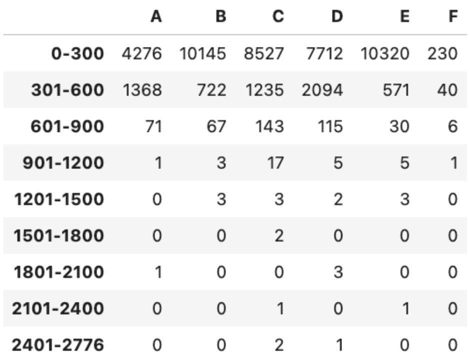
Artificial Neural Networks and Deep Learning 2023 Second Homework

Mehmet Emre Akbulut, Gamze Güliter, Eren Senoglu, Yavuz Samet Topcuoglu

December 2023

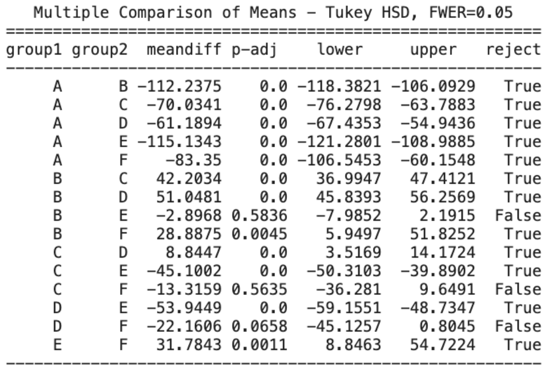
1 Data Analysis

Given the dataset's composition of 48,000 time series with different lengths and diverse categories, we wanted to analyze the data before starting any training. Our analysis highlighted the wide range of lengths present within the dataset. Namely, the shortest time series has a length of 24, and the longest has 2776. We've generated a table that displays the length distribution of time series categorized by their respective categories in order to provide a better view for further investigation and analysis.



	A	B	C	D	E	F
0-300	4276	10145	8527	7712	10320	230
301-600	1368	722	1235	2094	571	40
601-900	71	67	143	115	30	6
901-1200	1	3	17	5	5	1
1201-1500	0	3	3	2	3	0
1501-1800	0	0	2	0	0	0
1801-2100	1	0	0	3	0	0
2101-2400	0	0	1	0	1	0
2401-2776	0	0	2	1	0	0

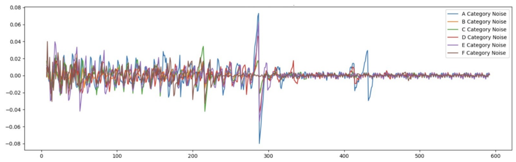
The table shows that a majority of time series within the training set have been zero-padded, since their original lengths are less than 300. We can also say that category F is dominated by other categories. Since a similarity in lengths across different categories is observed, we wanted to further examine the necessity of utilising category information during the training. For that purpose, we deployed Tukey's HSD Test in order to investigate pairwise similarities among categories.



```
  Multiple Comparison of Means - Tukey HSD, FWER=0.05
=========================================================
group1 group2  meandiff p-adj     lower     upper  reject
---------------------------------------------------------
     A      B -112.2375    0.0 -118.3821 -106.0929   True
     A      C  -70.0341    0.0  -76.2798  -63.7883   True
     A      D  -61.1894    0.0  -67.4353  -54.9436   True
     A      E -115.1343    0.0 -121.2801 -108.9885   True
     A      F    -83.35    0.0 -106.5453  -60.1548   True
     B      C   42.2034    0.0   36.9947   47.4121   True
     B      D   51.0481    0.0   45.8393   56.2569   True
     B      E   -2.8968 0.5836   -7.9852    2.1915  False
     B      F   28.8875 0.0045    5.9497   51.8252   True
     C      D    8.8447    0.0    3.5169   14.1724   True
     C      E  -45.1002    0.0  -50.3103  -39.8902   True
     C      F  -13.3159 0.5635   -36.281    9.6491  False
     D      E  -53.9449    0.0  -59.1551  -48.7347   True
     D      F  -22.1606 0.0658  -45.1257    0.8045  False
     E      F   31.7843 0.0011    8.8463   54.7224   True
---------------------------------------------------------
```

The test results show that for most pairs, there is a difference in length distribution between the two categories, rejecting the hypothesis 'Two categories are not different in terms of length distribution'. The hypothesis is accepted only for 3 pairs out of 15. And 2 in these 3 pairs, one of the groups is category F. The reason category F is deemed to be similar to C and D can be explained by the absurdly few number of samples in category F. The most significant insight that can be seen from this test is the confirmation that B and E have comparable length distributions. However, since all other pairs have distinct distributions according to the test, we believe that including category information alongside time series data during training could be useful.

Moreover, we concluded that an effective strategy would be to break down the signals into three components: **the trend**, which represents the series' ascending or descending pattern; **seasonality**, reflecting the clearly observable periodic component in our series; and **noise**. Our primary focus was on the first two elements – trend and seasonality – as these are key aspects we aimed to train the network on, considering that noise, in theory, is not predictable. We couldn't achieve better result in practice, however, in theory, this approach may lead better performance with a proper model.



2 Preprocessing

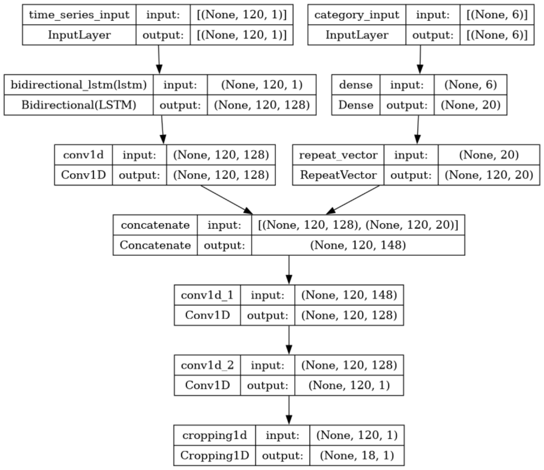
The padding is calculated in a way that the last point of each sequence is always included in the prediction range (telescope), regardless of the window, telescope, or stride settings. And it will be made only to the beginning of the sequences with zeros.

The sequences are created using only the valid periods of the time series. Each time series is divided into sequences of length equal to the window. Each sequence is shifted with the stride to create a new sequence. Since the context and categories of the data are unknown, we couldn't filter out the outliers by ourselves. Therefore, we opted for a statistical approach. We applied RobustScaler to make the model less sensitive to outliers. To make features comparable, we used StandardScaler in the preprocessing step. To tackle the right-skewed distribution in the data, we employed PowerTransformer. However, as these transformations failed to yield significant improvements to the predictions, we decided to discard them.

3 Models

3.1 BiLSTM with Convolutional Layers

For this model, Lab Session 7 was very helpful. We utilised the codes shared during the lecture to develop this model. This model has two input layers. One for time series input and the other for categorical information. The time series branch has a Bidirectional LSTM and a convolutional layer on top, whereas the category branch has only a dense layer. These two branches are concatenated and then followed by more convolutional layers.


time_series_input | input: | [(None, 120, 1)]
InputLayer | output: | [(None, 120, 1)]
category_input | input: | [(None, 6)]
InputLayer | output: | [(None, 6)]
bidirectional_lstm(lstm) | input: | (None, 120, 1)
Bidirectional(LSTM) | output: | (None, 120, 128)
dense | input: | (None, 6)
Dense | output: | (None, 20)
conv1d | input: | (None, 120, 128)
Conv1D | output: | (None, 120, 128)
repeat_vector | input: | (None, 20)
RepeatVector | output: | (None, 120, 20)
concatenate | input: | [(None, 120, 128), (None, 120, 20)]
Concatenate | output: | (None, 120, 148)
conv1d_1 | input: | (None, 120, 148)
Conv1D | output: | (None, 120, 128)
conv1d_2 | input: | (None, 120, 128)
Conv1D | output: | (None, 120, 1)
cropping1d | input: | (None, 120, 1)
Cropping1D | output: | (None, 18, 1)

The input layer in the time series is shaped as (120, 1) due to setting a window size of 120 for this specific run. Likewise, the output has a shape of (18, 1) because the telescope variable is set to 18 (to predict the next 18 data points).

During the first phase of the competition, this model proved to be our best performer, with MSE score of 0.0054.The result was attained using the following parameters: window = 100, telescope = 9, stride = 20. We have also cascaded BiLSTM's, increased their size and appended attention layers to the architecture to increase complexity to better capture the distribution and long range dependencies. As well as this, dropout and regularization is increased as a balance mechanism. However, the cascaded BiLSTM's didn't surpass the best model in the submissions.

3.2 RESNET with LSTM

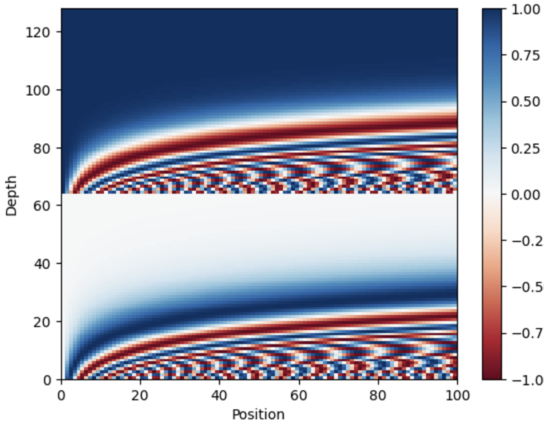
Even though bidirectional LSTM is good at understanding sequential data like time series by capturing patterns in both forward and backward directions, the success of the model was lower than expected. Therefore, we added simplified ResNet blocks the model, compared with dropout layers with a rate of 0.2 where the dropout layers with the rate of 0.2 help prevent overfitting, making the network better at understanding new data. The goal for adding ResNet blocks to the bidirectional LSTM model was to upgrade the network's ability to understand complex details within the sequential data. However, despite the slight improvement in the model, the outcome did not change significantly leaving the MSE of 0.0074 and we decided to proceed with other models.

3.3 Transformer

After the lectures and lab session on Sequence to Sequence Learning and Transformers, we realised that using a transformer mechanism can indeed be useful for this competition. Since the transformers library provided by HuggingFace is not allowed, we needed to build it from scratch. Apart from lectures and labs, PyLessons website was genuinely helpful.

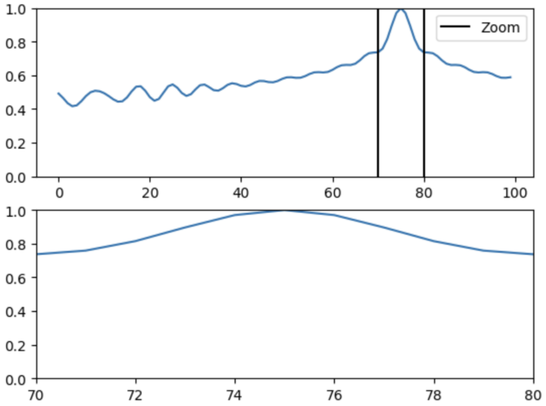
This model has an encoder but not a decoder. The encoder consists of a positional encoding layer, some encoding layers, and a drop-out layer.

Positional encodings were added to the input sequences to give the model information about the position of the tokens in a sequence.



This visualization helps in understanding how positional information is varied across different dimensions of the encoding.

Additionally, we can observe how positional encoding impacts a sequence with a similarity comparison.



In a sequence with length of 100, the element in the 75th position has the similarity to other positions shown in the graph above.

The number of encoding layers is determined with a parameter. Each encoder layer has a self-attention block and a feed-forward block. Self-attention blocks consist of multi-head attention layers and normalisation layers, whereas feed-forward blocks has sequential dense layers and normalisation layers.

This model was the second best during the first phase of the competition, and it provided an MSE score of 0.0055 in the submission.

4 Conclusion

In the end, our best performing submission was BiLSTM with Convolutional Layer with an MSE score of 0.010 in the final phase of the competition. Even though the transformer, that is built from scratch, is a larger model, it didn't manage to surpass the BiLSTM, getting a score of 0.011 in the final phase.

You can find all notebooks related to this challenge from this link.

5 Contributors

5.1 Mehmet Emre Akbulut

- Seasonality Analysis for the training data.
- New architecture research about LSTM and Convolution Layer
- Padding Tecniques Exploration
- Complex Model Research

5.2 Elif Gamze Güliter

- Implementation of ResNet. Researched ResNet implementation.
- Conducted research on LSTM and ResNet (Residual Neural Network) architectures.
- Conducted multiple model training sessions using various hyperparameters for BiLSTM and ResNet models.

5.3 Eren Şenoğlu

- Implemented Transformer and conducted experiments on it.
- Conducted experiments on data processing by utilizing scalers, and by transforming features.
- Trained ResNet and BiLSTM Convolutional Layers Model with different configurations.

5.4 Yavuz Samet Topcuoglu

- Data inspection and analysis with length distribution and HSD Test.
- Implementation of BiLSTM with Convolutional Layers Model.
- Implementation of helper functions: padding length calculation, sequence creation.
- Implementation of Transformer from scratch.