Time Series Forecasting - Challenge 2 Report

Anne-Lys LANORE, Jakub JASTRZEBSKI, Maxime LAFONT, Ahmed HARRABI - Team Name: kuba

Introduction

This document presents the outcomes of the second challenge conducted in the Artificial Neural Networks and Deep Learning course for the academic year 2023-2024 by the group of Anne-lys Lanore, Maxime Lafont, Ahmed Harrabi and Jakub Jastrzebski (Team Name: kuba). Task of homework was to develop a forecasting model predict future samples of the input time series.

The dataset consists of variable-length time series, uniformly padded to a length of 2776, resulting in a Nx2776 array. The dataset includes time series from 6 categories (Fig. 1.). The dataset contains mono-variate time series from diverse domains, requiring a model to generalize across these for predicting future samples in the hidden test set. The challenge was divided into two phases, in the first phase we have to predict the first 9 values of 60 time series (10 for each category) of length 200 time steps, and in the second phase we have to predict the 18 first values of the 60 time series.

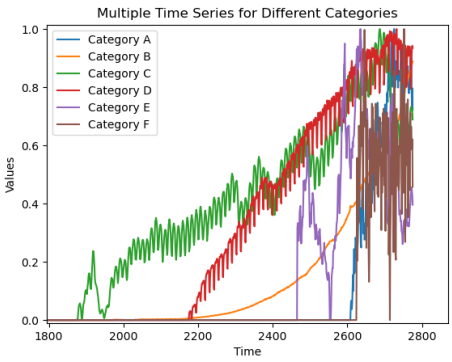

Fig. 1. Zoom of the not padded window of data series of 6 different categories

We tried different approaches to the models and data which are detailed in this report.
We based our work on the lectures given by Professor Matteucci, [5],[4] and Eugenio Lomurno laboratories [1],[2],[3].

Data Preparation

The project utilized a dataset consisting of 48,000 time series, each with a max length of 2776, summarized in Table 1. As we can see from these results, the time series are not equally distributed along the categories. This could be a source of issues for the challenge since in the hidden test set the categories are equally distributed. To better understand how data points are related to each other we plot the autocorrelation of one time series of one category in Fig. 2, all time series have a overall similar autocorrealation.
We see that the autocorrelation function has a plateau around the 150 lags, this observation will be useful for the choice

Category	Count	Min	Max	Average
A	5728	46	1943	278.18
B	10987	42	1484	165.94
C	10017	42	2708	208.15
D	10016	42	2641	216.99
E	10975	42	2776	163.05
F	277	24	1068	282.88

Table 1. Summary of Timeseries Data Across Different Categories, Count, Min, Max and Average denotes values across timeseries timestamps.

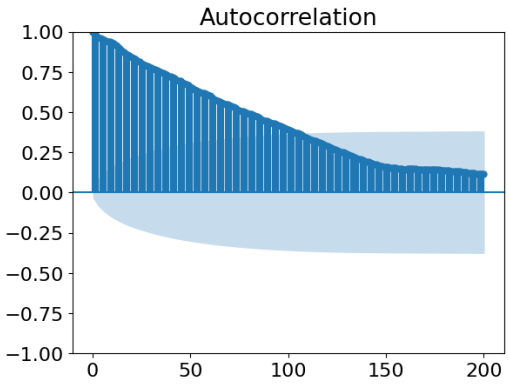

Fig. 2. Shape of the autocorrelation

of the window size. Since the time series provided are of different length, we decided to apply a window on each of these series to have a consistent input shape to the model for each time series. By analyzing the autocorrelation plot, we chose a window size of 100 time steps, 9 for the telescope and 9 for the stride. When a time series was not long enough to be windowed, we decided to just discard it. We could have chosen a wider window but increasing the window means having less data in our training set and discarding more "too short" time series.

Once every time series of each category was sampled, we constructed the training and validation sets by doing the usual split $80\%/20\%$. We chose to implement an overall model meaning that we did not make a different model for each category, this choice will be discussed later. Taking into account this fact the training and validation sets contain series from each category and with the same proportion as in the given dataset.

Model Architecture

A. Best Perfroming Model.

For the final model we used LSTM units as they are particularly suited for time series purposes. We computed all the models using TensorFlow functional API. The final model is sketched in Figure 3. The input of this model is an ten-

sor of size (None, 100, 1) and the output of size (None, 9). The model is composed of two branches. The first one (red box) is utilizing the inherent memory of LSTM to retrieve patterns in the series and a skip connection to enhance the performance of the branch. The second branch (orange box) is performing a self attention mechanism to capture contextual relationship among time series. The results of these two branches are concatenated and dense layers are then added to reduce the dimensionality of the output shape. Others alternatives model have been used and are briefly described in the next subsection.

B. Optimization.

We used the Adam optimizer and implemented Early Stopping callback with patience of 4 and a ReduceLROnPlateau callback with patience 2, an initial learning rate of 0.001, minimal one of 10^{-5} with a decay rate of 0.1. The patience are low because the training of LSTM is very slow. The batch size was of size 128 and maximal number of epochs was 200.

C. Other Models.

Another slightly less complex model that we tried combined a Bidirectional LSTM layer with a Self-Attention Mechanism and Conv1D layers, capturing dependencies and focusing on relevant sequence parts. Using an input of size (None, 100), it integrated these elements through concatenation and reshaping to give an output of size (None, 9).

Finally, we tried to apply the Transformer architecture to this problem since it supposedly achieves good results. We even tries different model based on transformer such as Informer model, which is more specific to Time series forecasting scenarios in taking account of the temporal nature in sequential data. However we struggled a lot to make the embedding of the input efficient and this solution was not adopted.

In the end the information about categories were not used in any of our models. We could have made a classification model to predict the categories of each time series, and then used the output of this model for a forecasting model, specialized for each category. We did not this option because there were a lot of models to train and we believe the architecture that we built was able by itself to differentiate the categories in its inner structure.

Performance

The models were evaluated on the Mean Squared Error (MSE), as mentioned in the instructions of the challenge, so all results are computed according to this metric.

D. First Phase.

During the 1st Phase we achieved a MSE of **0.00539571** for our best model, the one described in Fig 3. This level of accuracy, especially in time series forecasting, can be considered highly effective. It shows that the model is well-tuned and

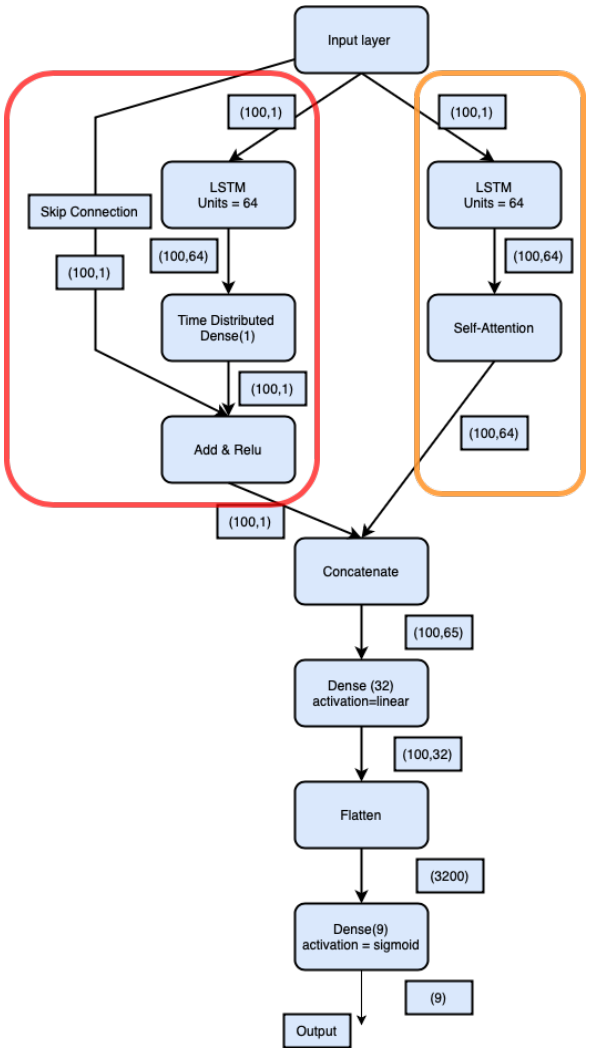

Fig. 3. Architecture of the final LSTM model

capable of capturing the underlying patterns in the data with minimal deviation. Overall, those were encouraging results for the final phase.
The second model described above was achieving a MSE of **0.0061147772**.

E. Final Performance - Second Phase.

The final best performance during the second phase of the competition on Codalab was **0.01020699**. Note that this model is in fact the same we used in the first phase, we predicted the 9 values for each time series and used these 9 predicted values to predict the 9 values after. The performance is "logical" since we double the number of samples to be predicted so the error is also doubled.
Note that we tried to train the same model with an output of 18 samples and a window of 120 and we had a worse performance, **0.0120964153**.

Possible Improvements

F. Handling Short Time Series.

When we apply the window we discard time series which are too small, this choice is of course debatable. Another

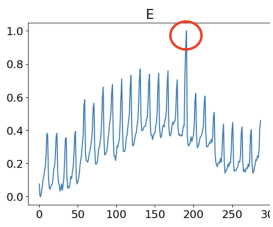

Fig. 4. A potential outlier

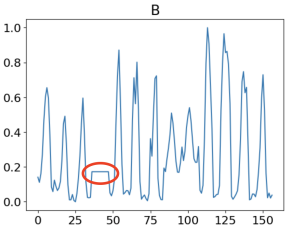

Fig. 5. Missing value ?

solution could have been to pad this short time series to make them long enough. This solution has been explored but this implied using a Masking Layer to let the network know that these zeros are not to be taken into account in the overall pattern of the time series, which would lead to misleading results. However using a Masking Layer made the training much slower considerably in addition to it already being very slow because of LSTM units.

G. Handling Outliers and Missing Values.

The initial dataset is composed of series with strange behavior. As shown in figures Fig. 4 and Fig. 5, some time series have peaks which can be associated to outliers or even plateau which can represent missing values.

A careful consideration of outliers and missing values could lead to better overall performance. We tried at the beginning to use Robust Scaling to take care of outliers, unfortunately the results were not that great and we decided to remove it from our data preprocessing. Another way to take care of them that has been tried is to declare these peaks and plateau as NaN and to interpolate this value using the function $interpolate$ of Numpy, unfortunately it was not making the results better.
In addition, preprocessing techniques like smoothing function such as B-spline, can be used to reduce the fluctuation on the data which may improve the learning algorithm and the overall performance.

H. Handling Under Represented Categories.

Finally the category F is clearly under represented as depicted in Table 1, knowing that the hidden test is balanced with respect to the categories, we tried to oversample category F and undersample the others categories to enhance the performance but the results were not conclusive.
An improvement of this project could be a better data preprocessing, including better detection of outliers and missing values, and a better handling of the small time series.

Conclusions

In this report we presented the result of the second challenge conducted in the Artificial Neural Networks and Deep Learning course. The homework consisted in developing a forecasting model to predict future samples of the input time series across six categories. Our approach was to make a single model to predict every time series using a dual-branch neural network featuring a windowing technique, LSTM units for time series pattern recognition and a self-attention mechanism for contextual understanding.
The final model showcased significant predictive accuracy with a very satisfying MSE of **0.01020699** in the final phase, demonstrating its ability to effectively forecast future time series samples.
During this challenge, we got the opportunity to experiment with different models of varying complexity, even dabbling with a transformer model although not conclusive.

We have encountered challenges in managing data complexities such as outliers, missing values, and category imbalances. Future improvements could focus on enhanced data preprocessing and exploring transformer architectures further.

In the end, the project provided valuable insights into deep learning applications in time series forecasting, highlighting the importance of comprehensive model design and data analysis in these fields.

References

[1] Eugenio Lomurno. *[2023-2024] AN2DL - Lecture 7*. 2023. URL: https://drive.google.com/drive/folders/1BwXymoD22VGvXNzyNhB4n6Wao4iTECcG?usp=drive_link.

[2] Eugenio Lomurno. *[2023-2024] AN2DL - Lecture 8*. 2023. URL: https://drive.google.com/drive/folders/1PGq0ycaBHujmvGQcnsWVEseh5P2BLgky?usp=drive_link.

[3] Eugenio Lomurno. *Logbook*. 2023. URL: https://docs.google.com/document/d/1qzSItCkkhiy9YkmOHlv6EeHsla4DXr-38NEe2Amn3TE/edit.

[4] Matteo Matteucci. *Beyond Sequence 2 Sequence Learning*. 2023. URL: https://chrome.deib.polimi.it/images/4/48/AN2DL_06_2324_AttentionAndTrasformers.pdf.

[5] Matteo Matteucci. *Seq2Seq and Word Embedding*. 2023. URL: https://chrome.deib.polimi.it/images/4/4b/AN2DL_04_2324_RecurrentNeuralNetworks.pdf.

Contributions

- **Anne-lys**: Model design and training, Attention and Bidirectional model, Report
- **Maxime**: Final model, Model design, try on Transformer, Report
- **Ahmed**: Pre-processing, Data analysis and Visualization, Informer Model, Report
- **Jakub**: Robust Scaler, Attention and Bidirectional model, Model training, Report