Comparative Analysis of LSTM-FCN on Multiple Datasets

Shanza Akhtar, Munam Ali Shah

[1]*COMSATS University Islamabad, Islamabad, Pakistan*
[2]*Department of Computer Science, COMSATS University Islamabad, Islamabad, Pakistan*
shanzekhan75@gmail.com, mshah@comsats.edu.pk

Keywords: TIME SERIES CLASSIFICATION, LONG SHORT-TERM MEMORY, CONVOLUTION NEURAL NETWORKS

Abstract

Classification of time series data is a critical problem. With the growth of time series data, several algorithms have been proposed. The deep learning technique Long Short-Term Memory (LSTM) with Fully Convolutional Networks (FCN) is widely used for the classification of time series data. The use of LSTM-FCN to improve fully convolutional networks. Through attention mechanism visualisation of context, the vector is performed and enhances the results of time series classification. The aim of this research is to compare the results of LSTM-FCN output on a multiple dataset. The results show that the selected technique is more effective at classifying time series. Visualisation is given for the performance analysis of the LSTM-FCN technique on all datasets.

1 Introduction

Over the last 10 years, interest in time series classification has evolved rapidly. However, it remains challenging due to continuous updates in data, the high data dimensions, and the size of data. It's a supervised learning problem that classifies a series of data collected over time intervals and presents them in sequential order [1]. Deep learning techniques have been successfully applied to a variety of time series data applications, especially in control systems [2,3], time series data is prevalent [4], for the brightness classification of a target star, in medical discipline to identify heart diseases [5] or the recognition of individual actions [6, 7], and in the field of computing sciences for speech recognition [8, 9]. Various approaches have been proposed to manage time series classification. These techniques are split into three groups: based on the model, features, and distance-based. There was a lot of research conducted through features-based methods.

In this analysis study, the deep learning model is applied to different datasets to classify time series data. All datasets perform well and do not need heavy pre-processing. Time series data represents special data like weather forecasts, biological records, health supervision data, and stock prices. The most important concern is to extract useful and valuable knowledge from time-series data. In the time series classification of data, the aim is to predict class labels. Early-classification-on-time-series (ECTS) and minimum-prediction-length (MPL) [10], Word-extraction-for-Time series Classification (WEASEL) [11], bag-of-sfa-symbols (BOSS) [12], Multi-scale convolutional neural networks (MCNN) [13], Conv Time Net (CTN) [14], have obtained promising results in the field. MPL and ECTS is a classification method based on the 1-nearest neighbour. The early predictions were made by ECTS and comparable to the accuracy of the 1NN classifier, which uses a full-length time series. WEASEL introduces a sliding window approach for converting data into feature vectors. These vectors are used by machine learning algorithms to categorise time series. BOSS presents a grouping of histograms and a classifier based on distance. A time series' substructures are represented by histograms. A Fourier approximation is used to build substructures. MCNN is a single framework that includes feature extraction. All features are extracted at different frequencies and scales which leads to higher feature representation. TN technique smartly revamps to new time series classification of target tasks via a little fine-tuning process through labeled instances from the target tasks. BOSS VS extends the model to decrease time complexity to sustain productivity by using a vector space model.

All these techniques and classifiers need huge feature extraction and feature engineering. The proposed LSTM-FCN model is applied in sequence to all datasets and does not need large pre-processing techniques on feature crafting and raw data. The results of the model on all datasets achieve outstanding performance.

2 Background Approaches

In this section, we provide a background study around different approaches.

2.1 *RNN*

Recurrent Neural Networks, abbreviated as RNNs, are a class of neural networks that permit previous outputs as an input

while having hidden states. It accomplishes a memoryless input-output conversion. The state-space enables us to learn sequentially extended dependencies over a long period.

$$h_t = f(W_{xh}x_t + W_{hh}h_t\text{-}1)$$

$$y_t = g(W_{hy}h_t)$$

where $h_i$ts the hidden state vector and y_t is the output layer vector. f illustrates activation function for hidden state and g indicates activation function for the output layer. Why are the weights that connect hidden units to the output layer, W_{xh} is the weights that connect the inputs layer to hidden units. X_t is the vector of inputs.

2.2 *Temporal convolution*

Temporal convolution networks are the variation of convolutional neural networks mostly used for sequence modelling tasks. According to Lea et al[15], X is the time series input signal where $X_t \mathcal{E} R^f$ is the feature as the input of length of vector f for time step t which may differ for different time series sequences.

The temporal convolution network uses 1D convolutional network architecture in which a 1D filter is applied to each of the layers, the length of hidden layers is the same as input layers. The batch normalisation is followed by the convolutional layer [16] Afterwards, the rectified linear unit is trailed as an activation function [17].

2.3 *LSTM*

Long Short Term Memory (LSTM) is a special kind of recurrent neural network (RNN) having long-term information learning ability. This is the most popular RNN that alleviates the vanishing gradient problem. The vanishing gradient issue is encountered in recurrent neural networks Hochreiter et al [18], LSTMS vanishes the gradients and makes it slower than general RNN. LSTMS holds information in a gated cell that makes decisions about when to write and reads information and what to store. The analogs gates are implemented with element-wise multiplication by activation function sigmoid. More specifically, Graves et al. [19] explain the time for computation mathematically at step t as follows:

$$g^c = \tanh(W^c h_{t-1} + I^c x_t)$$
$$g^f = \sigma(W^f h_{t-1} + I^f x_t)$$
$$g^u = \sigma(W^u h_{t-1} + I^u x_t)$$
$$g^o = \sigma(W^o h_{t-1} + I^o x_t)$$
$$h_t = \tanh(g_o m_t)$$
$$m_t = g^f m_{t-1} + g^u g^c$$

where W^o, W^u, W^c, W^f are the weights matrices, the σ represents the logistic sigmoid, while the I^o, I^u, I^c , I^f are the projection matrices.

2.4 *Transfer learning*

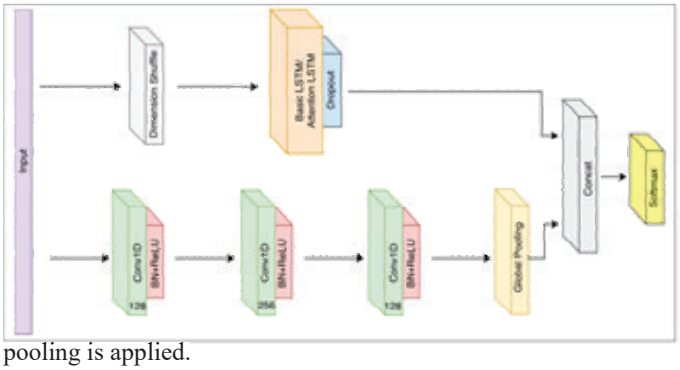
Transfer learning is a method that enables one to gain knowledge from the training model of one dataset and transfer it when training the model on another dataset. It takes features learned on one set of problems and uses them for a new set of a similar problem. The technique is usually applied for such datasets having small data to train a model from scratch [20].

3 LSTM Fully Convolutional Network

In this section, we briefly explain the framework of LSTM-FCN.

3.1 Framework

The temporal convolutional network is one of the famous and successful models for issues relating to time series classification [21]. Fully Convolutional Networks, which are made up of temporal convolutions, are commonly used for feature extraction. The global pooling computes the mean value for the feature map and provides it to SoftMax function This decreases the model's initial parameters for classification.



pooling is applied.

The input of time series data is passed into a layer called the dimension shuffle layer, which transforms the time series and then is conveyed to the LSTM block. The LSTM block consists of layers whether it be the Attention LSTM layer or the LSTM layer and dropout of the layers. The outcome of the LSTM block and global average pooling layer is concatenated and passes to the classification layer where SoftMax is applied.

3.2 Input

The time-series input moved into the LSTM block and fully convolutional block at the same time in various ways. In the selected model the LSTM block views the input in the form of multivariate time series which have a single time step. The

(Table 1 Accuracy result of all datasets)

SN	Dataset Name	Type	Sequence length	Classes	Training time (in hours)	Accuracy
1.	Melbourne Pedestrian	Traffic	24	10	8	97.08
2.	BME	Simulated	128	3	3	91.33
3.	Chinatown	Traffic	24	2	6	98.25
4.	Atrial Fibrillation	ECG	1279	3	8	26.66
5.	Electric Device Detection	Device	256	2	8	34.18

LSTM block takes input from the dimension shuffle layer which accomplishes this by taking transposes of the dimension of time series input. The fully convolutional block receives the time series data as a univariate time series having multiple time steps. If N is the length of the time series, there will be N time steps to receive data by a full convolution block.

This section provides the visual representation of the "Melbourne Pedestrian, BME, Chinatown, Crop, Atrial Fibrillation, Electric Device Detection" datasets of the Attention LSTM cell. All the classes in the dataset have the same weight at the point where all sequences of classes are squeezed together. Attention LSTM can identify the classes correctly at the point where all sequences are squeezed. Visual analysis of the actual time series backs up this argument. All classes can be distinguished from each other at the point where these all points are squeezed. Refinement improves the overall accuracy of the Attention LSTM Model on all the datasets.

The advantage of refining LSTM is that it requires much less pre-processing on the datasets. The downside of refinement is that it takes longer to train the dataset than a standard LSTM model because with the refinement we train the model with small batch sizes. Some datasets take six to eight hours while others take 10 to 13 hours of training. Despite the time consumption it also takes more computational power than the normal LSTM model.

3.3 Refinement

The procedure for the training model is divided into two phases. In the initial step, the best hyperparameter is selected for the model for a given dataset. The batch size is 128. In the second phase refinement method is applied to the initial trained model with new hyperparameters. During each refinement process, the transfer learning procedure is repeated. Model weight is initialised to each repetition and the learning rate is reduced to halved. In the refinement process, the learning rate is set to 1e-4 and the batch size is set to 32. The refinement process effectively increases the performance of the pre-trained model.

3.3 Algorithm

Algorithm for Refinement

```
1: for j < L do
2: weightsOfmodel ←init_weightsOfmodel
3: Training (model, Start_lr,batchsize)
4: init_ weightsOfmodel ← weightsOfmodel
5: j←j+1
6 Start_lr ←learningRateUpdation(Start_lr,j)
7: batchsize←batchsizeUpdation(batchsize,j)
```

3.4 Results and analysis

4 Experiments

The ALSTM_FCN model trained on six datasets taken from UCR time series datasets [26]. The datasets include the training set, and the testing set is used for training of model and validation. The range of hyperparameter search is from 8 cells to 128 cells which is an optimal range for LSTM cells. The block of FCN was kept constant for all experiments. Most of the models are trained at 2000 epochs while some of them having large data are trained on 110 epochs. This may increase the dataset when the algorithm takes more time to converge. All datasets have the same batch size of 128. After LSTM layers a high dropout of 80 percent is used to prevent the overfitting. The class weighing scheme is used to handle class imbalance issues. King and zeng [27]. Keras library is being used to train the models [28], with TensorFlow as a backend [29].

All models for datasets have been trained via stochastic gradient descent-based Adam optimiser [30]. Initially, the

learning rate was 1e-3 which was a decrease to 1e-4. All kernels of the convolution layer are initialised by following the initialisation procedure presented by He et al [31]. The training loss repeated up to 100 epochs if no improvement. The model only updated the weights through training loss. On each epoch accuracy of the model is reported. No extra pre-processing has been done on all datasets except checking empty columns, missing values, and datatype conversion.

4.1 Melbourne pedestrian dataset

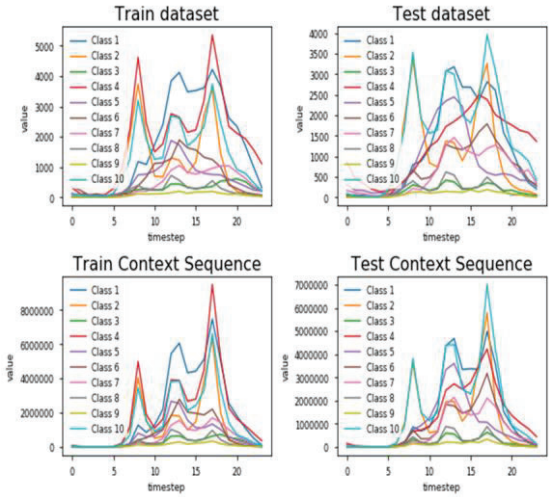


Figure 1 Context visualisation of Melbourne dataset

The dataset contains pedestrian data to understand pedestrian activities in different locations. The classes show the locations of the city and the time in which they visit. The above figure shows that there are 10 classes in the data, and the total sequence length is 24. Class 9, whose location is Tin Alley-Swanston St, are continuously declining while class 4, whose location is Flinders St Station, are increasing up to 17 then start decreasing. The data shows that all the multiple points are recorded at different timestamps. The classification helps in decision-making and better urban planning for the future.

4.2 BME Dataset

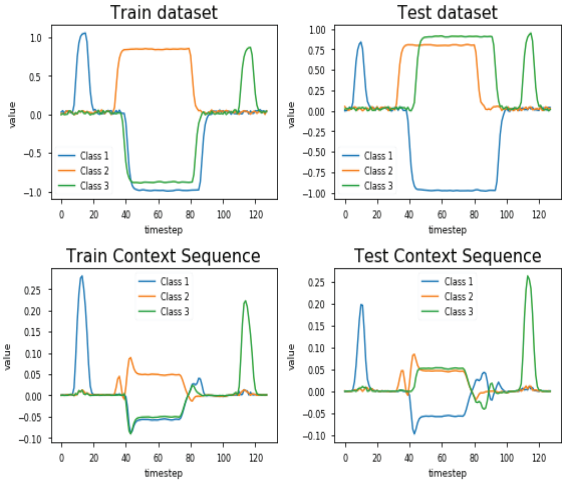


Figure 2 Context Visualisation of BME Dataset

The above figure represents the context visualisation of synthetic data having three classes and a sequence length of 128. Class 1 indicated by blue colour shows the beginning, class 2 (orange) in the middle, and class 3 (green) is the end. The data is positive in both the initial stage and end. However, the middle ranging from 40 to 80 may have negative values and positive values. The lines below 0.00 indicate the negative data which is only in the middle region.

4.3 Chinatown dataset

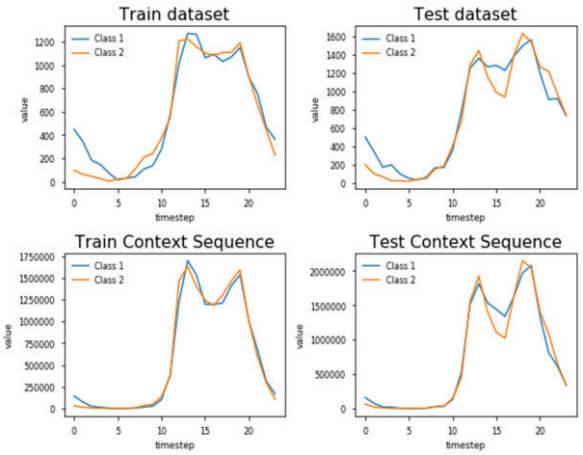


Figure 3 Context visualisation of Chinatown dataset

Chinatown dataset also contains pedestrian data like the Melbourne dataset. The sequence length is 24 and the number of classes is 2. The orange colour represents the weekend days while the other blue represents the weekdays. Most of the time, both classes show the same results except for some points which increase and decrease by both, which means that most of the time and locations are the same on both weekdays and weekends. The purpose is to classify the pedestrian data based on these two classes to make a better decision and adopt such a step for those locations where crowd increases at a specific time in those days.

4.4 Atrial fibrillation dataset

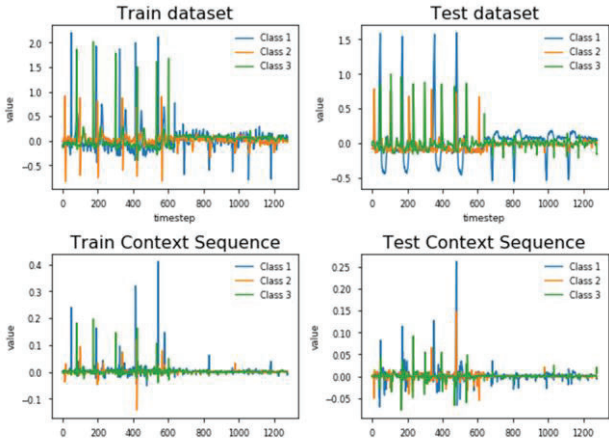


Figure 4 Context visualisation of atrial fibrillation

The above figure is the context visualisation of atrial fibrillation data. The data contains ECG recordings and created for the purpose to create such methods that predict the termination of atrial fibrillation. Class 1 is the non terminating atrial fibrillation, it takes some time to terminate when recording starts. Class 2 is the self-termination at a specific time, and lastly, class 3 is the termination immediately when the recording is finished.

4.5 Electric device detection dataset

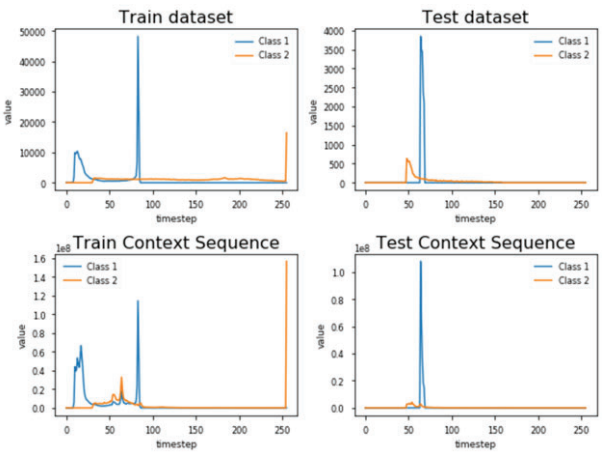


Figure 5 Context visualisation of EDD

The above figure shows that the dataset contains two classes and 256 sequence lengths. The data is about whether the electric device is present or not in the segments of 3D X-rays. Class 1 represents an electric device while Class 2 represents no device.

5 Conclusion

In this paper, we analyse the performance of the FCN model on different time series dataset taken from UCR time series classification. The results show that the FCN model performs outstandingly except for a few. The LSTM in the model effectively enhances the performance of the FCN model for the classification of time series. The results argued with context visualisation, class activation map, and final accuracy for all datasets. For future work, additional research can be done to investigate and enhance the low performance of some datasets through changes in the architecture of the model.

6 References

[1] E. A. Maharaj, P. D'Urso, and J. Caiado, Time Series Clustering, and Classification. Boca Raton, FL, USA: CRC Press, 2019.

[2] H. Wang, W. Sun, and P. X. Liu, ‘‘Adaptive intelligent control of nonaffine nonlinear time-delay systems with dynamic uncertainties,’’ IEEE Trans. Syst., Man, Cybern., Syst., vol. 47, no. 7, pp. 1474–1485, Jul. 2017

[3] X. Zhao, P. Shi, X. Zheng, and J. Zhang, ‘‘Intelligent tracking control for a class of uncertain high-order nonlinear systems,’’ IEEE Trans. Neural Netw. Learn. Syst., vol. 27, no. 9, pp. 1976–1982, Sep. 2016.

[4] L. Wei and E. Keogh, ‘‘Semi-supervised time series classification,’’ in Proc. 12th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, 2006, pp. 748–753.
[5] Wang S, Liu P, She MFH, et al. Bag of- word representation for biomedical time series classification. Biomedical Signal Processing and Control. 2013; 8(6):634-644
[6] Sadouk L, Gadi T, Essoufi EH. Convolutional neural networks for human activity recognition in the time and frequency domain. In: Proceedings of ICRTS. 2017. pp. 485-496
[7] Sadouk L, Gadi T, Essoufi EH. A novel deep learning approach for recognizing stereotypical motor movements within and across subjects on the autism spectrum disorder. Computational Intelligence and Neuroscience. 2018:16. DOI: 10.1155/ 2018/7186762. Article ID 7186762
[8] Abdel-Hamid O, Deng L, Yu D. Exploring convolutional neural network structures and optimization techniques for speech recognition. Interspeech. 2013;2013:1173-1175
[9] Abdel-Hamid O, Mohamed AR, Jiang H, Penn G. Applying convolutional neural networks concepts to hybrid nnhmm model for speech recognition. In: Proceedings of ICASSP IEEE. 2012. pp. 4277-4280
[10] Xing, Z., Pei, J., & Philip, S. Y. (2012). Early classification on time series. Knowledge and information systems, 31(1), 105-127.
[11] P. Schäfer and U. Leser. (2017). ‘‘Fast and accurate time series classification with WEASEL.’’ [Online]. Available: https://arxiv.org/ abs/1701.07681
[12] P. Schäfer, ‘‘The BOSS is concerned with time series classification in the presence of noise,’’ Data Mining Knowl. Discovery, vol. 29, no. 6, pp. 1505–1530, Nov. 2015.
[13] Cui, Zhicheng, Wenlin Chen, and Yixin Chen. "Multi-scale convolutional neural networks for time series classification." arXiv preprint arXiv:1603.06995 (2016).
[14] Kashiparekh, K., Narwariya, J., Malhotra, P., Vig, L., & Shroff, G. (2019, July). Continent: A pre-trained deep convolutional neural network for time series classification. In 2019 International Joint Conference on Neural Networks (IJCNN) (pp. 1-8). IEEE.
[15] C. Lea, R. Vidal, A. Reiter, and G. D. Hager, ``Temporal convolutional networks: A unified approach to action segmentation," in Computer Vision (Lecture Notes in Computer Science). Amsterdam, The Netherlands: Springer, 2016, pp. 4754.
[16] S. Ioffe and C. Szegedy, ``Batch normalization: Accelerating deep network training by reducing internal covariate shift," in Proc. 32nd Int. Conf. Int. Conf. Mach. Learn., 2015, pp. 448456.
[17] L. Trottier, P. Giguère, and B. Chaib-Draa. (May 2016). ``Parametric exponential linear unit for deep convolutional neural networks." [Online]. Available: https://arxiv.org/abs/1605.09332
[18] S. Hochreiter and J. Schmidhuber, ‘‘Long short-term memory,’’ Neural Comput., vol. 9, no. 8, pp. 1735–1780, 1997.
[19] A. Graves, Supervised Sequence Labelling With Recurrent Neural Networks, vol. 385. London, U.K.: Springer, 2012
[20] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, ‘‘How transferable are features in deep neural networks?’’ in Proc. Adv. Neural Inf. Process. Syst., 2014, pp. 3320–3328.
[21] Z. Wang, W. Yan, and T. Oates, ‘‘Time series classification from scratch with deep neural networks: A strong baseline,’’ in Proc. Int. Joint Conf. Neural Netw. (IJCNN), May 2017, pp. 1578–1585.
[22] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, ‘‘Dropout: A simple way to prevent neural networks from overfitting,’’ J. Mach. Learn. Res., vol. 15, no. 1, pp. 1929–1958, Jun. 2014.
[23] Z. Wang, W. Yan, and T. Oates, ‘‘Time series classification from scratch with deep neural networks: A strong baseline,’’ in Proc. Int. Joint Conf. Neural Netw. (IJCNN), May 2017, pp. 1578–1585.
[24] S. Ioffe and C. Szegedy, ‘‘Batch normalization: Accelerating deep network training by reducing internal covariate shift,’’ in Proc. Int. Conf. Mach. Learn., 2015, pp. 448–456.
[26]Y. Chen et al. (Jul. 2015). The UCR Time Series Classification Archive. [Online]. Available: http://www.cs.ucr.edu/~eamonn/time_series_data/
[27] G. King and L. Zeng, ``Logistic regression in rare events data," Political Anal., vol. 9, no. 2, pp. 137163, May 2001.
[28] F. Chollet et al. (2015). Keras. [Online]. Available: https://github.com/ fchollet/Keras
[29]M. Abadi et al.. (2015). TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. [Online]. Available: https://www.tensorow.org/
[30] D. P. Kingma and J. Ba. (2014). ``Adam: A method for stochastic optimization." [Online]. Available: https://arxiv.org/abs/1412.6980