entropy

MDPI

Article

Minimum Message Length in Hybrid ARMA and LSTM Model Forecasting

Zheng Fang [1], David L. Dowe [1,*], Shelton Peiris [2] and Dedi Rosadi [3]

[1] Department of Data Science and Artificial Intelligence, Monash University, Clayton, VIC 3800, Australia; zfan51@student.monash.edu
[2] School of Mathematics and Statistics, University of Sydney, Camperdown, NSW 2006, Australia; shelton.peiris@sydney.edu.au
[3] Department of Statistics, Gadjah Mada University, Sleman, Yogyakarta 55500, Indonesia; dedirosadi@gadjahmada.edu
* Correspondence: david.dowe@monash.edu

Abstract: Modeling and analysis of time series are important in applications including economics, engineering, environmental science and social science. Selecting the best time series model with accurate parameters in forecasting is a challenging objective for scientists and academic researchers. Hybrid models combining neural networks and traditional Autoregressive Moving Average (ARMA) models are being used to improve the accuracy of modeling and forecasting time series. Most of the existing time series models are selected by information-theoretic approaches, such as AIC, BIC, and HQ. This paper revisits a model selection technique based on Minimum Message Length (MML) and investigates its use in hybrid time series analysis. MML is a Bayesian information-theoretic approach and has been used in selecting the best ARMA model. We utilize the long short-term memory (LSTM) approach to construct a hybrid ARMA-LSTM model and show that MML performs better than AIC, BIC, and HQ in selecting the model—both in the traditional ARMA models (without LSTM) and with hybrid ARMA-LSTM models. These results held on simulated data and both real-world datasets that we considered.We also develop a simple MML ARIMA model.

Keywords: long short-term memory; minimum message length; time series; neural network; deep learning; Bayesian statistics; probabilistic modeling

Citation: Fang, Z.; Dowe, D.L.; Peiris, S.; Rosadi, D. Minimum Message Length in Hybrid ARMA and LSTM Model Forecasting. *Entropy* **2021**, *23*, 1601. https://doi.org/10.3390/e23121601

Academic Editors: Eric Nalisnick and Dustin Tran

Received: 30 September 2021
Accepted: 25 November 2021
Published: 29 November 2021

1. Introduction

Forecasting in time series is a difficult task due to the presence of trends and/or seasonal components. For example, economic time series data are highly impacted by seasonal factors and often show trends with long-run cycles. Such trends and seasonality are difficult to capture by the traditional Autoregressive Moving Average model (ARMA) [1]. The Bayesian Minimum Message Length (MML) principle [2], the Akaike Information Criterion (AIC) [3], Schwarz's Bayesian information criterion (BIC) [4] and Hannan–Quinn (HQ) [5] are often used in model selection for the ARMA model [6–8]. The models selected by MML87 [9] in ARMA time series have lower prediction errors than those from AIC, BIC, and HQ [10]. Schmidt previously showed that MML87 outperforms a variety of other (information-theoretic) approaches in ARMA time series modeling [11] (chapters 5 to 8). In this paper, we extended the traditional ARMA time series model to form the hybrid ARMA-LSTM by combining the neural network of long short-term memory (LSTM) in order to test the performance of MML in model selection. The results suggest that MML outperforms AIC, BIC, and HQ.

The ARIMA is used with integer differencing to achieve stationarity if the time series is not stationary. A time series with seasonal components can be modeled using the family of seasonal ARIMA (or SARIMA) models. On the other hand, this ARIMA family has been generated to include long memory time series using a suitable fractional order

differencing in (0, 0.5) to form the family of autoregressive fractionally integrated moving average (ARFIMA) models. Nevertheless, the deep learning LSTM technique might be more suitable to capture the information that is less obvious in the time series, as it allows for a much more general class of models. Time series analysts require a lot of effort to discover the appropriate model in order to identify the dependency in time series data [12]. Historically, the ARMA model was introduced by Box and Jenkins in 1976 [13], and it is popular and widely used in the time series science community and provides accurate forecasts in both in-sample and out-of-sample data when the parameters are correctly estimated [14]. It is a hybrid (or mixture) of autoregressive (AR) and moving average (MA) processes, but the ARMA model can only be used in stationary time series [15].

In parallel, machine learning has seen the development of neural network models in computer science, ultimately influencing statistics. Similar to the families of the ARMA model, deep learning also has several variants, such as a deep neural network (DNN), a convolutional neural network (CNN), and a recurrent neural network (RNN). This report investigates a particular form of the RNN called long short-term memory (LSTM), which is typically used in time series [16]. In recent years, LSTM has been shown to work well in forecasting for data with complex time dependency, such as the stock market and energy consumption prediction [17]. In this paper, we select the best ARMA(p,q) model and then train the LSTM model for the residuals through the ARMA model. The time-step order used in LSTM is the parameter q in ARMA(p, q) determined by different information-theoretic criteria [18,19].

Our results show that MML compares favorably with the other information-theoretic approaches, including AIC, BIC, and HQ, when conducting ARMA-LSTM. Further, we compare the ARMA-LSTM selected by MML with the ARMA model selected by MML. These results also show that MML outperforms when compared to AIC [20], BIC [20], and HQ [21] in terms of selecting a model with lower prediction error, and this holds whether our modeling is enhanced by LSTM or instead is ARMA unassisted by LSTM. The Bayesian information-theoretic MML principle provides more reliable and highly accurate results in the model selection of the hybrid ARMA-LSTM model than other traditional methods (AIC, BIC, HQ). When doing ARMA without a hybrid with LSTM, MML also performs better than other traditional methods (AIC, BIC, HQ). The best performing method considered is the hybrid MML ARMA-LSTM model. These results hold on simulated data and on the real-world datasets considered.

Section 2 introduces the Box and Jenkins theory for the ARIMA model and discusses its limitations. Section 3 introduces the information-theoretic Minimum Message Length criterion in model selection, and Section 4 introduces the deep learning model LSTM. Section 5 provides the algorithm of the hybrid ARMA-LSTM model, and Section 6 provides the experimental results with a comparison.

2. ARIMA Modeling

This section reviews the theory of Autoregressive Integrated Moving Average (ARIMA) modeling from Box and Jenkins (1970) [13,15]. Let $\{Y_t\}$ be a homogeneous nonstationary time series and suppose that the d^{th} $(d = 1, 2, \ldots)$ difference of the series is stationary and is given by $X_t = (1 - B)^d Y_t$,, where B is the backshift operator. Then a stationary ARMA(p,q) model can be fitted for $\{X_t\}$, satisfying

$$X_t = c + \sum_{i=1}^{p} \phi_i X_{t-i} + \epsilon_t + \sum_{i=1}^{q} \theta_i \epsilon_{t-i}, \tag{1}$$

where $\{\epsilon_t\} \sim WN(0, \sigma^2)$.

Let $\phi(B) = 1 - \phi_1 B - \ldots - \phi_p B^p$; $\theta(B) = 1 + \theta_1 B + \ldots + \theta_q B^q$, be two polynomials of degree p and q, respectively, such that the zeros of $\phi(B)$ and $\theta(B)$ are outside the unit circle. Then the ARMA(p,q) in Equation (1) can be written in a compact form as

$$\phi(B) X_t = c + \theta(B) \epsilon_t. \tag{2}$$

Now the corresponding ARIMA(p,d,q) model for the original series $\{Y_t\}$ is given by

$$\phi(B)(1-B)^d Y_t = c + \theta(B)\epsilon_t. \tag{3}$$

It is known that ARIMA is a form of a linear regression model with the lag order of time series data and corresponding residuals. In an application where the ARIMA model fits well for the given data, then the corresponding residuals through the model should form a random scatter plot with a constant mean and a constant variance over the time, see, for example, [13]. If the ARIMA model is not well fitted for the data or an incorrect model has been fitted, then the residuals will not show a random scatter plot and instead indicate autocorrelations within the residuals. This reveals that the information hidden in the data has not been completely captured by the fitted ARIMA model, and we consider refitting an alternative ARIMA model [22].

The above family of ARIMA models are also capable of modeling a wide range of seasonal data using slight modifications. A seasonal extension of Model (3) can be written for a set of time series data with seasonality m. Incorporating both the seasonal and nonseasonal components together with additional polynomials, a new model is

$$\phi(B)\Phi(B^m)(1-B)^d(1-B^m)^D Y_t = c + \theta(B)\Theta(B^m)\epsilon_t, \tag{4}$$

where $\Phi(B^m) = 1 - \Phi_1 B^m - \ldots - \Phi_P B^{mP}$, $\Theta(B^m) = 1 + \Theta_1 B^m + \ldots + \Theta_Q B^{mQ}$, and D is the degree of seasonal differencing. For simplicity, this is written as

$$Y_t \sim SARIMA(p,d,q)(P,D,Q)_m \tag{5}$$

Model (4) is known as the Seasonal ARIMA or SARIMA model.

To estimate the parameters of Model (4), it is important to identify the changes of variance in the autocorrelation function (ACF) plot of data. This ACF provides an indication of linear dependencies among the observation of time series, which is related to the order of the model. In addition, the corresponding partial autocorrelation function (PACF) can be used to confirm the approximate order required in the model.

In this study, we use non-seasonal ARIMA modeling because the non-seasonal degree of differencing d can be predetermined in practice. We consider the stationary time series data. Assuming the data are generated from a mean zero stationary ARMA(p,q) process with Gaussian errors, we use the fact that the distribution of data is a multivariate Gaussian distribution with mean $\mu = 0$.

Suppose that we have a sample of N observations $y = (y_1, ..., y_N)$ generated through Model (2), with $c = 0$, and let $\boldsymbol{\beta} = (\phi_1, ..., \phi_p, \theta_1, ..., \theta_q, \sigma^2)$ be the vector of all the parameters. Then the corresponding unconditional log-likelihood function, $L(y\beta)$, can be written as:

$$L(y\beta) = -\frac{N}{2}\log(2\pi\sigma^2) - \frac{1}{2}\log\Sigma - \frac{1}{2\sigma^2} y^T \Sigma^{-1} y, \tag{6}$$

where Σ is the determinant of Σ and $\sigma^2\Sigma$ is the $N \times N$ theoretical autocovariance matrix of y.

3. Minimum Message Length

The Bayesian information-theoretic Minimum Message Length (MML) principle [2,6,7,9,19,23] is based on coding theory and can be thought of in several equivalent ways. It can be thought of in terms of a transmitter encoding a two-part message and transmitting it to a receiver, where the first part of the message contains information encoding the model and the second part of the message encodes the data given the model. The length of the first part of the message can be thought of as the complexity of the model, and the length of the second part of the message (effectively, the statistical negative log-likelihood) is a measure of goodness of fit to the observed data. For example, with

$X = \{A, B, C, D\}$, possible encodings would be, e.g., $A = 00, B = 01, C = 10$, and $D = 11$, or instead, e.g., $A = 1, B = 01, C = 001$, and $D = 0001$, with the length of code represented as $I()$, e.g., with $A = 00$, $I(A) = 2$. The code length is typically (close to) the negative logarithm of the probability.

MML thus gives a quantitative information-theoretic trade-off between model complexity (length of first part of message) and goodness of fit (length of second part of message) [24]. A smaller MML value (or, equivalently, a shorter message length) indicates the model is less complex and highly fitted to the data [6]. In practice, minimizing the message length can be expressed as:

$$\arg\min_{\theta \in \Theta} \{\mathrm{I}(\theta) + \mathrm{I}(y^N \theta)\}, \tag{7}$$

where $I(\theta)$ is the length of encoding the assertion (or model), and $I(y^N\theta)$ is the length of encoding the detail (or data given the model). In MML, there is (Bayesian) prior knowledge (or a prior distribution), π, over the parameter space. Following Wallace and Freeman [9], MML has been shown to work well in time series models, such as autoregressive (AR) and moving average (MA) models [18,25,26]. We can thus estimate the parameters [7,9] by minimizing the message length:

$$MessLen(y, \beta) = -\log\left(\frac{h_3(\beta) f(y_1, ..., y_N \beta) \epsilon^N}{\sqrt{F(\beta)}}\right) + \frac{k}{2}(1 + \log \kappa_k) - \log h_1(p) - \log h_2(q), \tag{8}$$

where ϵ is measuring the accuracy of data, $h_3(\beta)$ is the Bayesian prior distribution over the parameter set β, we model the parameter set β using uniform prior [0, 1] in the stationarity region $h_3(\beta) = 1$, and $h_1(p) = 2^{-(1+p)}$ and $h_2(q) = 2^{-(1+q)}$ are the priors on the (non-negative integer) parameters p and q, $k = p + q + 1$ is the number of continuous-valued parameters, $f(y_1, ..., y_N \beta)$ is the standard statistical likelihood function, $L = -\log f$, $F(\beta)$ is the expected Fisher Information matrix (of expected second-order partial derivatives of L) and is a function of the parameter set β, $F(\beta)$ is the expected Fisher information, κ_k is the lattice constant (which accounts for the expected error in the log-likelihood function from ARMA model (Equation (6)) due to the quantization of the k-dimensional space, which is bounded above by $\frac{1}{12}$ and bounded below by $\frac{1}{2\pi e}$. For example, $\kappa_1 = \frac{1}{12}$, $\kappa_2 = \frac{5}{36\sqrt{3}}$, $\kappa_3 = \frac{19}{192 * 2^{1/3}}$, and $\kappa_k \to \frac{1}{2\pi e}$ as $k \to \infty$).

Ignoring the $-\log h_1(p)$, $-\log h_2(q)$, and $-N \log(\epsilon)$ terms, the message length for the ARMA model β can also be represented as:

$$I(y, \beta) = -\log h_3(\beta) + \frac{1}{2} \log F(\beta) + \frac{k}{2} \log \kappa_k + \frac{k}{2} - \log f(y\beta) \tag{9}$$

MML87 is model invariant and avoids explicitly constructing the quantized parameter space [7–9,23]. This is used for model selection and parameter estimation by choosing the model that minimizes the message length.

MML has been used for a variety of problems, including clustering and mixture modeling [27,28] ([19] Section 6.8), clustering of protein dihedral angles [29], decision graphs (as an extension of decision trees, allowing for disjunctions, or "or") [30] (Section 7.2.4 [19]) and multi-way joins in decision graphs with dynamic attributes [31], causal Bayesian nets (or Bayesian networks, or causal nets) ([19] Section 7.4) and Bayesian nets with decision trees in their (leaf) nodes [32,33], inference of probabilistic finite state automata (or probabilistic finite state machines, PFSAs, PFSMs) ([19] Section 7.1) and hierarchical PFSAs [34], and (given sufficient data and time, and based to whatever degree on the above-mentioned inference of Bayesian nets) automation of database normalization [35], etc.

Part of the reason for the above list is the universality of the MML approach [7] ([19] Chapter 2) (seeking the single best theory) and that of the predictive approach (seeking a Bayesian weighted combination of theories) of Solomonoff [36,37] ([38] Section 3.1).

The MML approach of Wallace and the algorithmic probability approach of Solomonoff both have many desirable properties, but they can be slow in practice, whereas deep learning often runs relatively quickly. This motivates us to combine these approaches, as we do using the deep learning approach of long short-term memory (LSTM). This gives us something of a combination of the simplicity and accuracy of MML and the speed of deep learning.

We note in passing that an earlier effort at combining MML with neural nets is [39]. We further note that some approaches to deep learning use a (suitably weighted) combination of a squared error term and a Kullback–Leibler divergence term. Given that squared error comes (or can come) from a Gaussian log-likelihood, this version of deep learning regularization bears similarities to D. F. Schmidt's MML approximation [11] ([6] footnotes 64 and 65).(The MMLD version of MML ([6] Section 0.2.2, p. 528) [40] ([19] Sections 4.10, 4.12.2 and 8.8.2, p. 360) modified MML87 [9] to allow for cases when the Bayesian prior is not approximately constant over the relevant region. D. F. Schmidt's MML approximation, just discussed, is a further modification, and explicitly introduces Kullback–Leibler divergence into the expression.) We also ask, for future work, whether our approach might be combined with graph neural networks [41] or (higher-dimensional) hyper-graph neural networks.

4. Long Short-Term Memory (LSTM)

With the development of computational power in electronic equipment, powerful computers provide many learning algorithms and approaches in time series forecasting [42–44]. Deep learning is one of the popular approaches in recent years; it provides a complex model that has at least the potential to capture (and often does capture) more general information from the predictors than a traditional model, such as ARMA. Long short-term memory (LSTM) is a special kind of recurrent neural network introduced by Hochreiter and Schmidhuber in 1997 [45]. LSTM manages the two state vectors, the short-term state h_t and long term state c_t, and uses the gating mechanism by adding linear components from the previous layer in order to provide the long memory. LSTM has been widely used in time series forecasting because it is able to capture more information in the time series data, particularly for the financial econometrics area, where the price of financial assets depends on various different factors that are difficult to represent by a linear model [44,46]. Each LSTM layer, including the cells of the forget gate, input gate, and output gate, is shown in Figure 1.

- Forget gate: $f_t = \sigma(U^f x_t + W^f h_{t-1} + b^f)$;
- Input gate: $i_t = \sigma(U^i x_t + W^i h_{t-1} + b^i)$;
- Output gate: $o_t = \sigma(U^o x_t + W^o h_{t-1} + b^o)$.

The forget gate uses a sigmoid function $\sigma(x)$ from Equation (10). It has a value between 0 and 1, and it determines how much information should be forgotten. If the result from the sigmoid function is close to 0, then more information should be forgotten, and if the result from the sigmoid function is close to 1, then less information should be forgotten.

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{10}$$

The input gate also uses the sigmoid function, the input gate controls the value input from the input function of $g_t = tanh(Wh_{t-1} + Ux_t + b)$ using the $tanh(x)$ function:

$$tanh(x) = \frac{sinh(x)}{cosh(x)} = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{11}$$

The input gate controls how much information should be remembered. The LSTM long-term state uses an element-wise operation with $c_t = f_t \odot c_{t-1} + g_t \odot i_t$, where $\odot$ is element-wise multiplication (of two matrices of the same dimension), also known as the Hadamard product.

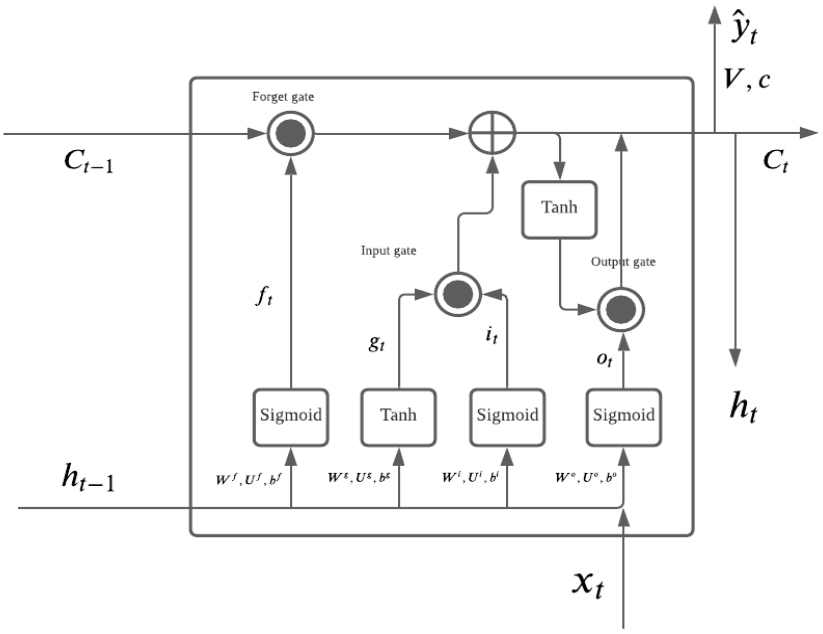

Figure 1. LSTM Structure.

The output gate o_t controls how much long-term information c_t should be carried forward to the next layer, and it also contributes to the short-term state of h_t. The result from the output gate function is also between 0 and 1, and the LSTM short-term state also uses element-wise multiplication, with $h_t = o_t \odot tanh(c_t)$. An LSTM with more than one layer is shown in Figure 2, and its structure enables the LSTM to capture long-term and short-term information in order to forecast. As usual, an LSTM is trained by back propagation as other neural network models are. An LSTM requires time series data to train the model, and its time series pattern will be modeled in every layer of the network.

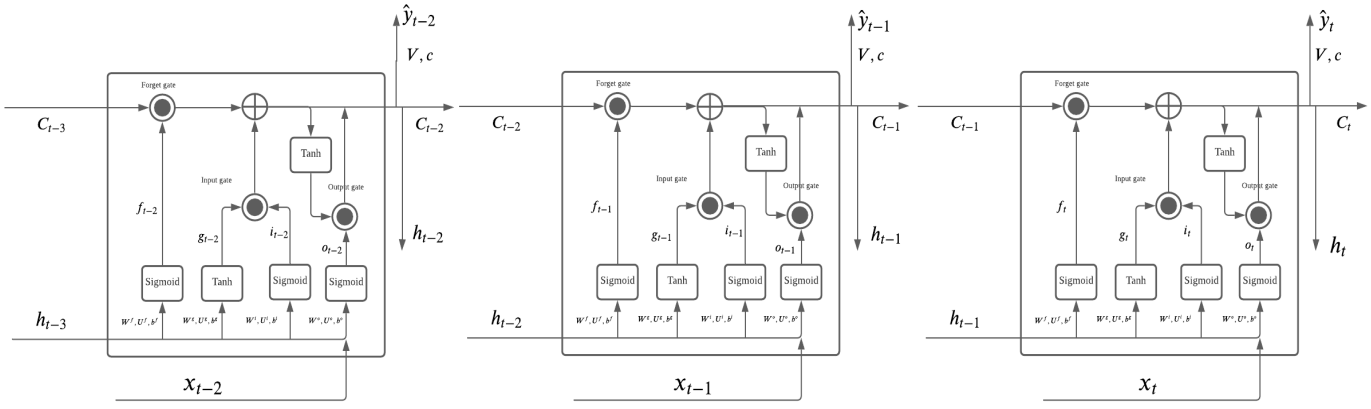

Figure 2. LSTM Overlapping.

5. Hybrid ARMA-LSTM Model

In recent years, LSTM and its variants—along with some hybrid models—have been thought by many to largely dominate the financial time series forecasting domain [44]. The LSTM is able to capture the dependency of residuals across time, and the LSTM is trained by the time step [47]. In this paper, we are using the Moving Average lag order q from ARMA parameters selected by MML87, AIC, BIC, and HQ—if $q = 0$, then we only use ARMA to forecast the time series data without LSTM. Our LSTM model is composed of

a single input layer with an input shape of MA order and the sequence learning features. The following LSTM layer also contains the sequence learning features, and the third LSTM layer with the same unit is followed by the fourth dense layer with one unit.

We developed Algorithm 1 based on [17] using a different loss function and activation function in the regression task. The hybrid ARMA-LSTM model trains the LSTM model by the residuals from the ARMA model. (This is similar in spirit to the discussion in ([7] (Section 5.1))). The simple point here is that the LSTM has at least the potential to find dependencies that the ARMA model (on its own) can not express. In this paper, MML87, AIC, BIC, and HQ have been used to select the model parameter orders from the ARMA model; so, this paper not only compares the errors of the hybrid ARMA-LSTM model with those from the single ARMA model but also the hybrid model in terms of the selection(s) of MML87, AIC, BIC, and HQ. The forecast from the ARMA model is the fitted mean μ_{t+1}. Because information is hidden in the residuals from the ARMA model (in a similar vein to ([7] (Section 5.1))), the forecast of the hybrid model will be

$$\hat{Y}_{t+1} = \mu_{t+1} + \mathrm{E}_{t+1} \tag{12}$$

where μ_{t+1} represents the linearity modeling of data from the ARMA model selected according to the information-theoretic MML87, AIC, BIC, and HQ. The term ϵ_t is the residual left by the ARMA model $Y_t - \hat{Y}_t$, and $\mathrm{E}_{t+1} = f(\epsilon_t) = f(Y_t - \hat{Y}_t)$, which is forecasted by the LSTM based on the past residual values $\epsilon_t, \epsilon_{t-1}, ..., \epsilon_{t-q}$, where the parameter q is selected by MML87, AIC, BIC, and HQ. The hybrid ARMA-LSTM model combines both linear and non-linear tendencies in time series data [48].

Algorithm 1 Algorithm 1 with the LSTM Model [17].

Require: number of epochs = 10
  while MA(q) order in order set selected by MML, AIC, BIC, and HQ **do**
    model.add(LSTM(30, return_sequences=True, input_shape=(q, 1)))
    model.add(LSTM(30, return_sequences=True))
    model.add(LSTM(30))
    model.add(Dense(1))

The algorithm of the hybrid model is shown below (Algorithm 2):

Algorithm 2 Algorithm 2 with the Hybrid ARMA-LSTM Model.

Require: number of data $n \geq 0$
  while $N \leq$ number of different simulations **do**
    while $n \leq$ number of dataset in simulation **do**
      while $i \in$ MA orders selected from MML, AIC, BIC, and HQ **do**
        if $i \neq 0$ **then**
          Train LSTM model by the residuals of ARMA model
          Rolling forecast the residual by LSTM
          Calculate root mean squared error by Y_{t+1}
        else if $i = 0$ **then**
          Calculate root mean squared error by forecast from ARMA only

6. Experiments

The experiments have been designed to compare the results of the ARMA model itself with the hybrid ARMA-LSTM model and also to compare different versions of the hybrid model with the parameters variously selected by the MML87, AIC, BIC, and HQ. In order to analyze the accuracy of forecasting, we are using the root mean squared error, $\mathrm{RMSE} = \sqrt{\frac{1}{T}\sum_{t=1}^{T}(y_t - \hat{y}_i)^2}$, to compare the different results, where T stands for the forecast window size, and we are using rolling forecast in this experiment. To elaborate and clarify,

for the financial data in Section 6.2, we do integer differencing with $d = 1$ to obtain stationarity before using the ARMA model and, as such, use an ARIMA or autoregressive integrated moving average model. We compare the performance of ARMA, ARIMA-LSTM, and LSTM alone on simulated dataset(s) (Section 6.1) and also on real-world financial (Section 6.2) and air pollution (Section 6.3) datasets.

We argue elsewhere (([6] footnotes 75 and 76) ([23] Section 3) ([38] Section 4.1)) about various uniqueness and invariance properties of log-loss (or logarithm loss). Squared error is a popular method and is also a variant of log-loss.

6.1. Simulated Dataset(s)

In this section, we perform experiments using various previously described modeling methods on simulated data, and we begin (in terms of LNPPP space ([6] Section 0.2.7)) by describing the experiments. We use a uniform distribution on $[-0.9, 0.9]$ (from minimum -0.9 to maximum 0.9) to randomize the parameters p and q of ARMA(p, q) for the data simulation by using the *arima.sim* function in R and then reject them if they are outside the stationarity region. There are $5 \times 2 = 10$ different parameter sets from $p_1, ..., p_5$ and q_1, q_2. The values in the table are the average RMSE over 100 runs (with standard deviation in brackets) in the simulated dataset corresponding to the particular parameters. The dataset includes N = 50, 100, 200, 300, and 500 time series data points in one dataset and also includes forecast windows of window size(s) $T = 3, 10, 30$, and 50. Table 1 shows the average of RMSE trained by LSTM alone (with different numbers of LSTM time steps) with different forecast window sizes, T. The results suggest that the LSTM alone does not work well in ARMA simulated data. For convenience of reading, we have moved Tables A1–A8 to the Appendix; each value in Table A1 is the average RMSE of forecast errors over the datasets (with standard deviation in brackets). The bold texts indicate the smallest forecast errors from the different kinds of models. Tables A1–A4 provide a comparison of different forecast window sizes (or window size) with T = 3, 10, 30, and 50.

Table 1. RMSE in LSTM for simulated data (p_1, q_1) with different time steps and N = 100.

No. of LSTM Time Steps	$T = 3$	$T = 10$	$T = 30$	$T = 50$
1	1.2519	1.3677	1.4962	1.3911
2	1.1794	1.2442	1.3863	1.2718
3	1.3372	1.6324	1.2256	1.3018
4	1.2195	1.2301	1.3284	1.3951
5	1.1341	1.6294	1.4276	1.4494

Table A2 shows the results for the average RMSE in the datasets for different simulated ARMA parameter sets, with the forecast window of $T = 10$. Table A3 provides the comparison of root mean squared error results of those datasets in different criteria, also comparing different simulated datasets with the forecast window of $T = 30$.

A large forecast window usually decreases the accuracy for the time series model. A window size of T = 50 (Data provided by Table A4) is 50% of the size of the in-sample set, and the MML87 hybrid model still outperforms its rivals. This indicates that the MML information criterion is efficient in model selection, and the algorithm of the hybrid model is also efficient in time series analysis, with the result of $T = 50$, as shown in Table A4. Table 2 shows the average of the ten different parameters of the simulated dataset in the forecast window sizes of $T = 3$ (Data provided by Table A1), 10 (Data provided by Table A2), 30 (Data provided by Table A3) and 50 (Data provided by Table A4) with the in-sample size of $N = 100$.

Table 2. Average of RMSE in forecast window size T = 3, 10, 30, and 50.

	Average of RMSE							
	ARMA				**ARMA-LSTM**			
	AIC	**BIC**	**HQ**	**MML87**	**AIC**	**BIC**	**HQ**	**MML87**
T = 3	1.086	1.076	1.090	1.072	1.121	1.123	1.134	1.115
T = 10	1.149	1.136	1.144	1.121	1.159	1.136	1.143	1.134
T = 30	1.338	1.331	1.340	1.325	1.234	1.221	1.224	1.220
T = 50	1.308	1.296	1.297	1.295	1.225	1.195	1.219	1.221

MML87 outperforms the rival methods in the in-sample size of $N = 100$ in all cases of $T = 3, 10, 30$, and 50. MML87 not only considers the goodness of fit of data but also considers the model complexity. Figure 3 shows that MML87 has a lower root mean squared error in most cases. The hybrid model selected by MML87 has the lowest error rate for $T = 3, 10$, and 30. These comparisons argue well for MML. The results of $N = 100$ with $T = 50$ seem to suggest that for a large size of the forecast window, the complex hybrid ARMA-LSTM model seems to perform better than the simple time series model. Given that the simulated data were generated from an ARMA model, it is not immediately apparent why adding LSTM to produce a hybrid model should be advantageous in the case of larger datasets (although we would typically expect this if not dealing with data that are purely from an ARMA model). Table 3 shows the average of the ten different parameters of the simulated dataset in the in-sample size of N = 50 (Data provided by Table A5), 100 (Data provided by Table A2), 200 (Data provided by Table A6), 300 (Data provided by Table A7), and 500 (Data provided by Table A8).

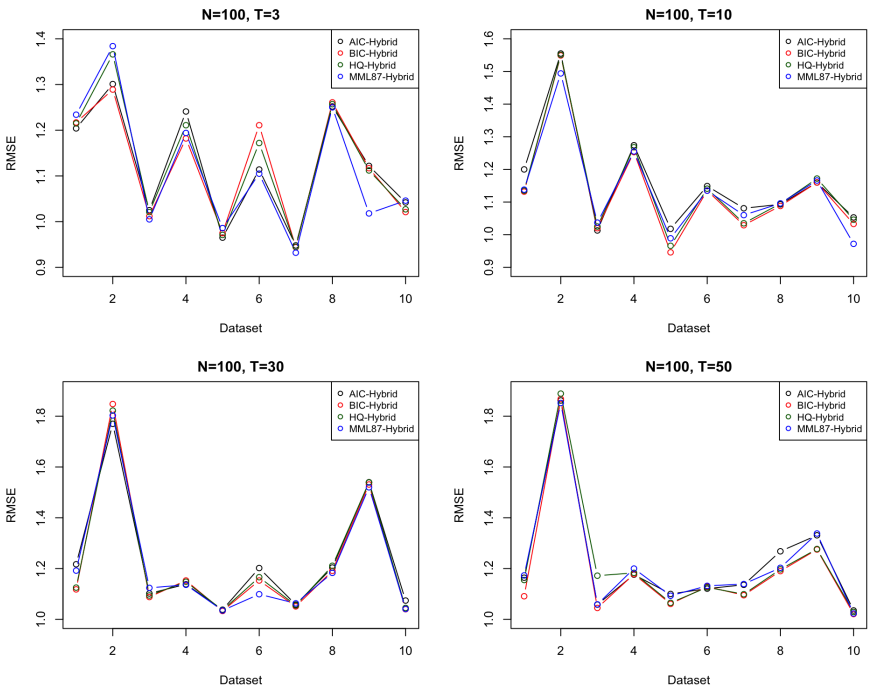

Figure 3. Comparison in forecast window sizes T = 3, 10, 30, and 50.

Table 3. Average of RMSE for in sample size N = 50, 100, 200, 300, and 500 with forecast window size T = 10.

	Average of RMSE (and Standard Deviation)							
	ARMA				**ARMA-LSTM**			
	AIC	**BIC**	**HQ**	**MML87**	**AIC**	**BIC**	**HQ**	**MML87**
N = 50	1.301	1.291	1.299	1.280	1.224	1.202	1.216	1.244
N = 100	1.149	1.136	1.144	1.121	1.159	1.136	1.143	1.134
N = 200	1.177	1.187	1.189	1.183	1.159	1.161	1.164	1.154
N = 300	1.163	1.152	1.155	1.147	1.131	1.125	1.124	1.123
N = 500	1.194	1.197	1.1984	1.196	1.180	1.173	1.179	1.181

Tables A5–A8 compare six different models or model selection techniques in the RMSE of the dataset in $N = 50, 200, 300$, and 500, with the forecast window size $T = 10$. AIC tends to overfit for small datasets, such as $N = 50$ (Data provided by Table A5 in the Appendix A). Through an increase in the amount for the in-sample dataset, the RMSE decreases in the hybrid ARMA-LSTM model because the larger size of data helps the LSTM to train and fit an accurate model. Thus, the results show the RMSE for the MML87 model is lower than the other models in the range $N = 100, 200$, and 300. Because of the efficiency in controlling the model complexity in MML87, the model can avoid the overfitting problem for small datasets.

The hybrid model with LSTM overfits when the in-sample size is small, basically because there is a larger amount of parameters that need to be estimated compared to the pure ARMA model. On the other hand, the hybrid model tends to perform well for a large in-sample size because the deep learning model is often better off for a large in-sample size, such as $N = 200$ (Data provided by Table A6), 300 (Data provided by Table A7), and 500 (Data provided by Table A8).

For a small in-sample size, such as $N = 50$, the BIC performance is good on the hybrid ARMA-LSTM because BIC is able to select the model well without overfitting. The MML87-Hybrid has the smallest average RMSE for $N = 100, 200$, and 300 for the different randomized datasets. The hybrid models work efficiently when there is enough in-sample data; otherwise, it can also overfit small datasets. In the meantime, by comparing the RMSE from MML87-ARMA, AIC-ARMA, BIC-ARMA, and HQ-ARMA, the results favor MML87 rather than AIC, BIC, and HQ. MML87 has a good performance in time series model selection and is able to select the ARMA model with lower forecasting errors. However, as noted earlier in this section, given that the simulated data were generated from an ARMA model, it is not immediately apparent why adding LSTM to produce a hybrid model should be advantageous in the case of larger datasets (although we would typically expect this if not dealing with data that are purely from an ARMA model). Figure 4 shows the comparison of RMSE in the in-sample size N = 50, 200, 300, and 500.

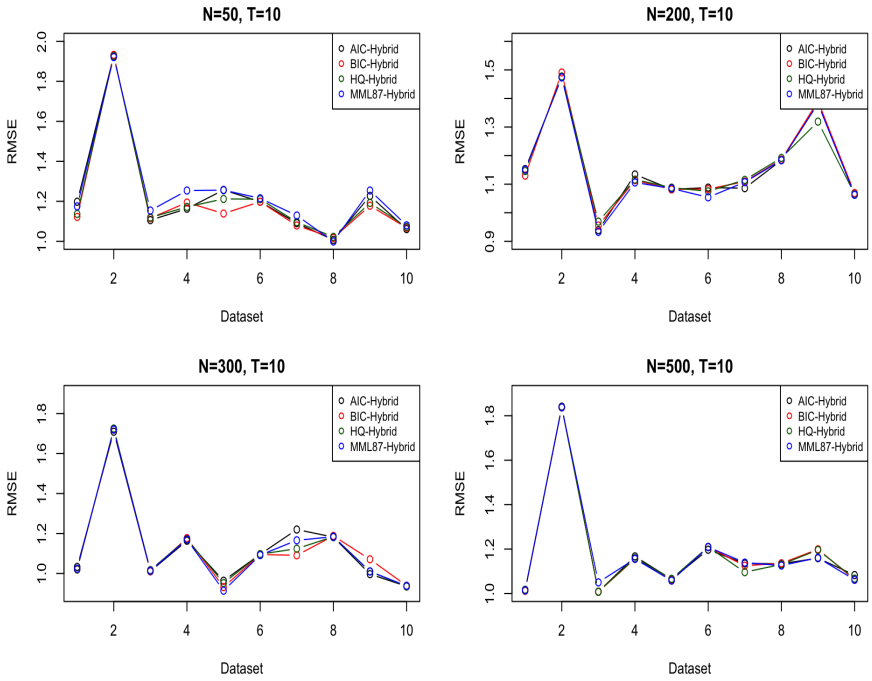

Figure 4. Comparison in the in-sample size N = 50, 200, 300, and 500.

6.2. Financial Data-and Extension to ARIMA Models

Stock return prediction is one of the most popular research topics in economics and finance [49,50]. This section studies the performance of the hybrid model from MML87; the hybrid models from AIC, BIC, HQ; and the ARIMA models selected by MML87, AIC, BIC, and HQ. The stock prices were selected from the components of the Dow Jones Industrial Average, including Apple (APPL), Boeing (BA), Cisco System (CSCO), Goldman Sachs (GS), IBM, Intel (INTC), Johnson & Johnson (JNJ), JPMorgan Chase (JPM), Coca-Cola (KO), and 3 M (MMM).The data selected start at 23 September 2016 and finish at 22 September 2021, with a total of 1258 trading days. This experiment studies the different performances in forecast window sizes $T = 3, 5, 10, 30, 50, 70, 100, 130, 150$, and 200. Table 4 shows the characteristic of stock prices selected, including mean, standard deviation, and partial autocorrelation.

Table 4. Mean, standard deviation, PACF lag 1 to 3 for ten selected stocks.

	Mean	S.D	PACF1	PACF2	PACF3
AAPL	66.440217	37.060808	0.996875	0.044454	−0.004848
BA	258.704781	82.478194	0.995870	−0.031231	−0.061804
CSCO	40.585947	8.595774	0.994585	0.073202	−0.016488
GS	227.095242	56.820929	0.993579	0.039741	−0.043412
IBM	124.851224	10.369478	0.982339	0.070195	−0.040622
INTC	46.269478	9.305502	0.992194	0.178757	−0.053398
JNJ	130.715314	18.399352	0.993930	0.050988	−0.031304
JPM	104.046116	24.467471	0.993854	0.067756	−0.049235
KO	44.519034	6.089778	0.993828	0.031639	−0.039178
MMM	173.550240	20.467854	0.991641	0.004475	0.026664

The empirical results show that the hybrid ARIMA-LSTM model can substantially outperform the traditional ARIMA (Autoregressive Integrated Moving Average) time series

model, particularly in the forecast window sizes of $T = 5, 30, 100, 130, 150$, and 200. Many studies demonstrated that the stock return depends on various factors, such as dividend yield, the book to market ratio, and/or interest rate [49,51,52]. However, traditional linear time series models are not able to take into account the effect of all those factors, thus requiring a more complex model to capture the information hidden in residuals from the ARIMA model. The hybrid model with LSTM is able to model publicly available and other information, which we have no reason to believe will be restricted, coming from a purely ARMA or ARIMA model. In order to make the stock price stationary in time series analysis, the ARIMA models are using the parameter $d = 1$ (or, equivalently, first-order differencing). As the experimental results show, MML87 outperforms the other information-theoretic criteria AIC, BIC, and HQ in terms of lower root mean squared error for out-of-sample forecasting. Figure 5 demonstrates the log prices for stock prices selected in this experiment.

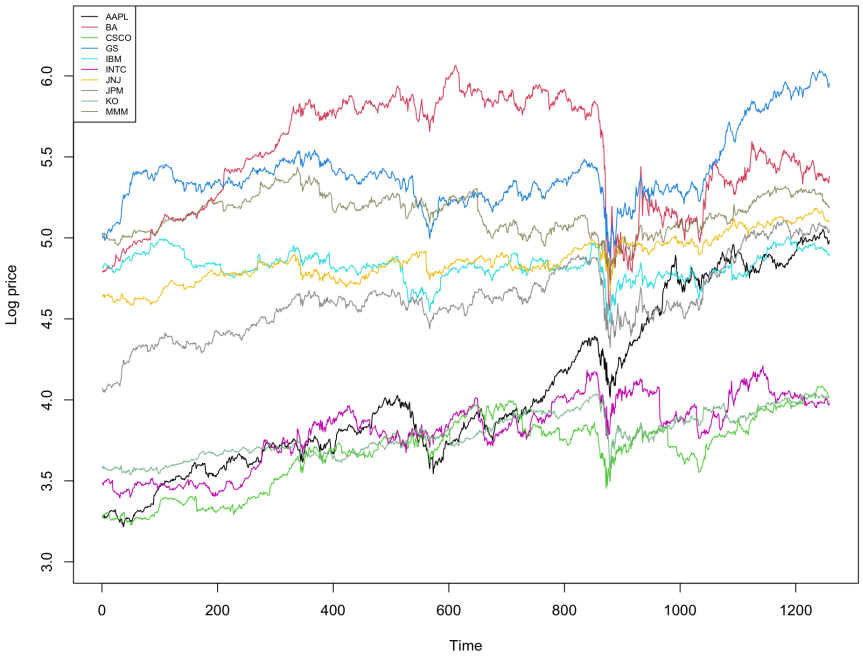

Figure 5. Log prices for ten selected stocks.

The hybrid model tends to outperform for a large forecast window size rather than the small forecast window size because a large lookahead in forecasting has higher uncertainty. For much—or perhaps even most—of the financial industry, there is high volatility in long forecasts. The notion of semi-strong market efficiency suggests that the stock price fully and fairly reflects publicly available information in the time horizontal in the forecast window and also reflects all past information (although by no means all authors agree with this in its entirety [53], partly due to principles of Solomonoff [37] and Wallace [7]). Thus, it is more likely that a complex model will at least be able to provide accurate results in predictions for a T greater than 100. Table 5 shows MML models have lower the RMSE in most cases for different forecast window sizes in financial data.

Table 5. RMSE for forecast window sizes T = 3, 5, 10, 30, 50, 70, 100, 130, 150, and 200.

	Average of RMSE (& Standard Deviation)							
	ARIMA				**ARIMA-LSTM**			
	AIC	**BIC**	**HQ**	**MML87**	**AIC**	**BIC**	**HQ**	**MML87**
T = 3	2.987 (3.446)	3.027 (3.555)	2.914 (3.567)	3.075 (3.572)	4.414 (4.75)	4.302 (4.608)	4.375 (4.757)	4.289 (4.616)
T = 5	4.024 (5.091)	4.077 (5.228)	4.126 (5.218)	4.163 (5.086)	4.024 (5.45)	3.966 (5.42)	4.081 (5.739)	3.907 (5.449)
T = 10	4.748 (4.707)	4.747 (4.858)	4.712 (4.815)	4.868 (4.347)	5.359 (5.429)	5.261 (5.268)	5.272 (5.443)	5.249 (5.262)
T = 30	5.872 (6.797)	5.867 (6.6)	5.91 (6.576)	5.994 (5.662)	5.754 (4.822)	5.628 (4.687)	5.726 (4.776)	5.643 (4.677)
T = 50	7.834 (7.511)	7.609 (7.298)	7.726 (7.269)	6.659 (6.966)	7.328 (6.787)	7.411 (6.879)	7.405 (6.789)	7.384 (6.898)
T = 70	9.991 (9.491)	9.909 (9.316)	10.024 (9.173)	9.645 (7.99)	10.393 (8.048)	10.221 (7.789)	10.42 (8.061)	10.085 (7.612)
T = 100	14.465 (17.187)	13.991 (15.428)	14.197 (13.637)	9.866 (10.854)	9.304 (9.256)	9.087 (9.35)	9.235 (9.396)	9.253 (9.486)
T = 130	14.482 (9.714)	14.301 (10.571)	17.672 (13.139)	13.551 (10.238)	13.768 (10.598)	13.811 (11.124)	13.9 (11.516)	14.581 (10.972)
T = 150	22.985 (28.173)	22.985 (28.077)	23.021 (28.071)	18.045 (17.856)	17.778 (16.771)	17.526 (16.582)	17.98 (16.734)	17.461 (15.931)
T = 200	31.144 (37.567)	30.502 (38.314)	30.712 (38.322)	30.286 (32.564)	26.831 (31.63)	26.424 (31.547)	26.662 (31.645)	26.507 (31.59)

Table 6 provides the average of RMSE for the selected stocks in different sizes, *T*, of the forecast window (shown in different columns) and numbers of LSTM time steps (shown in different rows). The LSTM models are trained by scalers in the range of 0 to 1, and the LSTM model performs worse in the case without scaling, which indicates that the neural network LSTM is scale insensitive and that combining the traditional ARMA time series model makes the neural network more scale-sensitive [54]. The results from Table 6 suggest that the LSTM model alone (unenhanced by ARMA and ARIMA) is not particularly able to capture the time series pattern for the stock price. The figures of the average RMSE are significantly higher than traditional ARMA and ARMA-LSTM models. Figure 6 shows the comparison between the ARIMA model and the hybrid ARIMA-LSTM model in this experiment.

Table 6. LSTM with different time steps for financial data in varying forecast windows.

No. Steps	**T = 3**	**T = 5**	**T = 30**	**T = 10**	**T = 50**	**T = 70**	**T = 100**	**T = 130**	**T = 150**	**T = 200**
1	8.5789	10.1965	56.7817	104.3681	123.4805	119.2673	151.1338	107.2951	114.8106	73.2335
3	5.7604	3.5166	3.6097	13.325	10.6368	31.9361	33.4419	26.0112	31.5578	26.6354
5	4.0695	3.0575	8.5064	11.9009	15.5075	17.3077	19.0942	48.0012	30.0622	36.6099
7	3.9708	6.4145	10.6368	6.8547	13.2163	16.5474	19.0724	32.7076	20.5954	44.1875
10	5.3985	6.4576	5.9597	13.8295	16.0972	20.6271	12.8859	28.2251	28.2803	25.5409

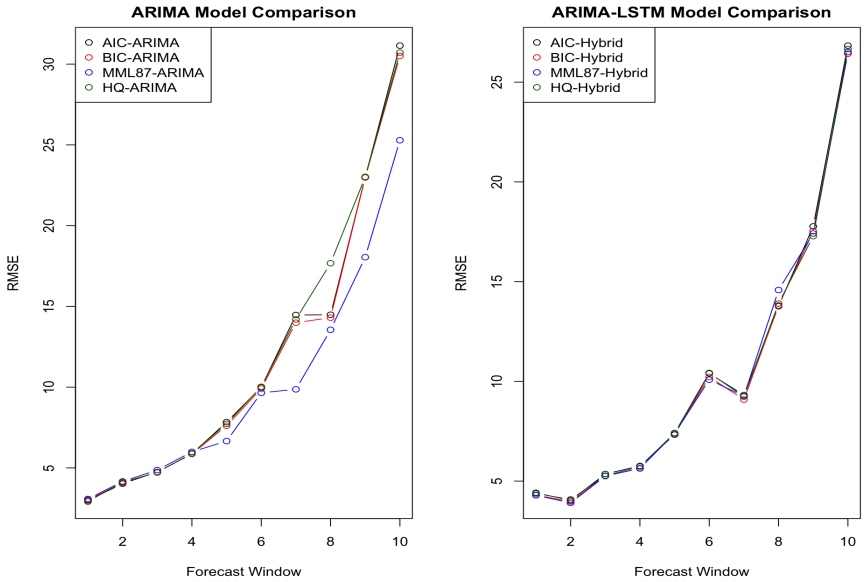

Figure 6. Comparison in different forecast windows.

6.3. PM2.5 Pollution Data

In this section, we use environmental data of PM2.5 pollution levels in the city of Beijing, China, with ten sensors located in different areas. The data are hourly PM2.5 levels in 53 days in 2013.

We are using the same data length and information-theoretic methods from Section 6.2 in order to demonstrate the performance of MML compared to rival methods. Table 7 shows the comparison between MML, AIC, BIC, and HQ. The hourly PM2.5 data have a seasonality; the level of PM2.5 reaches its highest near midday and decreases to its lowest near midnight. The results suggest that MML is a good model selection technique in this case.

Table 7. RMSE for forecast window sizes T = 3, 5, 10, 30, 50, 70, 100, 130, 150, and 200.

	Average of RMSE & Standard Deviation							
	ARIMA				**ARIMA-LSTM**			
	AIC	**BIC**	**HQ**	**MML87**	**AIC**	**BIC**	**HQ**	**MML87**
T = 3	26.805 (7.496)	26.689 (7.532)	26.569 (7.545)	23.104 (7.843)	25.768 (7.833)	23.066 (8.693)	24.791 (7.883)	22.965 (6.711)
T = 5	28.036 (6.986)	27.538 (6.805)	27.479 (7.186)	23.478 (8.426)	24.636 (8.518)	22.309 (7.596)	24.113 (8.584)	21.666 (7.424)
T = 10	30.633 (12.679)	31.502 (12.283)	31.585 (12.518)	30.074 (14.917)	26.970 (10.502)	27.566 (13.487)	27.924 (10.107)	25.102 (9.855)
T = 30	40.730 (14.001)	40.788 (13.372)	40.157 (14.195)	37.989 (19.180)	31.022 (11.409)	29.382 (12.196)	31.689 (14.124)	28.572 (12.229)
T = 50	39.097 (4.238)	38.662 (4.660)	39.007 (4.232)	42.986 (6.062)	35.639 (5.599)	33.335 (5.184)	36.036 (6.339)	40.568 (9.24)
T = 70	34.004 (4.223)	33.551 (4.105)	34.773 (3.514)	32.030 (9.404)	48.942 (12.377)	45.305 (10.567)	49.723 (12.068)	42.987 (8.705)
T = 100	32.002 (2.434)	32.444 (2.425)	31.170 (2.865)	37.925 (4.444)	56.024 (13.199)	51.543 (12.714)	59.705 (13.435)	49.513 (11.45)
T = 130	44.023 (2.583)	44.162 (2.576)	43.635 (2.836)	43.802 (1.853)	36.183 (7.184)	33.716 (4.591)	39.401 (8.488)	46.496 (7.168)
T = 150	44.463 (1.612)	44.736 (1.862)	43.928 (1.773)	41.150 (5.221)	32.574 (6.211)	31.225 (4.301)	30.923 (6.598)	33.584 (7.679)
T = 200	42.150 (2.620)	42.372 (2.522)	41.863 (2.787)	43.75 (4.07)	46.711 (13.86)	43.721 (11.349)	53.393 (12.458)	46.363 (12.472)

Table 8 shows the LSTM model alone in the PM2.5 data, and the results suggest that the LSTM model (on its own, unenhanced by ARMA and ARIMA) outperforms in the smaller-sized forecast windows, such as $T = 3, 5$, and 10. The RMSEs in larger window sizes ($T \geq 50$) are much larger for the LSTM than for the ARMA model and hybrid ARMA-LSTM.

Table 8. LSTM for PM2.5 Beijing data in different time steps and forecast windows.

No. Steps	T = 3	T = 5	T = 10	T = 30	T = 50	T = 70	T = 100	T = 130	T = 150	T = 200
1	3.0976	5.7806	16.5048	47.8431	53.7436	67.2412	81.7044	92.6897	73.4192	71.5536
3	4.4983	8.1565	17.0462	34.0492	36.3896	47.2558	64.68533	90.7986	78.9972	78.5648
5	4.7719	9.2208	18.7955	33.6065	50.4786	56.7465	59.3666	75.4321	102.0098	88.1695
7	5.9126	9.8355	15.3696	25.8551	38.6874	53.06845	50.962	87.998	92.101	101.2337
10	8.4289	11.4749	11.4479	38.3303	44.5138	65.6299	70.6415	74.6879	90.1211	84.0196

7. Conclusions

We have investigated time series modeling in the Minimum Message Length framework using Wallace and Freeman's (1987) approximation [9]. The hybrid ARMA-LSTM model has been compared with the traditional ARMA (Autoregressive Moving Average) time series model based on the information-theoretic approaches: AIC, BIC, HQ and MML87. We performed experiments on simulated data and also on two real-world datasets (financial and environmental data). We conducted the experiments based on hybrid ARMA-LSTM (with LSTM) and ARMA without LSTM (long short-term memory). This could be broadly thought of as constituting two experiments each on three datasets or with six experiments. For each of the six experiments, the results show that MML87 outperforms the other information-theoretic criteria. The hybrid ARMA-LSTM model performs better than the traditional ARMA model, and the MML hybrid ARMA-LSTM model performed best out of everything considered. It is worth noting that the LSTM model alone with unscaled data performed worse than everything else considered. In summary, MML87 is able to select the lower forecasting errors better than the AIC, BIC, and HQ, as the experimental results show.

Author Contributions: Conceptualization: Z.F.; methodology: Z.F. and D.L.D.; computation: Z.F. and D.R.; validation: Z.F., D.L.D. and S.P.; investigation: Z.F., S.P. and D.L.D.; writing and preparation: Z.F. and D.L.D.; writing and review: Z.F., D.L.D. and S.P.; supervision, D.L.D., S.P. and D.R. All authors have read and agreed to the published version of the manuscript and have endeavored to make the work as error-free as possible.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors thank three anonymous referees for their valuable comments and useful suggestions to improve the quality of this version of the paper. The (first two) authors would further like to thank the Department of Data Science and Artificial Intelligence, Faculty of IT, Monash University, for their support.

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

Appendix A

Table A1. Simulated data with $N = 100$ and $T = 3$ from Section 6.1.

Average of RMSE (and Standard Deviation)								
Order of Stationary ARMA	**ARMA**				**ARMA-LSTM**			
	AIC	**BIC**	**HQ**	**MML87**	**AIC**	**BIC**	**HQ**	**MML87**
p_1, q_1	0.982 (0.646)	1.108 (0.529)	1.112 (0.499)	1.033 (0.469)	1.204 (0.308)	1.217 (0.502)	1.215 (0.511)	1.234 (0.7)
p_1, q_2	1.133 (0.592)	1.053 (0.669)	1.172 (0.601)	1.166 (0.635)	1.301 (0.922)	1.289 (0.95)	1.366 (0.862)	1.384 (0.838)
p_2, q_1	1.027 (0.423)	1.024 (0.421)	1.029 (0.445)	1.023 (0.418)	1.025 (0.408)	1.012 (0.376)	1.021 (0.411)	1.005 (0.48)
p_2, q_2	1.333 (0.793)	1.278 (0.841)	1.286 (0.879)	1.271 (0.848)	1.241 (0.745)	1.182 (0.711)	1.211 (0.735)	1.194 (0.674)
p_3, q_1	0.955 (0.377)	0.956 (0.377)	0.951 (0.375)	0.944 (0.37)	0.965 (0.341)	0.975 (0.35)	0.971 (0.351)	0.986 (0.426)
p_3, q_2	1.293 (0.331)	1.241 (0.296)	1.245 (0.307)	1.238 (0.296)	1.114 (0.284)	1.211 (0.266)	1.172 (0.269)	1.105 (0.259)
p_4, q_1	0.901 (0.483)	0.916 (0.448)	0.913 (0.451)	0.871 (0.398)	0.948 (0.397)	0.944 (0.41)	0.945 (0.413)	0.932 (0.442)
p_4, q_2	1.207 (0.539)	1.226 (0.515)	1.224 (0.531)	1.206 (0.513)	1.252 (0.777)	1.261 (0.778)	1.257 (0.792)	1.251 (0.772)
p_5, q_1	1.006 (0.54)	0.907 (0.626)	0.911 (0.642)	0.903 (0.578)	1.122 (0.538)	1.117 (0.553)	1.112 (0.564)	1.018 (0.467)
p_5, q_2	1.026 (0.583)	1.052 (0.553)	1.054 (0.587)	1.061 (0.559)	1.042 (0.559)	1.021 (0.592)	1.027 (0.566)	1.046 (0.53)

Table A2. Simulated data with $N = 100$ and $T = 10$ from Section 6.1.

Average of RMSE (and Standard Deviation)								
Order of Stationary ARMA	**ARMA**				**ARMA-LSTM**			
	AIC	**BIC**	**HQ**	**MML87**	**AIC**	**BIC**	**HQ**	**MML87**
p_1, q_1	1.234 (0.178)	1.208 (0.165)	1.206 (0.167)	1.221 (0.293)	1.201 (0.404)	1.132 (0.184)	1.135 (0.188)	1.138 (0.317)
p_1, q_2	1.571 (0.375)	1.553 (0.386)	1.556 (0.391)	1.398 (0.304)	1.555 (1.109)	1.549 (1.125)	1.551 (1.123)	1.494 (0.834)
p_2, q_1	1.025 (0.194)	1.041 (0.203)	1.044 (0.216)	1.043 (0.193)	1.013 (0.174)	1.02 (0.182)	1.025 (0.174)	1.037 (0.265)
p_2, q_2	1.353 (0.438)	1.327 (0.373)	1.322 (0.391)	1.325 (0.368)	1.274 (0.213)	1.257 (0.206)	1.268 (0.211)	1.255 (0.205)
p_3, q_1	0.947 (0.194)	0.895 (0.129)	0.918 (0.157)	0.901 (0.134)	1.018 (0.135)	0.946 (0.116)	0.966 (0.151)	0.989 (0.154)
p_3, q_2	0.978 (0.266)	1.06 (0.239)	1.065 (0.225)	1.048 (0.226)	1.149 (0.328)	1.137 (0.338)	1.141 (0.352)	1.135 (0.328)
p_4, q_1	1.083 (0.206)	1.059 (0.2)	1.063 (0.218)	1.075 (0.179)	1.081 (0.261)	1.029 (0.179)	1.035 (0.219)	1.061 (0.128)
p_4, q_2	1.121 (0.192)	1.112 (0.17)	1.124 (0.186)	1.104 (0.174)	1.093 (0.212)	1.088 (0.191)	1.095 (0.252)	1.096 (0.181)
p_5, q_1	1.279 (0.322)	1.244 (0.296)	1.264 (0.251)	1.242 (0.29)	1.169 (0.335)	1.167 (0.327)	1.172 (0.335)	1.166 (0.306)
p_5, q_2	0.903 (0.078)	0.867 (0.067)	0.882 (0.059)	0.877 (0.074)	1.053 (0.231)	1.033 (0.192)	1.046 (0.188)	0.972 (0.126)

Table A3. Simulated data with $N = 100$ and $T = 30$ from Section 6.1.

	Average of RMSE (and Standard Deviation)							
Order of Stationary ARMA	**ARMA**				**ARMA-LSTM**			
	AIC	**BIC**	**HQ**	**MML87**	**AIC**	**BIC**	**HQ**	**MML87**
p_1, q_1	1.263 (0.167)	1.252 (0.156)	1.253 (0.173)	1.256 (0.159)	1.217 (0.295)	1.118 (0.119)	1.125 (0.133)	1.192 (0.247)
p_1, q_2	2.641 (0.905)	2.554 (0.838)	2.631 (0.972)	2.694 (0.961)	1.771 (1.135)	1.848 (0.739)	1.822 (0.959)	1.803 (1.373)
p_2, q_1	1.221 (0.139)	1.186 (0.096)	1.199 (0.121)	1.184 (0.102)	1.102 (0.084)	1.088 (0.083)	1.094 (0.089)	1.124 (0.101)
p_2, q_2	1.044 (0.091)	1.145 (0.108)	1.093 (0.117)	1.041 (0.088)	1.138 (0.255)	1.153 (0.211)	1.148 (0.262)	1.136 (0.256)
p_3, q_1	1.086 (0.181)	1.066 (0.19)	1.073 (0.195)	1.061 (0.182)	1.038 (0.172)	1.036 (0.171)	1.036 (0.188)	1.035 (0.145)
p_3, q_2	1.112 (0.295)	1.096 (0.309)	1.139 (0.331)	1.101 (0.306)	1.202 (0.38)	1.153 (0.328)	1.166 (0.369)	1.099 (0.264)
p_4, q_1	1.053 (0.22)	1.038 (0.189)	1.044 (0.167)	1.035 (0.185)	1.058 (0.14)	1.051 (0.124)	1.055 (0.139)	1.063 (0.152)
p_4, q_2	1.263 (0.2)	1.247 (0.194)	1.251 (0.229)	1.238 (0.21)	1.204 (0.133)	1.191 (0.114)	1.211 (0.138)	1.183 (0.152)
p_5, q_1	1.613 (0.27)	1.679 (0.301)	1.669 (0.343)	1.599 (0.342)	1.541 (0.884)	1.531 (0.609)	1.539 (0.915)	1.521 (0.848)
p_5, q_2	1.092 (0.132)	1.047 (0.234)	1.052 (0.337)	1.047 (0.114)	1.074 (0.144)	1.041 (0.117)	1.045 (0.196)	1.041 (0.115)

Table A4. Simulated data with $N = 100$ and $T = 50$ from Section 6.1.

	Average of RMSE (and Standard Deviation)							
Order of Stationary ARMA	**ARMA**				**ARMA-LSTM**			
	AIC	**BIC**	**HQ**	**MML87**	**AIC**	**BIC**	**HQ**	**MML87**
p_1, q_1	1.189 (0.217)	1.191 (0.228)	1.193 (0.221)	1.182 (0.222)	1.164 (0.304)	1.091 (0.212)	1.155 (0.292)	1.173 (0.241)
p_1, q_2	2.307 (0.458)	2.308 (0.457)	2.305 (0.464)	2.298 (0.466)	1.862 (1.169)	1.868 (1.16)	1.889 (1.171)	1.852 (1.073)
p_2, q_1	1.113 (0.087)	1.092 (0.103)	1.095 (0.107)	1.094 (0.104)	1.058 (0.073)	1.045 (0.096)	1.172 (0.225)	1.059 (0.139)
p_2, q_2	1.191 (0.096)	1.189 (0.103)	1.192 (0.107)	1.191 (0.1)	1.176 (0.24)	1.178 (0.259)	1.183 (0.285)	1.201 (0.289)
p_3, q_1	1.094 (0.159)	1.093 (0.157)	1.095 (0.156)	1.097 (0.155)	1.101 (0.192)	1.061 (0.144)	1.065 (0.177)	1.093 (0.115)
p_3, q_2	1.127 (0.06)	1.123 (0.055)	1.129 (0.057)	1.125 (0.058)	1.121 (0.134)	1.129 (0.155)	1.126 (0.143)	1.132 (0.153)
p_4, q_1	1.188 (0.182)	1.189 (0.188)	1.185 (0.187)	1.192 (0.186)	1.136 (0.137)	1.095 (0.181)	1.099 (0.173)	1.139 (0.113)
p_4, q_2	1.232 (0.165)	1.221 (0.133)	1.222 (0.137)	1.212 (0.134)	1.268 (0.457)	1.19 (0.269)	1.197 (0.234)	1.203 (0.319)
p_5, q_1	1.593 (0.304)	1.521 (0.199)	1.533 (0.216)	1.528 (0.209)	1.331 (0.428)	1.275 (0.234)	1.277 (0.214)	1.338 (0.383)
p_5, q_2	1.051 (0.083)	1.033 (0.064)	1.029 (0.096)	1.032 (0.063)	1.035 (0.055)	1.021 (0.067)	1.029 (0.077)	1.023 (0.069)

Table A5. Simulated data with $N = 50$ and $T = 10$ from Section 6.1.

	Average of RMSE (and Standard Deviation)							
Order of Stationary ARMA	ARMA				ARMA-LSTM			
	AIC	BIC	HQ	MML87	AIC	BIC	HQ	MML87
p_1, q_1	1.068 (0.147)	1.071 (0.118)	1.073 (0.125)	1.067 (0.115)	1.198 (0.222)	1.122 (0.305)	1.137 (0.336)	1.175 (0.259)
p_1, q_2	1.994 (0.655)	1.994 (0.655)	2.056 (0.692)	2.04 (0.705)	1.93 (1.553)	1.932 (1.563)	1.926 (1.572)	1.921 (1.566)
p_2, q_1	1.242 (0.213)	1.242 (0.213)	1.274 (0.252)	1.235 (0.17)	1.106 (0.193)	1.116 (0.196)	1.119 (0.189)	1.154 (0.278)
p_2, q_2	1.185 (0.355)	1.183 (0.359)	1.196 (0.361)	1.232 (0.476)	1.163 (0.386)	1.194 (0.499)	1.172 (0.534)	1.254 (0.601)
p_3, q_1	1.348 (0.557)	1.254 (0.604)	1.269 (0.661)	1.304 (0.575)	1.257 (0.499)	1.139 (0.605)	1.212 (0.657)	1.256 (0.449)
p_3, q_2	1.283 (0.234)	1.283 (0.234)	1.281 (0.265)	1.291 (0.233)	1.198 (0.27)	1.198 (0.27)	1.211 (0.298)	1.215 (0.285)
p_4, q_1	1.263 (0.461)	1.251 (0.469)	1.288 (0.477)	1.044 (0.172)	1.091 (0.264)	1.079 (0.27)	1.096 (0.288)	1.129 (0.243)
p_4, q_2	0.987 (0.132)	0.987 (0.132)	0.989 (0.132)	0.989 (0.137)	1.007 (0.137)	1.017 (0.126)	1.022 (0.139)	0.999 (0.138)
p_5, q_1	1.533 (0.457)	1.426 (0.535)	1.454 (0.561)	1.464 (0.509)	1.227 (0.442)	1.178 (0.445)	1.192 (0.496)	1.254 (0.434)
p_5, q_2	1.101 (0.153)	1.098 (0.151)	1.111 (0.186)	1.137 (0.185)	1.061 (0.168)	1.068 (0.175)	1.072 (0.183)	1.08 (0.117)

Table A6. Simulated data with $N = 200$ and $T = 10$ from Section 6.1.

	Average of RMSE (and Standard Deviation)							
Order of Stationary ARMA	ARMA				ARMA-LSTM			
	AIC	BIC	HQ	MML87	AIC	BIC	HQ	MML87
p_1, q_1	1.244 (0.365)	1.277 (0.42)	1.286 (0.417)	1.248 (0.404)	1.153 (0.381)	1.13 (0.376)	1.146 (0.392)	1.151 (0.353)
p_1, q_2	1.359 (0.445)	1.359 (0.445)	1.366 (0.462)	1.359 (0.445)	1.474 (0.813)	1.491 (0.882)	1.477 (0.893)	1.474 (0.813)
p_2, q_1	0.927 (0.183)	0.915 (0.172)	0.916 (0.185)	0.92 (0.182)	0.939 (0.126)	0.955 (0.15)	0.969 (0.163)	0.933 (0.128)
p_2, q_2	1.184 (0.41)	1.191 (0.398)	1.193 (0.366)	1.189 (0.402)	1.134 (0.368)	1.114 (0.393)	1.116 (0.407)	1.106 (0.37)
p_3, q_1	1.137 (0.347)	1.136 (0.347)	1.129 (0.351)	1.117 (0.355)	1.082 (0.314)	1.082 (0.316)	1.088 (0.361)	1.085 (0.325)
p_3, q_2	0.915 (0.198)	1.038 (0.08)	1.011 (0.081)	0.991 (0.093)	1.088 (0.184)	1.083 (0.172)	1.075 (0.199)	1.054 (0.161)
p_4, q_1	1.199 (0.558)	1.166 (0.557)	1.174 (0.531)	1.19 (0.562)	1.086 (0.591)	1.109 (0.507)	1.115 (0.691)	1.107 (0.732)
p_4, q_2	1.108 (0.196)	1.101 (0.191)	1.132 (0.615)	1.129 (0.24)	1.184 (0.358)	1.186 (0.359)	1.192 (0.379)	1.184 (0.36)
p_5, q_1	1.581 (0.481)	1.584 (0.475)	1.584 (0.422)	1.586 (0.48)	1.383 (0.83)	1.391 (0.802)	1.396 (0.811)	1.382 (0.832)
p_5, q_2	1.123 (0.263)	1.101 (0.174)	1.107 (0.155)	1.101 (0.174)	1.063 (0.234)	1.069 (0.133)	1.065 (0.129)	1.063 (0.128)

Table A7. Simulated data with $N = 300$ and $T = 10$ from Section 6.1.

	Average of RMSE (and Standard Deviation)							
Order of Stationary ARMA	ARMA				ARMA-LSTM			
	AIC	BIC	HQ	MML87	AIC	BIC	HQ	MML87
p_1, q_1	1.024 (0.312)	1.028 (0.332)	1.029 (0.321)	1.031 (0.322)	1.033 (0.316)	1.02 (0.27)	1.021 (0.291)	1.024 (0.32)
p_1, q_2	2.008 (1.123)	1.995 (1.024)	1.996 (1.031)	1.988 (1.028)	1.709 (0.918)	1.72 (0.896)	1.725 (0.812)	1.72 (0.854)
p_2, q_1	1.022 (0.144)	1.025 (0.138)	1.027 (0.132)	1.016 (0.133)	1.011 (0.125)	1.012 (0.121)	1.017 (0.126)	1.014 (0.297)
p_2, q_2	1.172 (0.398)	1.168 (0.383)	1.171 (0.326)	1.166 (0.384)	1.164 (0.422)	1.177 (0.443)	1.172 (0.461)	1.17 (0.413)
p_3, q_1	0.886 (0.198)	0.868 (0.205)	0.882 (0.217)	0.865 (0.215)	0.964 (0.261)	0.932 (0.183)	0.952 (0.191)	0.914 (0.188)
p_3, q_2	1.07 (0.408)	1.068 (0.412)	1.065 (0.407)	1.059 (0.401)	1.096 (0.284)	1.095 (0.289)	1.097 (0.277)	1.092 (0.284)
p_4, q_1	1.215 (0.445)	1.191 (0.468)	1.194 (0.462)	1.184 (0.464)	1.22 (0.621)	1.091 (0.42)	1.124 (0.468)	1.166 (0.453)
p_4, q_2	1.191 (0.338)	1.167 (0.308)	1.172 (0.311)	1.162 (0.278)	1.182 (0.427)	1.188 (0.473)	1.184 (0.113)	1.184 (0.433)
p_5, q_1	1.169 (0.225)	1.159 (0.216)	1.161 (0.232)	1.152 (0.216)	0.997 (0.131)	1.071 (0.213)	1.011 (0.159)	1.01 (0.146)
p_5, q_2	0.874 (0.25)	0.846 (0.249)	0.852 (0.297)	0.844 (0.247)	0.936 (0.213)	0.939 (0.197)	0.935 (0.199)	0.938 (0.196)

Table A8. Simulated data with $N = 500$ & $T = 10$ from Section 6.1.

	Average of RMSE (and Standard Deviation)							
Order of Stationary ARMA	ARMA				ARMA-LSTM			
	AIC	BIC	HQ	MML87	AIC	BIC	HQ	MML87
p_1, q_1	0.988 (0.229)	0.966 (0.233)	0.967 (0.252)	0.968 (0.232)	1.016 (0.178)	1.012 (0.182)	1.017 (0.169)	1.014 (0.179)
p_1, q_2	1.546 (0.728)	1.549 (0.713)	1.552 (0.736)	1.562 (0.703)	1.841 (0.915)	1.838 (0.875)	1.838 (0.876)	1.838 (0.877)
p_2, q_1	1.002 (0.37)	1.017 (0.349)	1.017 (0.355)	1.016 (0.351)	1.008 (0.329)	1.008 (0.325)	1.008 (0.334)	1.05 (0.349)
p_2, q_2	1.156 (0.188)	1.165 (0.176)	1.163 (0.182)	1.165 (0.176)	1.167 (0.337)	1.156 (0.355)	1.159 (0.363)	1.156 (0.355)
p_3, q_1	1.091 (0.175)	1.093 (0.18)	1.099 (0.175)	1.09 (0.176)	1.064 (0.225)	1.06 (0.157)	1.066 (0.173)	1.058 (0.22)
p_3, q_2	1.23 (0.372)	1.235 (0.364)	1.235 (0.364)	1.235 (0.364)	1.197 (0.365)	1.209 (0.393)	1.21 (0.378)	1.209 (0.393)
p_4, q_1	1.041 (0.272)	1.07 (0.25)	1.069 (0.261)	1.063 (0.257)	1.135 (0.342)	1.053 (0.239)	1.096 (0.298)	1.139 (0.343)
p_4, q_2	1.253 (0.265)	1.256 (0.265)	1.261 (0.273)	1.255 (0.266)	1.134 (0.218)	1.136 (0.214)	1.131 (0.243)	1.126 (0.235)
p_5, q_1	1.559 (0.363)	1.551 (0.385)	1.55 (0.374)	1.541 (0.365)	1.159 (0.331)	1.199 (0.421)	1.196 (0.411)	1.161 (0.292)
p_5, q_2	1.073 (0.179)	1.068 (0.188)	1.071 (0.192)	1.068 (0.188)	1.083 (0.136)	1.062 (0.167)	1.067 (0.166)	1.062 (0.167)

References

1. Siami-Namini, S.; Tavakoli, N.; Namin, A.S. A comparison of ARIMA and LSTM in forecasting time series. In Proceedings of the 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA), Orlando, FL, USA, 17–20 December 2018; pp. 1394–1401.
2. Wallace, C.S.; Boulton, D.M. An information measure for classification. *Comput. J.* **1968**, *11*, 185–194.
3. Akaike, H. A new look at the statistical model identification. *IEEE Trans. Autom. Control* **1974**, *19*, 716–723.
4. Schwarz, G. Estimating the dimension of a model. *Ann. Stat.* **1978**, *6*, 461–464.
5. Hannan, E.J.; Quinn, B.G. The determination of the order of an autoregression. *J. R. Stat. Soc. Ser. B Methodol.* **1979**, *41*, 190–195.
6. Dowe, D.L. Foreword re C. S. Wallace. *Comput. J.* **2008**, *51*, 523–560.
7. Wallace, C.S.; Dowe, D.L. Minimum message length and Kolmogorov complexity. *Comput. J.* **1999**, *42*, 270–283.
8. Wong, C.K.; Makalic, E.; Schmidt, D.F. Minimum message length inference of the Poisson and geometric models using heavy-tailed prior distributions. *J. Math. Psychol.* **2018**, *83*, 1–11.
9. Wallace, C.S.; Freeman, P.R. Estimation and inference by compact coding. *J. R. Stat. Soc. Ser. B Methodol.* **1987**, *49*, 240–252.
10. Fang, Z.; Dowe, D.L.; Peiris, S.; Rosadi, D. Minimum Message Length Autoregressive Moving Average Model Order Selection. *arXiv* **2021**, arXiv:2110.03250.
11. Schmidt, D.F. Minimum Message Length Inference of Autoregressive Moving Average Models. Ph.D. Thesis, Faculty of IT, Monash University, Melbourne, Australia, 2008.
12. Fathi, O. *Time Series Forecasting Using a Hybrid ARIMA and LSTM Model*; Velvet Consulting: Paris, France, 2019.
13. Box, G.E.; Jenkins, G.M.; Reinsel, G.C. *Time Series Analysis Prediction and Control*; John Wiley and Sons: Hoboken, NJ, USA, 1976.
14. De Gooijer, J.G.; Hyndman, R.J. 25 years of time series forecasting. *Int. J. Forecast.* **2006**, *22*, 443–473.
15. Box, G.E.; Jenkins, G.M.; Reinsel, G.C.; Ljung, G.M. *Time Series Analysis: Forecasting and Control*; John Wiley & Sons: New York, NY, USA, 2015.
16. Wang, J.Q.; Du, Y.; Wang, J. LSTM based long-term energy consumption prediction with periodicity. *Energy* **2020**, *197*, 117197.
17. Chen, K.; Zhou, Y.; Dai, F. A LSTM-based method for stock returns prediction: A case study of China stock market. In Proceedings of the 2015 IEEE International Conference on Big Data (Big Data), Santa Clara, CA, USA, 29 October–1 November, 2015; pp. 2823–2824.
18. Sak, M.; Dowe, D.L.; Ray, S. Minimum message length moving average time series data mining. In Proceedings of the In 2005 ICSC Congress on Computational Intelligence Methods and Applications, Istanbul, Turkey, 15–17 December, 2005; 6p.
19. Wallace, C.S. *Statistical and Inductive Inference by Minimum Message Length*; Springer: New York, NY, USA, 2005; pp. 93–100.
20. Aho, K.; Derryberry, D.; Peterson, T. Model selection for ecologists: The worldviews of AIC and BIC. *Ecology* **2014**, *95*, 631–636.
21. Grasa, A.A. *Econometric Model Selection: A New Approach*; Springer Science & Business Media: New York, NY, USA, 2013; Volume 16.
22. Hernandez-Matamoros, A.; Fujita, H.; Hayashi, T.; Perez-Meana, H. Forecasting of COVID19 per regions using ARIMA models and polynomial functions. *Appl. Soft Comput.* **2020**, *96*, 106610.
23. Dowe, D.L. MML, hybrid Bayesian network graphical models, statistical consistency, invariance and uniqueness. In *Handbook of the Philosophy of Science*; Volume 7: Philosophy of Statistics; Elsevier: New York, NY, USA, 2011; pp. 901–982.
24. Baxter, R.A.; Dowe, D.L. Model selection in linear regression using the MML criterion. In Proceedings of the Data Compression Conference, IEEE, Institute of Electrical and Electronics Engineers, Snowbird, UT, USA, 29–31 March 1994.
25. Fitzgibbon, L.J.; Dowe, D.L.; Vahid, F. Minimum message length autoregressive model order selection. In Proceedings of the International Conference on Intelligent Sensing and Information Processing, Chennai, India, 4–7 January 2004; pp. 439–444.
26. Schmidt, D.F. Minimum message length order selection and parameter estimation of moving average models. In *Algorithmic Probability and Friends*; Bayesian Prediction and Artificial Intelligence; Springer: Berlin/Heidelberg, Germany, 2013; pp. 327–338.
27. Wallace, C.S.; Dowe, D.L. Intrinsic classification by MML-the Snob program. In Proceedings of the 7th Australian Joint Conference on Artificial Intelligence World Scientific, Armidale, Australia, 1 January 1994; pp. 37–44.
28. Wallace, C.S.; Dowe, D.L. MML clustering of multi-state, Poisson, von Mises circular and Gaussian distributions. *Stat. Comput.* **2000**, *10*, 73–83.
29. Dowe, D.L.; Allison, L.; Dix, T.I.; Hunter, L.; Wallace, C.S.; Edgoose, T. Circular clustering of protein dihedral angles by minimum message length. In *Pacific Symposium on Biocomputing*; World Scientific: Singapore, 1996; pp. 242–255.
30. Oliver, J.J.; Dowe, D.L.; Wallace, C.S. Inferring decision graphs using the minimum message length principle. In Proceedings of the 5th Australian Joint Conference on Artificial Intelligence, Hobart, NSW, Australia, 16–18 November 1992; pp. 361–367.
31. Tan, P.J.; Dowe, D.L. MML inference of decision graphs with multi-way joins and dynamic attributes. In *Australasian Joint Conference on Artificial Intelligence*; Springer: Berlin/Heidelberg, Germany, 2003; pp. 269–281.
32. Comley, J.W.; Dowe, D.L. General Bayesian networks and asymmetric languages. In Proceedings of the 2nd Hawaii International Conference on Statistics and Related Fields, Honolulu, HI, USA, 5–8 June 2003; p. 18.
33. Comley, J.W.; Dowe, D.L. Minimum Message Length and Generalized Bayesian Nets with Asymmetric Languages. *Minimum* **2005**, *265*, 265–294.
34. Saikrishna, V.; Dowe, D.L.; Ray, S. MML learning and inference of hierarchical Probabilistic Finite State Machines. In *Applied Data Analytics: Principles and Applications*; River Publishers: Aalborg, Denmark, 2020; pp. 291–325.
35. Dowe, D.L.; Zaidi, N.A. Database normalization as a by-product of minimum message length inference. In *Australasian Joint Conference on Artificial Intelligence*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 82–91.
36. Li, M.; Vitányi, P. *An Introduction to Kolmogorov Complexity and Its Applications*; Springer: New York, NY, USA, 2008; Volume 3.

37. Solomonoff, R.J. Complexity-based induction systems: comparisons and convergence theorems. *IEEE Trans. Inf. Theory* **1978**, *24*, 422–432.
38. Dowe, D.L. Introduction to Ray Solomonoff 85th memorial conference. In *Algorithmic Probability and Friends. Bayesian Prediction and Artificial Intelligence*; LNAI 7070; Springer: Berlin/Heidelberg, Germany, 2013; pp. 1–36.
39. Makalic, E.; Allison, L.; Dowe, D.L. MML inference of single-layer neural networks. In Proceedings of the 3rd IASTED International Conferences Artificial Intelligence and Applications, Benalmadena, Spain, 8–10 September 2003; pp. 636–642.
40. Fitzgibbon, L.J.; Dowe, D.L.; Allison, L. Univariate polynomial inference by Monte Carlo message length approximation. In *International Conference Machine Learning*; ICML: Sydney, Australia, 2002; pp. 147–154.
41. Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; Philip, S.Y. A comprehensive survey on graph neural networks. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *32*, 4–24.
42. Chong, E.; Han, C.; Park, F.C. Deep learning networks for stock market analysis and prediction: Methodology, data representations, and case studies. *Expert Syst. Appl.* **2017**, *83*, 187–205.
43. Qiu, X.; Zhang, L.; Ren, Y.; Suganthan, P.N.; Amaratunga, G. Ensemble deep learning for regression and time series forecasting. In Proceedings of the 2014 IEEE Symposium on Computational Intelligence in Ensemble Learning (CIEL), Orlando, FL, USA, 9–12 December 2014; pp. 1–6.
44. Sezer, O.B.; Gudelek, M.U.; Ozbayoglu, A.M. Financial time series forecasting with deep learning: A systematic literature review: 2005–2019. *Appl. Soft Comput.* **2020**, *90*, 106181.
45. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780.
46. Li, J.; Bu, H.; Wu, J. Sentiment-aware stock market prediction: A deep learning method. In Proceedings of the 2017 International Conference on Service Systems and Service Management, Dalian, China, 16–18 June 2017; pp. 1–6.
47. Zhang, X.; Tan, Y. Deep stock ranker: A LSTM neural network model for stock selection. In *International Conference on Data Mining and Big Data*; Springer: Cham, Switzerland, 2018; pp. 614–623.
48. Bukhari, A.H.; Raja, M.A.Z.; Sulaiman, M.; Islam, S.; Shoaib, M.; & Kumam, P. Fractional neuro-sequential ARFIMA-LSTM for financial market forecasting. *IEEE Access* **2020**, *8*, 71326–71338.
49. Cheng, T.; Gao, J.; Linton, O. *Nonparametric Predictive Regressions for Stock Return Prediction*; Working Paper; University of Cambridge: Cambridge, UK, 2019.
50. Gao, J. Modelling long-range-dependent Gaussian processes with application in continuous-time financial models. *J. Appl. Probab.* **2004**, *41*, 467–482.
51. Fama, E.F.; French, K.R. *Dividend Yields and Expected Stock Returns*; University of Chicago Press: Chicago, IL, USA, 2021; pp. 568–595.
52. Keim, D.B.; Stambaugh, R.F. Predicting returns in the stock and bond markets. *J. Financ. Econom.* **1986**, *17*, 357–390.
53. Dowe, D.L.; Korb, K.B. Conceptual difficulties with the efficient market hypothesis: Towards a naturalized economics. In Proceedings of the Information, Statistics and Induction in Science, World Scientific, Melbourne, Australia, 20–23 August 1996; pp. 212–223.
54. Lai, G.; Chang, W.C.; Yang, Y.; Liu, H. Modeling long-and short-term temporal patterns with deep neural networks. In Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, New York, NY, USA, 8–12 June 2018; pp. 95–104.