Time Series Classification
ITEC5920 Final Project Report

Georges Ankenmann
School of Information Technology
Carleton University
Ottawa, Canada
GeorgesAnkenmann@cmail.carleton.ca

Davood Akhavannasab
School of Information Technology
Carleton University
Ottawa, Canada
DavoodAkhavannasab@cmail.carleton.ca

Abstract—In this project we are tasked to come up with various deep learning models that are to be trained on a provided dataset. This paper talks about our approach, our chosen models, and the results of training the models on the data.

Index Terms—deep learning, machine learning, time, series, classification, neural network, cnn, lstm, mlp, vgg, convolutional neural network, multilayer perceptron, long short-term memory, very deep convolutional networks, n-beats

I. Introduction

Nowadays, sensors can be seen in many devices such as smartphones, automobiles, and industrial instruments. Also, sensor information plays a critical role in a wide range of industries from health to safety. This information usually comes in a time-series format can be helpful in different fields such as disaster prevention, human health alerts, and air quality control by measurement interpretation. Some sensor information may contain noise (usually environmentally). We can find patterns by classifying the sensor information. The most common challenging issues of classification application are when we face imbalanced data and lack of training data. For example, if you consider a time series data set containing vibration data of a bridge, we will face a lack of earthquake information while it is a rare phenomenon. In several related works, CNN (Convolutional Neural Network), Multilayer Perceptrons (MLP), and LSTM (Long Short Term Memory network) have succeeded in time-series classification problems.

II. Research and Literature Review

This section contains literature for deep learning for time series classification.

The authors of article [13] have proposed a model for sensor classificatoin by encoding time series data into colored images (two-dimentional). Then they did image concatenation to image feature identity and then did image classification by Convolutional Neural Network (CNN). Angular Difference Field (GADF), Gramian Angular Summation Field (GASF), and Markov Transition Field (MTF) has been utilized for time series data transformation into images. In fact they have used these functions to prepare multivariate sensor data to obtain two-dimensional images. They have proved that their proposed model improved the classification results in comparison with other deep learning classification methods. They have used ECG and Wafer datasets in order to evaluate their model.

In another work ([14]), an outstanding model for network trafic classification has been proposed. Their work contains three tasks for predicting future network volume traffic, future packet protocol and future packet distribution through Recurrent Neural Network(RNN), Long Short Term Memory(LSTM) and Gated Recurrent Units(GRU). In the second task(future packet protocol prediction), they have focused on layer seven protocols such as SNMP, and HTTPS. As a matter of fact, they have considered some network packet features such as packet legnth, packet header information, and statistical information such as source IP and destination port as input and have considered packet protocol as output. They have evaluated their proposed models by GEANT and Abilene public datasets. Also, they have collected real networking data from network nodes by themselves. The results show that RNN and LSTM had amazing improvements in network traffic prediction in comparison with other proposed models such as Autoregressive Integrated Moving Average (ARIMA).

The authors of one work ([?]) have defined a DApp fingerprinting technique utilizing Graph Neural Networks (GNN) to identify different DApps by observing the user traffic stream. Their proposal consists of an information graph structure named Traffic Interaction Graph (TIG). TIG represents each traffic flow, comprising of packets resulting from client-server communications. The main advantage of TIG is that it shows much data like the original stream, such as packet length, packet direction, and packet order. They proposed a GNN-base classifier that extracts input TIG's features to recognize graph structures.

Several works have focused on online shopping prediction by deep learning. In one article([15]) RNN has been used on recommender systems, while these systems have one common issue to make decisions based on short session-based Information and usually do not track the session information based on customer user IDs. They

have utilized two datasets. The first one belongs to RecSys Challenge 2015, and the second one has been collected from an online service platform. They have eliminated 1-click sessions and combined the buyer's and clickers' Information to prepare the dataset. They have tried to focus on classifying user intent to make a purchase. Their results show that their work has remarkably improvemened recommender systems by applying RNNs. The authors of [16] have utilized RNN for customer behaviour prediction using clickstream information and the have proved RNN has better results than Logistic Regression. Also, they have admitted that the only disadvantage of RNNs is longer training time, but they mentioned the point that in RNNs, we do not need much feature engineering. The authors of other related work ([17]) have utilized LSTM RNNs(emphasizing single RNNs architectures) for shopping behaviour prediction using clickstream information.

Several researchers such as [18] and [19] have utilized Neural Networks to optimize inventory management systems through sales prediction. Suppose companies know about the number of their future sales. In that case, they can optimize their inventory management system. In first paper ([18]) MPNN(Multilayer Perceptron Neural Network) has been implemented to predict future sales to anticipate the required space of the company warehouse in the future. They have used sales information of a company for one year to predict future sales and see how many warehouses should be constructed in the future. They have admitted that MPNN does not work precisely in large time-series datasets for behaviour prediction. However, still, MPNN outperforms traditional machine learning methods in small datasets for non-linear behaviour prediction. The have utilized RMSE in model evaluation. In second work ([19]) LSTM and ARIMA have been utilized to predict grocery sales and then optimize the inventory system to reduce the number of waste foods. They have trained the models through sales information of a chain grocery company from 2013 to 2017. They have proved that LSTM has better results in comparison with ARIMA in predicting the coming seven days of given data. RMSE and MAE have been used for model evaluation.

III. Methodology

We are given a dataset that contains a matrix of 336 measurements with each measurement contains 3000 data samples. With this data, we must train a deep learning model that will allow for automatic classification of new data. Data is already standardised, so we do not need to reduce it further. We will then proceed with training and comparing various models, notably Multilayer Perceptrons (MLP), Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM), and a hybrid of models. We want to have multiple models to choose from as one model might be able to more accurately classify new sensor data over another given the available dataset. Once we have a trained model, we will want to validate the model by feeding it existing data to confirm the expected results, making adjustments accordingly. Once we have confirmed that the models are trained correctly, we will be feeding it new data and provide results.

IV. Results

See table I for the results of the data without augmentation. See table II for the results of the data with augmentation.

V. Discussion

A. Source Code

All the source code for this project is included in the submission and it is also available on the project's GitHub repository [7].

B. Data

The data was provided to us for this project. It consisted of sensor measurements from 336 sensors, with 3000 measurements for each sensor. We initially identified that the data was unbalanced and proceeded to fix this with random over sampling. The Keras Python libraries [8] were used for most of the project. Tensorflow and other libraries were used to augment the Keras libraries. Early stoppage was implemented for all models with idendical stoppage properties to allow for consistent training. The Root Mean Square Error (RMSE) values were monitored and training was stopped if after 15 epochs there was no significant decrease in the RMSE values. According to our research, a "better" model has a lower RMSE value. Table I and table II contains the min, max, average, and mean, of the tracked metrics. This data in this table will be discussed shortly in the next sections. A Graphical Processor Unit (GPU) was used as much as possible to speed up the training of the various models.

C. Data Augmentation and Unbalanced Data

Upon feedback from the presentation from the Professor and other students, we proceeded to augment the data using an open source Python library called Tsaug [3] and used the SMOTEENN algorithm [4] to fix the unbalanced data and the data augmentation issues noted during the presentation. The SMOTEENN algorithm uses a combination of over sampling and under sampling techniques. It uses the Synthetic Minority Over-sampling Technique (SMOTE) [5] for over sampling, and Edited Nearest Neighbours (ENN) for under sampling [6]. After using the two techniques to fix the unbalanced data and using the Tsaug library for data augmentation, we proceeded to train the models on the provided dataset.

D. CNN Model

The first model we designed was a Convolutional Neural Network (CNN) based model. Upon playing around with the parameters and mixing and matching layers, we found that a 12 layer model with 10 hidden layers offered the best results for our dataset. The hidden layers consisted

of alternating Keras Conv1D layers and Keras MaxPooling1D layers with decreasing units. The output layer is a single Keras Dense layer with one single unit.

E. MLP Model

The second model we designed was a Multilayer Perceptron (MLP) based model. Upon playing around with the parameters and mixing and matching layers, we found that a 12 layer model with 10 hidden layers offered decent performance. All of the layers consisted of the Keras Dense layers with a decreasing amount of units ("neurons") for each layer. The output layer is a single Keras Dense layer with one single unit.

F. LTSM Model

The third model we designed was a Long short-term memory (LSTM) based network. LSTM is a type of a Recurrent Neural Network (RNN). Similar to the previous models, we found that a 12 layer model with 10 hidden layers offers decent performance. The hidden layers consisted of one Keras LSTM layer with 50 units which then feed into 10 Keras Dense layers with 32 units each. The output layer is a single Keras Dense layer with one single unit.

G. N-BEATS Model

Part of this project, we were tasked with essentially making or finding a "better" model. One of the first ways we tried to make a better model was using the best found standard model (LSTM model for our project) and stacking a bunch of layers hoping that the results would improve. We also tried to change the activation functions along with changing how many units are in each layer. Upon further research it seems that stacking a bunch in a RNN based model seems to be the right approach according to the Neural Basis Expansion Analysis for Time Series (N-BEATS) [9] approach. N-BEATS is a RNN style approach specifically designed for time series classification with a minimum of 256 hidden layers. A Python N-BEATS implementation was used for this project to gather results [10].

H. Data Augmentation

After some feedback from the presentation, it was brought to our attention that we should be also augmenting the data as the dataset is slightly smaller than desired. The Tsaug [3] library was used to augment the data in this project. The Tsaug library provides multiple different types of augmenters for time series datasets. One of the augmenters used is a reverse augmenter, basically it reverses the time line of series. Another augmenter used is the add noise augmenter, basically it adds noise to every point of a time series and is independent and identically distributed. The third augmenter used in this project is the drift augmenter. This third augmenter drifts the value of time series from its original values randomly and smoothly.

I. Results

1) Without data Augmentation: Table I contains the results after training the various models on the data provided using two different algorithms to solve the perceived unbalanced data issue. Out of all the models we tried, the LSTM model performed the best with an average accuracy score of about 68% with Random Over Sampling while using the SMOTEENN method to solve the unbalanced data problem it provided an accuracy of about 80%. For both methods of dealing with unblanaced data, the LSTM model performed best out of all the models tested. One notable fact is that when using a non augmented dataset, the N-BEATS performed similar to a CNN and MLP based model. The fact that the LSTM model performed best with our data does conincide with our research that RNN based models, such as LSTM, perform best for time series classification [12].

2) With data Augmentation: Table II contains the results after training the various models on the data provided using two different algorithms to solve the perceived unbalanced data issue while also using the Tsaug data augmentation. After using the Tsaug data augmentation technique, it seems to have impacted all models and produced worst results when comparing to the previous training performed on non augmented data. In this part, all models produced almost identical results across all metrics, there is no clear winner for which model is better when you augment the data with the Tsaug libraries. When using the Random Over Sampling method to deal with the unbalanced data, the CNN and N-BEATS produced slightly better results. When using the SMOTEENN method to deal with the unbalanced data, the N-BEATS model is the best, and the LSTM model is close to optimal.

VI. Contribution

Both group members contributed equally to this project. We leveraged each others strengths to allow for an effective and concentrated group effort. The work for the models were equally distributed.

References

[1] A. Shrestha and . J. Dang, "Deep Learning-Based Real-Time Auto Classification of Smartphone Measured Bridge Vibration Data," 2020.

[2] D. P. Francis, M. Laustsen and H. Babamoradi, "Classification of colorimetric sensor data using time series," 2021.

[3] M. A. Jethva, "Tsaug: An open-source python package for time series augmentation," Medium, 22-Nov-2020. [Online]. Available: https://mineshj1291.medium.com/tsaug-an-open-source-python-package-for-time-series-augmentation-85ab98 [Accessed: 14-Apr-2022].

[4] "Smoteenn," SMOTEENN - Version 0.9.0. [Online]. Available: https://imbalanced-learn.org/stable/references/generated/imblearn.combine.SMOTEENN.html. [Accessed: 14-Apr-2022].

[5] N. V. Chawla, K. W. Bowyer, L. O.Hall, W. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," Journal of artificial intelligence research, 321-357, 2002.

[6] G. Batista, R. C. Prati, M. C. Monard. "A study of the behavior of several methods for balancing machine learning training data," ACM Sigkdd Explorations Newsletter 6 (1), 20-29, 2004.

[7] G. Ankenmann and D. Akhavannasab, ITEC5920 Project GitHub Repository. [Online]. Available: https://github.com/devagent42/ITEC5902-Project. [Accessed: 15-Apr-2022].
[8] Keras, "Keras. Simple. flexible. powerful.," Keras. [Online]. Available: https://keras.io/. [Accessed: 15-Apr-2022].
[9] B. N. Oreshkin, D. Carpov, N. Chapados, and Y. Bengio, "N-beats: Neural basis expansion analysis for interpretable time series forecasting," arXiv.org, 20-Feb-2020. [Online]. Available: https://arxiv.org/abs/1905.10437. [Accessed: 15-Apr-2022].
[10] P. Remy, Philipperemy/N-Beats: Keras/pytorch implementation of N-beats: Neural basis expansion analysis for interpretable time series forecasting. [Online]. Available: https://github.com/philipperemy/n-beats. [Accessed: 15-Apr-2022].
[11] J. Brownlee, "LSTMs for human activity recognition time series classification," Machine Learning Mastery, 27-Aug-2020. [Online]. Available: https://machinelearningmastery.com/how-to-develop-rnn-models-for-human-activity-recognition-time-series-classification/. [Accessed: 15-Apr-2022].
[12] M. D. Pra, "Time series forecasting with deep learning and attention mechanism," Medium, 05-Nov-2020. [Online]. Available: https://towardsdatascience.com/time-series-forecasting-with-deep-learning-and-attention-mechanism-2d001fc871fc. [Accessed: 15-Apr-2022].
[13] Yang, Chao-Lung and Chen, Zhi-Xuan and Yang, Chen-Yi, Sensor classification using convolutional neural network by encoding multivariate time series as two-dimensional colored images, vol. 20. MDPI, 2019, pp.168.
[14] Ramakrishnan, Nipun and Soni, Tarun, Network traffic prediction using recurrent neural networks, 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA), 2018, pp.187-193.
[15] Hidasi, Balázs and Karatzoglou, Alexandros and Baltrunas, Linas and Tikk, Domonkos, Session-based recommendations with recurrent neural networks, arXiv preprint arXiv:1511.06939, 2015
[16] Lang, Tobias and Rettenmeier, Matthias, Understanding consumer behavior with recurrent neural networks, Workshop on Machine Learning Methods for Recommender Systems, 2017
[17] Toth, Arthur and Tan, Louis and Di Fabbrizio, Giuseppe and Datta, Ankur, Predicting shopping behavior with mixture of RNNs, eCOM@ SIGIR, 2017
[18] Croda, Rosa María Cantón and Romero, Damián Emilio Gibaja and Morales, Santiago-Omar Caballero, Sales prediction through neural networks for a small dataset, Vol. 5, NO. 4, pp. 35-41, UNIR-Universidad Internacional de La Rioja, 2019
[19] Elmasdotter, Ajla and Nyströmer, Carl, A comparative study between LSTM and ARIMA for sales forecasting in retail, 2018
[20] Shrestha, Ashish and Dang, Ji, Deep learning-based real-time auto classification of smartphone measured bridge vibration data,Vol. 20, NO. 9, Multidisciplinary Digital Publishing Institute, 2020

VII. Appendix

Data Table of the results without data augmentation																								
Random Over Sampling																								
	Loss				MSE				Accuracy				RMSE				Validation Loss				Validation Accuracy			
Model	Min	Max	Avg	Mean	Min	Max	Avg	Mean	Min	Max	Avg	Mean	Min	Max	Avg	Mean	Min	Max	Avg	Mean	Min	Max	Avg	Mean
CNN	0.250	0.392	0.255	0.255	0.250	0.395	0.255	0.255	0.450	0.516	0.495	0.495	0.500	0.630	0.505	0.505	0.249	0.300	0.254	0.254	0.457	0.543	0.493	0.493
MLP	0.250	0.364	0.256	0.256	0.250	0.365	0.256	0.256	0.461	0.522	0.495	0.495	0.500	0.605	0.506	0.506	0.248	0.315	0.255	0.255	0.457	0.543	0.491	0.491
LSTM	0.126	0.324	0.183	0.183	0.126	0.324	0.183	0.183	0.439	0.796	0.682	0.682	0.354	0.569	0.424	0.424	0.127	0.260	0.179	0.179	0.394	0.851	0.702	0.702
N-BEATS	0.249	0.355	0.254	0.254	0.249	0.335	0.254	0.254	0.446	0.527	0.495	0.495	0.499	0.581	0.504	0.504	0.248	0.273	0.253	0.253	0.456	0.545	0.484	0.484
SMOTEENN																								
	Loss				MSE				Accuracy				RMSE				Validation Loss				Validation Accuracy			
Model	Min	Max	Avg	Mean	Min	Max	Avg	Mean	Min	Max	Avg	Mean	Min	Max	Avg	Mean	Min	Max	Avg	Mean	Min	Max	Avg	Mean
CNN	0.248	0.502	0.257	0.257	0.248	0.502	0.257	0.257	0.456	0.539	0.498	0.498	0.498	0.711	0.506	0.498	0.250	0.400	0.258	0.258	0.484	0.516	0.498	0.498
MLP	0.257	0.554	0.282	0.282	0.257	0.554	0.282	0.282	0.460	0.564	0.491	0.491	0.508	0.564	0.529	0.529	0.258	0.505	0.277	0.277	0.484	0.515	0.495	0.495
LSTM	0.043	0.379	0.133	0.133	0.043	0.379	0.133	0.133	0.470	0.960	0.801	0.801	0.208	0.616	0.349	0.349	0.078	0.541	0.178	0.178	0.485	0.882	0.721	0.721
N-BEATS	0.249	0.406	0.256	0.256	0.249	0.406	0.256	0.256	0.430	0.531	0.505	0.505	0.499	0.639	0.505	0.505	0.250	0.267	0.252	0.252	0.484	0.516	0.509	0.509

TABLE I

Data table of the results from the models without data augmentation

Data Table of the results with tasug data augmentation																								
Random Over Sampling																								
	Loss				MSE				Accuracy				RMSE				Validation Loss				Validation Accuracy			
Model	Min	Max	Avg	Mean	Min	Max	Avg	Mean	Min	Max	Avg	Mean	Min	Max	Avg	Mean	Min	Max	Avg	Mean	Min	Max	Avg	Mean
CNN	0.250	0.300	0.253	0.253	0.250	0.300	0.253	0.253	0.436	0.514	0.497	0.497	0.500	0.549	0.503	0.503	0.249	0.254	0.254	0.476	0.456	0.544	0.476	0.476
MLP	0.250	0.367	0.258	0.258	0.250	0.367	0.258	0.258	0.442	0.514	0.487	0.487	0.500	0.606	0.508	0.508	0.248	0.294	0.255	0.255	0.456	0.544	0.485	0.485
LSTM	0.249	0.316	0.254	0.254	0.249	0.316	0.254	0.254	0.414	0.525	0.490	0.490	0.499	0.562	0.504	0.504	0.250	0.256	0.251	0.251	0.468	0.521	0.490	0.490
N-BEATS	0.250	0.325	0.254	0.254	0.250	0.325	0.254	0.254	0.437	0.536	0.491	0.491	0.500	0.571	0.504	0.504	0.250	0.257	0.251	0.251	0.479	0.521	0.490	0.490
SMOTEENN																								
	Loss				MSE				Accuracy				RMSE				Validation Loss				Validation Accuracy			
Model	Min	Max	Avg	Mean	Min	Max	Avg	Mean	Min	Max	Avg	Mean	Min	Max	Avg	Mean	Min	Max	Avg	Mean	Min	Max	Avg	Mean
CNN	0.243	0.424	0.255	0.255	0.243	0.424	0.255	0.255	0.517	0.603	0.578	0.578	0.492	0.647	0.504	0.504	0.244	0.416	0.253	0.253	0.530	0.600	0.576	0.576
MLP	0.243	0.348	0.255	0.255	0.243	0.348	0.255	0.255	0.347	0.603	0.572	0.572	0.491	0.597	0.504	0.504	0.244	0.278	0.249	0.249	0.530	0.600	0.581	0.581
LSTM	0.239	0.363	0.245	0.245	0.239	0.363	0.245	0.245	0.362	0.603	0.594	0.594	0.489	0.602	0.495	0.495	0.240	0.301	0.243	0.243	0.350	0.600	0.590	0.590
N-BEATS	0.239	0.387	0.245	0.245	0.239	0.387	0.245	0.245	0.397	0.603	0.599	0.599	0.488	0.632	0.495	0.495	0.240	0.339	0.243	0.243	0.400	0.500	0.596	0.596

TABLE II

Data table of the results from the models with tsaug data augmentation