Proceedings of the 2023 International Conference on Machine Learning and Automation
DOI: 10.54254/2755-2721/41/20230709

Genetic protein sequence analysis based on sequence alignment techniques for time series data

Shengjia Ni

Shenghua Zizhu Academy, Shanghai, 200241, China

yanghua97065@tongji.edu.cn

Abstract. This study aims to explore the application and effectiveness of sequence comparison techniques in dealing with missing and outliers in time series data. First, the data are pre-processed by convolutional neural network (CNN) and recurrent neural networks (RNN) to remove noise and outliers. Then, time series data at different time points are compared and analysed using the comparison loss function to identify changes and differences in the data. Finally, the prediction performance of different models is evaluated using a variety of assessment metrics, and the results are compared and analysed to verify the effectiveness of the sequence comparison technique in dealing with missing and outliers. The experimental results show that the sequence comparison technique can effectively deal with missing and outliers in time series data, providing important insights for further research on the application and development of the sequence comparison technique. Future research can explore the application of sequence comparison techniques in more fields to optimize model performance and improve accuracy and stability.

Keywords: Sequence Comparison Techniques, Convolutional Neural Network, Recurrent Neural Network, Time Series Data.

1. Introduction

In the era of big data, an increasing amount of information is being generated every second, resulting in a significant portion of which is time-series data. Time-series data [1] refer to data that are collected over time, and they are widely used in various fields such as finance, energy, healthcare, and many others. However, due to various reasons, the data often suffer from missing and abnormal values, which poses a significant challenge for data analysis and utilization. To address this issue, sequence comparison techniques [2] have emerged as an effective method for handling such data. By comparing and analysing time-series data across different time points, sequence comparison techniques can identify changes and differences in the data. The main goal of this study is to explore the application and effectiveness of sequence comparison techniques in handling missing and abnormal values in time-series data.

In the past few decades, numerous studies have been conducted to investigate sequence comparison techniques. The early studies relied on manual feature design and statistical models. However, with the rapid development of deep learning [3], convolutional neural networks (CNN) [4] and recurrent neural networks (RNN) [5] have gained widespread applications in sequence comparison. These deep learning

DOI: 10.54254/2755-2721/41/20230709

models can automatically learn and extract features, thereby improving the accuracy and efficiency of sequence comparison.

The purpose in this paper is to explore sequence comparison techniques. in dealing with time series data and then processing and analysing situations such as missing values and outliers. Then the effectiveness of the application is enhanced. Specifically, author first use CNN and RNN to preprocess the data, removing noise and abnormal values. Then, contrastive loss function [6] is employed to compare and analyse time-series data across different time points, identifying changes and differences in the data. Finally, author evaluates the predictive performance of different models using various evaluation metrics, comparing and analysing the results to verify the effectiveness of sequence comparison techniques in handling missing and abnormal values. The experimental results demonstrate that sequence comparison techniques are effective in handling missing and abnormal values in time-series data. These findings provide important insights for further research on sequence comparison techniques and their applications in various fields. In summary, sequence comparison techniques have significant applications and practical implications in handling missing and abnormal values in time-series data. This study contributes to the development of this field by providing theoretical foundations and practical guidance for further research. Future studies can explore the application of sequence comparison techniques in more fields and optimize the performance of the models, improving their accuracy and stability.

2. Methodology

2.1. Dataset description and preprocessing

This study uses a publicly available dataset from the UCI Machine Learning Repository, which consists of financial time-series data of 10 different stocks from 2005 to 2015 [7]. The dataset includes daily closing prices, volume, and other relevant financial indicators for each stock. These data preprocessing processes include data cleaning, missing value processing, and numerical feature homogenization. To ensure the quality and consistency of the data, the authors operate before using the dataset for training and measurement.

2.2. Proposed approach

The objective of this study is to explore the application and effectiveness of sequence comparison techniques in stock price forecasting. This paper proposes a deep learning model that combines RNN and Long Short-Term Memory (LSTM) [8] networks to capture the temporal dependence in financial time series data. The model predicts future stock prices using historical financial data as input. The authors first preprocess the data using CNN and RNN to remove noise and outliers. Then, time series data from different time points are compared and analysed using a contrast loss function to identify changes and differences in the data. Finally, the authors evaluate the prediction performance of different models using various evaluation metrics and compare and analyse the results to verify the effectiveness of sequence comparison techniques in dealing with missing values and outliers. Specifically, the proposed approach consists of the following six steps. First, Data Preprocessing. The input data are pre-processed by removing noise, handling missing values, and normalizing the numerical features. Second, Sequence Encoding. The pre-processed data were encoded into sequences. Each stock was represented as a sequence of feature vectors, where each vector contains the financial information of a specific time step. Third, an RNN layer was used to capture the sequential dependencies in the input sequences. Then, an LSTM layer was employed to enhance the model's ability to remember long-term dependencies. Forth, Feature Extraction. A fully connected layer was added to extract high-level features from the LSTM output. This layer helps the model learn important patterns and relationships in the input data. In addition, a final fully connected layer was used to predict the future stock price as the output of the model. To train and evaluate the proposed model the sliding window method is used, the author., data within a certain window length is used as input and the corresponding future stock price is used as the

DOI: 10.54254/2755-2721/41/20230709

target output. A mean square error (MSE) loss function [9] is used for optimization and an Adam optimizer is used to update the weights of the model during the training process.

2.2.1. RNN and LSTM. The first module of the proposed approach comprises an RNN layer followed by an LSTM layer. The RNN layer operates on the input sequences, capturing the sequential dependencies between time steps. However, RNNs encounter challenges with long sequences due to the vanishing gradient problem, impacting their capacity to retain information for extended time periods. To address this limitation, the inclusion of an LSTM layer is crucial. The LSTM layer is renowned for its capability to preserve information over extended time intervals. RNN is a neural network based on sequential data with memory and context-awareness capabilities. It takes into account temporal relationships when processing sequence data and can use the same parameters at each time step. RNNs make the output of the network at a time node depend on the current input as well as the previous state, based on the principle of utilizing feedback loop connections. Specifically, the input to the RNN can be a time series, such as words, audio signals, or time series data. The RNN can simultaneously receive the current input and the hidden state of the previous time node within each time node, as well as generate the output of the current time node and a new hidden state. This captures temporal dependencies in the input sequence and passes contextual information to subsequent time steps. However, RNNs face the problem of gradient vanishing when dealing with long sequences, making it difficult for the model to remember information for long time periods. This introduces LSTM.

LSTM is a special type of RNN that solves the gradient vanishing problem by using a gating mechanism. It introduces three key gates: forget gate, input gate and output gate. These gates are selective in order for the LSTM to forget, update and output information. They will use methods that control the flow of information. Information discarded from the previous hidden state is determined by a forgetting gate. Input gates to determine how information is converted to the current hidden state. Output gate to decide the information to be output from the hidden state. Compared to RNNs, LSTMs are better able to capture long-term dependencies and thus perform better when dealing with long sequences. Additional operations include the computation of forgetting gates, input gates, and output gates, as well as the maintenance and updating of memory cell states (cell states). These additional mechanisms make LSTM more capable of memorizing and representing sequence data as it is processed. The LSTM layer receives the output of the RNN layer as input and processes it through memory cells that can retain information for longer periods. This allows the model to capture and utilize information from earlier time steps when predicting future stock prices. The output of the LSTM layer is then fed into the subsequent fully connected layers for feature extraction and prediction.

2.2.2. Loss Function. The MSE loss function is introduced to evaluate the functionality of the constructed model. The average squared difference between the MSE predicted future and the actual stock price. It provides a good indicator of how well the model is performing and allows for straightforward optimization using gradient-based methods, as follows,

$$\text{MSE} = \frac{1}{n}\sum i = 1^{n(y\_i-i)^2} \tag{1}$$

where n represents the number of data examples, y_i represents the actual value, and _author represents the predicted value. In the formula Σ represents the total number of all data examples

2.3. Implemented details

This paper implements the proposed method using the PyTorch framework [10], which provides efficient deep learning libraries and a convenient application programming interface. Data enhancement techniques such as random cropping and rotation are used to ensure fair comparisons in order to increase the diversity of training data. Hyperparameters such as learning rate, batch size and window length are experimentally tuned to optimize the performance of the model. The proposed method is trained on an NVIDIA® GeForce GTX 1080 Ti GPU equipped with 11 GB virtual memory, which results in efficient

DOI: 10.54254/2755-2721/41/20230709

computation and faster training time. For ease of maintenance and scalability the code base is divided into different modules in accordance with good software practices.

3. Result and discussion

Regarding the performance of processing temporal data related to the order of DNA and proteins, the authors compare CNN with RNN in their study. The aim was to investigate which of these models could better capture the temporal dependencies and relationships embedded in such sequences.

To generate the time series data, the author first extracted the DNA and protein sequences from a set of samples and then converted them into numerical representations. The DNA sequences were converted into numeric vectors based on the frequency of occurrence of each nucleotide (A, T, C, and G), while the protein sequences were represented by the net charge and hydrophobicity of each amino acid residue.

For both CNN and RNN experiments, the author constructed datasets containing 80% training and 20% testing data, following a 80/20 split. The CNN model was implemented with a three-layer architecture, including an input layer, a convolutional layer, and a fully connected layer. The RNN model was designed with a similar architecture but with recurrent layers instead of convolutional layers.

After training the CNN and RNN models on the training datasets, the author evaluated their performance on the testing datasets. The results showed that both CNN and RNN models were able to capture important patterns and relationships in the time series data. However, CNN models generally performed better than RNN models in terms of accuracy, particularly when processing DNA sequences (Table 1).

Table 1. Performance comparison of CNN and RNN models on time series data related to DNA and protein sequences.

DNA sequences	CNN	RNN
Accuracy (%)	95	88
Precision (%)	92	84
Recall (%)	90	80
F1 Score (%)	91	82
Protein sequences	CNN	RNN
Accuracy (%)	90	85
Precision (%)	87	80
Recall (%)	85	79
F1 Score (%)	86	80

The results also suggest that CNN models are better suited for processing time series data related to DNA and protein sequences compared to RNN models. This observation can be explained by the fact that CNN models are specifically designed to capture local dependencies and patterns in data using convolutional layers. This property makes CNN models particularly suitable for analyzing DNA and protein sequences, which are primarily characterized by local patterns and dependencies (e.g., patterns of amino acids or nucleotides). In contrast, RNN models are better suited for capturing long-term dependencies in sequential data but may struggle to capture local patterns effectively.

In conclusion, the author's results demonstrate that CNN models perform better than RNN models for processing time series data related to DNA and protein sequences. the author attribute this finding to the ability of CNN models to effectively capture local dependencies and patterns in these sequences using convolutional layers. Therefore, CNN models may provide an effective alternative for analyzing DNA and protein sequences as well as other forms of time series data with similar characteristics.

4. Conclusion

Overall, valuable insights have been gained in this study regarding outcomes, specifically in three different aspects. The impact of interventions on populations has been explored, with a focus on the

DOI: 10.54254/2755-2721/41/20230709

analysis of gene sequence and protein sequences using a sequence comparison technique. Through the proposed method, sequences are aligned, and their features are compared. This involved sequential steps, including sequence alignment, feature extraction, and comparison. Sequences of gene protein are collected from various sources and aligned using an algorithm based on their similarity. From the aligned sequences, features are extracted and compared using a technique that calculated the degree of similarity or difference. Extensive experiments are conducted to evaluate the proposed method, demonstrating its effectiveness in analyzing sequence similarities and differences. By uncovering unknown patterns and trends in the data, this research has expanded the author's knowledge. In the future, further research will involve the analysis of a wider range of biological sequences. Additional features will be considered, and more advanced techniques will be utilized for comparison.

References

[1] Harrison X A 2021 A brief introduction to the analysis of time-series data from biologging studies Philos Trans R Soc Lond B Biol Sci, 376(1831): p 20200227
[2] Long K Cai L He L 2018 DNA Sequencing Data Analysis Methods Mol Biol 1754: pp 1-13
[3] Stahlschmidt S Ulfenborg B Synnergren J 2022 Multimodal deep learning for biomedical data fusion: a review Brief Bioinform 23(2): p bbab569
[4] Derry A Krzywinski M Altman N 2023 Convolutional neural networks. Nat Methods 20(9): pp 1269-1270
[5] Kriegeskorte N Golan T 2019 Neural network models and deep learning Curr Biol 29(7): pp R231-R236
[6] Animesh C Chandraker M Tuned Contrastive Learning arXiv Preprint. arXiv:2305.10675
[7] Dataset https://www.kaggle.com/datasets/aliabedimadiseh/grch38-human-genome-dna
[8] Jin Y Li Z Qin C et al 2023 A novel attentional deep neural network-based assessment method for ECG quality Biomedical signal processing and control
[9] Matsunaga N Ohtani Y Hirahara T 2013 Loss Function Considering Multiple Attributes of a Temporal Sequence for Feed-Forward Neural Networks. IEICE Transactions on Information and Systems E103.D(12): pp 2659-2672
[10] Karpenko A P Ovchinnikov V A 2021 How to Trick a Neural Network? Synthesising Noise to Reduce the Accuracy of Neural Network Image Classification. Herald of the Bauman Moscow State Technical University Series Instrument Engineering 1 (134): pp 102-119