Deep Learning Techniques for Missing Data Imputation in Time Series

WANG Xiangyu, WONG Chak Kei Jack

Hong Kong University of Science and Technology

# ABSTRACT

We empirically compare four state-of-art deep learning networks for sequence learning and applied in imputation of missing data in time series. Although inherently suitable for the financial time series predictions, they are less commonly utilized in this domain. All algorithms were re-implemented in a uniform development environment. We gather the dataset comes from the Hang Seng Index (HSI) which is formed by ten stocks and deploy above models for imputing random missing data blocks of daily returns from 2017 to 2022. Through the experimental evaluation, it allows us to identify the limitations of the current model of deep learning networks in financial applications and suggests the future research directions.

# INTRODUCTION

In this paper, we primarily focus on deep learning for imputing missing data of five-year daily returns from ten stocks of HSI. We present an experimental evaluation of missing data recovering techniques. The chosen algorithms are summarized in Table 1, which cover the full gamut of state-of-art deep learning techniques currently available to recover large missing blocks.

Table : Missing Data Imputation Techniques

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Components | | | Implementation | |
|  | Sampling algorithm | Loss function | Original | Framework |
| Deep Learning | SSIM [1] | VLSW | MSE | Python 3.6 | Torch 1.10.1 |
| DeepMVI [2] |  |  | Python 3.8 | Torch 1.7.0 |
|  |  |  |  |  |
|  |  |  |  |  |

***Results***.

Problem description and classification:

* Echo sequence prediction problem:

Sequences must be framed as a supervised learning problem when using neural networks,

* + Generation of random sequences
  + Transformation of random sequences into a supervised learning problem

The problem then can be framed as making a prediction based on a function of current and previous timesteps.

* Handling missing sequence data
* Learning with missing sequence values

There are many ways of preparing time series data for training.

* GSW
* VLSM

# Task overview and formulation

Just like an entire process of financial data analysis, our task consists of five steps:

1). Data acquisition. Ten real-world stocks were acquired as our missing data imputing targets.

2). Data preparation. We first clean up the raw data for extracting valid information and randomly remove 5% the data in these time series. Then based on the specific deep learning models, tailored sampling algorithms, and train-impute strategies for the generation of “study period”. Finally, forecasting and imputing the missing data.

3). Deep learning model construction.

4). Model deployment.

5). Evaluation and discussion.

## Raw data acquisition

For the empirical application, we chose ten stocks containing past five-year daily data blindly from “Top 30 Components” of Hang Seng Index as the real-world datasets at *Yahoo Finance*[3]. To eliminate haphazard, historical daily data in the past five years of ten stocks as the real-world datasets were acquired by *pandas-datareader*[4]. One sample list (0012.HK) printed in the terminal from January 2017 to December 2021 was shown in the following Appendix 1(a).

## Problem overview

***Daily return.*** As requested, we are going to fit RNN models to the daily returns of obtained ten sets of time series jointly which means just column “*Adj Close*” is needed to calculate the features – daily return sequences.

Let be defined as the price process of stock at time step during the whole period . Let denote the simple return for one stock over periods, i.e.,

|  |  |  |
| --- | --- | --- |
|  |  | (1) |

where and as one-day returns for each day and one of ten stocks. After stacking them in one large feature table as shown in Appendix 1(b), we obtain the feature vector of dimension .

***Data missing scenario.*** The pattern ofmissing blocks in a time series dataset can be quite arbitrary and varied. An entire contiguous block of entries might be missing within a time series, and across multiple time-series. The signals from the rest of the dataset that are most useful for imputing a missing block would depend on the size of the block, patterns within a series.

In this paper, we use missing completely at random (MCAR) scenario to generate random number with fixed seed and will produce the same blocks every run. Based on the MCAR pre-set, 5% time series are incomplete. Since we stipulate the missing block size is ten, then the block number in our case is six. For example, in Figure 1, we show one dimension series of a fixed length. Six missing blocks starting a random point from corresponding six intervals. Each missing block contains continuous ten-day missing data.

|  |
| --- |
|  |

Figure . Time series with six continuous missing data blocks. Grey-shaded blocks highlight the missing time series data, while the available time series data (daily returns of 0012.HK) is plotted with green and red bars in the white blank spaces.

# Selected Imputation Algorithms

## Variable -Length Sliding Window Algorithm

Due to the conventional approach, like *Generic Sliding Window* Algorithm (GSW), can only learn to reconstruct the missing data based on the previous information so that it will cause the lack-of-utilization of half of the useful information. VLSM algorithm [1] was proposed to make use of information from both past and future time index with variable padding length.

Unlike original paper padding with zeros which is very easy to be confused with zeros embedded in the training data itself, we use a more special number e.g., -10 as dummy value to pad the output gap.

# Imputation Model

## Sequence-to-sequence with Attention

**SSIM** [1] is a sequence-to-sequence imputation model for recovering missing data in a wireless sensor network. The architecture with attention mechanism utilized by SSIM was depicted in Figure 2(a), where the encoder and decoder are two key functional components. In the SSIM work, they chose the *Bi-directional Long Short-Term Memory* (BiLSTM*)* network[5]and the *Long Short-Term Memory* (LSTM) with attention mechanism[6] for both the encoder and decoder. The encoder in the SSIM is a BiLSTM, which is comprised of a forward LSTM and a backward LSTM. The decoder in the SSIM is a unidirectional LSTM. The dropout layers are also included in both encoder and decoder. A masking layer is added to remove the zero-padded vectors in the input sequences. A dense layer with a linear activation function is stacked to produce predictions with continuous values.

Seq2Seq as one form of RNN model, it is hard to parallelize which means needs to read the entire series information, then get the outputs.

## Temporal transformer

**DeepMVI**

|  |  |
| --- | --- |
|  |  |
| (a) | (b) |
|  |  |
| (c) | (d) |

Figure . Four kinds of designed architectures of deep learning networks.

# Evaluation

This section evaluates the previously selected algorithms through a number of experiments designed to compare their results based on the following three metrics – parameterization, accuracy, and efficiency.

## Evaluation Metrics

We evaluate the performance of recovering missing data based upon the following four metrics:

1. Root mean square error (RMSE)

|  |  |  |
| --- | --- | --- |
|  |  | () |

1. Mean absolute error (MAE)

|  |  |  |
| --- | --- | --- |
|  |  | (3) |

1. Mean absolute percentage error (MAPE)

|  |  |  |
| --- | --- | --- |
|  |  | (4) |

1. Symmetric MAPE (SMAPE)

|  |  |  |
| --- | --- | --- |
|  |  | (5) |

## Model parameters

## Implementation Notes

I conducted all the experiments on my laptop equipped with a 4.8GHz (Max. Boost Clock) AMD Ryzen 9 5900HX CPU (which consists of 8 cores with a 16MB L3 cache), an NVIDIA GeForce RTX 3080 Laptop GPU, and 32GM of RAM. The code was compiled with Python 3.9.7 64-bit interpreter. Most of the calculation of sample training was depended on the framework *TensorFlow Core 2.7.0* and *Keras* neural network API[7].

As mentioned, I implemented all the algorithms in Python3 and TensorFlow2. In Table 1, the *Implementation*column describes the original coding language and deep learning framework or package with corresponding versions in which each algorithm was used.

I use the same advantage algorithms mentioned above across all techniques.

|  |  |
| --- | --- |
|  |  |
| (a) |  |

Figure . Training and validation loss of different models. (a) SSIM.

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

Figure . Recovering six missing blocks of 0012.HK by using re-implemented SSIM. Green bars and orange bars represent the ground truth data and the imputations generated by SSIM, respectively. The ten day’s data comes from the before and after corresponding period are used as inputs.

# References

[1] Y. F. Zhang, P. J. Thorburn, W. Xiang, and P. Fitch, “SSIM - A Deep Learning Approach for Recovering Missing Time Series Sensor Data,” *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 6618–6628, Aug. 2019, doi: 10.1109/JIOT.2019.2909038.

[2] P. Bansal, P. Deshpande, and S. Sarawagi, “Missing Value Imputation on Multidimensional Time Series,” Mar. 2021, [Online]. Available: http://arxiv.org/abs/2103.01600

[3] “Yahoo Finance.” https://finance.yahoo.com/ (accessed Dec. 30, 2021).

[4] “pandas-datareader.” https://github.com/pydata/pandas-datareader.git (accessed Dec. 25, 2021).

[5] M. Schuster and K. K. Paliwal, “Bidirectional recurrent neural networks,” *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997, doi: 10.1109/78.650093.

[6] M. T. Luong, I. Sutskever, Q. v. Le, O. Vinyals, and W. Zaremba, “Addressing the Rare Word Problem in Neural Machine Translation,” *ACL-IJCNLP 2015 - 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, Proceedings of the Conference*, vol. 1, pp. 11–19, Oct. 2014, doi: 10.3115/v1/p15-1002.

[7] F. Chollet and & others, “Keras,” *GitHub. Retrieved from https://github.com/fchollet/keras*, 2015.

Appendix

|  |
| --- |
|  |
| (a) |
| Calendar  Description automatically generated |
| (b) |

Appendix . Raw data sample format. (a) A snippet of 0012.HK time-series daily data printed in the terminal; (b) Daily return sequences over past five years of picked ten stocks.

|  |
| --- |
|  |

Appendix . Daily returns of target ten HSI stocks over past five years as training datasets.