NTNU

Kunnskap for en bedre verden

DEPARTMENT OF COMPUTER SCIENCE

TDT4173 - ASSIGNMENT 1

Time Series Prediction

Long Short Term Memory Networks for predicting the future

Group:
Group 9

Authors:
Espen Løvhaugen (Espenblo)
Haakon Flaarønning (Haakotf)
Kristian Bondevik Hardang (Krisbha)

September 20, 2022

Abstract

In this paper, we argue for the effectiveness of using a long short term memory (LSTM) neural network for time series prediction. We discuss the LSTM network's fundamental concepts, what makes it good for time series forecasting, and touch on some frequently applied optimization techniques. In addition, we explain how the LSTM network may be improved by implementing a statistical method to aid in the forecasting process. We dive into using exponential smoothing as a pre-processing step to deal with seasonality in the data, and explain how it helps the LSTM network in the learning process. Then we discuss some recent studies that have been made in the field of time series prediction and display how LSTM networks have been applied to make profits and even save lives. Finally, we discuss some further work which should be made within the field of time series forecasting.

Table of Contents

List of Figures

1 Introduction

Time series forecasting is a technique in machine learning which aims to predict future events given a sequence of past events. The ability to predict the future gives a significant competitive advantage for all businesses across all industries. Time series forecasting is commonly used in most fields within science and engineering and can for example be used to predict future sales and earnings or the health-condition of key system components within a business. With the ability to achieve a significant competitive advantage, the question of what prediction model is best has become a popular field of discussion.

The first real application of statistical models for the purpose of time series prediction dates back to late 1920s [Zoubir, 2017]. At that time, simple linear methods were applied, and the moving average model was introduced as an effective way to remove fluctuations in time series data. In the late 1930s the Autoregressive Moving Average (ARIMA) model was discovered and frequently applied for time series forecasting. Statistical linear methods continued to produce the most accurate predictions all the way to the late 1980s and early 1990s when machine learning methods established themselves as a genuine contender to classical statistical models [Bontempi et al., 2013].

1.1 Machine learning in time series prediction

The Makridakis competition in 2018 marked a big breakthrough in time series prediction as for the first time in the history of the forecasting competition, the winning model was one based primarily on machine learning [Barker, 2020]. Prior to the competition, the general consensus was that statistical methods produce more accurate forecasts than those based on machine learning. However, the machine learning based model did not just win - it had a prediction accuracy that was more than 10.8% higher than that of the second place contestant.

The rest of this method paper will revolve around a machine learning based model for time series forecasting, namely prediction using an LSTM neural network. The strength of LSTM networks is its ability to capture long-term dependencies and to perform cross-learning, i.e using many series to train a single model. Still, like all neural networks, LSTM networks require well thought of preprocessing to forecast accurately. In section 3 of this paper, we propose using a statistical method to aid in the preprocessing step.

1.2 Taxonomy of machine learning models

Contextualizing time series forecasting in the overall machine learning taxonomy is difficult, as models for time series prediction can be based both on supervised and unsupervised learning. However, in this paper we discuss using LSTM neural networks for forecasting on time series data, which predominantly is based on supervised learning. Hence, we choose to place the model within the scope of supervised learning (1).

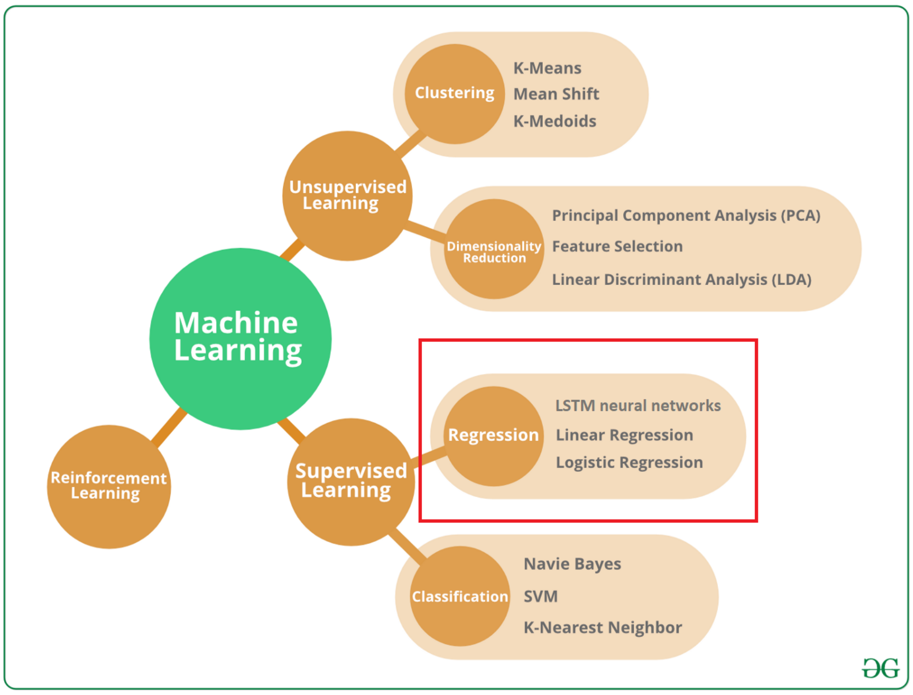


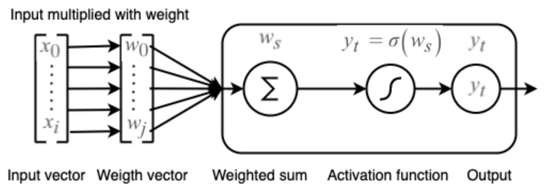
Figure 1: Taxonomy of machine learning models

2 Foundations

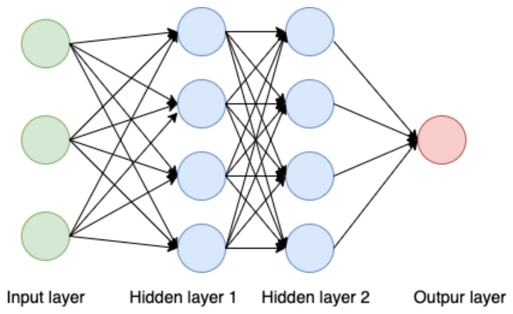
This section presents the core idea and theoretical foundations of LSTM networks for time series prediction. A brief introduction to artificial neural networks and recurrent neural networks is given to better explain why LSTM networks are a good fit for time series forecasting. Then, a detailed description of an LSTM neural network is presented, and finally some optimization techniques are discussed.

2.1 Artificial neural networks

Most machine learning models applied on time series data contain some form of an artificial neural network (ANN). An artificial neural network is a computational model which is heavily inspired by the human brain architecture. It is used to build intelligent agents through many iterations of learning. At its core, a neural network consists of layers of perceptrons which are simple units that pass weighted inputs through an activation function and outputs a single value. It can have different types of activation functions like step functions and linear or non-linear functions. The role of the activation function is to determine whether a neuron should be activated or not.[Sharma, 2017] A neural network has an input layer, an output layer and one or more hidden layers each of which contains one or more perceptrons. If the neural network has more than one hidden layer it is called a deep neural network, giving rise to the term deep learning. Neural networks are trained to detect and understand patterns in the data it receives, which is why it has proved so effective at predicting time series data. Figure 2a illustrates a simple perceptron and Figure 2b a neural network with two hidden layers.



(a) An artificial neuron (perceptron)

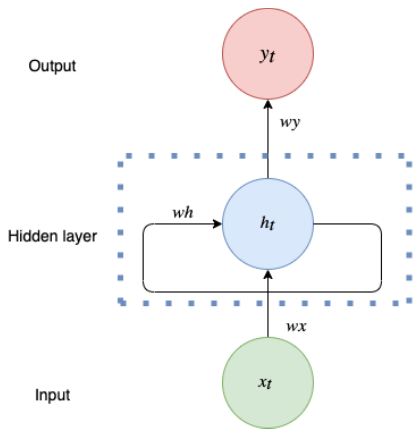


(b) A neural network with 2 hidden layers

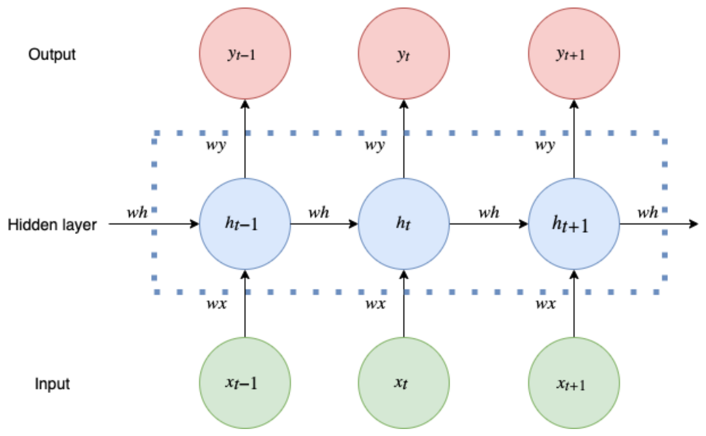
Figure 2: Overview of an artificial neural network

2.2 Recurrent neural network

A recurrent neural network (RNN) is a type of neural network that makes use of the sequential information that the input data may have [Liang and Bose, 1996]. Standard feed-forward neural networks assume that the inputs are independent and for many tasks that assumption holds true. However, for problems regarding sequential data, like time series, the input sequence contains a lot of information that must be captured to make accurate predictions. An RNN is a neural network where the hidden layers have connections back to themselves, through so-called loops. The loops make it possible to utilize historical information by passing information from an earlier time step in the calculation for the current time step. That way, the RNN is able to capture the underlying sequential information from the data. Figure 3b illustrates an RNN, where the loop is unfolded to better understand the nature of the network. At each time step t, we feed in the current input x_t along with the output at the previous time step, h_{t-1} which are combined to calculate the new output at time t. The w_h,w_y and w_x are weights for the respective recurrent input, output and input. The weights can be initialized different, but for sigmoid and tanh it is used $w = random(n)/sqrt(n)$ where n is the n number of neuron in each layer. This must be repeated in each layer.[Bianchi, 2017]



(a) A simple recurrent neural network



(b) An unfolded recurrent neural network

Figure 3: Recurrent neural networks

2.3 The vanishing and exploding gradient problem

Although RNNs are good at capturing the underlying sequential information, they still have problems with learning long-range dependencies in the data. This is because of the exploding and vanishing gradient problem. The problem occurs for the weights of the first layers in the neural network. During back-propagation with stochastic gradient descent, the weights in the network are updated proportionally to the gradient of the loss function with respect to the weights in the network. The chain rule is applied to obtain the network gradient by multiplying the gradient of each layer from the final layer down to the initial layer. If the gradients are small, the network gradient becomes a product of many small gradients [Flaarønning and Tuhus, 2020]. A very small gradient means an insignificant change to the network's weights, which in turn means that the change will not contribute much to minimizing the error. The network will therefore not train properly. The opposite can occur if the gradient evolves to become extremely large, and it is then called an exploding gradient. An LSTM network helps to solve these problems.

2.4 Long short-term memory

A long short term memory (LSTM) neural network is a specialized recurrent neural network that prevents the problem of vanishing and exploding gradients. It contains hidden LSTM layers which consist of specialized LSTM cells that are specifically designed to be able to capture long-term dependencies. Accurately capturing the level and magnitude of the trend and seasonal components of time series data is critically important in order to perform accurate forecasting.

2.5 Understanding the LSTM cell

An LSTM cell consists of 3 gates, namely an input gate, an output gate, and a forget gate. Each gate helps the network with learning long-term dependencies by determining what information to retain, and what information to forget. Additionally, the cell has an internal cell state, c_t, which acts as the cell's long-term memory, and is continuously updated as new data runs through the cell. Finally, the cell contains a hidden state, h_t, which acts as the short-term memory of the cell, and is simply the output from the cell at the previous time step.

The calculations performed in the cell consists of the four following steps [Olah, 2015], as illustrated in Figure 4:

The first step is to decide what information to throw away from the cell state, which is determined by the forget gate. The previous hidden state, h_{t-1}, and the new input x_t is multiplied by the weights of the forget gate, added with the bias of the forget gate, and passed through a sigmoid function (equation 1). The sigmoid function outputs a final value, f_t, between 0 and 1, where a value close to 0 represents forgetting the information, while a value close to 1 means retain the information.

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \tag{1}$$

The next step is to decide what new information to store in the cell state, which is determined by the input gate. A sigmoid layer, i_t, decide what values to update, while a hyperbolic tangent layer, C'_t, creates a vector of candidate values that could potentially be added to the new cell state. Finally, these values are multiplied together to obtain the new candidate values for the cell state scaled by how much the sigmoid layer decided to update each state value.

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \tag{2}$$

$$C'_t = \tanh(W_c[h_{t-1}, x_t] + b_f) \tag{3}$$

$$G_t = i_t * C'_t \tag{4}$$

Where W_i and b_i are the weights and bias of the input layer, and W_c and b_c are the weights and bias of the hyperbolic tangent (tanh) layer.

The third step is to update the cell state. This is simply done by multiplying the old state, C_{t-1}, with the values from the forget gate, f_t, hence forgetting the information that the forget gate deemed unimportant. Then we simply add the new candidate values G_t and we are left with the new cell state, C_t.

$$C_t = C_{t-1} * f_t + G_t \tag{5}$$

The last step is to decide what the cell should output, which is determined by the output gate. The cell's previous hidden state, h_{t-1} is passed alongside the current input, x_t, through a sigmoid function to decide what parts of the cell state that the cell should output, o_t. Then, the new cell state, C_t, is passed through a hyperbolic tangent function and multiplied by o_t to produce the output and new hidden state, h_t.

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{6}$$

$$h_t = o_t \tanh(c_t) \tag{7}$$

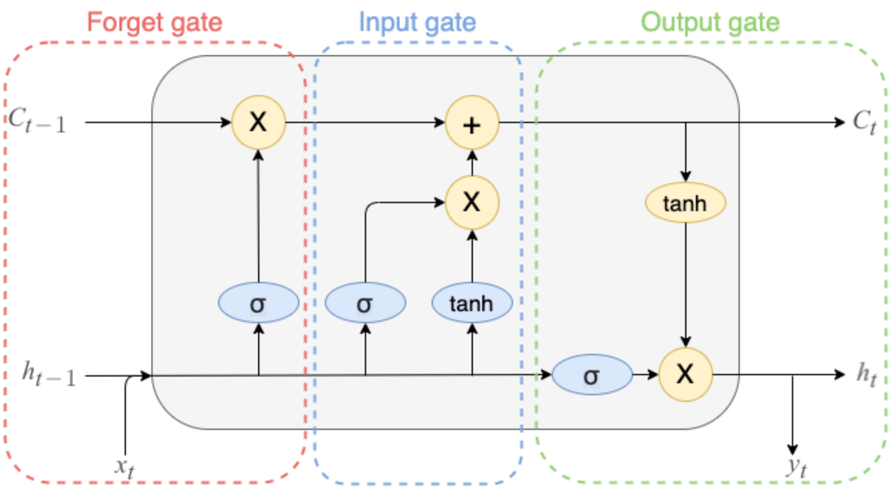


Figure 4: Overview of an LSTM cell

2.6 Addressing optimization

Like most neural networks, LSTM networks struggle with the problem of overfitting. The dilemma involves finding an appropriate balance between the accuracy of the prediction, and the generalization of the model. A model can perform very well on the dataset it is trained on, showing an error close to zero, however, when it is tested on data that it has not seen before, it performs much worse. In that case, the model represents the training data too well and fails to generalize when tested on unseen data. The issue is especially apparent when the LSTM network is trained on a small dataset.

Detecting overfitting in a model for time series prediction can be done by splitting the dataset into a test set and a training set. We make the model train on the training set for multiple iterations, called epochs. After each epoch, we allow the model to make predictions on the test set, and measure its accuracy. The most common evaluation indicators used for time series data are mean absolute percentage error and root mean squared error. If, after an epoch, the prediction accuracy on the test set decreases compared to the previous epoch, we know that the model is starting to overfit on the training data, and we can stop the training process.

Another much used regularization technique is to experiment with different network architectures. The combination of layers and number of units per layer can have a big impact on the performance of the model. The right sequence of layers must correspond to the amount of data available, and the amount of features that is taken into consideration by the model.

3 Improvements to the Method

Models for time series prediction are commonly categorized as either purely statistical or purely machine learning based. Figure 5 illustrates examples of both statistical and ML based models used for time series prediction. Both categories have their advantages and disadvantages, but historically speaking, statistical methods have proved to be more accurate for time series forecasting [Bontempi et al., 2013]. Recent studies, on the other hand, have shown that a hybrid approach combining both a statistical and a machine learning model performs better than a purely statistical or purely ML-based model [Barker, 2020]. In this section, we identify the shortcomings of using the pure machine learning model described above, namely the LSTM neural network. Then we introduce a couple of statistical methods for time series forecasting, and discuss their advantages and disadvantages. Finally, we propose a hybrid model combining an LSTM network with the use of a statistical method which addresses the shortcomings of each model individually.

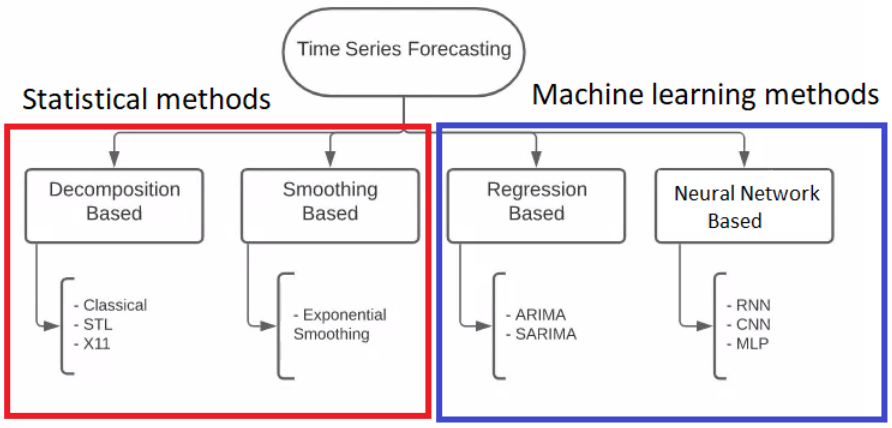
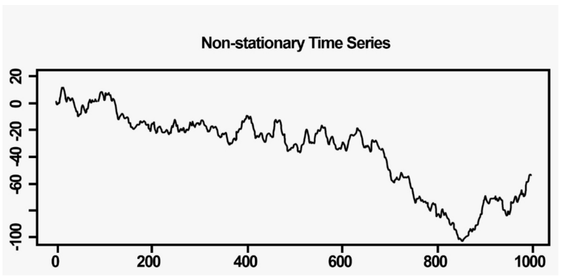


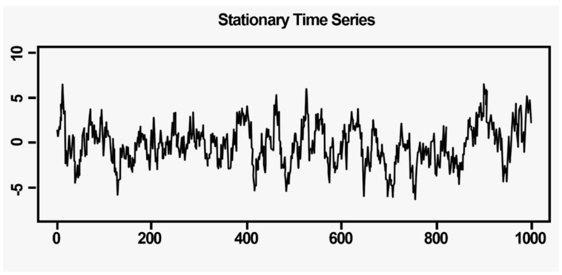
Figure 5: Statistical and machine learning models for time series prediction. Note that the ARIMA and SARIMA models can in some cases be classified as either statistical or machine learning based.

3.1 Pre-processing: Deseasonalization

For a machine learning model to be able to train efficiently and predict accurately it has to be trained on data that is processed appropriately to fit what the model should receive as input. An important preprocessing step specific for time series data is to account for seasonality. Generally, neural networks struggle to deal with seasonality, and a standard remedy is to apply some deseasonalization technique on the data before feeding it to the network. Claveria, Monte and Torra [Claveria et al., 2017] found that applying preprocessing to account for seasonality on time series data reduced the mean absolute percentage error (MAPE) produced by the neural network by an average of more than 60%. Although LSTM networks are better than most neural networks at



(a) Non-stationary time series



(b) Stationary time series: mean and variance are constant across the time series

Figure 6: Non-stationary versus stationary time series

catching long-term dependencies, it still struggles to deal with high or varying degrees of seasonality. The reason is that neural networks operate under the assumption that the input and output data is stationary, which means they derive from a single distribution. Seasonal data, on the other hand, is not stationary but comes from multiple distributions depending on the season. Figure 6a and 6b illustrates the differentiation between stationary and non-stationary data.

3.2 Statistical methods for time series forecasting

The forecasting literature is heavily dominated by statistical methods based on linear processes, such as ARIMA or exponential smoothing. In the recent work of [Makridakis et al., 2018] the authors present evidence that classical statistical methods systematically outperform machine learning methods for univariate time series forecasting. In the paper, the author's compare 10 machine learning models, including an LSTM model, against 8 statistical methods. The best performing statistical models were a Holt-Winters model, which is based on multiple exponential smoothing calcualtions, and an ARIMA model. The ARIMA and Holt-Winters models displayed a mean average percentage error (MAPE) on the time series data of 7.12% and 7.32% respectively, while the LSTM neural network had a MAPE of 11.67%. A couple of reasons are brought forth as to why the statistical methods outperform the ML-methods: First of all, the statistical methods seemed to be better at capturing the seasonal, level and trend components of the time series. Secondly, statistical methods such as exponential smoothing and ARIMA are not prone to overfitting in the same way that neural networks are. The time series used in the research of [Makridakis et al., 2018], had an average number of observations of only 118, which could explain the clear signs that the LSTM model was struggling with overfitting. This notion was further elaborated upon when Cerqueria, Torgot and Soares [Cerqueira et al., 2019] attempted to recreate the research of [Makridakis et al., 2018], by using the same models to predict on similar time series data, but which contained at least 1000 observations. In their experiment, the machine learning models obtained a better forecasting accuracy than that of the statistical models. The author's concluded with the hypothesis that neural networks outperform statistical methods when fed with enough data, because of its ability to perform cross-learning, which means that patterns in the time series data can be learned across multiple series. However, statistical methods proved far better at dealing with high or varying degrees of seasonality, especially models such as Holt-Winters which is based on exponential smoothing calculations.

3.3 Using exponential smoothing to account for seasonality

Exponential smoothing is a technique for smoothing time series data using an exponential moving window function. The idea is to use weighted averages of past observations, giving more weight to recent observations, and exponentially smaller weights to older observations, to ultimately predict future observations. The Holt-Winters method consists of performing multiple exponential smoothing calculations to capture the level, seasonality and trend components of the time series. The idea is to make the time series data stationary, so that the mean and variance are constant (see Figure 6b). Below is the Holt-Winter's formulas for capturing seasonality, level and trend respectively:

$$s_t = \frac{\gamma y_t}{l_{t-1} + b_{t-1}} + (1 - \gamma)s_{t-1} \tag{8}$$

$$l_t = \frac{\alpha y_t}{s_{t-1}} + (1 - \alpha)(l_{t-1} + b_{t-1}) \tag{9}$$

$$b_t = \beta(l_t - l_{t-1}) + (1 - \beta)b_{t-1} \tag{10}$$

where y_t is the value of the series at point t, l_t, s_t, and b_t are the level, seasonality and trend components, and α, β and γ are smoothing coefficients.

3.4 Combining an LSTM network with exponential smoothing

To improve the forecasting accuracy of the sole LSTM network, we propose a hybrid model combining an LSTM network with the use of Holt-Winters exponential smoothing method. The idea is to use the formulas presented above (8), (9) and (10) to deseasonalize the data before feeding it to the LSTM network. The two models complement each other perfectly: The smoothing calculations deseasonalize the data which allows the LSTM network to perform cross-learning on appropriately pre-processed data. To make use of the full potential of deseasonalization, the formulas for level, trend and seasonality can be calculated dynamically for each individual time series. During prediction, these are fitted along with the global weights of the neural network. In that regard, the hybrid model is hierarchical as it combines training global parameters across many time series (the weights of the neural network), with local smoothing coefficients and seasonality components calculated on a per-series basis. Ultimately, the level, trend and seasonality of the time series are captured precisely, thus enhancing the forecasting accuracy.

Possibly the most important components to capture are the level and seasonal components, as these tend to be the most volatile in time series data. LSTM networks are usually good at capturing the overall trend due to its ability to capture long-term dependencies. In the M4 competition, the winning model used formulas 8 and 9 during pre-processing to capture the seasonal and level components, while relying on the LSTM network to capture the overall trend [Smyl, 2019].

3.5 3 steps of forecasting

The hybrid model steps through three main elements when producing forecasts [Smyl, 2019], as illustrated in figure 7.

3.5.1 Deseasonalization and adaptive normalization

For each time series, the seasonal, level and trend components are calculated through the Holt-Winters equations above. These are fitted alongside the global weights of the neural network through stochastic gradient descent to calculate the global seasonality, level and trend components. Finally, the global components are used to deseasonalize and normalize the time series.

3.5.2 Generate forecast

The next step is to feed the normalized and deseasonalized data to the LSTM network, and allow it to make a prediction on the data.

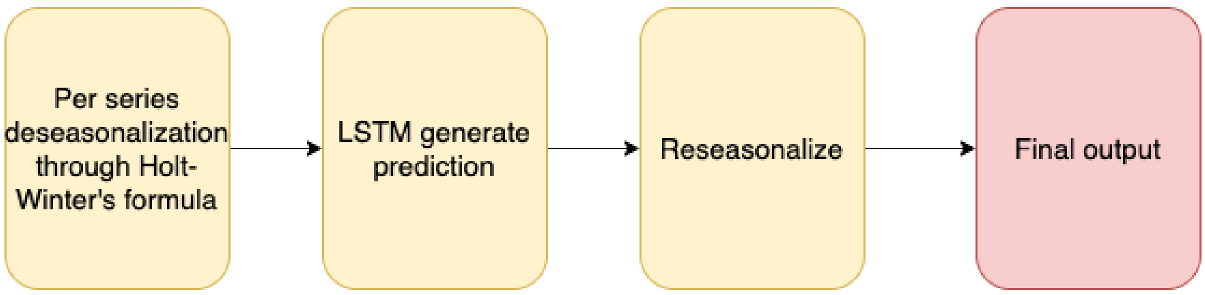


Figure 7: 3 steps of forecasting using the hybrid method

3.5.3 Re-seasonalize and re-normalize

The third and last step is to re-seasonalize and re-normalize the data by adding the seasonal components back to the neural network. The result will be the network's scaled prediction.

4 Current Applications

In the past years LSTM networks have been used for time series prediction in many fields. In this section we will take a look at recent research that have used LSTM networks to predict the future. We present four areas where LSTM networks were applied successfully.

4.1 The M4 Competition

The use of LSTM networks was highly successful in the M4 competition [Smyl, 2019]. The M4 competition (Makridakis Fourth Competition) is a competition with intentions to evaluate and compare different forecasting methods. Spyros Markidakis is a researcher in forecasting and is leading the competition where participants compete on developing the most accurate forecasting method for a given data set.

The winner of the M4 competition used a hybrid method combining exponential smoothing with LSTM networks. Slawek Smyl, the winner, argued that a hybrid model could exploit the advantages of pure statistical and machine learning methods while avoiding their drawbacks. With exponential smoothing models, you can effectively capture the level and seasonality of the time series. In addition, the LSTM network can cross-train and capture nonlinear trends. The result of this led to a hybrid forecasting method that could utilize more out of the data and therefore enhancing the prediction accuracy.

4.2 COVID-19

Another interesting way that LSTM networks have been used is to predict COVID-19 transmission. Many research papers have been published lately where they try to predict how many COVID-19 cases there will be in an area.

Vinay Chimmula and Lei Zhaing used data provided by John Hopkins University and Canadian health authorities to create a time series forecast using a LSTM network to predict the COVID-19 outbreak in Canada [Chimmula and Zhang, 2020]. The LSTM network provided a good short-term prediction with a root mean square error(RMSE) of 34.83 and an accuracy of 93.4%. For the long term prediction, the RMSE was 45.70 with an accuracy of 92.67%.

Another research paper written by Parul Arora, Himanshu Kumar and Bijaya Panigrahi also used LSTM networks to predict positive cases of COVID-19 in India [Arora et al., 2020]. Instead of using basic LSTM networks they used different variants of LSTM. Stacked LSTM, bi-directional

LSTM(Bi-LSTM) and convolutional LSTM were used. Stacked LSTM is a network with multiple hidden layers, also called deep LSTM networks. Bi-LSTM networks take the time series data input both forwards and backwards. These methods have been used to predict positive cases in 32 states in India. When comparing prediction errors for the methods the convolutional LSTM had the highest prediction error. On the other hand, the bi-directional LSTM network provided the best forecast and on short term prediction (1-3 days) the error was less than 3%.

Farah Shahid, Aneela Zameer and Muhammad Muneeb did a similar research paper [Shahid et al., 2020]. In this paper they focused on short term prediction models for predicting the number of COVID-19 cases, deaths and recoveries for ten major countries. Here they compared autoregressive integrated moving average (ARIMA), support vector regression (SVR), LSTM, and bi-directional LSTM. The total result ranked Bi-LSTM most accurate, followed by LSTM. They concluded that Bi-LSTM and LSTM have robustness and can predict COVID-19 cases with high prediction.

All three papers have shown how LSTM and versions of LSTM can be used in time series prediction of COVID-19, and on short-term the accuracy is quite high. These predictions can be useful for states and national governments when developing strategies to reduce the transmission of COVID-19.

4.3 Uber

A third very interesting way LSTM networks currently are used for time series prediction is at Uber [Erran et al., 2017, Laptev et al., 2017]. Uber uses forecasting to predict user demand. More precisely they want to know when, where and how many rides they will receive in the future. LSTM networks are here mostly used for extreme event forecasting or anomaly detection. Extreme events for Uber is when there is a big peak in demand. Factors like holidays, cultural events and external factors like weather and population growth play a big role. The extreme events with peaks in demand are very important for Uber because these days are the busiest days for the company. However, peak days like new years eve only happen once in a year making the data sparse because Uber is a young company with only a few years of data.

The LSTM network method was selected because of its end-to-end modelling, easy to take in external factors and the ability to automatically extract features. However, Uber found out that the normal LSTM model performed worse than other models they already used. Therefore, they built an architecture based on LSTM networks. The first part was to make the architecture ready with auto feature extraction from an LSTM network. This is important for capturing the time series dynamics, and normally this is done manually. Next step was to aggregate the feature vectors with an ensemble technique to create a final vector. This vector is then combined with new input data and used as input in a new LSTM network. The result improved with 14.09% from the LSTM network that was trained over sets of raw inputs.

4.4 Floods

LSTM networks have also been used to predict floods in China [Y.Ding et al., 2020]. In this research paper the authors used a model based on LSTM. They present the challenge of forecasting floods where high accuracy and interpretability is needed. Hydrological models have achieved low prediction accuracy because of nonlinearity and natural uncertainty of floods. On the other side, many machine learning models fail to capture the physical interpretability of floods. The researchers therefore created a Spatio-Temporal Attention LSTM (STA-LSTM) model, a LSTM network with attention mechanism. Attention mechanisms are based on focusing on certain factors when processing data. In this case, spatial and temporal data have attention weights so they are paid more attention to.

The model was compared with other forecasting methods like historical average, fully connected networks, convolution neural networks, graph convolutional networks, LSTM networks, spatial attention LSTM networks and temporal attention networks. Overall, in most cases the STA-LSTM model outperformed the other models, reflecting that LSTM networks with spatio-temporal attention are beneficial for this kind of time series prediction.

5 Conclusions and Further Work

Time series prediction can be very valuable for businesses across all industries. The ability to forecast the future from historical data can increase profits and even save lives. In this paper, we described how long short term memory neural networks can be used for time series forecasting. The LSTM network was not originally created for time series prediction, yet it has become one of the more popular machine learning models for this purpose. As presented in section 2, this is much due to its ability to capture long-term dependencies when cross-trained over multiple time series. However, the LSTM network struggles with high or varying degrees of seasonality and therefore requires accurate pre-processing to provide the best forecasting accuracy. In section 3 we discussed how to implement a statistical method to aid the LSTM network in the pre-processing step. Finally, in section 4 we looked at how LSTM networks have successfully been applied for time series forecasting. We described how it was implemented to win the prestigious M4 competition, how it was used to predict COVID-19 transmissions, and how it was used for predicting future floods. From our research, we conclude that LSTM networks work very well for predicting the future, but for it to succeed it is crucial to understand what kind of data you have, and what pre-processing steps that are necessary. Future research should explore additional pre-processing steps that can further enhance prediction accuracy, and look at more opportunities where LSTM networks can be used for forecasting.

Bibliography

P. Arora, H. Kumar, and B. Panigrahi. Prediction and analysis of covid-19 positive cases using deep learning models: A descriptive case study of india, 2020. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7205623/.

Jocelyn Barker. Machine learning in m4: What makes a good unstructured model? 2020.

Filippo Maria Bianchi. Recurrent neural networks, 2017. URL https://www.sintef.no/contentassets/9689621c8cda44d9b2bc91c5fba21135/bianchi.pdf/.

Gianluca Bontempi, Yann Borgne, and Souhaib Taieb. Machine learning strategies for time series forecasting. 2013.

Vitor Cerqueira, Luis Torgo, and Carlos Soares. Machine learning vs statistical methods for time series forecasting: Size matters, 2019. URL https://www.researchgate.net/publication/336146790_Machine_Learning_vs_Statistical_Methods_for_Time_Series_Forecasting_Size_Matter.

Vinay Chimmula and Lei Zhang. Time series forecasting of covid-19 transmission in canada using lstm networks, 2020. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7205623/.

Oscar Claveria, Enric Monte, and Salvador Torra. Data pre-processing for neural network-based forecasting: Does it really matter?, 2017. URL https://www.researchgate.net/publication/283823960_Data_pre-processing_for_neural_network-based_forecasting_Does_it_really_matter.

L. Erran, N. Laptev, and S. Smyl J. Yosinski. Time-series extreme event forecasting with neural networks at uber, 2017. URL http://www.cs.columbia.edu/~lierranli/publications/TSW2017_paper.pdf/.

Kolbjørn Flaarønning and Elida Tuhus. Initial integration of data-driven health-indicators in the petroleum industry. 2020.

N. Laptev, S. Smyl, and S. Shanmugam. Engineering extreme event forecasting at uber with recurrent neural networks, 2017. URL https://eng.uber.com/neural-networks/.

P Liang and NK Bose. Neural network fundamentals with graphs, algorithms and applications. 1996.

Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakupoulos. Statistical and machine learning forecasting methods: Concerns and ways forward. 2018.

Christopher Olah. Understanding lstm networks, 2015. URL https://colah.github.io/posts/2015-08-Understanding-LSTMs/.

F. Shahid, A. Zameer, and m. Muneeb. Predictions for covid-19 with deep learning models of lstm, gru and bi-lstm, 2020. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7437542/.

Sagar Sharma. Activation functions in neural networks, 2017. URL https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6.

Slawek Smyl. A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting, 2019. URL https://www.sciencedirect.com/science/article/pii/S0169207019301153.

Y.Ding, Y. Zhu, J. Feng, P. Zhang, and Z. Cheng. Interpretable spatio-temporal attention lstm model for flood forecasting, 2020. URL https://www.sciencedirect.com/science/article/pii/S0925231220307530.

Leila Zoubir. A brief history of time series analysis. 2017.