.06$	
		$\epsilon = 0.17$	
Seeds	Stand-Alone Random Forest	$n\_estimators = 100$	0.918
		$max\_depth = 5$	
	ChaosFEX + Random Forest	$n\_estimators = 1000$	0.941
		$max\_depth = 5$	
		$q = 0.020$	
		$b = 0.070$	
		$\epsilon = 0.238$	
FSDD	Stand-Alone Random Forest	$n\_estimators = 1000$	0.9479
		$max\_depth = 9$	
	ChaosFEX + Random Forest	$n\_estimators = 1000$	0.921
		$max\_depth = 8$	
		$q = 0.340$	
		$b = 0.499$	
		$\epsilon = 0.178$	

Appendix C.0.3. AdaBoost

Following are the hyperparameters tuned for Adaptive Boosting (AdaBoost):

1. **n_estimators**: An upper limit for the number of stumps being grown. The values are tuned from the array $[1, 10, 50, 100, 500, 1000, 5000, 10000]$.

All remaining hyperparameters offered by scikit-learn are retained in their default forms. The results of hyperparameter tuning for AdaBoost are available in Table C.25.

Table C.25: ***AdaBoost***: Tuned hyperparameters for all nine datasets. The performance metric used for the provided results is macro F1-score.

Dataset	Implementation	Tuned Hyperparameters	F1 Score
Iris	Stand-Alone AdaBoost	$n\_estimators = 50$	0.929
	ChaosFEX + AdaBoost	$n\_estimators = 10$	0.915
		$q = 0.21$	
		$b = 0.969$	
		$\epsilon = 0.03$	
Ionosphere	Stand-Alone AdaBoost	$n\_estimators = 50$	0.929
	ChaosFEX + AdaBoost	$n\_estimators = 500$	0.921
		$q = 0.21$	
		$b = 0.969$	
		$\epsilon = 0.02$	
Wine	Stand-Alone AdaBoost	$n\_estimators = 10$	0.896
	ChaosFEX + AdaBoost	$n\_estimators = 10$	0.893
		$q = 0.21$	
		$b = 0.969$	
		$\epsilon = 0.05$	
Bank Note Authentication	Stand-Alone AdaBoost	$n\_estimators = 500$	0.999
	ChaosFEX + AdaBoost	$n\_estimators = 500$	0.934
		$q = 0.080$	
		$b = 0.250$	
		$\epsilon = 0.233$	
Haberman's Survival	Stand-Alone AdaBoost	$n\_estimators = 10$	0.620
	ChaosFEX + AdaBoost	$n\_estimators = 1$	0.639
		$q = 0.81$	
		$b = 0.14$	
		$\epsilon = 0.003$	
Breast Cancer Wisconsin (Heart)	Stand-Alone AdaBoost	$n\_estimators = 1000$	0.962
	ChaosFEX + AdaBoost	$n\_estimators = 100$	0.952
		$q = 0.930$	
		$b = 0.490$	
		$\epsilon = 0.159$	
Statlog (Heart)	Stand-Alone AdaBoost	$n\_estimators = 10$	0.816
	ChaosFEX + AdaBoost	$n\_estimators = 50$	0.815
		$q = 0.08$	
		$b = 0.06$	
		$\epsilon = 0.17$	
Seeds	Stand-Alone AdaBoost	$n\_estimators = 500$	0.563
	ChaosFEX + AdaBoost	$n\_estimators = 10$	0.918
		$q = 0.020$	
		$b = 0.070$	
		$\epsilon = 0.238$	
FSDD	Stand-Alone AdaBoost	$n\_estimators = 50$	0.086
	ChaosFEX + AdaBoost	$n\_estimators = 50$	0.170
		$q = 0.340$	
		$b = 0.499$	
		$\epsilon = 0.178$	

Appendix C.0.4. Support Vector Machine

Following are the hyperparameters tuned for Support Vector Machine (SVM):

1. **C**: Controls the bias-variance trade-off of the algorithm, known as the *"Regularization Parameter"*. It is tuned from 0.1 to 100.0 with a step-size of 0.1

All remaining hyperparameters offered by scikit-learn are retained in their default forms. The results of hyperparameter tuning for Support Vector Machine are available in Table C.26.

Table C.26: ***Support Vector Machine (SVM)***: Tuned hyperparameters for all nine datasets. Only FSDD uses the 'linear' kernel. All remaining datasets, use the 'rbf' kernel. The performance metric used for the provided results is macro F1-score.

Dataset	Implementation	Tuned Hyperparameters	F1 Score
Iris	Stand-Alone SVM	$C = 4.4$	0.954
	ChaosFEX + SVM	$C = 38.7$	0.958
		$q = 0.21$	
		$b = 0.969$	
		$\epsilon = 0.13$	
Ionosphere	Stand-Alone SVM	$C = 2.6$	0.934
	ChaosFEX + SVM	$C = 3.4$	0.926
		$q = 0.21$	
		$b = 0.969$	
		$\epsilon = 0.37$	
Wine	Stand-Alone SVM	$C = 0.6$	0.975
	ChaosFEX + SVM	$C = 16.0$	0.958
		$q = 0.21$	
		$b = 0.969$	
		$\epsilon = 0.01$	
Bank Note Authentication	Stand-Alone SVM	$C = 1.3$	1.0
	ChaosFEX + SVM	$C = 16.8$	0.965
		$q = 0.080$	
		$b = 0.250$	
		$\epsilon = 0.233$	
Haberman's Survival	Stand-Alone SVM	$C = 19.8$	0.597
	ChaosFEX + SVM	$C = 15.7$	0.609
		$q = 0.81$	
		$b = 0.14$	
		$\epsilon = 0.003$	
Breast Cancer Wisconsin (Diagnostic)	Stand-Alone SVM	$C = 16.0$	0.981
	ChaosFEX + SVM	$C = 20.3$	0.948
		$q = 0.930$	
		$b = 0.490$	
		$\epsilon = 0.159$	
Statlog (Heart)	Stand-Alone SVM	$C = 0.2$	0.852
	ChaosFEX + SVM	$C = 2.5$	0.825
		$q = 0.08$	
		$b = 0.06$	
		$\epsilon = 0.17$	
Seeds	Stand-Alone SVM	$C = 0.9$	0.948
	ChaosFEX + SVM	$C = 2.4$	0.909
		$q = 0.020$	
		$b = 0.070$	
		$\epsilon = 0.238$	
FSDD	Stand-Alone SVM (linear kernel)	$C = 0.7$	0.952
	ChaosFEX + SVM (linear kernel)	$C = 6.1$	0.978
		$q = 0.340$	
		$b = 0.499$	
		$\epsilon = 0.178$	

Appendix C.0.5. k-Nearest Neighbors

Following are the hyperparameters tuned for k-Nearest Neighbors:

1. k: Defines the number of nearest training data samples to the testing data sample to be chosen. It is tuned from 1 to 6 with a step-size of 2.

All remaining hyperparameters offered by scikit-learn are retained in their default forms.The results of hyperparameter tuning for k-Nearest Neighbors are available in Table C.27.

Table C.27: ***k*-Nearest Neighbors (*k*-NN)**: Tuned hyperparameters for all nine datasets. The performance metric used for the provided results is macro F1-score.

Dataset	Implementation	Tuned Hyperparameters	F1 Score
Iris	Stand-Alone k-NN	$k = 5$	0.944
	ChaosFEX + k-NN	$k = 1$	0.936
		$q = 0.21$	
		$b = 0.969$	
		$\epsilon = 0.12$	
Ionosphere	Stand-Alone k-NN	$k = 1$	0.814
	ChaosFEX + k-NN	$k = 1$	0.886
		$q = 0.21$	
		$b = 0.969$	
		$\epsilon = 0.11$	
Wine	Stand-Alone k-NN	$k = 5$	0.957
	ChaosFEX + k-NN	$k = 1$	0.966
		$q = 0.21$	
		$b = 0.969$	
		$\epsilon = 0.10$	
Bank Note Authentication	Stand-Alone k-NN	$k = 5$	0.998
	ChaosFEX + k-NN	$k = 3$	0.967
		$q = 0.080$	
		$b = 0.250$	
		$\epsilon = 0.233$	
Haberman's Survival	Stand-Alone k-NN	$k = 1$	0.562
	ChaosFEX + k-NN	$k = 5$	0.659
		$q = 0.81$	
		$b = 0.14$	
		$\epsilon = 0.003$	
Breast Cancer Wisconsin (Diagnostic)	Stand-Alone k-NN	$k = 5$	0.964
	ChaosFEX + k-NN	$k = 1$	0.932
		$q = 0.930$	
		$b = 0.490$	
		$\epsilon = 0.159$	
Statlog (Heart)	Stand-Alone k-NN	$k = 5$	0.803
	ChaosFEX + k-NN	$k = 5$	0.791
		$q = 0.08$	
		$b = 0.06$	
		$\epsilon = 0.17$	
Seeds	Stand-Alone k-NN	$k = 5$	0.930
	ChaosFEX + k-NN	$k = 1$	0.876
		$q = 0.020$	
		$b = 0.070$	
		$\epsilon = 0.238$	
FSDD	Stand-Alone k-NN	$k = 1$	0.834
	ChaosFEX + k-NN	$k = 1$	0.909
		$q = 0.340$	
		$b = 0.499$	
		$\epsilon = 0.178$	

Appendix C.0.6. ChaosNet

1. q: Initial Neural Activity, it is varied from 0.01 to 0.49 with a step-size of 0.01.

2. b: Discrimination Threshold, it is varied from 0.01 to 0.49 with a step-size of 0.01.
3. ϵ: Noise Intensity, it is varied from 0.001 to 0.499 with a step-size of 0.001.

Appendix C.1. Results

Appendix C.1.1. Decision Tree

The results of all experiments using Decision Tree in the high training sample regime are shown in Table C.28.

Table C.28: ***Decision Tree***: Experimental results (test data macro F1-scores) for all nine datasets in the high training sample regime.

Dataset	Implementation	F1 Score (Test Data)
Iris	Stand-Alone Decision Tree	0.967
	ChaosFEX + Decision Tree	1.0
Ionosphere	Stand-Alone Decision Tree	0.881
	ChaosFEX + Decision Tree	0.891
Wine	Stand-Alone Decision Tree	0.904
	ChaosFEX + Decision Tree	0.822
Bank Note Authentication	Stand-Alone Decision Tree	0.933
	ChaosFEX + Decision Tree	0.978
Haberman's Survival	Stand-Alone Decision Tree	0.516
	ChaosFEX + Decision Tree	0.482
Breast Cancer Wisconsin (Diagnostic)	Stand-Alone Decision Tree	0.696
	ChaosFEX + Decision Tree	0.882
Statlog (Heart)	Stand-Alone Decision Tree	0.697
	ChaosFEX + Decision Tree	0.878
Seeds	Stand-Alone Decision Tree	0.880
	ChaosFEX + Decision Tree	0.880
FSDD	Stand-Alone Decision Tree	0.603
	ChaosFEX + Decision Tree	0.579

Appendix C.1.2. Random Forest

The results of all experiments using Random Forest in the high training sample regime are shown in Table C.29.

Table C.29: ***Random Forest***: Experimental results (test data macro F1-scores) for all nine datasets in the high training sample regime.

Dataset	Implementation	F1 Score (Test Data)
Iris	Stand-Alone Random Forest	1.0
	ChaosFEX + Random Forest	1.0
Ionosphere	Stand-Alone Random Forest	0.909
	ChaosFEX + Random Forest	0.924
Wine	Stand-Alone Random Forest	0.966
	ChaosFEX + Random Forest	0.943
Bank Note Authentication	Stand-Alone Random Forest	0.974
	ChaosFEX + Random Forest	0.978
Haberman's Survival	Stand-Alone Random Forest	0.560
	ChaosFEX + Random Forest	0.398
Breast Cancer Wisconsin (Diagnostic)	Stand-Alone Random Forest	0.919
	ChaosFEX + Random Forest	0.918
Statlog (Heart)	Stand-Alone Random Forest	0.838
	ChaosFEX + Random Forest	0.838
Seeds	Stand-Alone Random Forest	0.877
	ChaosFEX + Random Forest	0.828
FSDD	Stand-Alone Random Forest	0.970
	ChaosFEX + Random Forest	0.937

Appendix C.1.3. Adaptive Boosting (AdaBoost)

The results of all experiments using AdaBoost in the high training sample regime are shown in Table C.30.

Appendix C.1.4. Support Vector Machine (SVM)

The results of all experiments using Support Vector Machine (SVM) in the high training sample regime are shown in Table C.31.

Table C.30: ***AdaBoost***: Experimental results (test data macro F1-scores) for all nine datasets in the high training sample regime.

Dataset	Implementation	F1 Score (Test Data)
Iris	Stand-Alone AdaBoost	0.967
	ChaosFEX + AdaBoost	0.865
Ionosphere	Stand-Alone AdaBoost	0.926
	ChaosFEX + AdaBoost	0.925
Wine	Stand-Alone AdaBoost	0.833
	ChaosFEX + AdaBoost	0.846
Bank Note Authentication	Stand-Alone AdaBoost	0.985
	ChaosFEX + AdaBoost	0.910
Haberman's Survival	Stand-Alone AdaBoost	0.505
	ChaosFEX + AdaBoost	0.609
Breast Cancer Wisconsin (Diagnostic)	Stand-Alone AdaBoost	0.858
	ChaosFEX + AdaBoost	0.881
Statlog (Heart)	Stand-Alone AdaBoost	0.777
	ChaosFEX + AdaBoost	0.717
Seeds	Stand-Alone AdaBoost	0.746
	ChaosFEX + AdaBoost	0.873
FSDD	Stand-Alone AdaBoost	0.080
	ChaosFEX + AdaBoost	0.397

Table C.31: ***Support Vector Machine (SVM)***: Experimental results (test data macro F1-scores) for all nine datasets in the high training sample regime.

Dataset	Implementation	F1 Score (Test Data)
Iris	Stand-Alone SVM	0.966
	ChaosFEX + SVM	0.933
Ionosphere	Stand-Alone SVM	0.924
	ChaosFEX + SVM	0.909
Wine	Stand-Alone SVM	0.928
	ChaosFEX + SVM	0.896
Bank Note Authentication	Stand-Alone SVM	0.993
	ChaosFEX + SVM	0.978
Haberman's Survival	Stand-Alone SVM	0.437
	ChaosFEX + SVM	0.447
Breast Cancer Wisconsin (Diagnostic)	Stand-Alone SVM	0.824
	ChaosFEX + SVM	0.918
Statlog (Heart)	Stand-Alone SVM	0.844
	ChaosFEX + SVM	0.801
Seeds	Stand-Alone SVM	0.924
	ChaosFEX + SVM	0.827
FSDD	Stand-Alone SVM	0.952
	ChaosFEX + SVM	0.978

Appendix C.1.5. k-Nearest Neighbors

The results of all experiments using k - Nearest Neighbors in the high training sample regime are shown in Table C.32.

Table C.32: ***k*-Nearest Neighbors (*k*-NN)**: Experimental results (test data macro F1-scores) for all nine datasets in the high training sample regime.

Dataset	Implementation	F1 Score (Test Data)
Iris	Stand-Alone k-NN	1.0
	ChaosFEX + k-NN	0.902
Ionosphere	Stand-Alone k-NN	0.821
	ChaosFEX + k-NN	0.939
Wine	Stand-Alone k-NN	0.943
	ChaosFEX + k-NN	0.873
Bank Note Authentication	Stand-Alone k-NN	0.993
	ChaosFEX + k-NN	0.971
Haberman's Survival	Stand-Alone k-NN	0.480
	ChaosFEX + k-NN	0.455
Breast Cancer Wisconsin (Diagnostic)	Stand-Alone k-NN	0.954
	ChaosFEX + k-NN	0.935
Statlog (Heart)	Stand-Alone k-NN	0.847
	ChaosFEX + k-NN	0.739
Seeds	Stand-Alone k-NN	0.924
	ChaosFEX + k-NN	0.854
FSDD	Stand-Alone k-NN	0.885
	ChaosFEX + k-NN	0.938

Appendix C.1.6. Gaussian Naive Bayes

The results of all experiments using Gaussian Naive Bayes (GNB) in the high training sample regime are shown in Table C.33.

Table C.33: ***Gaussian Naive Bayes (GNB)***: Experimental results (test data macro F1-scores) for all nine datasets in the high training sample regime.

Dataset	Implementation	F1 Score (Test Data)
Iris	Stand-Alone GNB	0.966
	ChaosFEX + GNB	0.933
Ionosphere	Stand-Alone GNB	0.862
	ChaosFEX + GNB	0.771
Wine	Stand-Alone GNB	1.0
	ChaosFEX + GNB	0.976
Bank Note Authentication	Stand-Alone GNB	0.785
	ChaosFEX + GNB	0.781
Haberman's Survival	Stand-Alone GNB	0.572
	ChaosFEX + GNB	0.535
Breast Cancer Wisconsin (Diagnostic)	Stand-Alone GNB	0.858
	ChaosFEX + GNB	0.812
Statlog (Heart)	Stand-Alone GNB	0.826
	ChaosFEX + GNB	0.788
Seeds	Stand-Alone GNB	0.846
	ChaosFEX + GNB	0.849
FSDD	Stand-Alone GNB	0.919
	ChaosFEX + GNB	0.687

References

[1] Harikrishnan Nellippallil Balakrishnan, Aditi Kathpalia, Snehanshu Saha, and Nithin Nagaraj. Chaosnet: A chaos based artificial neural network architecture for classification. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 29(11):113125, 2019.

[2] NB Harikrishnan and Nithin Nagaraj. Neurochaos inspired hybrid machine learning architecture for classification. In *2020 International Conference on Signal Processing and Communications (SPCOM)*, pages 1–5. IEEE, 2020.

[3] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[4] R. A. FISHER. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(2):179–188, 1936.

[5] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[6] Vincent G Sigillito, Simon P Wing, Larrie V Hutton, and Kile B Baker. Classification of radar returns from the ionosphere using neural networks. *Johns Hopkins APL Technical Digest*, 10(3):262–266, 1989.

[7] Michele Forina, Riccardo Leardi, Armanino C, and Sergio Lanteri. *PARVUS: An Extendable Package of Programs for Data Exploration*. 01 1998.

[8] Eugen Gillich and Volker Lohweg. Banknote authentication. 11 2010.

[9] Shelby J Haberman. The analysis of residuals in cross-classified tables. *Biometrics*, pages 205–220, 1973.

[10] W Nick Street, William H Wolberg, and Olvi L Mangasarian. Nuclear feature extraction for breast tumor diagnosis. In *Biomedical image processing and biomedical visualization*, volume 1905, pages 861–870. SPIE, 1993.

[11] Zohar Jackson, César Souza, Jason Flaks, Yuxin Pan, Hereman Nicolas, and Adhish Thite. Jakobovski/free-spoken-digit-dataset: v1.0.8, August 2018.

[12] NB Harikrishnan and Nithin Nagaraj. When noise meets chaos: Stochastic resonance in neurochaos learning. *Neural Networks*, 143:425–435, 2021.