ROCKET

Albert Sayapin [1]

Abstract

Most methods for time series forecasting that reach state-of-the-art performance have high computational complexity as their architecture is neural. On the other hand, the other set of methods focuses on a single type of feature such as shape or frequency that can restrict the performance.

However, the recent success of convolutional neural networks for time series data showed that random convolutional kernels (such that 1D convolutions) allow to achieve state-of-the-art accuracy with much less computational time and manual feature engineering. The main purpose of this work is to compare ROCKET model (Dempster et al., 2019) based on aforementioned 1D convolutions with some general machine learning and statistical models for regression in terms of time complexity and prediction quality using open source Airline time series data (Box & Jenkins, 1990).

1. Introduction

Many methods for time series regression that achieve state-of-the-art accuracy have high computational complexity, requiring significant training time even for smaller datasets, and simply do not scale to large datasets. Mostly, they are based on neural architectures like LSTM (Greff et al., 2015) or GRU (Chung et al., 2014).

The other bundle of methods for time series regression can be characterized by focusing on a single representation feature such as shape, frequency, variance or their combination. These features can be constructed manually or generated in automatic way and fed into general machine learning regression models e.g. Decision Trees (Rokach & Maimon, 2005) or Ridge regression model (Marquardt & Snee, 1975).

Generally speaking, manual constructing features can be time consuming and intricate. That is why, convolution operation can be introduced. From the research in the area of image processing we know that convolution kernels constitute a single mechanism which can capture many of the features that would require their own specialized techniques to be constructed.

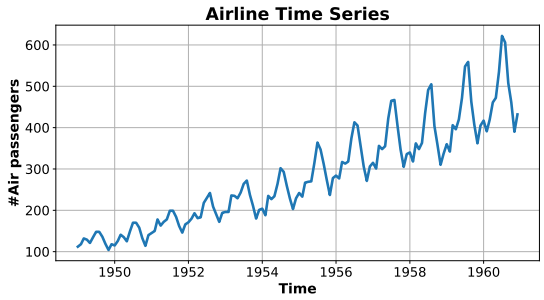

Figure 1. The classic Box & Jenkins airline data. Monthly totals of international airline passengers, 1949 to 1960. Dimensionality: univariate Series length: 144 Frequency: Monthly Number of cases: 1. This data shows an increasing trend, non-constant (increasing) variance and periodic, seasonal patterns.

According to the latest research, using random convolution kernels can get the same or even better performance results than learning kernels from data (Dempster et al., 2019). It means that the model transforms raw time series into features using random 1D convolutions to train a model of the second layer. The model can be a linear regression or more complex regressor based on Decision Trees (Rokach & Maimon, 2005) or Support Vector Machines (Evgeniou & Pontil, 2001). This overall framework is called ROCKET (Dempster et al., 2019).

It is worth mentioning that, one can use state-of-the-art machine learning (ML) models such as XGBoost (Chen & Guestrin, 2016), Random Forest (Breiman, 2001) and Support Vector Regression (Basak et al., 2007) without any neural preprocessing. However, it is necessary to construct hand-crafted features to use these general purpose ML models. Some examples of these features can include statistical values like mean or median, maximum, minimum vales, the number of zeros. Information about time like day, month, year can also be included in these models to improve the

[1]Department of Computational Intelligence, Skolkovo Institute of Science and Technology, Moscow, Russia. Correspondence to: Albert Sayapin <Albert.Sayapin@skoltech.ru>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

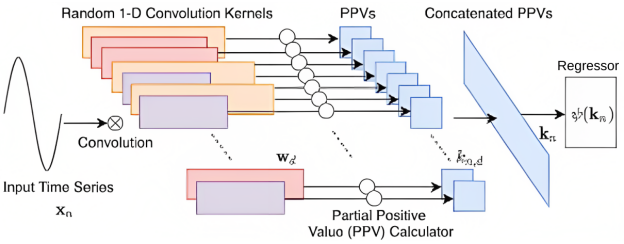

Figure 2. ROCKET model for regression. Random convolution kernels for feature extraction and regression. (The image is based on (Salehinejad et al., 2022))

final accuracy.

The main purpose of the paper is to present a comparison of different models for time series forecasting/regression. Namely, ROCKET model (Figure 2) is compared against several competitors like: Moving Average (MA) (Tunnicliffe Wilson, 2016), XGBoost (Chen & Guestrin, 2016), LSTM (Greff et al., 2015). They are compared based on computational complexity and prediction quality.

The rest of this paper is structured as follows. In section 2, the review of relevant related works is presented. Section 3 explains plan for the project in more details. In section 4 the set of conducted experiments and model settings are described. Section 5 gives some interpretations on the results of the experiments. Conclusion of results is depicted in section 6. Finally, the paper is summed up in section 7.

One can find implementation and evaluation details in the GitHub repository: `https://github.com/AlbMLpy/ROCKET_MSD_2022`

2. Literature Review

In the last decade, there has been an increasing interest in time series analysis research, in particular time series classification (TSC) (Bagnall et al., 2017; Dempster et al., 2019) and time series forecasting (TSF) (Hyndman, 2019; Tunnicliffe Wilson, 2016). TSC is the task of predicting a discrete label for a time series that classifies the time series into some finite discrete categories. On the other hand, TSF aims to predict future values of a series based on recent or seasonal values.

Generally speaking, Time Series Regression (TSR) is the problem of how to predict numerical values that depend on the whole series, rather than depending more on recent than past values. With respect to TSF, regression usually means fitting the historical time series data with a regression model such as ARIMA (Tunnicliffe Wilson, 2016).

Classical regression models are designed for tabular data. These models learn a mapping function from input features to the target variable. These features typically do not take into account the temporal dimension which is important for time series data. Hence, these models need to be adapted for TSR problems. One have to apply hand-crafted feature engineering in order to use such models. For example, XGBoost (Chen & Guestrin, 2016) is a decision tree based ensemble learning algorithm that aims to reduce the variance and bias. Different from RF (Breiman, 2001) that uses bagging, XGBoost uses gradient boosting with regularisation to avoid overfitting. Another model, Support Vector Regression (Basak et al., 2007) is designed for regression. SVR tries to fit the error rate within a threshold. It works by mapping the data into a higher-dimensional space so that it is linearly separable using a kernel function such as Gaussian Radial Basis Function. Then it fits a hyperplane through the data bounded by two boundary lines which are distance apart from the hyperplane. The boundary lines are formed by support vectors which are datapoints that are closest to the boundary.

Last but not least set of models that can be used for TSR is based on neural architectures. Deep learning models are capable of predicting both discrete labels (classification) and continuous values (regression). Fundamentally, the output of a neural network is a continuous value. Recently, several deep learning models have been developed and benchmarked for TSC (Fawaz et al., 2018; 2019). They can easily be adapted for TSR task using a linear activation layer in the end.

On the other hand, as time series have a sequential nature than one could use the power of sequential models like RNN (Lindemann et al., 2021) or LSTM (Lindemann et al., 2021). RNNs are able to capture nonlinear short-term time dependencies. They can be used to find and model relations and extract contextual information from time series. Based on this foundation, numerous extensions of RNN have been developed in recent years to tackle a wide range of problems. In (Motazedian & Safavi, 2019) a discrete wavelet transformation is integrated into an RNN to replace the regular activation functions. A comparison is drawn to classical activation functions, such as sigmoid as well as tangens hyperbolicus, and it is emphasized that an enhancement in modeling accuracy could be achieved due to an improved control of the information flow. Nevertheless, RNN have the disadvantage of vanishing gradients. Thus, non-stationary dependencies that occur over a long period of time are less well captured by RNN.

Thus, gating mechanisms are developed to replace the classical activation functions. LSTM cells possess three gates, an input, a forget and an output gate, that allow to make changes on a cell state vector that is propagated iteratively to capture long-term dependencies. This controlled informa-

tion flow within the cell enables the network to memorize multiple time dependencies with different characteristics.

Recently, (Dempster et al., 2019) proposed the ROCKET classifier that achieves state-of-the-art accuracy in TSC with a fraction of the computational expense of existing methods. ROCKET transforms time series using a large number of random convolution kernels and trains a ridge regression classifier. These kernels have random length, weights, bias, dilation, and padding, and when applied to a time series produce a feature map. Then the maximum value and the proportion of positive values are computed from each feature map, producing two real-valued numbers as features per kernel. With the default 10,000 kernels, ROCKET produces 20,000 features. Rocket was found to be the most accurate TSC classifier compared with other state-of-the-art models such InceptionTime (Fawaz et al., 2019). In this work, ROCKET is adapted by replacing the classifier with a regression model.

3. Project Plan

3.1. Experiment

The experiment consists of the following stages:

1. **Load and preprocess the Airline time series data** (Box & Jenkins, 1990). Figure 1 shows original Airline time series. Preprocessing depends on a model and will be described in section 3.3.3.
2. **Divide dataset into train, validation and test**. Validation is used to find optimal hyperparameters. After the hyperparameters are found a model is retrained on concatenated train and validation subsets and evaluated on test subset to get the final metrics.
3. **Find optimal hyperparameters of the following models:**
   - Moving average (MA(q)) (Tunnicliffe Wilson, 2016). Parameter - order q.
   - ROCKET + Ridge regression (Hoerl & Kennard, 2012). Parameter - the number of kernels.
   - XGBoost (Chen & Guestrin, 2016). Parameter - the number of estimators.
   - LSTM (PyTorch) (Greff et al., 2015). Parameter - the hidden dimensionality.
4. **For each model, get graphs of the target metric** - R^2 (Barten, 1987) from different forecast horizons (1 month, 2 months e.t.c.) and lengths of the history. Plot the influence of number of the kernels on the final result for ROCKET model.
5. **Calculate training time for all the models**. Compare them.
6. **Based on the results of the work, conclusions must be drawn**. Which model performed best on each of the forecast horizons? Reasonable assumptions must be put forward why the results are the way they are.

3.2. Hypotheses

1. What is the most computationally efficient model? MA?
2. What is the most accurate model? LSTM?
3. What is the optimal model? ROCKET?
4. How to construct features for XGBoost? What features are the best?
5. How to construct features for LSTM? What features are the best?
6. What if to use XGBoost as a regressor for ROCKET?

3.3. Peculiarities of the Experiment

There are some peculiarities of the experimental settings that are worth mentioning.

3.3.1. Problem of one time series

As there is the only one time series it is impossible to train a machine learning model as it is. Only statistical MA(q) model can be trained from one time series. That is why, in order to train the other models I had to generate more time series chunks with rolling window. The procedure generated several time series such that all of them have the same length of 10 timestamps (10 month). Therefore there are 134 time series generated. They represent the independent variable **X**. In order to get a target dependent variable **y**, I shifted target column by 10 points.

3.3.2. Data splitting

Data splitting on train, validation and test was conducted as the following: the first 91 rows of **X**, **y** (approximately 70% of the data) go to train subset, the next 5 rows of **X**, **y** (approximately 3.4% of the data) go to validation subset and the last 38 points (approximately 26.6% of the data) go to test subset. In order to get results for different history length the ratio of points in a train subset can be more or less than 70%, e.g. 60%.

3.3.3. Preprocessing and Feature engineering

First of all, let us talk about MA(q) model. This model does not need any extra features or elaborate preprocessing in order to train and predict. It just needs a column with values and timestamps of time series in order to train parameters.

Secondly, a ROCKET model needs extra preprocessing in order to be launched. It uses generated data **X** as an input. However, the first part of the model itself is a preprocessor that gives matrix of instances×features as an output. It can be used by the model of the second layer. In case of this paper, a Ridge regression model was used, because it is lightweight and interpretable.

Thirdly, an XGBoost model is similar to ROCKET and it uses generated data **X**. However, it order to increase potential quality automatic feature extractor - tsfresh was used. After that the most important features were selected to train XGBoost regressor. In total 117 features were used to train this model.

Finally, in order to train LSTM model not only rolling window was used but also features from timestamps like: day, week, sin_day_of_week, etc. In total 20 features were used to train LSTM model. However, not only these features can be used but also aforementioned automatic features from tsfresh.

4. Conducted Experiments

4.1. Prediction Quality Comparison

The most important experiment is comparison of 4 models in terms of performance/prediction quality. There are different strategies of how to compare the models. The one I have chosen is based on using R^2 (Barten, 1987) from different forecast horizons (1 month, 2 months e.t.c.) and lengths of the history. The main idea is to train a model on some data of fixated history length (e.g. 91 month. Section 3.3.2). After that to gather forecasts for different horizons like 1, 2, 3, etc months.

What is really important, in order to use R^2 score one have at least 2 data points. That is why, I applied the following procedure to generate R^2 scores: firstly, generate prediction for the 1 and 2 months from a test set and calculate metric for two two-dimensional points - predicted and target, then generate prediction for the 1, 2, 3 months and calculate metric for two three-dimensional points and so on. After all, you get 38 metric results (Section 3.3.2) that show the dynamics of the forecast. This allows to compare the models in equal settings.

Moreover, in order to get more experimental results and understand the prediction dynamics of different models more precisely, one can choose different train history lengths. For the sake of this experiment, 60% and 70% of data were used to construct train subset and get extra results prediction quality of model depending on prediction horizon.

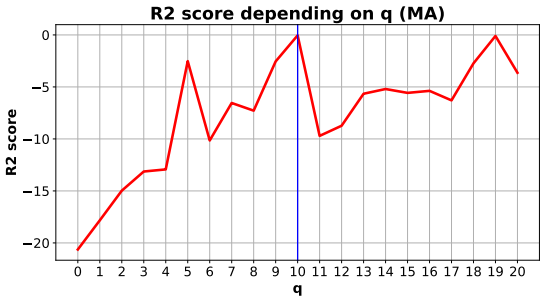

Figure 3. MA (q) model, train ratio = 0.7. Dependence of R^2 score on order q on validation subset. The optimal hyperparameter is equal 10.

4.2. Computational Performance Comparison

The one more experiment is connected with computational properties of time series prediction models. As we know, neural models usually take more time to train and it can become a bottleneck e.g. in real industrial problem. That is why, the main idea of the experiment to get if a ROCKET model is faster than LSTM in terms of training time. In addition to it, the comparison with the MA(q) and XGBoost are also considered for consistency of the experiment.

5. Results of the Experiments

One can find implementation and evaluation details in the GitHub repository: `https://github.com/AlbMLpy/ROCKET_MSD_2022`

5.1. Hyperparameters Search

This section depicts how target metric (R^2) depends on hyperparameters on validation subset and comparison of prediction and original time series on a test subset.

5.1.1. Moving Average

Figure 3 shows the behavior of target metric depending on the order of Moving Average model. As we can see the higher the order the better the result up to the moment of q = 10. After that we can see a decrease in the quality. Figure 4 shows that MA(q) model prediction is considerably worse than the original data. We do not see any trends or peaks. The pattern was lost.

5.1.2. ROCKET

Figure 5 shows the behavior of target metric depending on the number of kernel in ROCKET model. The other parameter are fixated both for preprocessing part and predicting Ridge regressor. We can see that all the values of parameter

starting from 2000 give approximately the same results and the number of kernels influences the metric results not so drastically as order q for MA(q) model. In comparison to the previous results of MA(q) model, we can see on Figure 6 that ROCKET model prediction is comparable to original data. We can find trends and peaks in almost the same location. The overall pattern is saved by the ROCKET model and the quality of prediction is likely to be high, especially for the horizon of the first 10 months.

5.1.3. XGBoost

Speaking of the next model, on Figure 7 we can see almost the same behavior as for ROCKET model. The metric results almost do not change after some threshold equal to 80 estimators for XGBoost model. XGBoost prediction given on Figure 8 is pretty close to original data at least compared to MA(q) model. We can find some trends and peaks that do not coincide with real data but still are very close. The overall pattern is saved in average and the quality of this prediction is probably to be higher than for MA(q). However, the prediction is worse than of the ROCKET model at first sight.

5.1.4. LSTM

Generally speaking, the behavior of LSTM model is the most unpredictable. As we can see from Figure 9 the optimal hidden dimension is equal to 16. Nothing strange up to here. However, if we look at Figure 10, we can see that prediction of the LSTM model is constant everywhere and is equal to 130 approximately. My assumption is that there is not enough data to train this neural model that is why the possible optimum lies in predicting something constant. On the other hand, there might be some implementation peculiarities that influenced the solution.

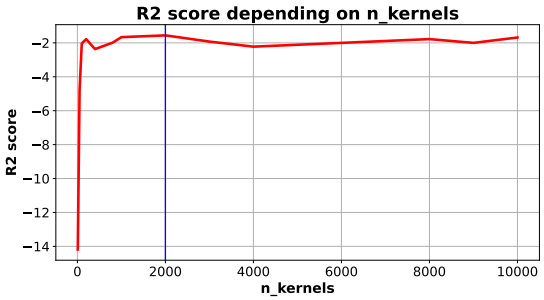

Figure 5. ROCKET model, train ratio = 0.7. Dependence of R^2 score on the number of kernels on validation subset. The optimal hyperparameter is equal 2000.

5.2. Prediction Quality

As we have already seen from Figure 6 for example, the model prediction is different from the original data for all the models. However, in order to compare the prediction quality of different models one should look at how the target metric changes with the growth of forecast horizon measured in months for Airline time series.

From the Figure 11 and Figure 12 we can see that LSTM shows the worst results. It is clear that the constant forecast will not lead to good quality for non-constant target.

The second model in terms of bad accuracy is MA(q). As we have seen on Figure 4 the prediction is close to a constant value. Therefore, the metric score should behave as LSTM models score. It is just the behavior we see on Figure 11.

One of the best models is a ROCKET and it shows pretty good results especially for the experiment with train ratio of 0.7. As we see the R^2 score lies close to 0 and above which tells about a good quality regardless the forecast horizon. Speaking of Figure 12 the results are also better than for the previous competitors.

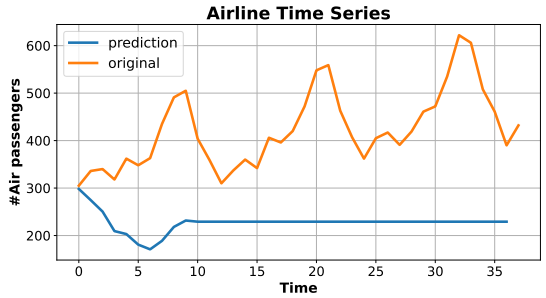

Figure 4. MA (q) model, train ratio = 0.7. Prediction of the model vs original time series data.

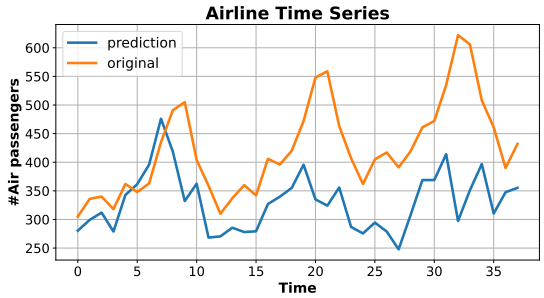

Figure 6. ROCKET model, train ratio = 0.7. Prediction of the model vs original time series data.

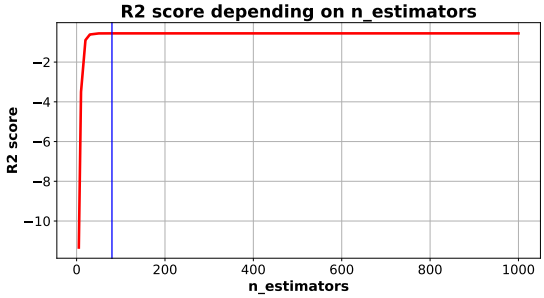

Figure 7. XGBoost model, train ratio = 0.7. Dependence of R^2 score on the number estimators on validation subset. The optimal hyperparameter is equal 80.

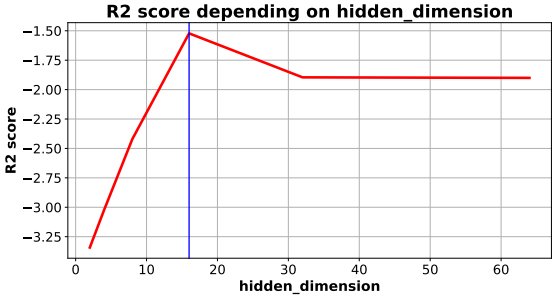

Figure 9. LSTM model, train ratio = 0.7. Dependence of R^2 score on the hidden dimension on validation subset. The optimal hyperparameter is equal 16.

Finally, the best model is XGBoost as we can sum up from Figure 11 and Figure 12. It lies in the neighbourhood of zero all the time and even more than zero for some horizons.

5.3. Computational Performance

The other important question is connected with computational time of the competing models. How much time do they need to train?

You can find the answer to the question in the Table 1. As we can see the fastest model is MA. Well, it is expected due to the nature of the model. The next model is ROCKET and it lived up to its name. XGBoost is 1.5 times slower than ROCKET in average as we can see from 2 columns of Table 1. After all, the slowest model is LSTM. It is not a surprise, as we have already discussed the neural models to have computational bottlenecks.

Table 1. Comparison of training time (seconds) for different models.

MODEL	TRAIN RATIO = 0.6	TRAIN RATIO = 0.7
MA	0.003	0.004
ROCKET	0.057	0.0517
XGBOOST	0.068	0.108
LSTM	14.443	16.449

6. Conclusion on the Results

Now, it is high time to discuss the results and answer the questions I have pointed out in Section 3.2.

1. Actually, the most computationally efficient model is MA as expected. It is a statistical model that needs only values of time series and timestamp identification. No preprocessing is needed.

2. The most accurate model in terms of R^2 metric is XGBoost. It is not LSTM as I expected, but the problem may be little training data or implementation issues. All in all, XGBoost is a general tool for classification and regression problems that works well. Time series forecasting problem is not an exception.

3. The optimal model in terms of both training time and prediction quality is a ROCKET as was assumed. It is

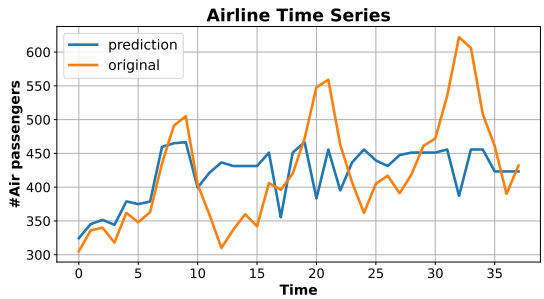

Figure 8. XGBoost model, train ratio = 0.7. Prediction of the model vs original time series data.

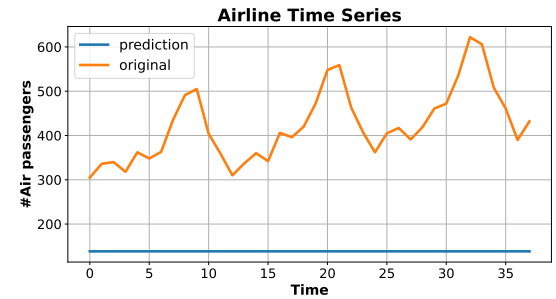

Figure 10. LSTM model, train ratio = 0.7. Prediction of the model vs original time series data.

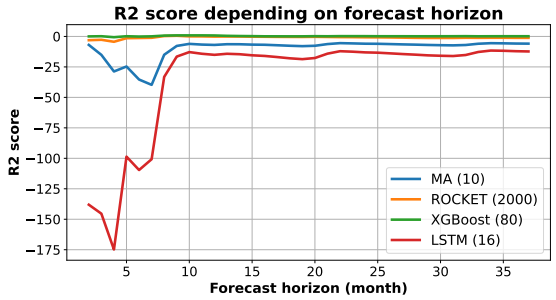

Figure 11. Train ratio = 0.7. Dependence of R^2 score on forecast horizon (measured in months) for competing models.

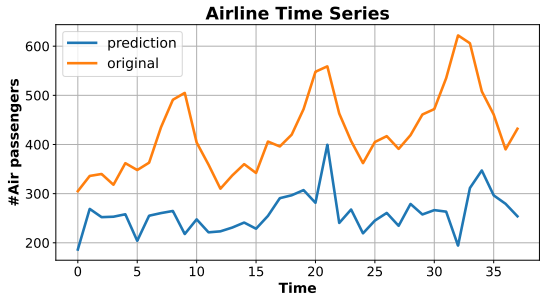

Figure 13. ROCKET + XGBoost model, train ratio = 0.7. Prediction of the model vs original time series data.

faster than XGBoost and shows almost similar results.

4. One can construct features either manually or automatically using tsfresh. The best features can be selected based on heuristics or by measuring the relation between feature and target. Some examples of the best features are maximum/minimum number of passengers, sum and mean values.

5. The features for LSTM model can be constructed the same way as for XGBoost model.

6. As we can see from Figure 6 and Figure 13 using XGBoost as a model of the second step does not lead to better prediction quality.

It is worth mentioning that there is no visible difference depending on the length of the history. One can find additional plots in Appendix B

7. Conclusion

Convolutional kernels are a single, powerful instrument which can capture many of the features used by existing methods for time series forecasting. The comparison of different models for time series forecasting like ROCKET, MA, XGBoost, LSTM showed that the most optimal model in terms of computational complexity and prediction quality is ROCKET after all.

It proves one more time that random kernels have very low computational requirements, making learning and forecasting extremely fast. ROCKET makes key use of the proportion of positive values (or ppv) to summarise the output of feature maps, allowing a regressor to weight the prevalence of a pattern in a given time series. Training a simple linear ridge regressor using features generated from a bank of random convolution kernels (without training the kernels) is a fast and efficient approach for time series forecasting as a framework.

Despite the superior performance of Rocket in total, machine learning models such as XGBoost and Random Forest are equally competitive as well. Therefore, this suggests much research is needed to develop better algorithms to improve the accuracy on time series forecasting problems.

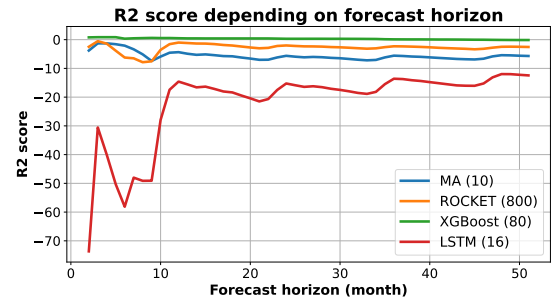

Figure 12. Train ratio = 0.6. Dependence of R^2 score on forecast horizon (measured in months) for competing models.

References

Bagnall, A., Lines, J., Bostrom, A., Large, J., and Keogh, E. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery*, 31, 05 2017. doi: 10.1007/s10618-016-0483-9.

Barten, A. P. The coefficient of determination for regression without a constant term. 1987.

Basak, D., Pal, S., and Patranabis, D. Support vector regression. *Neural Information Processing – Letters and Reviews*, 11, 11 2007.

Box, G. E. P. and Jenkins, G. *Time Series Analysis, Forecasting and Control*. Holden-Day, Inc., USA, 1990. ISBN 0816211043.

Breiman, L. Random forests. *Machine Learning*, 45:5–32, 10 2001. doi: 10.1023/A:1010950718922.

Chen, T. and Guestrin, C. Xgboost: A scalable tree boosting system. *CoRR*, abs/1603.02754, 2016. URL http://arxiv.org/abs/1603.02754.

Chung, J., Gülçehre, Ç., Cho, K., and Bengio, Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555, 2014. URL http://arxiv.org/abs/1412.3555.

Dempster, A., Petitjean, F., and Webb, G. I. ROCKET: exceptionally fast and accurate time series classification using random convolutional kernels. *CoRR*, abs/1910.13051, 2019. URL http://arxiv.org/abs/1910.13051.

Evgeniou, T. and Pontil, M. Support vector machines: Theory and applications. volume 2049, pp. 249–257, 09 2001. ISBN 978-3-540-42490-1. doi: 10.1007/3-540-44673-7_12.

Fawaz, H. I., Forestier, G., Weber, J., Idoumghar, L., and Muller, P. Deep learning for time series classification: a review. *CoRR*, abs/1809.04356, 2018. URL http://arxiv.org/abs/1809.04356.

Fawaz, H. I., Lucas, B., Forestier, G., Pelletier, C., Schmidt, D. F., Weber, J., Webb, G. I., Idoumghar, L., Muller, P., and Petitjean, F. Inceptiontime: Finding alexnet for time series classification. *CoRR*, abs/1909.04939, 2019. URL http://arxiv.org/abs/1909.04939.

Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., and Schmidhuber, J. LSTM: A search space odyssey. *CoRR*, abs/1503.04069, 2015. URL http://arxiv.org/abs/1503.04069.

Hoerl, A. and Kennard, R. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12: 55–67, 04 2012. doi: 10.1080/00401706.1970.10488634.

Hyndman, R. A brief history of forecasting competitions. *International Journal of Forecasting*, 36, 06 2019. doi: 10.1016/j.ijforecast.2019.03.015.

Lindemann, B., Müller, T., Vietz, H., Jazdi, N., and Weyrich, M. A survey on long short-term memory networks for time series prediction. *Procedia CIRP*, 99:650–655, 2021. ISSN 2212-8271. doi: https://doi.org/10.1016/j.procir.2021.03.088. URL https://www.sciencedirect.com/science/article/pii/S2212827121003796. 14th CIRP Conference on Intelligent Computation in Manufacturing Engineering, 15-17 July 2020.

Marquardt, D. and Snee, R. Ridge regression in practice. *American Statistician - AMER STATIST*, 29:3–20, 02 1975. doi: 10.1080/00031305.1975.10479105.

Motazedian, Z. and Safavi, A. A. Nonlinear and time varying system identification using a novel adaptive fully connected recurrent wavelet network. In *2019 27th Iranian Conference on Electrical Engineering (ICEE)*, pp. 1181–1187, 2019. doi: 10.1109/IranianCEE.2019.8786669.

Rokach, L. and Maimon, O. *Decision Trees*, volume 6, pp. 165–192. 01 2005. doi: 10.1007/0-387-25465-X_9.

Salehinejad, H., Wang, Y., Yu, Y., Jin, T., and Valaee, S. S-Rocket: Selective Random Convolution Kernels for Time Series Classification. *arXiv e-prints*, art. arXiv:2203.03445, March 2022.

Tunnicliffe Wilson, G. Time series analysis: Forecasting and control,5th edition, by george e. p. box, gwilym m. jenkins, gregory c. reinsel and greta m. ljung, 2015. published by john wiley and sons inc., hoboken, new jersey, pp. 712. isbn: 978-1-118-67502-1. *Journal of Time Series Analysis*, 37:n/a–n/a, 03 2016. doi: 10.1111/jtsa.12194.

A. Team Description

Albert Sayapin:

- Organized the paper.
- Conducted all the experiments.
- Made conclusions on the results.
- Prepared project presentation.

B. Additional Plots

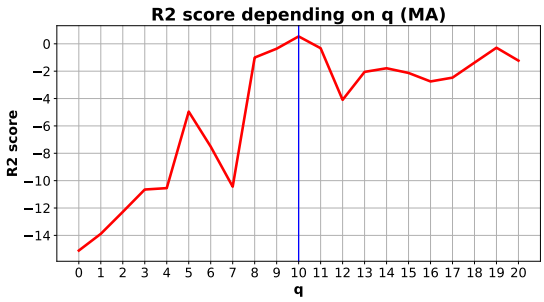

Figure 14. MA (q) model, train ratio = 0.6. Dependence of R^2 score on order q on validation subset. The optimal hyperparameter is equal 10.

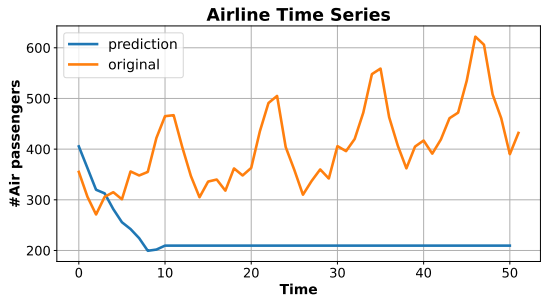

Figure 15. MA (q) model, train ratio = 0.6. Prediction of the model vs original time series data.

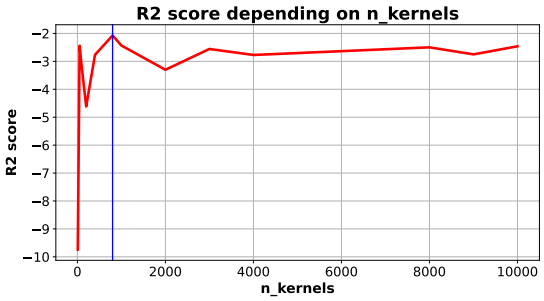

Figure 16. ROCKET model, train ratio = 0.6. Dependence of R^2 score on the number of kernels on validation subset. The optimal hyperparameter is equal 800.

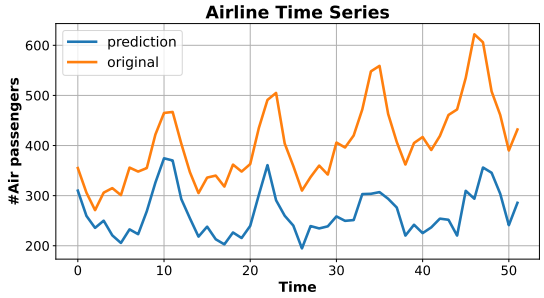

Figure 17. ROCKET model, train ratio = 0.6. Prediction of the model vs original time series data.

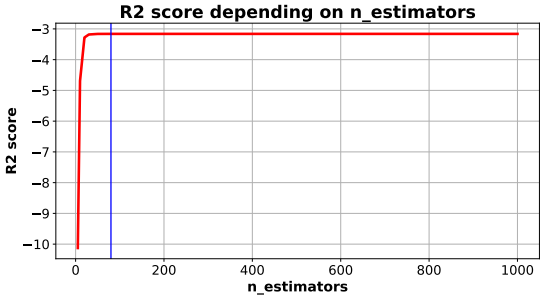

Figure 18. XGBoost model, train ratio = 0.6. Dependence of R^2 score on the number estimators on validation subset. The optimal hyperparameter is equal 80.

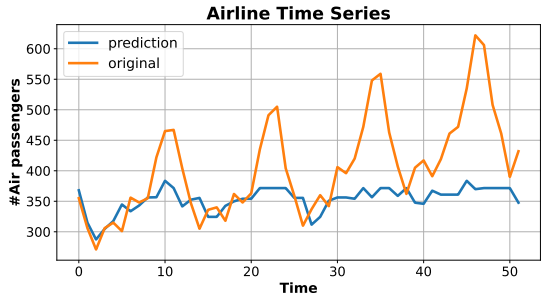

Figure 19. XGBoost model, train ratio = 0.6. Prediction of the model vs original time series data.

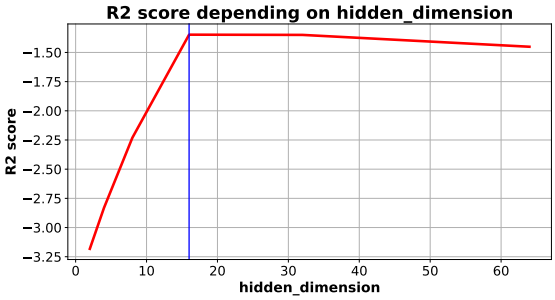

Figure 20. LSTM model, train ratio = 0.6. Dependence of R^2 score on the hidden dimension on validation subset. The optimal hyperparameter is equal 16.

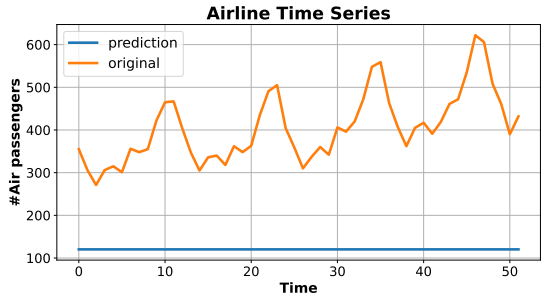

Figure 21. LSTM model, train ratio = 0.6. Prediction of the model vs original time series data.