Mini-Project (ML for Time Series) - MVA 2023/2024

Guillaume Levy levyguillaume10@yahoo.fr
Clement Wang clementwang2001@gmail.com

January 9, 2024

1 Introduction and contributions

Despite the remarkable success of deep learning models across different data types, their performance often falters when confronted with time series data. One plausible explanation for this limitation lies in the representation of the data itself, typically presented as a sequence of floating-point numbers.

The concept behind Adaptive Brownian Bridge-based Symbolic Aggregation (ABBA) revolves around a more concise representation of time series data, aiming to capture the broader trends rather than focusing on values at specific timestamps. This methodology transforms the signal into a shorter sequence of symbols from a learned dictionary. This condensed representation mitigates computational costs, both during the training phase of neural networks and at inference. ABBA symbolic representation is employed for time series forecasting purposes in the rest of the report.

For this project, we reimplemented two papers, namely [1] and [2]. The first paper describes the functionality of ABBA in detail, while the second paper leverages ABBA's symbolic representation for training LSTM neural networks [5].

Our motivation came from identifying inconsistencies within both the papers and the code provided by the original authors.

1. ABBA includes learning the symbolic representation initially. However, the original paper determines these symbols before partitioning the time series into a train set and a test set. Therefore, the symbolic representation depends on the test set, which biases the evaluation.
2. In the original implementation, we were taken aback to discover that the test set is incorporated into the train set of the network.
3. We observed an inherent unfairness in the comparison of both methods. k symbols were used to train the LSTM network with ABBA, and k timestamps were employed to train the LSTM network on the raw time series. However, it's crucial to note that a symbol represents more than one timestamp which is unfair for evaluation.

Clement Wang's involvement centered on reimplementing both papers to rectify these discrepancies. He assessed the revised implementation using the identical sinusoidal signal featured in the original paper. Guillaume Levy built upon Clement Wang's reimplementation to analyze the ABBA LSTM forecasting algorithm applied to noisy and complex real-world data. He studied the dataset and the hypothesis that it holds and see how well the algorithm can be applied to it.

The dataset used is the Monthly Sunspots dataset [4] created by the SIDC (Solar Influences Data Analysis Center) which can be found here.

2 Method

ABBA comprises three primary functional components:

- Quantization/Symbolization: This phase processes a time series input to establish the symbols forming the alphabet to be utilized.
- Inference: Here, given a time series input, it condenses it into a shorter sequence of symbols.
- Inverse transformation: This step works with a sequence of symbols as input, approximating the original time series from this symbolic representation.

Initially, the quantization and inference processes occurred simultaneously. However, we restructured this to ensure that the evaluation remains distinctly separate from the training phase.

Quantization. We initially approximate the time series T using a piecewise linear continuous time series. The goal of the method is to find iteratively the biggest intervals where the time series can be approximated by a piecewise linear function with an error smaller than a given threshold. To achieve this, at each loop, the algorithm starts at the timestamp i_k and finds the largest timestamp i_{k+1} such that:

$$\forall j \in [|i_k, i_{k+1}|] \sum_{l=i_k}^{j} \left(t_k + \frac{t_i - t_k}{i - i_k} l - i_k - t_l \right)^2 < (j - i_k)\epsilon^2$$

From the obtained piecewise linear continuous time series, we generate a sequence of tuples: $(\text{inc\_x}_1, \text{inc\_y}_1), (\text{inc\_x}_2, \text{inc\_y}_2), \ldots, (\text{inc\_x}_n, \text{inc\_y}_n)$. Subsequently, we employ a K-means clustering algorithm on this set of couples to derive our alphabet. Before the clustering, the values are divided by their standard deviation. The resulting centroids of these clusters serve as the alphabet.
Inference. The inference phase follows a similar procedure: using the same method to obtain a piecewise linear continuous approximation and categorizing these segments into clusters.

Inverse transform. The inverse process is relatively simple: reconstructing a piecewise linear function from a sequence of symbols.

We used LSTM networks to assess forecasting performance both with and without the integration of ABBA symbolic representation (See figure 4).

The LSTM networks utilize either k previous timestamp points to predict the next one or k' symbols to predict the subsequent symbol. In contrast to the original paper's approach of using $k = k'$, we deliberately select k and k' to align with a comparable number of timestamps, ensuring a fair evaluation of the forecasting performance.

3 Data

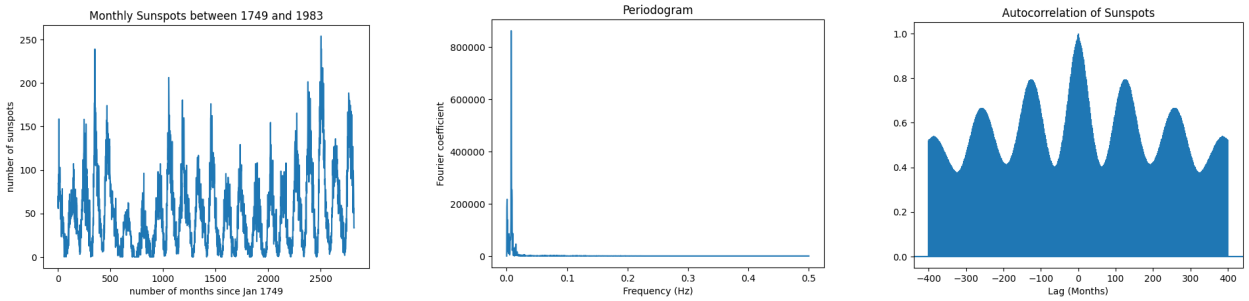
The previously mentioned method is tested on two separate datasets: a toy dataset that represents a sinusoidal time series and a dataset formed on real data: the Sunspots dataset.

3.1 Sinusoidal dataset

The dataset is composed of a sinusoidal function of frequency 0.01Hz sampled with the frequency F_s = 1Hz. The number of total samples in the dataset is 1000 which is sufficient to contain 10 periods of the sinusoidal function. Since the function is simple and periodic, no preprocessing was done before giving it to the algorithm.

3.2 Sunspots dataset

For the second experiment, we had to verify the capacity of the method to work with real data that contains noise and is not perfectly regular. For this, we used the Monthly Sunspots dataset [4] created by the SIDC (Solar Influences Data Analysis Center). It consists of the number of recorded sunspots each month between 1749 to 1983. The time series can be found in figure 1a below:



(a) The number of sunspots as the number of months

(b) The periodogram of the sinusoidal function

(c) The plot of autocorrelation of the sinusoidal function

Figure 1: Analysis of the sinusoidal function

Since the sun follows the solar cycle, its magnetic field, responsible for the sunspots, is reset every 11 years to almost its original value. So we expect the time series to be periodic. To verify this assumption, the autocorrelation and the periodogram of the signal are plotted on figure 1b and on figure 1c.

From these plots, we see that the signal is indeed almost periodic with a period of 132 Months and a frequency of around 0.008Hz, which corresponds to the 11 years of the solar cycle mentioned before. Since the dataset is periodic, we expect the model to be efficient when quantifying the time series and precise when forecasting the next values.

4 Results

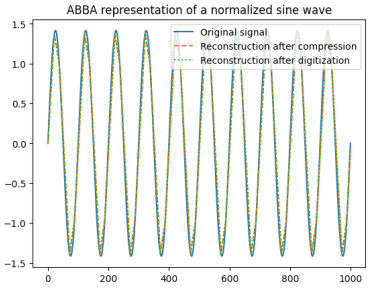
We evaluate the method through two experiment setups:

- Autoregressive method: Utilizing the last k timestamps from the training set, we autoregressively compute predictions for the n test set points. Subsequently, we measure the Dynamic Time Warping (DTW) [3] distance between these k' autoregressively predicted timestamps and the n ground truth test time series.
- Non-autoregressive method: For each of the n test points, we utilize the ground truth of the preceding k timestamps as input for the network.

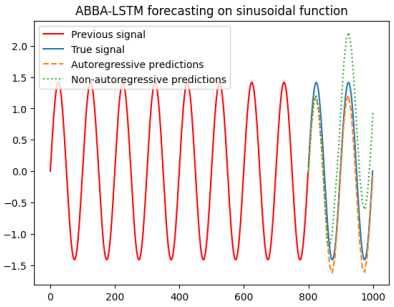
Our evaluation primarily focuses on assessing the performance in terms of Dynamic Time Warping (DTW).

4.1 Sinusoidal dataset

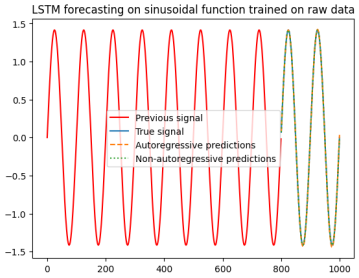
The first experiment done was on the toy dataset: the sinusoidal dataset. To see the performance of the method, we first look at the quality of the reconstruction of the signal after the linear piecewise approximation and the quantization. The results can be found on the figure 2a below. The resulting symbols obtained for this time series are: "facbdacbdacbdacbdacbdacbdacbdacbdacbdacbe"



(a) The reconstruction of the sinusoidal function after compression and digitization



(b) Forecasting the sinusoidal function using ABBA-LSTM



(c) Forecasting the sinusoidal function using LSTM trained on raw data

Figure 2: Results on the sinusoidal dataset

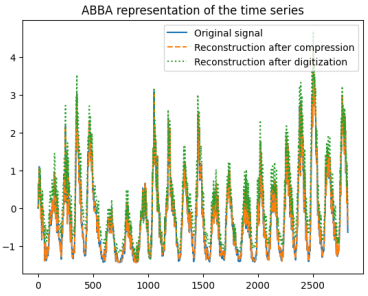
As expected, for the simple case, we obtain a very good reconstruction. The DTW distance for the reconstruction after compression is 0.016 and the reconstruction after digitization is 0.017 which is very small and close to each other. Since there is no noise in this dataset and since the sinusoidal function is a periodic function that can be well approximated by only a few linear functions.

We then compare the efficacy of the forecasting compared to the forecasting using regular LSTM and raw data. The two results are available on the figures 2b and 2c.

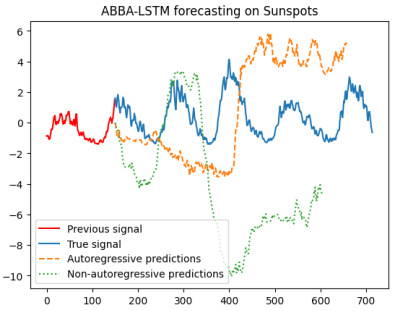
As expected, the raw LSTM is achieving a much better result with a DTW distance of 0.008 and 0.0009 for the autoregressive and non-autoregressive parts. Since the function is simple, the model has no problem forecasting the rest. This is not the case for the ABBA-LSTM which suffers from the accumulating errors of its prediction. The autoregressive ABBA-LSTM is still able to achieve decent results. However since the raw LSTM achieves better performance than the reconstruction after compression or digitization, ABBA-LSTM is thus not able to do as well.

4.2 Sunspots dataset

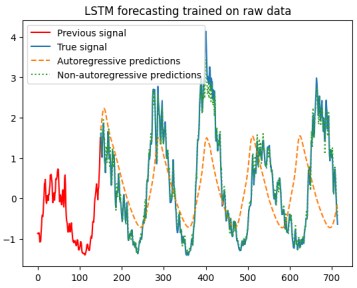
Now that we have tested the toy dataset, we then try to see if the method can be adapted to real data. Just as before we first look at the reconstruction of the signal after compression and digitization.



(a) Reconstruction of the time series after compression and digitization



(b) Forecasting with ABBA-LSTM



(c) Forecasting with a raw LSTM

Figure 3: Results on the Sunspots dataset

Even for real data, the compression and digitization achieves good performance but at the cost of losing the efficiency of the digitization. To obtain decent results we must use a lot more symbols and that represents a small number of timestamps(around 3 per symbol).

The great number of symbols makes it difficult for the LSTM to give an accurate forecast of the next symbol. We can see in figure 3a that the ABBA-LSTM model is not able to give a decent prediction of the time series. The underlying problem of the method is that it is fundamentally auto-regressive and though accumulates errors over time. When compared to LSTM trained on raw data, the model performs very poorly as we can see on figures 3b and 3c. Though the raw LSTM is not perfect, with a DTW distance of 0.194 and 0.068 for the autoregressive and non-autoregressive part when compared to the part of the time series used for testing.

4.3 Conclusion

So we study in this report the ABBA-LSTM method which quantifies a time series to forecast symbols instead of the raw values. We have seen that it is working efficiently for simple time series like a sinusoidal function but lacks robustness when working on real-world data. Even when comparing periodic time series that inherently enables the use of a small diversity of symbols, the method is less accurate when compared to an LSTM trained on the raw values. The time gained by digitizing the time series, by around 3, is not worth the high decrease in performance.

References

[1] Steven Elsworth and Stefan Güttel. Abba: Adaptive brownian bridge-based symbolic aggregation of time series. *Data Mining and Knowledge Discovery*, 34(4):1175–1200, 2020.

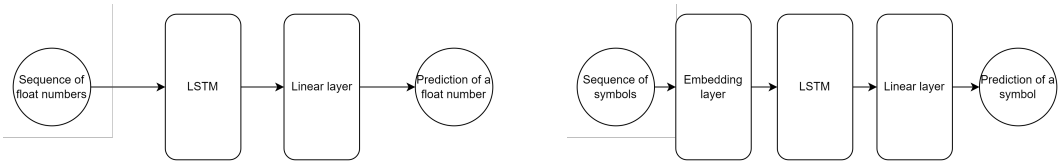
[2] Steven Elsworth and Stefan Güttel. Time series forecasting using lstm networks: A symbolic approach. *arXiv preprint arXiv:2003.05672*, 2020.

[3] H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 26(1):43–49, 1978.

[4] SILSO World Data Center. The international sunspot number. *International Sunspot Number Monthly Bulletin and online catalogue*, 1749-2018.

[5] Ralf C. Staudemeyer and Eric Rothstein Morris. Understanding LSTM - a tutorial into long short-term memory recurrent neural networks. *CoRR*, abs/1909.09586, 2019.

A LSTM architecture



(a) LSTM on raw time series

(b) LSTM on symbols input

Figure 4: LSTM networks architectures for forecasting

B Performance

	Sinusoidal dataset	Sunspots dataset
Reconstruction after compression	0.016	0.020
Reconstruction after digitization	0.017	0.100
ABBA-lstm autoregressive	0.034	0.319
ABBA-lstm non-autoregressive	0.134	0.357
Raw-lstm autoregressive	0.015	0.022
Raw-lstm non-autoregressive	0.001	0.068

Table 1: DTW distance between the different reconstruction and the original time series