Estimating Australian beer production one month ahead

Erik Kuitunen, Pauli Anttonen, Joona Lappalainen

January 17, 2025

1 Introduction

In decision making and risk management, forecasting the behavior of a physical model or complex system is of great importance. The ability to essentially predict possible outcomes of a phenomena gives invaluable aid in strategy and planning in plethora of fields and applications, from finance to healthcare.

Machine learning (ML) can be utilized to achieve fast and precise forecasts in nowadays applications. Especially, long short-term memory (LSTM), along with the other recurrent neural network (RNN) architectures, is very suitable and widely applied method for these purposes. For example, in production forecasting, it is essential to capture long-term dependencies while ensuring accuracy across different time scales. [1]

The goal of this project is to predict the beer production in Australia one month ahead. The data set contains time-series data in two columns: dates and monthly beer production between years 1956-1995. Thus, the data is very low-dimensional, and dimensionality reduction techniques will not be needed. The data along with smaller subset of the first five years are shown in Figure 1. There is clear seasonality to be observed. In addition, an increasing trend can be seen from the full data.

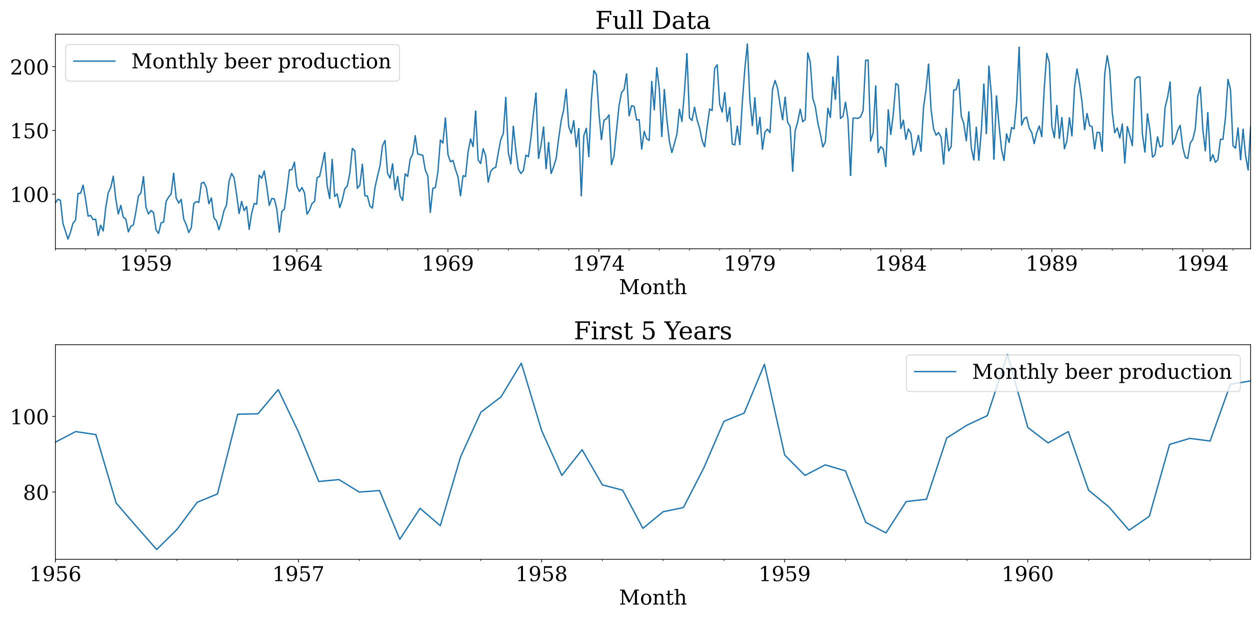

Figure 1: Whole data set and subset of first five years.

All of the data has been recorded with the same frequency and it consists of continuous measurements with no missing values. Each year has 12 data points, one for each month, except for the last year, which has only eight data points. Since we have only one variable, asynchronous data between variables is not a problem.

2 Methods and baseline modeling

2.1 Time Series Decomposition

In time series decomposition, the time series data is divided into into components. In this project we use the Seasonal and Trend decomposition using Loess (STL) method [2], which divides the data into three components: trend, seasonal, and residual components. The data along its decomposition can be seen in Figure 2. It can be seen that there is clear seasonality in the data, with peak beer production occurring in the winter months each year. Trend through the years 1956-1975 is rising, after which settles to a somewhat stable region of values, and in the 90s, even decreasing a bit. In the 1970s the residuals' variance seems to increase, settling back close to the original in the 90s. No clear outliers can be spotted from the STL decomposition.

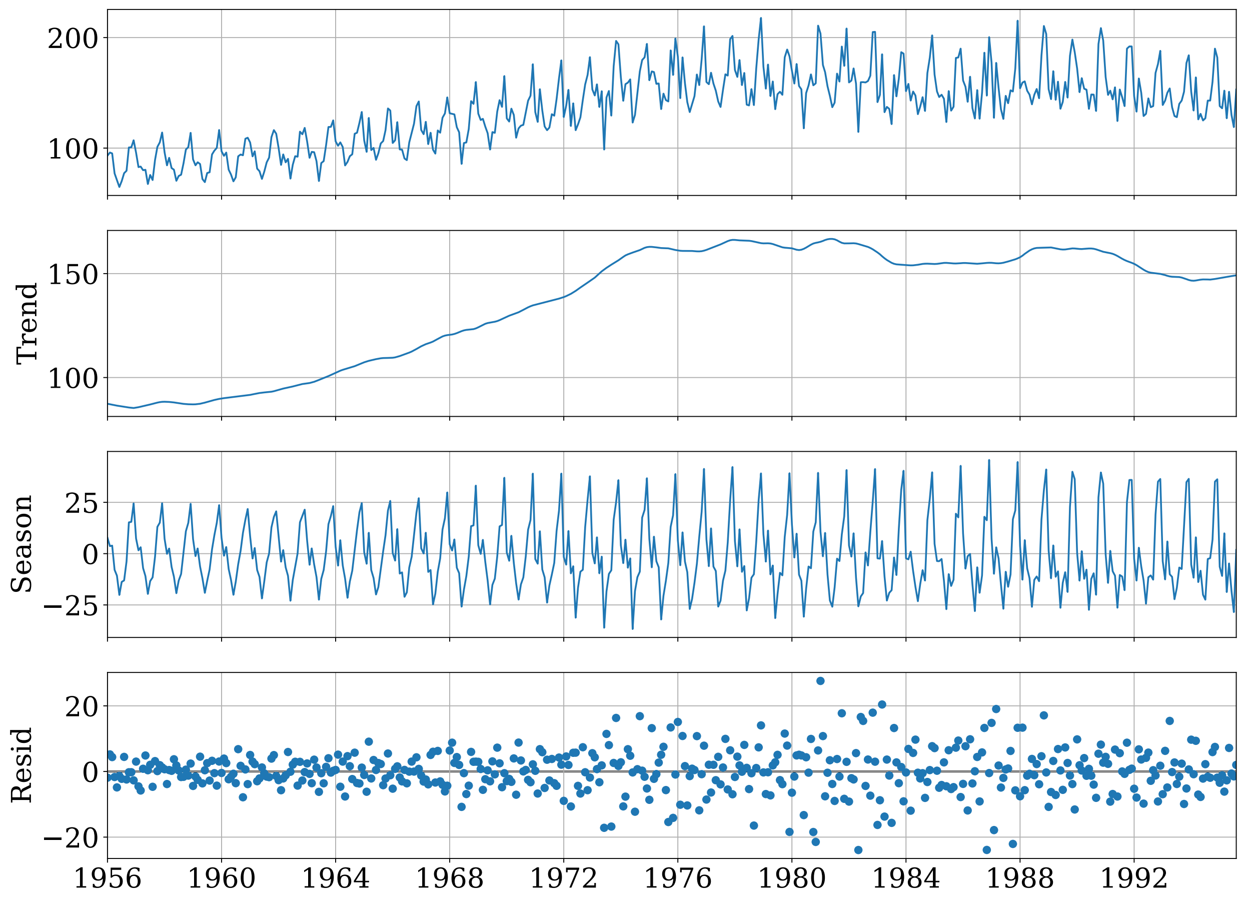

Figure 2: STL decomposition of beer data.

2.2 Autocorrelation analysis

The autocorrelation function indicates a clear yearly seasonality in the data. The current value is significantly influenced by the value from the previous year and several years prior. When building the model, it is essential to broaden the scope enough to utilize all available information effectively. The autocorrelation function for the whole data set is shown in Figure 3.

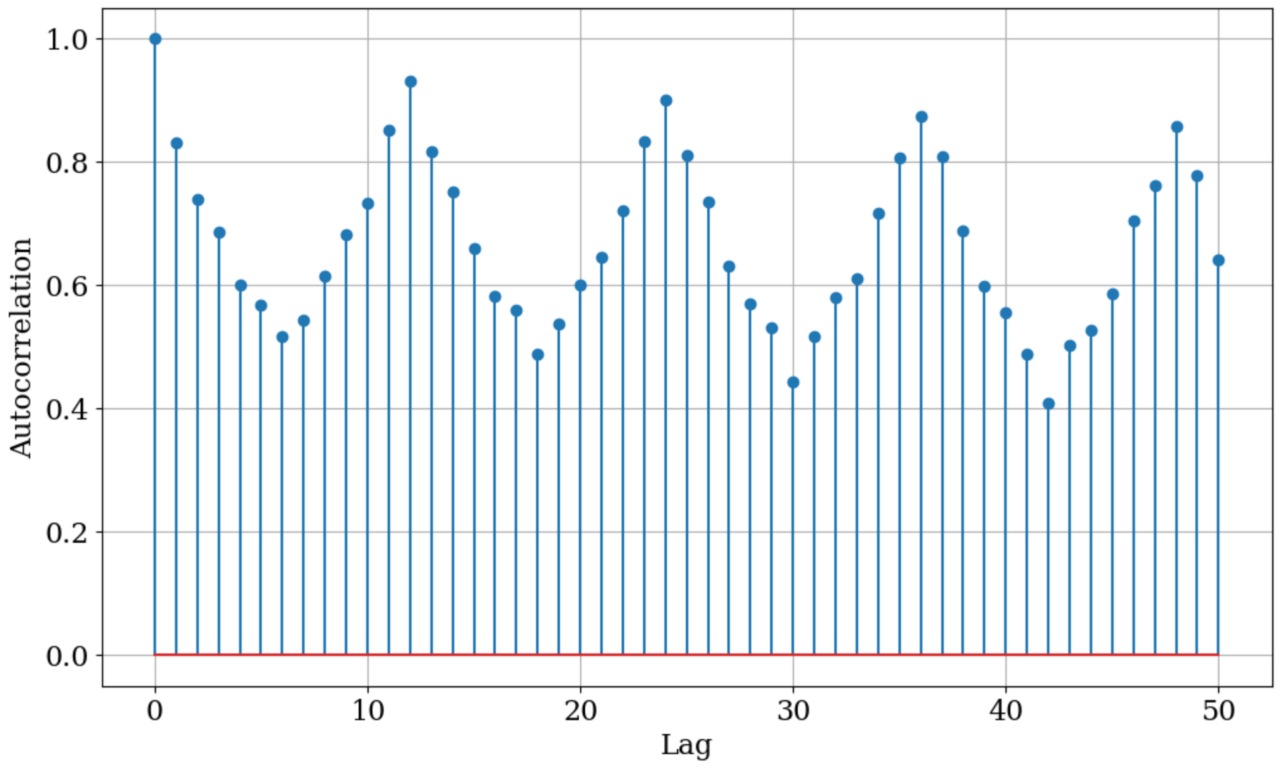

Figure 3: Autocorrelation function for entire dataset

2.3 Plan for partitioning the time-series data

Evaluation is crucial when building a machine learning model. Cross-validation (CV) is a statistical method which can reduce the chance of overfitting. Especially, k-folds-Cross-Validation evaluates the performance of the model much better than a simple train-test. However, when it comes to time-series data, CV becomes less trivial. For example, random samples from the future cannot be used to forecast values in the past. Several methods have been developed to introduce effective CV for time-series data. The suitable method for this project is blocked CV, since it does not create leakage from future data to the model. This method can be computationally heavy, but since the data is very low-dimensional and there are only 476 data points, it should not be a problem. [3]

2.4 Sub-samplings division and effects of seasonality

LSTM is a model, which expects three-dimensional input while using the Keras Python library for deep learning. Also, long time series data must be divided into smaller samples and reshaped for LSTM model since LSTMs work properly within the range from 200 to 400 time steps.[4] Sub-sampling can be done with various ways. One of them is removing time steps from the beginning, from the end, or at random. Also, it is possible to use truncated backpropagation through time (TBPTT) method, which utilizes the estimated gradient from a subset. The simplest method is to do nothing. That is, the long sequence of data is left as is. [5] In addition, in the case of seasonality, it is important to do the sub-sampling so that full seasons are entirely included in the samples. Thus, the effects of the seasons can be accurately modeled and season-specific details do not get mixed.

In our case, we could leave the data as it is since it's not too long for LSTM. Still, we probably should divide the data for three or four samples so that it comes fluently with CV partitioning.

If the time series contains seasonal fluctuations, a single trained model will not perform very well in general predictions. Thus, they need to be incorporated into the model or taken into account when pre-processing the data for model training. In case of LSTM, a variant called STL-LSTM can be used [6]. In short, after STL decomposition, the LSTM network is used to predict each component separately, after which the predictions are combined to achieve a final prediction.

2.5 Standardization methods

Several suitable standardization methods exist for time series forecasting using Neural Networks models. The most common is probably *Z-score normalization*, defined as

$$z_i = \frac{x_i - \bar{x}}{\sigma},$$

where x_i is a training sample, $\bar{x}$ is the mean of the training samples, and σ is the standard deviation of the samples. It is especially useful, when the data has several variables that have differing scales. It also makes the units of the variables comparable. [7]

Another method is *linear transformation* [7] (also known as Min-Max scaling), which is used to transform the data between a range, usually $[0,1]$ or $[-1,1]$:

$$z_i = \frac{x_i - x_{\min}}{x_{\max} - x_{\min}}.$$

One more sophisticated scaling method is *Box-Cox Transformation* [8], which in essence tries to find a value, $\lambda \in [-5,5]$ for which, after the transformation, the data best follow the Gaussian distribution. The Box-Cox transformation is defined as

$$\begin{aligned} z_i &= \frac{x_i^\lambda}{\lambda}, \quad \lambda \neq 0, \\ z_i &= \log x_i \quad \lambda = 0. \end{aligned}$$

2.6 Baseline model

A simple baseline model is built to use for comparison: more sophisticated models should be able to perform at least as well as the baseline model, prefererably significantly better. As a baseline model, we are using a basic autoregressive model. At each timestep, we give an input with 12 past values and use these to predict the next value for each timestep. For this model we will be using the first 66% of the data as training partition and the latter 33% as test partition. The division is subject to change to match the final model as we center our focus on a more sophisticated methods for choosing the partitions.

Current baseline model consist from single connected layer of weights and a bias

$$T_{i+1} = \sum_{n=0}^{11} k_n T_{i-n} + b,$$

where k_n and b are trainable parameters.

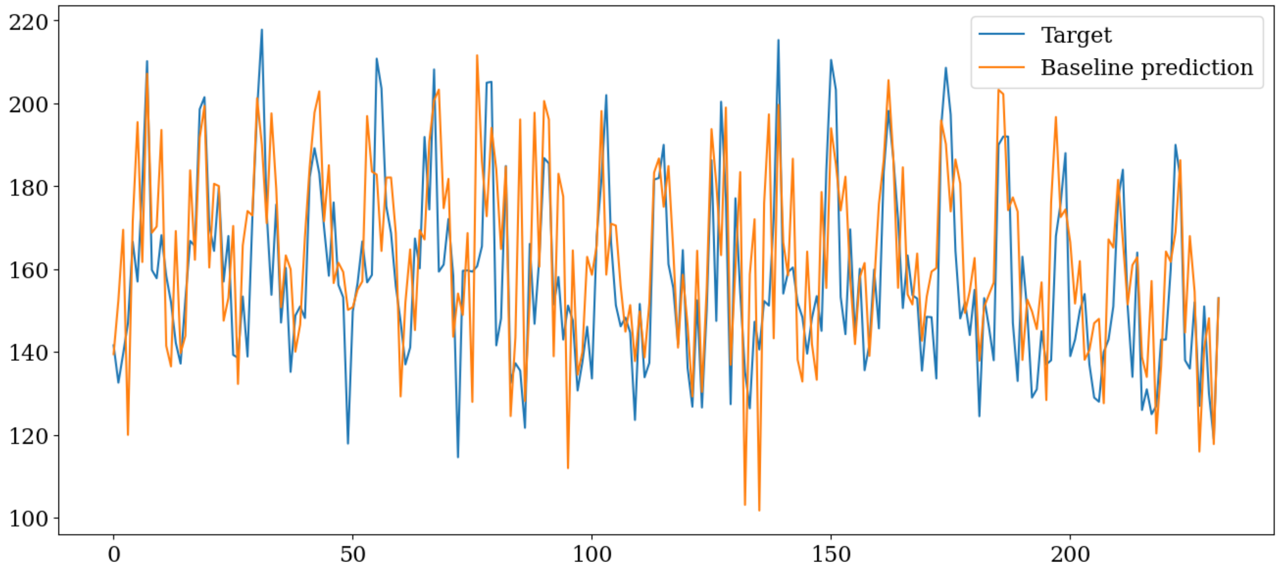

Figure 4: Baseline model predictions on the test data.

3 Modeling strategy

In this project, we will be building two models to forecast beer production: LSTM and Transformer models.

3.1 Data pretreatment

We are going to use Min-Max scaling as a data pretreatment method for our models . It is very suitable for our dataset since there are no outliers which could have a negative influence to the scaling. Also, Min-Max scaling helps LSTM algorithms to work properly. [9]

3.2 LSTM

LSTM is a Neural Network architecture, which utilizes recurring modules that comprise of several layers, each layer having their own functionality and effectively enabling the network to ”remember” past information. [10]

LSTM model architecture

Our LSTM model is going to have an input layer, LSTM layers, a dense layer, and an output layer. The architecture of the model is shown in Figure 5.

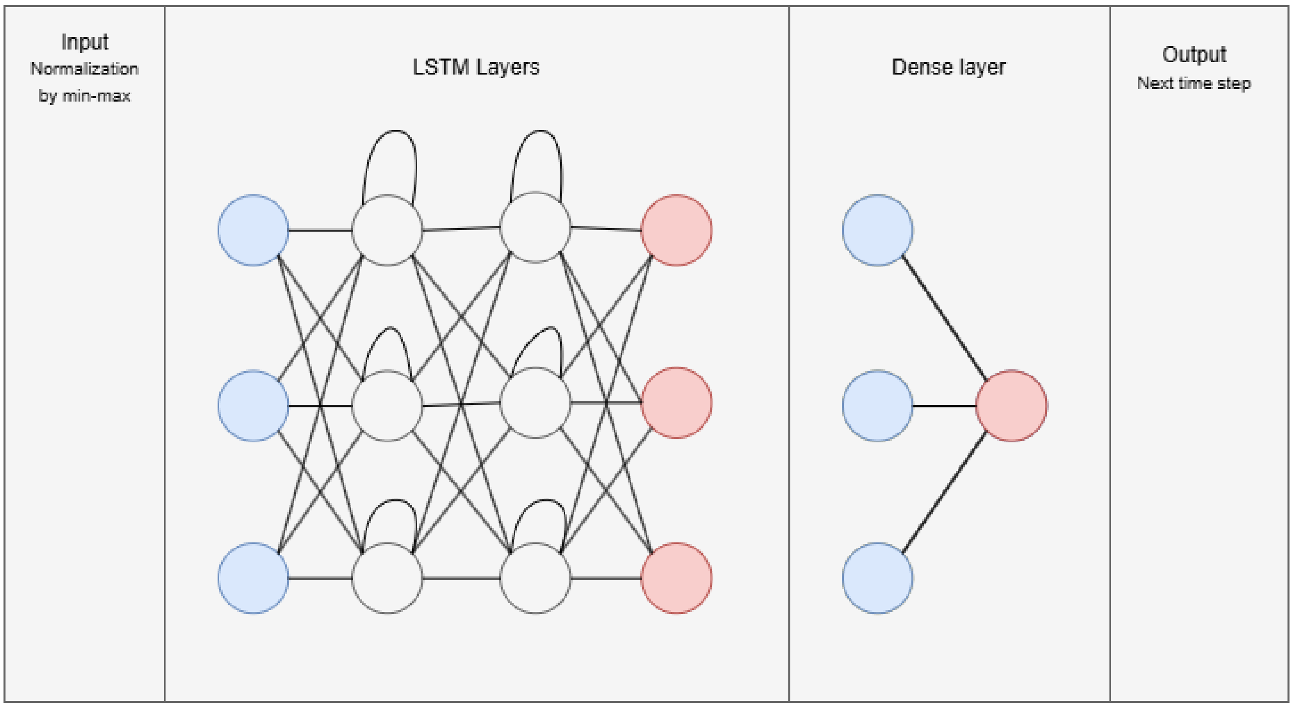

Figure 5: A layer graph of the LSTM model architecture.

3.3 Transformer models

Transformer models are state-of-the-art machine learning architecture proposed by Google scientists in 2017. It is based on the attention mechanism, which in essence is a method for determining relative importance between all components in a sequence. Transformer models are the foundation of modern Large Language Models such as ChatGPT. Transformer model architecture can be seen in Figure 6. More in-depth discussion on architecture is outside the scope of this work, but can be found from the original paper. [11]

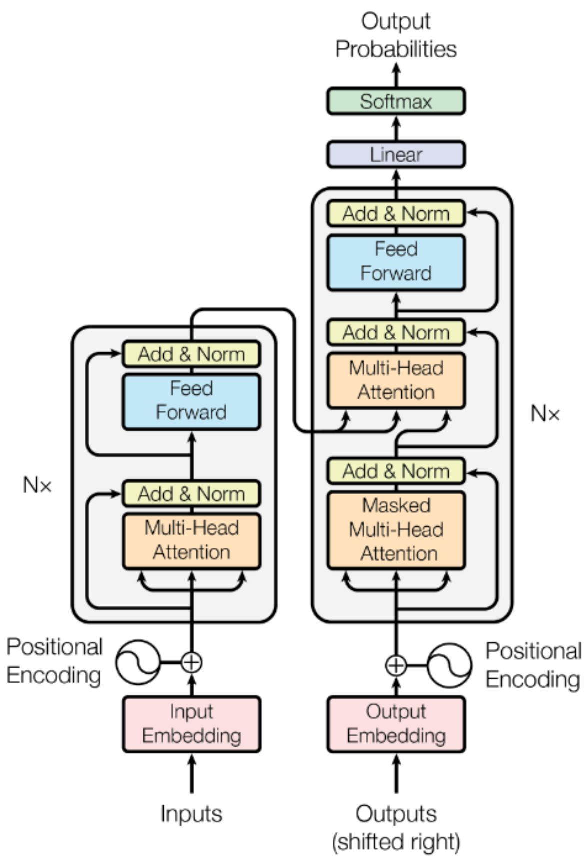

Figure 6: Transformer model architecture. Figure from [11].

3.4 Model evaluation and optimization

The data has been divided into training-validation and testing sets. Since we are interested in predicting the future values of the time-series, the data is divided in a way that testing set contains only time-series data from later times than training-validation set. As with the loss function of the model, Mean Squared Error (MSE) is used as the primary metric to evaluate the performance of the model along with the output accuracy.

3.4.1 Model optimization

A study for optimization needs to be conducted in order to produce accurate and robust model. Model accuracy and RMSE will be the main metrics that will be followed during the optimization process. There exists several possible approaches for optimization.

Neural network layer configurations

The Neural Network consists of a number of layers, each with their own parameters. The amount of hidden layers could be altered, although adding layers will also increase the computational cost of the training. Different sizes for the hidden layers will be tested and their impact examined. Adding gradually more complexity to the model will enable finding a sweet spot for the accuracy and computational cost.

Optimizing training parameters

We will be testing several combinations for model training parameters to find optimal set of values for them. We will be focusing on learning rate and batch size, since they are easy to modify and have potentially great impact on the convergence and accuracy of the mode.

Feature engineering

The dataset can be augmented by introducing time-based features that depend on the original data. Lagged variables can be introduced, giving information on what kind of effect the past has to the measurements of the present. Lagged variables are defined as

$$Y^{t-n}(t) = Y(t-n),$$

where n is the amount of lag added to to the variable. Another possibility is moving average, defined as

$$MA_{k,n} = \frac{1}{k}\sum_{i=n-k+1}^{n} Y_i,$$

where k is the size of the sliding window and n denotes the nth point of the data set, at which the average is calculated.

In practice, the optimization will be conducted using Grid Search algorithm.

3.4.2 Grid Search

In order to find the best model, hyperparameter tuning can be applied. Grid search is a common algorithm used for this purpose. On every training iteration, it tries different combination of the provided hyper-parameter values. In the end, the best values are chosen. In our case, using grid search is very beneficial. That is because our data set is so small thus it also have only a few parameters. In addition, it is automated so that the amount of manual implementation is reduced but it still allows to control the desired combinations of hyperparameters. [12]

3.5 Roles

Pauli Anttonen	Model creation
Joona Lappalainen	Evaluating
Erik Kuitunen	Optimization

4 Results

4.1 LSTM

LSTM model shows quite promising results. Optimal hyperparameters and layer configuration was obtained through grid search, results can be seen in Table 1. Feature engineering was not utilized in training of this model. Using the optimized parameters, an optimal model was trained. Results of the actual sales and corresponding predictions can be seen in Figure 7. MSE Loss during the training can be seen in Figure 8. The training loss converges relatively quickly, having only minor improvements after only 60 epochs. Similar behavior is observed on the validation loss. Minimum values for training and validation losses were 0.0037 and 0.0086, respectively.

Table 1: Grid search-optimized parameters for LSTM.

Parameter	**Value**
Number of LSTM layers	1
Size of hidden layers	64
Batch size	32
Sequence length	24
Learning rate	0.01

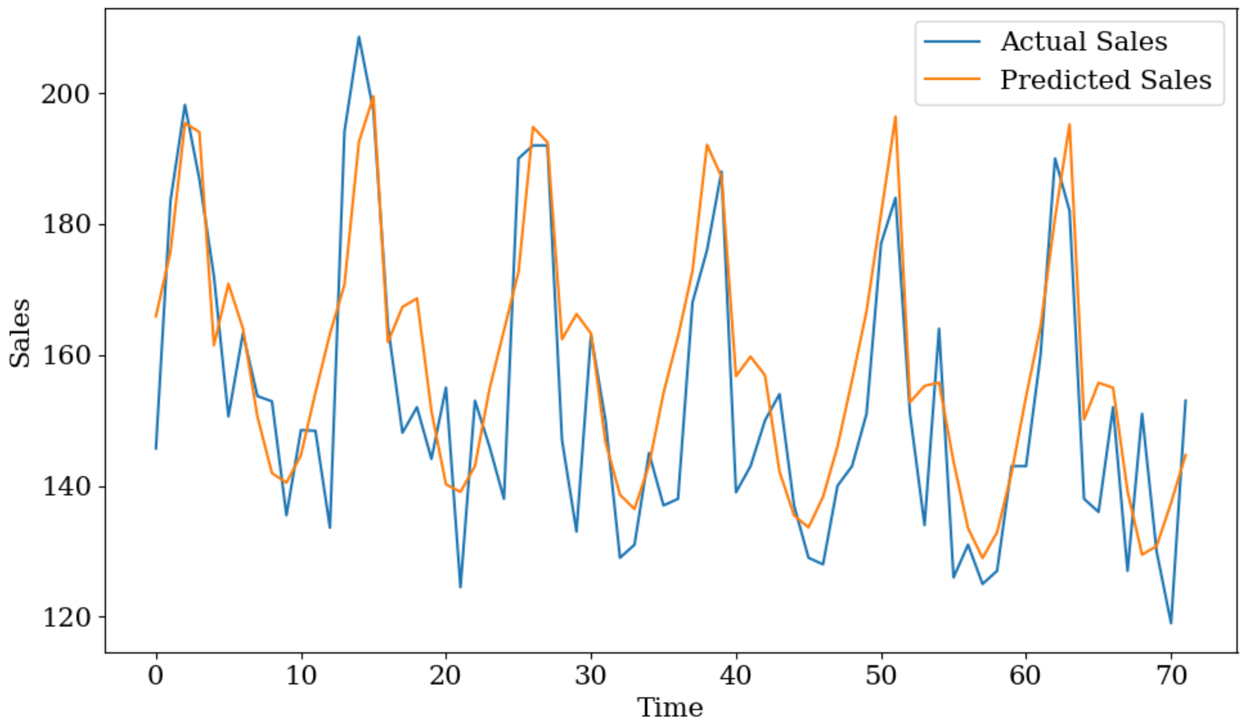

Figure 7: Ground truth and LSTM forecast of beer sales.

Because of grid search, our model is already quite well optimized. Thus further improvements and parameter tuning might not be needed.

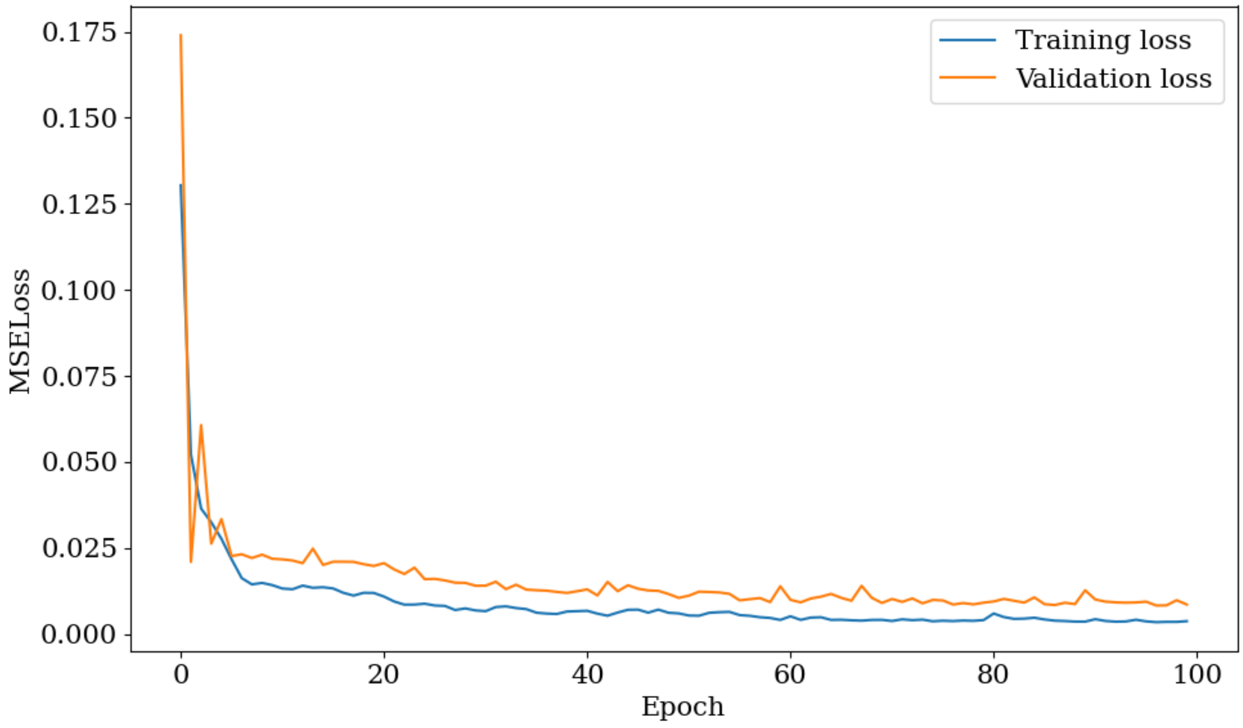

Figure 8: Training and validation losses of the model against number of epochs trained.

4.2 Transformer model

Good results were obtained with the Transformer model as well. Grid search found the model's optimal parameters, which can be seen in Table 2. Transformer seems to forecast somewhat larger values than LSTM, and LSTM managed to capture the small downward trend in our time frame better than Transformer. Notably, the Transformer loss converges even faster than LSTM loss but stays a very small amount higher even after convergence. The ground truth

Table 2: Grid search-optimized parameters for Transformer model.

Parameter	**Value**
Number of layers	1
Size of hidden layers	16
Batch size	64
Sequence length	24
Learning rate	0.01

and forecast results can be seen in Figure 9. Also, MSE Training Loss is plotted in Figure 10. The smallest training and validation losses were 0.0038 and 0.0112, respectively.

Once again, thanks to grid search, neither further improvements nor parameter tuning is needed.

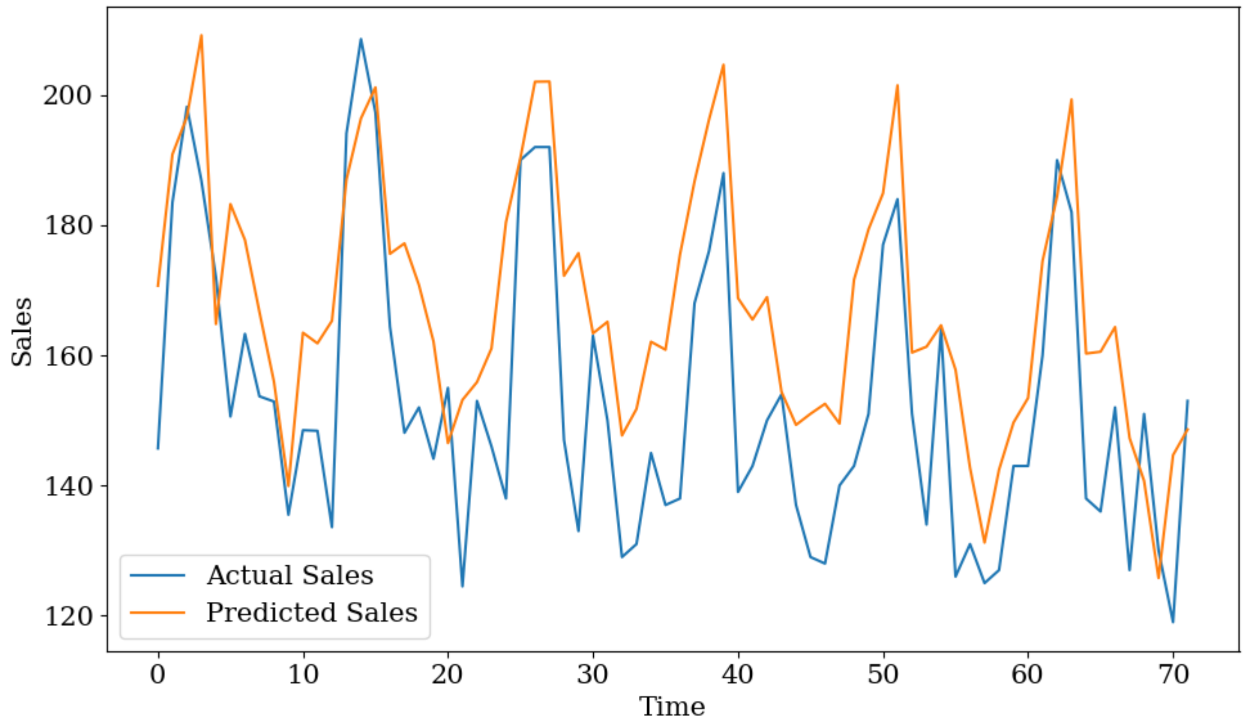

Figure 9: Ground truth and Transformer forecast of beer sales.

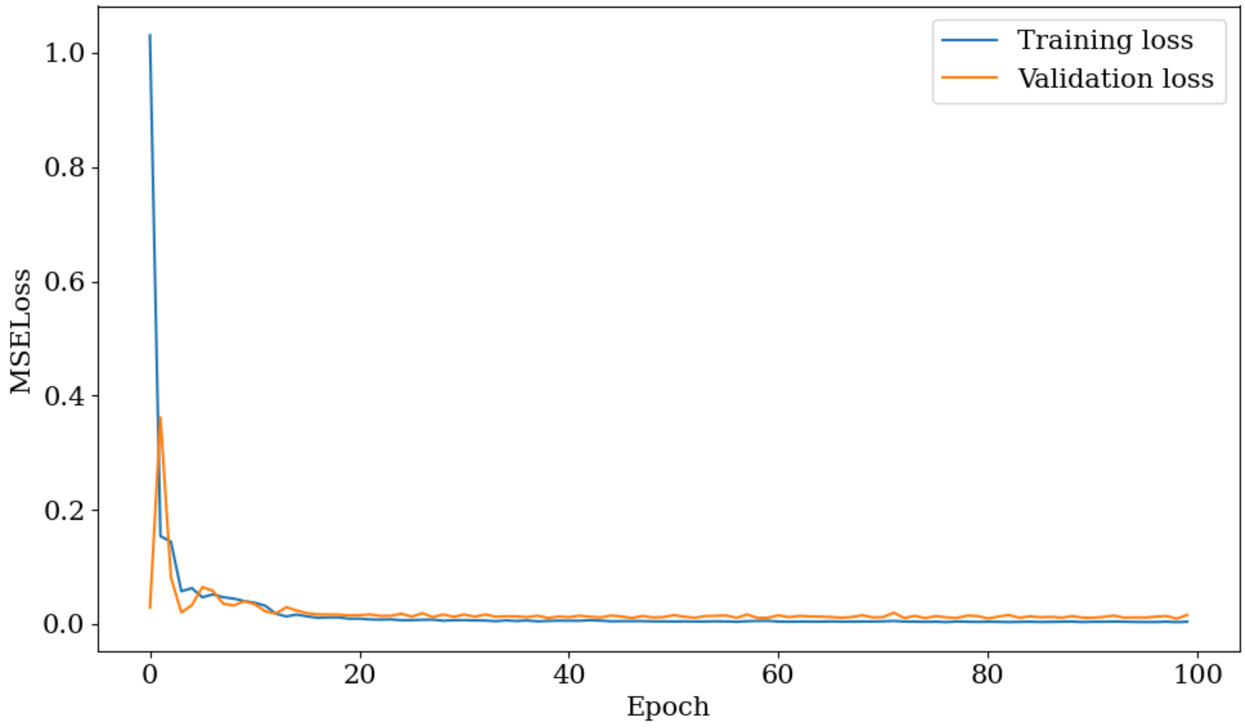

Figure 10: Training and validation losses of the model against number of epochs trained.

5 Conclusions

As expected, LSTM and transformer models are well-fitted for sequential data. Both methods give great increases in accuracy compared to the baseline model. The baseline model gives predictions based solely on the previous year's values, whereas LSTM and transformer models can find more complex patterns in the data. The LSTM model generalizes the problem better, whereas the transformer model gives more variating results.

Overall, both methods give good prediction sales and are well-fitted for the application. Based on this data, we would choose the LSTM model to predict the sales for this company.

References

[1] Saif Ur Rehman. *LSTM for Production: Part 1.* URL: https://jpt.spe.org/twa/lstm-for-production-part-1.

[2] Robert B. Cleveland, William S. Cleveland, Jean E. McRae, and Irma Terpenning. "STL: A Seasonal-Trend Decomposition Procedure Based on Loess (with Discussion)". In: *Journal of Official Statistics* 6 (1990), pp. 3–73.

[3] Soumya Shrivastava. *Cross Validation in Time Series.* URL: https://medium.com/@soumyachess1496/cross-validation-in-time-series-566ae4981ce4.

[4] Jason Brownlee. *How to Prepare Univariate Time Series Data for Long Short-Term Memory Networks.* URL: https://machinelearningmastery.com/prepare-univariate-time-series-data-long-short-term-memory-networks/.

[5] Jason Brownlee. *Techniques to Handle Very Long Sequences with LSTMs.* URL: https://machinelearningmastery.com/handle-long-sequences-long-short-term-memory-recurrent-neural-networks/.

[6] Dewang Chen, Jianhua Zhang, and Shixiong Jiang. "Forecasting the Short-Term Metro Ridership With Seasonal and Trend Decomposition Using Loess and LSTM Neural Networks". In: *IEEE Access* 8 (2020), pp. 91181–91187. DOI: 10.1109/ACCESS.2020.2995044.

[7] M. Shanker, M.Y. Hu, and M.S. Hung. "Effect of data standardization on neural network training". In: *Omega* 24.4 (1996), pp. 385–397. ISSN: 0305-0483. DOI: https://doi.org/10.1016/0305-0483(96)00010-2. URL: https://www.sciencedirect.com/science/article/pii/0305048396000102.

[8] Md. Rashedul Islam, Md. Nasim Akhtar, and Momotaz Begum. "Long short-term memory (LSTM) networks based software fault prediction using data transformation methods". eng. In: *2022 International Conference on Advancement in Electrical and Electronic Engineering (ICAEEE).* IEEE, 2022, pp. 1–6. ISBN: 9781665469449.

[9] Kamlesh Kumar Rangi. *How Min-Max Scaler Works.* URL: https://medium.com/@iamkamleshrangi/how-min-max-scaler-works-9fbebb9347da.

[10] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-Term Memory". In: *Neural Computation* 9.8 (Nov. 1997), pp. 1735–1780. ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735. eprint: https://direct.mit.edu/neco/article-pdf/9/8/1735/813796/neco.1997.9.8.1735.pdf. URL: https://doi.org/10.1162/neco.1997.9.8.1735.

[11] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention is all you need". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems.* NIPS'17. Long Beach, California, USA: Curran Associates Inc., 2017, pp. 6000–6010. ISBN: 9781510860964.

[12] dremio. *Grid Search.* URL: https://www.dremio.com/wiki/grid-search/.