Predicting Future Events in a Complex Time Series Using an Adaptive Deep LSTM Ensemble

Anthony Morast
South Dakota School of Mines and Technology
anthony.morast@mines.sdsmt.edu

Abstract

Deep learning methods are typically based on the assumption that a dataset's underlying data generating process does not change over time. However, for many important datasets these assumptions don't hold resulting in non-stationarity. Learning algorithms operating in non-stationary settings often produce unreliable and spurious results leading to poor understanding and forecasting. Two common remedies to this ailment are to i) train a combination of methods and perform online model selection and ii) retrain a model repeatedly throughout time. In this paper an Adaptive Deep LSTM Ensemble (ADLE) is introduced that takes advantage of both of the aforementioned methodologies. It is shown that ADLE performs competitively with state-of-the-art LSTM and ARIMA models.

1 Introduction

Since its inception, deep learning has proved wildly successful on many classification and regression tasks. However, deep learning methods are typically based on the assumption that the underlying data generating process does not change over time. This property is known as stationarity and, in many sufficiently complex time series, the assumption does not hold. Using non-stationary data in time series models often produces unreliable and spurious results and leads to poor understanding and forecasting.

Non-stationary time series data is characterized by changing statistical properties in its data generating function. Put another way, through time, the data making up the time series will be drawn from distributions parameterized by different statistics, e.g. normal distributions parameterized by different means (μ) and variances (σ^2). Machine learning algorithms typically have no mechanism to account for this change in statistical properties making learning in non-stationary environments difficult.

Two possibilities for effectively handling non-stationary data are to use a combination of different models and perform online model selection and to retrain a model repeatedly on either a finite window into the past or on all available data. When implementing state-of-the art models for non-stationary time series, such as the ARIMA model, one usually takes advantage of the latter. The proposed adaptive ensemble combines both of these strategies into a model for one-step ahead predictions of non-stationary time series data.

This work introduces an adaptive ensemble of deep (stacked) LSTM-RNNs to model a non-stationary time series. The ensemble trains a set of LSTM networks on different subsets of the available training data to be used for forecasting. The outputs of the networks are combined by a function taking into consideration the statistical properties of the data segment used to train a network and the current statistical properties of a sliding window over the data. As new data become available additional networks are added to the ensemble increasing its effectiveness over time and allowing the ensemble to adapt. It's shown that this strategy yields competitive results when compared to ARIMA models and single LSTMs.

32nd Conference on Neural Information Processing Systems (NIPS 2018), Montréal, Canada.

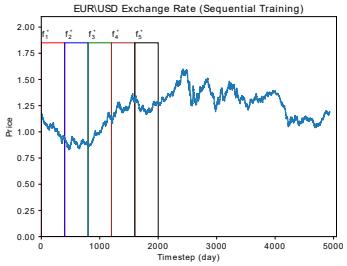

Figure 1: Segments of the training data when training the ensemble on sequential segments.

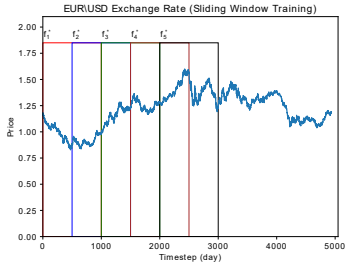

Figure 2: Segmentation of the training data with overlap on the various segments.

2 Related Work

One particularly challenging aspect of learning in non-stationary environments is that changes in non-stationary time series may be abrupt or gradual, random or systematic, and may even be cyclical. Much work has been done to handle these particular difficulties and typically include mechanisms to detect the changes, learn the changes, or forget past trends in the time series. Ensemble algorithms, in particular, typically try to update the weighting mechanisms of a fixed set of methods, use new data to update an ensemble of methods, and/or add new methods to the ensemble as training data become available. A review of strategies for building ensembles to operate in non-stationary environments is provided by Kuncheva [7].

In the past different variations of ensemble methods have been applied to learning non-stationary time series data, many of which focus on classification rather than regression. Heeswijk [12] uses an ensemble of extreme learning machines (ELMs) for prediction in both stationary and non-stationary environments. The ensemble adapts by adjusting the weights of the ensemble's aggregation methodology. In this ensemble, the non-stationarity is dealt with by creating the ELMs with various hyperparameters making in an attempt to find the hyperparameters that work best in the current environment. A major drawback of this ensemble is the need to retrain at each time step.

Polikar [9] introduces an ensemble for classification in non-stationary environments. The ensemble adds new classifiers as training data become available and, like Heeswijk [12], uses a weighting scheme based on each classifier's current performance. Blum [1] proposes an ensemble method focusing on different weighting schemes for the ensemble. Yet another approach at adaptation is the removal of ensemble methods once a change is detected (Chu [2]) or on other heuristics (Street [11]), keeping the ensemble a fixed size. Other ensembles rely heavily on training data for the preprocessing required to create the ensemble (Yu [13]) and combine aggregation methods to create a better classification methodology (Kotsiantis [6]).

The proposed method takes advantage of two of the three mechanisms typically used to deal with non-stationary data. An ensemble is created which uses a dynamic weighting mechanism based on certain statistical properties of the data. Then, as new data become available, new methods will be added to the ensemble to more accurately represent the changes in the data. The proposed method does not retrain any of its members with newly acquired data which, after initial training, makes use of the algorithm more efficient.

3 Method

In traditional neural networks it is assumed that inputs are independent of one another. **Recurrent neural networks** (RNNs) remove this restriction by allowing current computations to be dependent on input, output, or hidden layer computations from previous time steps, making them ideal for modeling sequential data. Because of this dependency, RNNs are particularly susceptible to problems with vanishing and exploding gradients as errors are back-propagated through time. The accepted solution to these gradient problems in RNNs is to build the network with **long short-term memory** (LSTM) cells which use input, output, and forget gates to determine which data is relevant to the network and cell state [4].

Because learning algorithms operating in non-stationary environments rely on up-to-date data, the model requires the ability to change with the environment, i.e. to adapt. This is usually done by retraining the model(s) as new data become available. This approach to adaptive learning is called **passive adaption** and typically requires frequently performing the lengthy training process consuming time and resources.

Ensemble methods improve machine learning results by aggregating the predictions of several base models which offers improved generalization and robustness over a single estimator. A particular kind of ensemble learning called **stacking** combines the predictions of several models trained on the available data with a combiner method, such as k-nearest neighbors or a neural network. Stacking ensemble methods have been successfully applied to many supervised and unsupervised machine learning tasks.

The proposed adaptive deep LSTM ensemble (ADLE), combines the prediction of an ensemble of LSTM recurrent neural networks. The networks are trained using one of two methodologies: i) sliding window wherein the network's training data is taken from a sliding window over the training data set and ii) sequential where the training data is broken into s segments of equal size and uses one segment per network. These methodologies are depicted in Figures 1 and 2.

The aggregation of the predictions is done via a distance weighted k-NN algorithm. Weights are assigned to each network's prediction based on the distance the network's statistical properties are from the current statistical properties of a sliding window over the dataset. The i^{th} network's weight is determined as

$$w_i = \frac{1}{d(x_q, x_i)^2}$$

where $d(x_q, x_i)$ is the distance function defined as $d(x_q, x_i) = \sqrt{(\mu_q - \mu_i)^2 + (\sigma_q^2 - \sigma_i^2)^2}$ where σ_q^2 and μ_q are the variance and mean of the sliding window and σ_i^2 and μ_i are the variance and mean of the i^{th} network. Then the final prediction is,

$$\hat{y} = \frac{\sum_{i=1}^{k} w_i \hat{y}_i}{\sum_{i=1}^{k} w_i}$$

where $\hat{y}_i$ is the prediction from the i^{th} network and k is the number of neighbors being considered. In this prototypical implementation of ADLE, k is equal to the number of networks in the ensemble. That is, each network's prediction is considered in the final prediction. Further work would include trying different values of k.

To adapt in changing non-stationary environments ADLE continuously adds new models to the ensemble as training data become available. As predictions are made with the ensemble, the time series values x_t and their target values y_t are stored in a list of historical points. As soon as the list contains enough feature/target pairs, $\{x_t, y_t\}$, to train a new LSTM network an LSTM is trained and added to the ensemble. This procedure provides the ensemble functional approximations of the most recent data as well as additional robustness as the new data segments may have statistical properties not yet seen by the ensemble's other networks.

Note that, in practice, training new ensembles will have little effect on the use of time and computational resources since data will become available gradually. Consider using the ensemble for one-step-ahead predictions on daily foreign exchange rate data. Then, if each ensemble method is trained on 500 examples, it would take 501 days until the list of historical points contains enough data to train a new network (501 rather than 500 since $y_t = x_{t+1}$). This means training a new ensemble would be rare and not take too heavy a toll on computational resources.

Essentially, ADLE attempts to approximate different functions, f_i^*, using different segments of the training data to try and capture the changes in statistical properties of the time series data. Each method is added to a 'dictionary' tying the statistical properties of the training segments to a network trained to approximate that segment, $\{\sigma_i, \mu_i : f_i^*\}$. The dictionary is then queried for predictions from networks trained over data with similar statistical structures as a sliding window over the most recent data. Each selected network's prediction is then weighted based on this distance.

The key benefits of ADLE over other ensemble methods are the train-once mentality, weights being determined instantaneously rather than being trained over time, and the use of strong learners *viz.* LSTM-RNNs. Many ensemble methods require constant retraining, some prior to every prediction,

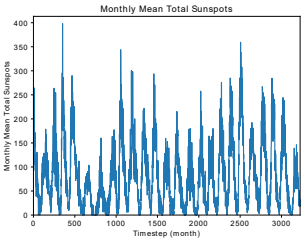

Figure 3: Monthly mean total sunspots between January 31, 1749 to August 31, 2017 dataset.

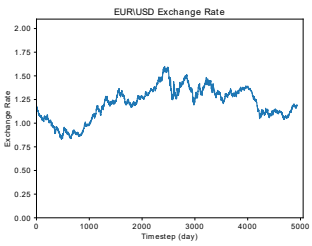

Figure 4: Daily closing prices of the EUR/USD foreign exchange rate.

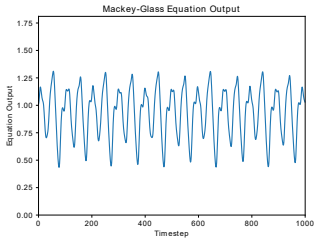

Figure 5: Mackey-Glass nonlinear time delay differential equation output.

which uses time and computational resources. The networks in ADLE are trained once on their segment of the data and the ensemble adapts by adding new methods to the ensemble. Considering the statistical properties of the time series as a weighting mechanism allows changes, gradual or abrupt, to be determined as quickly as they occur and, unlike many other ensembles, the methods that ADLE consists of are well-suited to learn very complex as well as very simple data generating functions.

4 Experiments

4.1 Datasets

Three datasets are used to determine the effectiveness of this ensemble model. The datasets have different statistical properties to demonstrate how the ensemble performs in different environments. The datasets selected for these experiments are monthly mean total sunspots, the euro to United States dollar foreign exchange rate daily closing price, and the Mackey-Glass nonlinear time delay differential equation's output.

The sunspot dataset is a univariate time series of the monthly average number of sunspots from January 31, 1749 to August 31, 2017 providing 3224 data samples. The dataset is taken from the Solar Influences Data Analysis Center which is the solar physics research department of the Royal Observatory of Belgium [3]. The dataset, shown in Figure 3, shows seasonality with large differences betweens seasons and also exhibits some signs of periodicity approximately every 11 years.

The EUR/USD foreign exchange rate dataset consists of the closing price of the exchange rate's ticker between January 1, 1999 and November 30, 2017 giving 4935 examples for training and testing. Predicting foreign exchange rates is an important task in finance and economics but, due to the complexity of the data, most methods perform poorly on out-of-sample data when compared to a simple random walk model. A plot of this dataset is shown in Figure 4.

The final dataset is the Mackey-Glass time-delay differential equation output introduced by Mackey and Glass [8]. The dataset is generated by the Mackey-Glass equation which produces a noisy non-periodic, non-convergent time series that serves as an example of deterministic chaos. This dataset is taken from the *frbs* package in R [10] which contains 1000 data points and is shown in Figure 5.

To study the ensemble method the dataset is segmented as outlined above after being split into train and test datasets. One-step ahead predictions are made with the trained ensemble and, as data become available, new methods are added to the ensemble. Error measures between predicted and actual values for ADLE, a single LSTM network, and a linear ARIMA model are compared to demonstrate the competitiveness of the model.

4.2 Results

The datasets described above were split into test and training datasets and used to train an ARIMA model, a single LSTM, and ADLE. To find an acceptable set of hyperparameters for the single LSTM an algorithm similar to population based training, introduced by Jaderberg [5], was used. To compare

	ARIMA		LSTM		ADLE	
	MAE	MSE	MAE	MSE	MAE	MSE
Sunspots	-0.343	639.594	19.909	786.056	19.515	727.079
EUR/USD Exchange Rate	$2.810e^{-5}$	$6.470e^{-5}$	0.007	$9.317e^{-5}$	0.0097	$1.690e^{-4}$
Mackey-Glass Equation	-0.1110	0.0634	0.0278	0.0011	0.0274	0.00107

Table 1: Mean squared error (MSE) and mean average error (MAE) of ARIMA, LSTM, and ADLE over the three datasets.

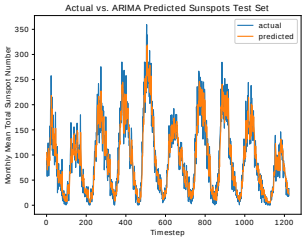

Figure 6: Predicted ARIMA values vs actual values for the sunspots dataset.

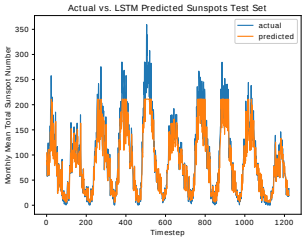

Figure 7: Predicted LSTM values vs actual values for the sunspots dataset.

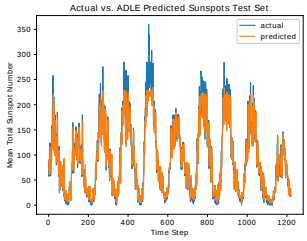

Figure 8: Predicted ADLE values vs actual values for the sunspots dataset.

performance of the three methods, predictions of the test set are made and mean squared error (MSE) and mean average error (MAE) are computed. These error metrics are defined as

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|\hat{y}_i - y_i|$$

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2.$$

For this initial implementation of ADLE k in the weighted k-NN aggregation is set as the number of networks in the ensemble. Tuning this hyperparameter may produce better results than those summarized in Table 1.

The results of the ensemble compared to a linear ARIMA model and a single LSTM model are summarized in Table 1. It is shown that in two out of three datasets ADLE performs better than a single LSTM and outperforms both of the other benchmarks on the Mackey-Glass dataset. On the datasets where ADLE underperforms the other methods the results remain competitive and, with more fine tuning, ADLE may have the capability of outperforming the other methods. Due to the nature of discrete time series data, such as the Sunspots and EUR/USD exchange rate datasets, ARIMA models can be expected to fit well for one-step ahead predictions (with retraining at every time step). This is because steps between points is actually a straight line so a good linear measure determining where the line is heading is bound to give good predictions.

4.2.1 Sunspots

Shown in Figure 6, Figure 7, and Figure 8 are the predicted values from ARIMA, LSTM and ADLE vs actual values of the sunspots dataset. It appears that the single LSTM and ADLE do a better job at predicting the mean total sunspots number during periods of 'typical' behavior but fail when attempting to predict extraordinarily high values for the number of sunspots. This indicates the models may be restricted in some way when attempting to predict these values which warrants further investigation.

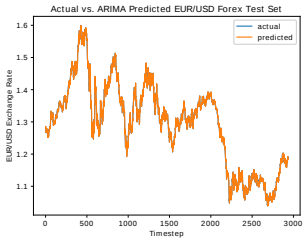

Figure 9: Predicted ARIMA values vs actual values for the EUR/USD exchange rate dataset.

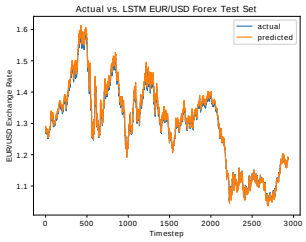

Figure 10: Predicted LSTM values vs actual values for the EUR/USD exchange rate dataset.

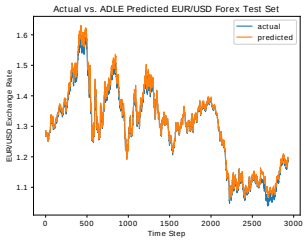

Figure 11: Predicted ADLE values vs actual values for the EUR/USD exchange rate dataset.

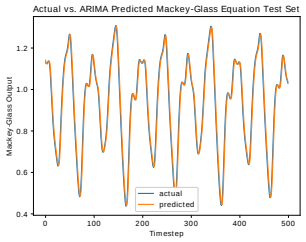

Figure 12: Predicted ARIMA values vs actual values for the Mackey-Glass equation output dataset.

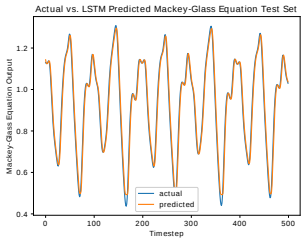

Figure 13: Predicted LSTM values vs actual values for the Mackey-Glass equation output dataset.

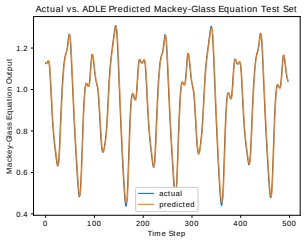

Figure 14: Predicted ADLE values vs actual values for the Mackey-Glass equation output dataset.

4.2.2 EUR/USD Exchange Rate

The predicted values compared to actual values of the EUR/USD foreign exchange rate dataset can be seen in Figure 9, Figure 10, and Figure 11 for the ARIMA, LSTM, and ADLE models, respectively. As was seen with the sunspots dataset in subsubsection 4.2.1, when attempting to predict values that are particularly low or high ADLE has issues meeting these datasets. In both cases for the EUR/USD dataset ADLE's predictions are higher than the expected values indicating too much weight may be placed on the methods predicting high values during these periods (since the initial implementation uses all networks for prediction).

4.2.3 Mackey-Glass Equation Output

Figure 12, Figure 13, and Figure 14 show the actual time series values plotted against the ARIMA, LSTM, and ADLE models, respectively. In these figures it's hard to distinguish between the actual and predicted values which indicates the dataset is likely easy to learn. This is expected based on the summarized results in Table 1 where it can be seen that all three models have approximately the same performance on this dataset.

5 Conclusions and Future Work

As shown in subsection 4.2, in this limited test ADLE outperforms or performs competitively when compared to standard methods on these datasets. The ensemble works best on the Mackey-Glass and Sunspots datasets which both show signs of periodicity and seasonality indicating that the method may outperform other models on data with similar trends. As with most things, there is no "one size fits all" method for time series forecasting but with further refinement it's believed ADLE is a good competitor for a particular type of dataset. One thing to note is the results above are preliminary due to the time constraints of the project. Given more time the hyperparameters of each ensemble could be tuned to potentially give better prediction accuracy.

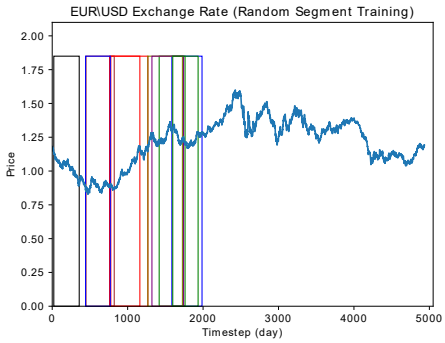

Figure 15: Training segments when start points and sizes are randomly generated.

Some interesting avenues for future work include testing the ensemble on multivariate time series, implementing another training methodology wherein random starting locations and sizes are chosen for each data segment in an attempt to capture the data's structural changes (Figure 15), a parallel implementation of the algorithm to decrease training times and improve the efficiency of the method, and the application of different aggregation functions, e.g. a neural network with, online training to weight each individual model's prediction at each time step.

In the current implementation of ADLE the ensemble consists of LSTM-RNNs as the base model. Other promising methods such as convolutional LSTMs, support vector machines, convolutional neural networks and deep belief networks, which have all been shown to be good predictors of time series data, could potentially boost the performance of ADLE. Additional updates to the training procedure could prove beneficial as well. For instance, in the current implementation each LSTM uses the same set of hyperparameters but in reality certain segments of data might be better approximated by networks with fewer or additional hidden layers and training epochs. That is, more linear segments may require shallow, narrow networks while complex segments might need deeper, wider networks to effectively approximate their data generating functions.

Perhaps the most promising future work is to implement a type of change detection to better determine when the underlying structure of the data changes. The goal of ADLE is to find models that best fit the data as statistical properties of the dataset change. This acts as a workaround to the stationarity assumption of machine learning methods. Determining when these changes occur could prove critical to the success of the method.

References

[1] Avrim Blum. Empirical support for winnow and weighted-majority based algorithms: results on a calendar scheduling domain. In *Machine Learning Proceedings 1995*, pages 64–72. Elsevier, 1995.

[2] Fang Chu and Carlo Zaniolo. Fast and light boosting for adaptive mining of data streams. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 282–292. Springer, 2004.

[3] Especuloide. Sunspots, Sep 2017. URL `https://www.kaggle.com/robervalt/sunspots`.

[4] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

[5] Max Jaderberg, Valentin Dalibard, Simon Osindero, Wojciech M. Czarnecki, Jeff Donahue, Ali Razavi, Oriol Vinyals, Tim Green, Iain Dunning, Karen Simonyan, Chrisantha Fernando, and Koray Kavukcuoglu. Population based training of neural networks. *CoRR*, abs/1711.09846, 2017. URL `http://arxiv.org/abs/1711.09846`.

[6] S Kotsiantis, Kiriakos Patriarcheas, and M Xenos. A combinational incremental ensemble of classifiers as a technique for predicting students' performance in distance education. *Knowledge-Based Systems*, 23(6):529–535, 2010.

[7] Ludmila I Kuncheva. Classifier ensembles for changing environments. In *International Workshop on Multiple Classifier Systems*, pages 1–15. Springer, 2004.

[8] Michael C Mackey and Leon Glass. Oscillation and chaos in physiological control systems. *Science*, 197(4300):287–289, 1977.

[9] Michael D Muhlbaier and Robi Polikar. An ensemble approach for incremental learning in nonstationary environments. In *International Workshop on Multiple Classifier Systems*, pages 490–500. Springer, 2007.

[10] Lala Septem Riza, Christoph Bergmeir, Francisco Herrera, and José Manuel Benítez. frbs: Fuzzy rule-based systems for classification and regression in R. *Journal of Statistical Software*, 65(6):1–30, 2015. URL `http://www.jstatsoft.org/v65/i06/`.

[11] W Nick Street and YongSeog Kim. A streaming ensemble algorithm (sea) for large-scale classification. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 377–382. ACM, 2001.

[12] Mark Van Heeswijk, Yoan Miche, Tiina Lindh-Knuutila, Peter AJ Hilbers, Timo Honkela, Erkki Oja, and Amaury Lendasse. Adaptive ensemble models of extreme learning machines for time series prediction. In *International Conference on Artificial Neural Networks*, pages 305–314. Springer, 2009.

[13] Lean Yu, Shouyang Wang, and Kin Keung Lai. Forecasting crude oil price with an emd-based neural network ensemble learning paradigm. *Energy Economics*, 30(5):2623–2635, 2008.