Comparison of Different Approaches of Machine Learning Methods with Conventional Approaches on Container Throughput Forecasting

Shuojiang Xu 1, Shidong Zou 1, Junpeng Huang 1, Weixiang Yang 1 and Fangli Zeng 2,*

1 School of Artificial Intelligence, Guilin University of Electronic Technology, Guilin 541004, China
2 Logistics and E-Commerce College, Zhejiang Wanli University, Ningbo 315104, China
* Correspondence: fangli.zeng@zwu.edu.cn

Abstract: Container transportation is an important mode of international trade logistics in the world today, and its changes will seriously affect the development of the international market. For example, the COVID-19 pandemic has added a huge drag to global container logistics. Therefore, the accurate forecasting of container throughput can make a significant contribution to stakeholders who want to develop more accurate operational strategies and reduce costs. However, the current research on port container throughput forecasting mainly focuses on proposing more innovative forecasting methods on a single time series, but lacks the comparison of the performance of different basic models in the same time series and different time series. This study uses nine methods to forecast the historical throughput of the world’s top 20 container ports and compares the results within and between methods. The main findings of this study are as follows. First, GRU is a method that can produce more accurate results (0.54–2.27 MAPE and 7.62–112.48 RMSE) with higher probability (85% for MAPE and 75% for RMSE) when constructing container throughput forecasting models. Secondly, NM can be used for rapid and simple container throughput estimation when computing equipment and services are not available. Thirdly, the average accuracy of machine learning forecasting methods is higher than that of traditional methods, but the accuracy of individual machine learning forecasting methods may not be higher than that of the best conventional traditional methods.

Keywords: comparison; machine learning; conventional approach; time series; forecasting; container throughput

1. Introduction

Container shipping is an important form of international trade logistics, and a great deal of goods are transported from the origin to the consumption place far across the ocean by container shipping [1]. However, the spread of COVID-19 has had a profound impact on container shipping and will even overturn the trend of container shipping in the future [2]. Since the third quarter of 2020, there has been a global shortage in the supply of empty containers, and major shipping companies are short of shipping space [3]. The advance booking period of Sino-European routes is about two weeks, and even the Sino-American routes are sold out. The empty container supply and lack of capacity directly leads to a rapid rise in container service charge, and the Shanghai Containerised Freight Index (SCFI) and the Freight Baltic Index (FBX) are obviously up. For example, the price of a container shipped from China to Europe has risen from $2000 to $15,000, a 7.5 times higher transportation cost than before [4]. The rapidly rising price of container transportation has brought heavy burden to international trade, and the price of all kinds of goods transported by container has also risen sharply. Therefore, improving the efficiency of container shipping is an important way to better the performance of international trade and reduce trade costs. Previous practitioners and scholars have conducted studies on improving the efficiency of container shipping from many aspects [5–8]. This study focuses on the forecasting of port container throughput, because the accurate forecasting results of port
container throughput can provide decision support for shipping companies, port owners, freight forwarders, and other container shipping participants. With the development of machine learning, a variety of sophisticated forecasting models were proposed based on machine learning algorithms. However, many up-to-date forecasting methods were not applied to forecast container throughput. It is necessary to compare the performance of advanced machine learning methods and conventional methods on container throughput forecasting. Therefore, the research question of this study is which of the existing forecasting methods is more accurate in forecasting container throughput.

The main contributions of this study are as follows. First, the performance of nine different time series forecasting methods on a single time series is compared, including conventional methods and machine learning methods. Secondly, the forecasting method GRU, which is accurate for short time series forecasting results, is obtained through comparison, which provides experience for future forecasting research. Thirdly, it is found that the forecasting results of machine learning algorithms on short time series are not necessarily better than those of conventional methods, and the more complex models tend to produce less ideal forecasting results.

2. Literature Review

From the perspective of learning mechanisms of forecasting models, we can divide them into two categories: conventional forecasting models and machine learning forecasting models. Conventional forecasting models are those that use simple rules or methods to forecast future values, such as the naïve method (NM), moving average (MA), autoregressive (AR) and autoregressive integrated moving average (ARIMA), etc. Machine learning forecasting models are those that employ more complex computational methods and model structures to extract underlying patterns from the data, such as multilayer perceptron (MLP), recurrent neural network (RNN), convolutional neural network (CNN) and Transformer, etc. The summary of the literature review is presented in Table 1.

Among the conventional forecasting models, the naïve method is the simplest but most effective time series forecasting method [9]. It takes the actual value at time $t - 1$ as the forecasting value at time $t$. In actual production, many enterprises choose to use the naïve method as the basic forecasting method to guide their operations plan. The naïve method is also used as a benchmark for the evaluation of the performance of other forecasting methods [10]. For any designed forecasting model, the method is valid if its accuracy is higher than the naïve method’s, and vice versa. It is similar to random guess in classification problems. The moving average is another method commonly used to forecast future values [11]. It uses the average of a group of recent actual values to forecast future values, such as demand and capacity, etc. However, this method can only be used when the demand is neither rapid growth nor rapid decline, and there is no seasonal factor. Previous studies investigated optimal MA length for forecasting future demand. Their findings suggest that optimal MA length is related to the frequency of occurrence of the structural change [12]. The autoregressive model is developed from linear regression in regression analysis and used to deal with time series [13]. It uses the historical values of the same variable ($y_{t-1}$ to $y_{t-n}$) to forecast the current $y_t$. Because an autoregression model only uses the historical value of a variable to forecast its future value, it does not use other variables, so it is called autoregressive. Many studies have analysed and improved AR [13–16]. Furthermore, Box and Jenkins integrated AR and MA methods and added an integrated method to put forward the ARIMA time series forecasting model [17]. On this basis, ARIMAX and SARIMA were designed to handle multivariate input data and seasonal input data, respectively. Many studies use ARIMA and its derived models to forecast the future value of the target and obtain acceptable forecasting accuracy [18–20]. The traditional method is used by many enterprises because of its simple deployment and fast computing speed. However, these methods are difficult to obtain complex influence relationships from a large number of influencing factors, so scholars put forward more complex and effective forecasting models called machine learning (ML) [21].
MLP is a kind of neural network machine learning model which attracts a great deal of attention [22]. It is a fully connected feedforward artificial neural network and has been employed as a benchmark to test the forecasting performance of other forecasting models [23–25]. MLP was improved by integrating other forecasting models [26–29]. The concept of deep learning originates from the development of the artificial neural network [30]. MLP with multiple hidden layers can be considered as a deep learning structure [31]. By combining low-level features, deep learning can form more abstract high-level attributes or features to discover distributed feature representations of data [32]. There are many architectures for deep learning, among which RNN is a common architecture. Many complex and well-performing deep learning architectures are based on RNN [33]. RNN has good processing ability for sequential structure data and is often used in language processing problem. Gated recurrent unit (GRU) and long short-term memory (LSTM) are two representative RNN architectures. For instance, Noman et al. proposed a GRU based model to forecast the estimated time of arrival for vessels. Their experimental results show that the GRU-based model can produce the best forecasting accuracy compared to other methods [34]. Moreover, Chen and Huang employed Adam-optimised GRU (Adam-GRU) to forecast port throughput. Their findings can be concluded as Adam-GRU can produce relatively accurate forecasting results [35]. Shankar et al. built a container throughput forecasting model by using LSTM. Their experiment showed that LSTM can also generate accurate forecasting results [36]. CNN is another commonly used deep learning architecture. It was originally used to solve computer vision problems, such as image recognition, and later some scholars applied CNN to the analysis and forecasting of sequence data. For instance, Chen et al. proposed a temporal CNN to estimate probability density of time series [37]. There are many studies that employed CNN to build time series forecasting model [38–41]. More recently, Transformer, another deep learning architecture, was first proposed by Google Brain in 2017 to solve the sequential data problem, such as natural language processing (NLP) [42]. It features all input data into the model at once, and uses positional encodings, attention, and self-attention mechanisms to capture the patterns from the data. Based on Transformer, scholars also put forward powerful NLP models such as GPT-3 [43], BERT [44], T5 [45], etc. Later, some scholars applied Transformer to time series forecasting, because time series data and text data are both sequential data [46]. Experimental results show that Transformer can produce more accurate results in time series forecasting than previous work. There have been a number of recent studies using Transformer for forecasting. All these studies suggest that Transformer has a good performance in time series forecasting [46–49].

However, these studies only assessed some of these methods’ performance, but no research has investigated the performance of these methods on the same time series simultaneously. Thus, which method performs better on the same time series for container throughput remains unclear. In this context, the aim of this study is to compare several existing forecasting methods for the container throughput in the same port. Then, insights for selecting an appropriate method can be suggested.

| Literature | Methods | Data | Main Finding |
|------------|---------|------|--------------|
| [18]       | ARIMA, ANN | Wolf’s sunspot data, the Canadian lynx data, and the British pound = US dollar exchange rate data | The combined model can be an effective way to improve forecasting accuracy achieved by either of the models used separately. |
| [19]       | ARIMA    | Spanish electricity market, Californian electricity market | The Spanish model needs 5 h to predict future prices, as opposed to the 2 h needed by the Californian model. |
| [21]       | SARIM, SVR | Aviation factors of China | The SARIMA-SVR can provide the best forecasting results. |
Table 1. Cont.

| Literature | Methods | Data | Main Finding |
|------------|---------|------|--------------|
| [24]       | Particle-swarm-optimized multilayer perceptron (PSO-MLP) model | Landslides of Shicheng County in Jiangxi Province of China | Proposed PSO-MLP model addresses the drawbacks of the MLP-only model performs better than conventional artificial neural networks (ANNs) and statistical models. |
| [25]       | MLP, linear regression (LR) | Covid-19 positive case from March to mid-August 2020 in West Java | MLP reaches optimal if it used 13 hidden layers with learning rate and momentum = 0.1. The MLP had a smaller error than LR. |
| [26]       | random forest, MLP | Electrical load data of six years from a university campus | Hybrid forecast model performs better than other popular single forecast models. |
| [27]       | MLP, Whale optimization algorithm | Read gold price | The proposed WOA–NN model demonstrates an improvement in the forecasting accuracy obtained from the classic NN, PSO–NN, GA–NN, GWO–NN, and ARIMA model. |
| [28]       | Dynamic regional combined short-term rainfall forecasting approach, MLP | Actual height, temperature, temperature, dew point difference, wind direction and wind speed at 500 hPa height | DRCF outperforms existing approaches in both threat score (TS) and root mean square error (RMSE). |
| [29]       | local MLP | Simulated data | A greater degree of decomposition leads to the greater reduction in forecast errors. |
| [34]       | GRU | Vessels that travel on the inland waterway | GRU provides the best prediction accuracy. |
| [35]       | Adam-GRU | Guangzhou Port | Adam-GRU outperformed all other methods. |
| [36]       | LSTM | Port of Singapore | LSTM outperformed all other benchmark methods. |
| [37]       | DeepTCN | JD-demand, JD-shipment, electricity, traffic and parts | The framework compares favorably to the state-of-the-art in both point and probabilistic forecasting. |
| [38]       | CNN | Bid and ask | CNNs are better suited for this kind of task. |
| [39]       | LSTM, CNN | Electric load dataset in the Italy-North Area | The experimental results demonstrate that the proposed model can achieve better and stable performance in STL. |
| [40]       | CNN | Australian solar PV power data | Convolutional and multilayer perceptron neural networks performed similarly in terms of accuracy and training time, and outperformed the other models. |
| [41]       | Nonpooling CNN | Simulated data, daily visits to website | Convolutional layers tend to improve the performance, while pooling layers tend to introduce too many negative effects. |
| [46]       | Transformer | ILI data from the CDC | Transformer-based approach can model observed time series data as well as phase space of state variables through time delay embeddings. |
| [47]       | Enhancing the locality of Transformer, breaking the memory bottleneck of Transformer | Electricity-f (fine), electricity-c (coarse), traffic-f (fine), traffic-c (coarse), wind | It compares favorably to the state of the art. |
| [48]       | Transformer | Electricity transformer temperature, electricity consuming load, weather | The experiments demonstrated the effectiveness of Transformer for enhancing the prediction capacity in LSTF problem. |
| [49]       | customized transformer neural network | Electricity consumption dataset, traffic dataset | In terms of long-term estimation up to eight times more resistant and in terms of estimation accuracy about 20 percent improvement, compare to other well-known methods, is obtained. |

3. Materials and Methods

This study compares the performance of nine different time series forecasting methods on the same time series, including traditional methods, which are the naïve method (NM), moving average (MA), autoregressive (AR) and autoregressive integrated moving average (ARIMA), and machine learning methods, which are multilayer perceptron (MLP), recurrent neural network (RNN), convolutional neural network (CNN) and Transformer. This section explains the technical details of these nine methods, such as calculation methods, flow charts, parameter definitions, etc.
3.1. Conventional Approaches

Conventional forecasting approaches mainly refer to methods with simple calculation process, few adjustable parameters, fast calculation speed and poor learning ability for complex nonlinear relations, such as NM, MA, AR, and ARIMA. This subsection is to explain the technical details of these conventional approaches.

3.1.1. Naïve Method

The expression of NM is shown in Equation (1),

\[ y'_t = y_{t-1}, \]  

where \( y'_t \) is the forecasted result of target variable at time \( t \), and \( y_{t-1} \) is the real value of target variable at time \( t - 1 \).

3.1.2. Moving Average

The expression of MA is shown in Equation (2),

\[ y'_t = \frac{1}{n} \sum_{i=1}^{n} y_{t-i}, \]  

where \( y'_t \) is the forecasting result at time \( t \), \( y_{t-i} \) is the real observation at time \( t - i \), and \( n \) is the size of the moving windows.

3.1.3. Autoregressive

The expression of autoregressive method is shown in Equation (3) [50],

\[ \phi_p \cdot Y_t = a_t, \]  

where \( \phi_p \) is the autoregressive operator, \( p \) is the autoregressive order, \( Y_t \) is the real time series at time \( t \), and \( a_t \) is the Gaussian white noise with zero mean and \( \sigma^2 \).

3.1.4. AutoRegressive Integrated Moving Average

ARIMA consists of three parts: AR, integration (I), and MA, and the corresponding parameters are \( p, d, q \) respectively. The general ARIMA model is called ARIMA \((p, d, q)\). The expression of ARIMA is shown in Equation (4) [21],

\[ \phi_p(B) \cdot \nabla^d \cdot Y_t = \theta_q(B) \cdot a_t, \]  

where \( B \) is the back-shift operator, and \( a_t \) is the Gaussian white noise with zero mean and \( \sigma^2 \). The expression of each parameter is shown in Table 2 [21].

### Table 2. Name and definition of each parameter in ARIMA.

| Name              | Parameter | Operator         | Equation                           |
|-------------------|-----------|------------------|------------------------------------|
| Autoregressive    | \( p \)   | \( \phi_p(B) \)  | \( 1 - \phi_1 B^1 - \phi_2 B^2 - \ldots - \phi_p B^p \) |
| Integration       | \( d \)   | \( \nabla^d \)   | \( (1 - B^1)^d \)                  |
| Moving Average    | \( q \)   | \( \theta_q(B) \)  | \( 1 - \theta_1 B^1 - \theta_2 B^2 - \ldots - \theta_q B^q \) |
| Back Shift        | \( B \)   | \( B \)          | \( B^r \cdot Y_t = Y_{t-n} \)      |
| Gaussian White Noise | \( \sigma^2 \) | \( a_t \) | \( a_t \sim (0, \sigma^2) \)        |

3.2. Machine Learning

Machine learning forecasting methods mainly refer to methods with a complex calculation process, many adjustable parameters, slow calculation speed, and strong learning ability for complex nonlinear relations. These methods, such as MLP, RNN, CNN, and Transformer, can obtain better fitting results by adjusting a large number of parameters.
3.2.1. MLP

MLP is an interconnected network composed by many simple neurons. When the input signal to the neuron exceeds the threshold, this neuron will be at excitatory state and then send information to downstream neurons and repeat the above steps. The basic structure of MLP is shown in Figure 1. The input data is connected to the neurons in input layer \((L_1)\), and there is a full-connection architecture between the neurons in input layer \((L_n)\) and the neurons in hidden layer \((H_n)\). Each connection to the downstream neurons is weighted. Similarly, neurons in hidden layer \((H_n)\) and neurons in output layer \((O_n)\) are fully connected with weighted lines [51].

First, the values in each layer are vectorised:

\[
\text{Input: } x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}
\]

\[
\text{Output of Hidden Layer: } a^H = \begin{bmatrix} a^H_1 \\ a^H_2 \\ \vdots \\ a^H_n \end{bmatrix}
\]

\[
\text{Output of Hidden Layer: } a^O = \begin{bmatrix} a^O_1 \\ a^O_2 \\ \vdots \\ a^O_n \end{bmatrix}
\]

The output of the input layer is

\[
a^H = \sigma(w^H \cdot x + b^H),
\]

where \(\sigma\) is the activation function, \(w^H\) is the vector of the weight of the linkage between the input layer and the hidden layer, and \(b^H\) is the vector of the threshold value of the neurons in hidden layer.

The output of the hidden layer is

\[
a^O = \sigma(w^O \cdot x + b^O),
\]

where \(w^O\) is the vector of the weight of the linkage between the hidden layer and the output layer, \(b^O\) is the vector of the threshold value of the neurons in the hidden layer.

![Figure 1. Basic structure of MLP.](image-url)
3.2.2. GRU

As mentioned earlier, a GRU is an RNN structure, and the recurrent model of a common RNN is shown in Figure 2. RNN is commonly composed of one or more units (the green rectangle A in the Figure 2), and the learning model is constructed by iteratively updating the parameters in the units. The basic structure of a GRU unit is shown in Figure 3. The calculation expressions of the parameters are shown in Equations (10)–(13) [52].

\[
\begin{align*}
    z_t &= \sigma_g(W_z x_t + U_z h_{t-1} + b_z) \\
    r_t &= \sigma_g(W_r x_t + U_r h_{t-1} + b_r) \\
    \hat{h}_t &= \phi_h(W_h x_t + U_h (r_t \odot h_{t-1}) + b_h) \\
    h_t &= z_t \odot \hat{h}_t + (1 - z_t) \odot h_{t-1},
\end{align*}
\]

where \(x_t\) is the input vector, \(h_t\) is the output vector, \(\hat{h}_t\) is the candidate activation vector, \(z_t\) is the update gate vector, \(r_t\) is the reset gate vector, \(W, U\) and \(b\) are parameter matrices and vectors, and \(\sigma_g\) and \(\phi_h\) are the activation functions.

![Figure 2. Basic structure of RNN.](image)

![Figure 3. Basic structure of GRU.](image)

3.2.3. LSTM

LSTM is another type of RNN with the same recurrent model as Figure 2. Figure 4 presents the common structure of an LSTM unit. There are three types of gates in the unit, which are the input gate, forget gate, and output gate. The calculation expressions of the parameters of LSTM are shown in Equations (14)–(19) [53].

\[
\begin{align*}
    f_t &= \sigma_f(W_f x_t + U_f h_{t-1} + b_f) \\
    i_t &= \sigma_g(W_i x_t + U_i h_{t-1} + b_i) \\
    o_t &= \sigma_g(W_o x_t + U_o h_{t-1} + b_o) \\
    \hat{C}_t &= \phi_c(W_c x_t + U_c h_{t-1} + b_c) \\
    C_t &= f_t \odot C_{t-1} + i_t \odot \hat{C}_t
\end{align*}
\]
where $x_t$ is the input vector, $f_t$ is the forget gate’s activation vector, $i_t$ is the update gate’s activation vector, $o_t$ is the output gate’s activation vector, $h_t$ is the output vector, $\hat{C}_t$ is the cell input activation vector, $C_t$ is the cell state vector, $W$, $U$ and $b$ are parameter matrices and vectors, $\sigma_g$, $\phi_c$ and $\phi_h$ are activation functions.

$$h_t = o_t \odot \phi_h(C_t),$$

where $x_t$ is the input vector, $f_t$ is the forget gate’s activation vector, $i_t$ is the update gate’s activation vector, $o_t$ is the output gate’s activation vector, $h_t$ is the output vector, $\hat{C}_t$ is the cell input activation vector, $C_t$ is the cell state vector, $W$, $U$ and $b$ are parameter matrices and vectors, $\sigma_g$, $\phi_c$ and $\phi_h$ are activation functions.

**Figure 4.** Basic structure of LSTM.

### 3.2.4. CNN

The CNN is constructed by an input layer, convolution layer, pooling layer, fully connected layer, and output layer. The input data in the input layer is first convoluted by a convolution kernel to make a convolution layer. Then, the pooling layer is to use the pooling method, such as max pooling, average pooling, etc., to effectively reduce the size of the parameter matrix, thereby reducing the number of parameters in the fully connected layer. Therefore, adding the pooling layer can speed up the calculation and prevent overfitting. After the pooling process, the pooled data is fed into the fully connected layer, which can be treated as the traditional multi-layer perceptron. The input of the fully connected layer is the feature extracted by the convolution layer and the pooling layer. The last output layer can use logistic regression, softmax regression, or even support vector machine to generate the final output. The network model adopts the gradient descent method to minimise the loss function to reverse-adjust the weight parameters in the network layer by layer, and improves the accuracy of the network through frequent iterative training.

The CNN is originally designed to deal with computer vision problems and the default input is the RGB image. This type of CNN is called as 3DCNN, because the RGB image can be filtered into three sub-image with RGB colours. If the input data is time series, then the CNN is called as 1DCNN. The basic structure of 1D-CNN is shown in Figure 5 [54].

**Figure 5.** Basic structure of 1DCNN.
3.2.5. Transformer

Transformer is the first transformation model that fully relies on self-attention to compute input and output representations without using recurrent or convolution mechanism. Self-attention is sometimes called as intra-attention. When a dataset is fed into the transformer, the data will first pass through the encoder module to encode the data, and then the encoded data will be sent to the decoder module for decoding. After decoding, the processed result will be obtained. The basic structure of Transformer is shown in Figure 6 [46]. It can be seen that the encoder input is fed into the input layer in the encoder, and then the positional encoding is used to inject some information about the relative or absolute position of the tokens in the sequence [42]. Then the encoder layer 1 and encoder layer 2 are used to encode the data. Here, the number of the encoder layers in the encoder can be defined by users. After then encoding process, the encoder output is fed into the decode layer 1 in the decoder. At the same time, decoder input is fed into the input layer of the decoder. Then the output of input layer is also fed into the decoder layer 1. After the process by decoder layer 2 and linear mapping, the final output can be obtained. Similarly, the number of the decoder layers in the decoder can also be defined by the users.

![Figure 6. Basic structure of Transformer.](image)

3.3. Process of Comparison

The comparison process is shown in Figure 7. The first step is to send the top 20 container ports’ throughput into the methods that need to be compared. Then the forecasting results are analysed from the perspectives of intra-method and inter-method, respectively. As an example, the pseudocode of learning and forecasting processes of MLP is presented in Algorithm 1. In the line 1, a range of the hidden layer size of the MLP is predefined. Then a variable named output, with an empty value, is predefined to hold the results generated by MLP methods with different hidden layer sizes. From line 3 to line 14, there are two for loops to get the forecasting results. More details about the searching range of each method can be found in Table 3. The source code of each forecasting method and the comparative drawing can be found at: https://github.com/tdjuly?tab=repositories (accessed on 20 September 2022).
Figure 7. Process of comparison.

Algorithm 1 An algorithm with caption.

```
1: hidden_size ← [8, 16, 32, 64, 100, 128, 256, 512]  ▶ To hold the model output
2: output ← []  ▶ hz is the size of the hidden layer of MLP
3: for hz in hidden_size do  ▶ epoch number, hz, learning rate, optimiser, etc.
4:   set model parameters  ▶ raw_set ← top 20 container ports’ throughput
5:   for time_series in raw_set do  ▶ train/test partition, min-max normalisation, etc.
6:     data processing
7:     define training model  ▶ test_MAPE, test_RMSE, etc.
8:     training
9:     load fitted model
10:    testing
11:   calculate assessment criteria
12:   output ← results
13: end for
14: save results
15: end for
```

Table 3. Searching range of each method.

| Learning Method | Searching Parameter       | Searching Range          |
|-----------------|---------------------------|--------------------------|
| NM              | No applicable            | No applicable            |
| MA              | Size of moving window    | 2                        |
| AR              | Autoregressive order     | 1                        |
| ARIMA           | No applicable            | No applicable            |
| MLP             | Size of hidden layer     | 8, 16, 32, 64, 100, 128, 256, 512 |
| GRU             | Size of hidden layer     | 8, 16, 32, 64, 100, 128, 256, 512 |
| LSTM            | Size of hidden layer     | 8, 16, 32, 64, 100, 128, 256, 512 |
| CNN             | Number of filters        | 8, 16, 32, 64, 100, 128, 256, 512 |
| Transformer     | Number of encoder/decoder layer | 1, 2, 3, 4 |

3.4. Data Description

In this study, annual container throughput from 2004 to 2020 was obtained from the official websites of the world’s top 20 container ports. For each port, there are 17 observations. The statistical description of the data is shown in Table 4 and the time plots of the
The container throughput of the world’s top 20 container ports are shown in Figure 8. It can be seen from the figure that the annual container throughput of most ports shows a trend of gradual increase, such as Antwerp, Guangzhou, Qingdao, Ningbo, Busan, etc. However, some, such as Hong Kong, showed a downward trend. Some ports, such as Dalian and Dubai, showed a trend of increasing first and then decreasing.

Before the experiment, the obtained data should be divided into the training set and testing set. The training set is used to tune the parameters of the model, so that the forecasting results of the model will be closer to the real value. The testing set is used to test the accuracy of the trained model on the new data. According to Al-Musaylh et al. (2018), 80/20 is a common ratio of training and testing sets [55]. Therefore, the training set includes 13 observations (76%) from 2004 to 2016. The testing set includes four observations (24%) from 2017 to 2020.

Table 4. Summary statistics of top 20 container ports’ throughput.

| Port     | Mean  | SD    | Max    | Min    |
|----------|-------|-------|--------|--------|
| Ningbo   | 1617.4| 786.05| 2872.00| 400.50 |
| Shanghai | 3167.23| 865.51| 4350   | 1455.4 |
| Singapore| 3031.71| 467.16| 3720   | 2132.9 |
| Shenzhen | 2222.69| 374.38| 2774   | 1365.5 |
| Hong Kong| 2187.56| 188.18| 2449.4 | 1830   |
| Busan    | 1664.03| 384.83| 2199   | 1149.2 |
| Guangzhou| 1442.5| 612.93| 2323   | 330    |
| Qingdao  | 1387.51| 510.66| 2200   | 514    |
| Dubai    | 1257.11| 271.97| 1573   | 642.9  |
| Tianjin  | 1154.67| 435.45| 1835   | 381.6  |
| Rotterdam| 1167.74| 193.12| 1482   | 829.2  |
| Kelang   | 984.91 | 54.38 | 1373   | 524.4  |
| Kaohsiung| 982.51 | 1099.3| 858.1  |
| Dalian   | 686.26 | 1021  | 220    |
| Hamburg  | 871.68 | 80.18 | 973.7  | 700.3  |
| Antwerp  | 891.72 | 1204  | 606.3  |
| Xiamen   | 719.56 | 281.92| 1141   | 287.2  |
| Tanjung  | 715.77 | 184.87| 985    | 402    |
| Los Angeles| 823.28| 77.63 | 946    | 710.3  |
| Long Beach| 682.15| 82.03 | 811    | 506.7  |

Figure 8. The container throughput of the world’s top 20 container ports from 2004 to 2020.
4. Results and Discussion

In this study, according to the flow of Algorithm 1, we completed the comparison of nine forecasting methods on the data collected. This chapter analyses the forecasting results from both intra-method and inter-method perspectives. In the intra-method comparison, we focus on comparing the forecasting performance of the same method in different time series, and analyse the reasons for this observation. In the inter-method comparison, we focus on the forecasting performance of different methods in the same time series, and analyse the reasons for this observation. Finally, according to the phenomena and reasons obtained, we draw conclusions and guide the subsequent forecasting research.

4.1. Intra-Method Comparison

Figure 9 presents the MAPEs and RMSEs of MLPs with different hidden sizes. It can be seen that 80% (16/20) of the container throughput time series can find lower MAPE and RMSE by increasing the hidden layer size of MLP, which include Antwerp, Busan, Dalian, Dubai, Guangzhou, Hong Kong, Kaohsiung, Kelang, Long Beach, Los Angeles, Ningbo, Rotterdam, Shanghai, Shenzhen, Tanjung, and Xiamen. In addition, many time series find the minimum error when the number of MLP layers is small, such as Antwerp, Kelang, Long Beach, Tanjung, etc. This observation indicates that increasing the size of the hidden layer is useful for finding models with higher forecasting accuracy when using MLP to build forecasting models. The number of MLP layers corresponding to the optimal result may not be too large.

Figure 10 presents the MAPEs and RMSEs of GRUs with different hidden sizes. One obvious experimental result is that MAPE and RMSE values of all ports decrease with the increase of hidden layer size, which means that the GRU forecasting accuracy of all ports becomes more accurate with the increase of hidden layer size. However, the growth rate of forecasting accuracy decreases rapidly at a certain stage (hidden layer size ≈ 100). This observation suggests that when using GRU to build the forecasting model, 100 can be selected as the initial hidden layer size considering the forecasting accuracy, computational complexity, and other factors, and then the hidden layer size can be modified to find the most appropriate hidden layer size.

Figure 9. MAPE and RMSE of MLP with different hidden size.
The MAPEs and RMSEs of LSTMs with different hidden sizes are presented in Figure 11. It can be seen that the values of MAPE and RMSE increase as the hidden layer size increases, which means that the LSTM forecasting accuracy of all ports becomes worse with the increase of hidden layer size. The possible reason for this situation is that LSTM model is good at analysing time series with a long time span and a large number of observations. For container throughput data, the LSTM model with a small time span and limited number of observations cannot accurately obtain the rules in container throughput, and it is easy to produce under-fitting results, which leads to lower accuracy with the increase of LSTM hidden layer size.

This observation suggests that when LSTM is used for forecasting model construction, it is not necessary to select a large hidden layer size.

Figure 11. Comparison between MAPE and RMSE of LSTM under different hidden layer size in the test set.
Figure 12 presents the MAPEs and RMSEs of CNNs with a different number of filters. It can be seen that around 90% of the CNN forecasting models can find the minimum MAPE and RMSE values under the increasing number of filters, which are Busan, Dalian, Dubai, Guangzhou, Hamburg, Hong Kong, Kaohsiung, Kelang, Los Angeles, Ningbo, Qingdao, Rotterdam, Shanghai, Shenzhen, Singapore, Tanjung, Tianjin, and Xiamen. There are also some unsynchronized changes in MAPE and RMSE, which are Guangzhou, Qingdao, Shenzhen, and Singapore. The unsynchronized change may be due to the change of extreme value in the forecasting results, because RMSE is more sensitive to the change of extreme value. Overall, this observation suggests that when CNN is used to build the container throughput forecasting model, it is necessary to increase the number of filters to search for the model that can produce the most accurate results.

Figure 12. MAPE and RMSE of CNN with different number of filters.

Figure 13 presents the MAPEs and RMSEs of Transformers with different number of layers. It can be seen from the figure that the forecasting accuracy of Transformer does not show the same rule as GRU, which is the forecasting error decreases with the increase of model size. However, we can find that by increasing the size of the Transformer, we can find the best model settings in the process of increasing the size. Among the 20 subfigures, 17 subfigures indicate that the minimum MAPE and RMSE have been found in the process of increasing the size of the Transformer. These ports are Antwerp, Busan, Dalian, Dubai, Guangzhou, Hamburg, Kaohsiung, Kelang, Long Beach, Los Angeles, Ningbo, Rotterdam, Shanghai, Shenzhen, Singapore, Tanjung, and Xiamen. This observation suggests that we can find better model parameters by increasing the size of the Transformer.
4.2. Inter-Method Comparison

Table 5 presents the test sets MAPE obtained by nine methods based on the results of the optimal training set on the container throughput time series of 20 ports. It can be seen that 17 of the 20 port container throughput time series obtained the minimum MAPE by using GRU. For the remaining three container throughput time series, the minimum MAPE of two series is generated by ARIMA, and the minimum MAPE of one series is generated by Transformer. The possible reason is that the length of the container throughput time series is too short to be explored by methods other than GRU. Similar observations suggest that GRU is able to perform better on certain smaller, less frequent datasets [56,57]. As a method with similar structure to GRU, LSTM performs worse than GRU. The possible reason is that due to the short length of time series used in this study, LSTM cannot obtain enough patterns from too short time series. For the time series of Guangzhou Port, the method to generate the minimum MAPE is ARIMA, but its result is very close to that of GRU, which are 2.2039 and 2.2658, respectively. This indicates that GRU can also produce relatively accurate results, but ARIMA is slightly more accurate.

In terms of the average MAPE of 20 ports, the best performing method is GRU, followed by CNN and NM. Surprisingly, the simplest NM method ranked third in the forecasting accuracy. Considering the simplicity, convenience, and ease of operation of the NM method, NM can be used for rapid and simple container throughput estimation when computing equipment and services are not available. This finding is consistent with previous studies, which also found that although NM results are not as good as other methods, the accuracy is very close [58]. Another finding is that the average performance of machine learning methods is better than the average performance of traditional methods, 7.89 and 8.39, respectively.
Table 5. MAPE obtained by nine methods based on the results of the optimal training set on the container throughput time series.

| Port     | NM   | MA   | AR   | ARIMA | MLP   | GRU   | LSTM  | CNN  | Transformer |
|----------|------|------|------|-------|-------|-------|-------|------|-------------|
| Antwerp  | 4.2967 | 8.5629 | 12.6199 | 2.6308 | 42.1258 | 1.2568 | 4.7404 | 3.7207 | 4.8954 |
| Busan    | 4.3966 | 8.9598 | 1.8104 | 1.8189 | 2.1596 | 1.2305 | 5.8232 | 2.9297 | 1.2705 |
| Dalian   | 15.5175 | 16.6758 | 20.6046 | 27.7902 | 15.0450 | 2.0448 | 8.0274 | 18.9709 | 20.5270 |
| Dalian   | 3.9585 | 8.9939 | 10.8113 | 22.6040 | 7.8605 | 1.3193 | 6.3575 | 5.9389 | 15.0013 |
| Guangzhou| 5.0806 | 16.6106 | 8.3624 | 2.0393 | 7.6721 | 2.2658 | 9.8339 | 3.3537 | 8.5603 |
| Hamburg  | 3.7098 | 10.6753 | 1.9775 | 2.5863 | 5.4786 | 0.7035 | 3.3343 | 2.8165 | 9.3777 |
| Hong Kong| 6.6044 | 6.4270 | 11.6346 | 33.6413 | 6.6200 | 2.4633 | 2.4470 | 5.6522 | 5.3062 |
| Kaohsiung| 3.0441 | 3.6576 | 3.0417 | 3.0994 | 3.9087 | 0.5937 | 2.1216 | 3.2833 | 4.3407 |
| Kelang   | 6.7878 | 11.4642 | 16.2776 | 12.6127 | 14.8417 | 1.5906 | 7.1609 | 8.7083 | 18.1385 |
| Long Beach| 7.1401 | 9.9497 | 15.2853 | 17.3519 | 16.2451 | 0.9496 | 3.4095 | 10.6295 | 35.0854 |
| Los Angeles| 2.2923 | 4.8444 | 14.9724 | 8.4657 | 105.9678 | 0.8928 | 3.2123 | 2.4537 | 5.2540 |
| Ningbo   | 6.8160 | 17.5070 | 7.4184 | 5.7412 | 5.5024 | 2.2467 | 10.1937 | 2.4335 | 10.0223 |
| Qingdao  | 4.8482 | 13.5448 | 4.4866 | 2.4632 | 5.3027 | 1.9691 | 9.0470 | 4.5223 | 7.8437 |
| Rotterdam| 5.1438 | 5.9598 | 14.5569 | 6.7150 | 14.6435 | 1.5742 | 10.2919 | 4.3124 | 7.5989 |
| Shanghai | 3.8454 | 11.0333 | 10.3761 | 2.3899 | 4.7858 | 2.1551 | 10.5326 | 2.5710 | 1.6235 |
| Shenzhen | 6.1554 | 8.4958 | 10.7214 | 3.7892 | 6.1664 | 1.4057 | 9.8344 | 5.4836 | 3.3679 |
| Singapore| 4.8712 | 8.1805 | 13.9995 | 14.0775 | 19.7786 | 1.0961 | 4.7537 | 4.0374 | 11.4153 |
| Tianjin  | 4.2048 | 10.3170 | 4.8853 | 1.8982 | 2.8838 | 1.5207 | 8.1054 | 4.4786 | 3.3916 |
| Tianjin  | 5.7244 | 13.9282 | 6.2496 | 1.5497 | 16.1054 | 2.0482 | 9.1047 | 3.0897 | 3.1095 |
| Xiamen   | 5.9579 | 13.9844 | 4.2122 | 3.7042 | 4.9255 | 1.9265 | 8.2918 | 4.7534 | 5.6281 |
| Average  | 5.4541 | 10.4530 | 9.6689 | 7.9463 | 16.7704 | 1.4780 | 6.8717 | 5.2247 | 9.0806 |

Table 6 presents the test sets RMSE obtained by nine methods based on the results of the optimal training set on the container throughput time series of 20 ports. It can be seen that the results of RMSE are similar to those of MAPE, and the best forecasting method is still GRU. However, there are also slight differences. The number of best performing ARIMA has increased from 2 to 3, and the number of best performing Transformer has increased from 1 to 2. The possible reason is that for some time series, GRU produces larger errors where the actual value is higher, which leads to larger differences. Moreover, RMSE is highly sensitive to the extreme value of errors, the results of RMSE for some time series are not ideal when GRU performs well in terms of MAPE.

| Port     | NM   | MA   | AR   | ARIMA | MLP   | GRU   | LSTM  | CNN  | Transformer |
|----------|------|------|------|-------|-------|-------|-------|------|-------------|
| Antwerp  | 60.71 | 84.19 | 158.48 | 39.46 | 489.99 | 21.43 | 56.63 | 51.65 | 71.64 |
| Busan    | 71.54 | 150.09 | 53.32 | 56.43 | 54.75 | 38.15 | 124.66 | 78.38 | 28.91 |
| Dalian   | 184.44 | 227.76 | 286.75 | 171.12 | 28.44 | 78.12 | 214.64 | 215.15 |
| Dalian   | 58.82 | 129.84 | 173.05 | 356.00 | 127.00 | 33.35 | 106.95 | 83.54 | 217.56 |
| Guangzhou| 126.61 | 219.79 | 198.95 | 54.49 | 174.70 | 70.79 | 216.35 | 74.79 | 214.67 |
| Hambur  | 41.96 | 110.20 | 24.63 | 33.68 | 55.36 | 9.93 | 32.52 | 31.05 | 103.45 |
| Hong Kong| 132.96 | 244.37 | 683.97 | 22.42 | 60.87 | 133.99 | 221.12 | 121.12 |
| Kaohsiung| 42.11 | 45.59 | 34.84 | 35.74 | 50.68 | 7.62 | 22.80 | 44.69 | 57.20 |
| Kelang   | 99.65 | 113.62 | 241.45 | 189.05 | 229.73 | 30.38 | 95.26 | 138.23 | 242.65 |
| Long Beach| 56.84 | 76.80 | 123.72 | 139.15 | 318.97 | 51.31 | 164.20 | 61.71 | 61.24 |
| Los Angeles| 30.06 | 50.01 | 140.40 | 81.09 | 57.11 | 30.21 | 91.21 | 59.05 | 74.58 |
| Average  | 5.4541 | 10.4530 | 9.6689 | 7.9463 | 16.7704 | 1.4780 | 6.8717 | 5.2247 | 9.0806 |

5. Conclusions

This research is a comparison study of nine forecasting methods on container throughput time series, four of which are traditional regression-based methods, and five of which are machine learning-based methods. The main finding of this study is that GRU is a method that can produce more accurate results with higher probability when constructing container throughput forecasting models. Another finding is that NM can be used for rapid and simple container throughput estimation when computing equipment and services are not available. The study also confirmed that machine learning methods are still the better choice over some traditional methods. An important conclusion that can be drawn from
the analysis of experimental results is that machine learning methods are useful for training forecasting models, but the characteristics of the data can affect the performance of the methods. Therefore, machine learning methods are not necessarily better than traditional forecasting methods. In other words, one should be cautious about using machine learning methods to build forecasting models. This study compares the performance of different methods on multiple time series, and these time series are characterised by short observation period and small number of observations. Therefore, the conclusion of this study is applicable to any time series with the same time characteristics as this study.

Although this study explores the performance of nine different methods in forecasting the throughput of the world’s top 20 container ports, there are still limitations. As the hub of world trade, the change of port throughput is not only determined by the port city, but also determined by the operation situation of various ports around the world and the development of world trade market. This study only uses historical port throughput data as the data source. Therefore, the future research direction is to add the above influencing factors such as the development of port facilities, economic data of port cities, transportation between the port and other ports into the forecasting model and analyse their impact on port container throughput.

Author Contributions: Conceptualization, S.X. and F.Z.; methodology, S.X.; software, S.X. and S.Z.; validation, S.Z., J.H., and W.Y.; formal analysis, S.X.; investigation, S.X.; resources, S.X. and F.Z.; data curation, S.X.; writing original draft preparation, S.X.; writing review and editing, F.Z.; visualization, S.Z., J.H., and W.Y.; supervision, S.X. and F.Z.; project administration, F.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Publicly available datasets were analysed in this study. This data can be found here: https://github.com/tdjuly/Port-container-throughput (accessed on 20 September 2022).

Acknowledgments: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References
1. Grzelakowski, A. Global container shipping market development and Its impact on mega logistics system. TransNav Int. J. Mar. Navig. Saf. Sea Transp. 2019, 13, 529–535. [CrossRef]
2. Heidari, A.; Tousanj, S.; Navimipour, N.J.; Unal, M. A privacy-aware method for COVID-19 detection in chest CT images using lightweight deep conventional neural network and blockchain. Comput. Biol. Med. 2022, 145, 105461. [CrossRef] [PubMed]
3. Toygar, A.; Yildirim, U.; Inegol, G.M. Investigation of empty container shortage based on SWARA-ARAS methods in the COVID-19 era. Eur. Transp. Res. Rev. 2022, 14, 1–17. [CrossRef]
4. Goncalves, P. Global Cargo Shortage: How Iron Boxes Became Money Magnets. 2022. Available online: https://uk.finance.yahoo.com/news/global-cargo-shortage-how-iron-boxes-became-money-magnets-084858021.html?guccounter=1&guce_referrer=aHR0cHM6Ly93d3cuZ29vZ2xlLmNvbS5oay8&guce_referrer_sig=AQAABAEmEad6Py72PQZAcyonGjijCYn1SXd1Z6gx4QZosQIDBnniHItsJAU66aq5KyB70o0WEFH73FQ7TQpdktreWEHfiQszuw9-gPJcf0Dx0RgaJwvJ4d1D-W-bTaFdcUUpeaRI3rnHwGtt0Xlew4bpBXTScnk43NHo6lvSes3ijs-3a_ (accessed on 29 September 2022)
5. Du, P.; Wang, J.; Yang, W.; Niu, T. Container throughput forecasting using a novel hybrid learning method with error correction strategy. Knowl. Based Syst. 2019, 182, 104853. [CrossRef]
6. Yang, C.H.; Chang, P.Y. Forecasting the demand for container throughput forecasting using a mixed-precision neural architecture based on CNN–LSTM. Mathematics 2020, 8, 1784. [CrossRef]
7. Moscoso-Lopez, J.A.; Urda, D.; Ruiz-Aguilar, J.J.; Gonzalez-Enrique, J.; Turias, I.J. A machine learning-based forecasting system of perishable cargo flow in maritime transport. Neurocomputing 2021, 452, 487–497. [CrossRef]
8. Justo-Silva, R.; Ferreira, A.; Flintsch, G. Review on machine learning techniques for developing pavement performance prediction models. Sustainability 2021, 13, 5248. [CrossRef]
9. Fildes, R.; Allen, P. Econometric forecasting: Strategies and techniques. In Principles of Forecasting: A Handbook for Researchers and Practitioners; Armstrong, J.S., Ed.; Kluwer Academic Publishers: Dordrecht, The Netherlands, 2001.
10. Hyndman, R.J.; Koehler, A.B. Another look at measures of forecast accuracy. Int. J. Forecast. 2006, 22, 679–688. [CrossRef]
41. Liu, S.; Ji, H.; Wang, M.C. Nonpooling convolutional neural network forecasting for seasonal time series with trends. *IEEE Trans. Neural Netw. Learn. Syst.* 2019, 31, 2879–2888. [CrossRef] [PubMed]

42. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is all you need. In Proceedings of the Advances in Neural Information Processing Systems 30 (NIPS 2017), Long Beach, CA, USA, 4–9 December 2017.

43. Floridi, L.; Chiatti, M. GPT-3: Its nature, scope, limits, and consequences. *Minds Mach.* 2020, 30, 681–694. [CrossRef]

44. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* 2018, arXiv:1810.04805.

45. Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; Liu, P.J. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.* 2020, 21, 1–67.

46. Wu, N.; Green, B.; Ben, X.; O’Banion, S. Deep transformer models for time series forecasting: The influenza prevalence case. *arXiv* 2020, arXiv:2001.08317.

47. Li, S.; Jin, X.; Xuan, Y.; Zhou, X.; Chen, W.; Wang, Y.X.; Yan, X. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. In Proceedings of the Advances in Neural Information Processing Systems 32 (NeurIPS 2019), Vancouver, BC, Canada, 8–14 December 2019.

48. Zhou, H.; Zhang, S.; Peng, J.; Zhang, S.; Li, J.; Xiong, H.; Zhang, W. Informer: Beyond efficient transformer for long sequence time-series forecasting. In Proceedings of the AAAI Conference on Artificial Intelligence, Virtual Event, 2–9 February 2021; pp. 11106–11115.

49. Mohammadi Farsani, R.; Pazouki, E. A transformer self-attention model for time series forecasting. *J. Electr. Comput. Eng. Innov. (JECEI)* 2021, 9, 1–10.

50. Klein, J.L. *Statistical Visions in Time: A History of Time Series Analysis, 1662–1938*; Cambridge University Press: Cambridge, UK, 1997.

51. Noriega, L. *Multilayer Perceptron Tutorial*; School of Computing, Staffordshire University: Stoke-on-Trent, UK, 2005.

52. Cho, K.; Van Merriënboer, B.; Bahdanau, D.; Bengio, Y. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv* 2014, arXiv:1409.1259.

53. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* 1997, 9, 1735–1780. [CrossRef]

54. Huang, S.; Tang, J.; Dai, J.; Wang, Y. Signal status recognition based on 1DCNN and its feature extraction mechanism analysis. *Sensors* 2019, 19, 2018. [CrossRef]

55. Al-Musaylh, M.S.; Deo, R.C.; Adamowski, J.F.; Li, Y. Short-term electricity demand forecasting with MARS, SVR and ARIMA models using aggregated demand data in Queensland, Australia. *Adv. Eng. Inform.* 2018, 35, 1–16. [CrossRef]

56. Chung, J.; Gulcehre, C.; Cho, K.; Bengio, Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv* 2014, arXiv:1412.3555.

57. Gruber, N.; Jockisch, A. Are GRU cells more specific and LSTM cells more sensitive in motive classification of text? *Front. Artif. Intell.* 2020, 3, 40. [CrossRef] [PubMed]

58. Lynch, C.J.; Gore, R. Application of one-, three-, and seven-day forecasts during early onset on the COVID-19 epidemic dataset using moving average, autoregressive, autoregressive moving average, autoregressive integrated moving average, and naïve forecasting methods. *Data Brief* 2021, 35, 106759. [CrossRef] [PubMed]