Signal Processing 183 (2021) 108026

Contents lists available at ScienceDirect

Signal Processing

journal homepage: www.elsevier.com/locate/sigpro

AR and ARMA model order selection for time-series modeling with ImageNet classification

Jihye Moon, Md Billal Hossain, Ki H. Chon*

Department of Biomedical Engineering, University of Connecticut, 260 Glenbrook Road, Unit 3247 Storrs, CT 06269-3247, USA

ARTICLE INFO

Article history:
Received 4 May 2020
Revised 11 November 2020
Accepted 2 February 2021
Available online 4 February 2021

Keywords:
ImageNet classification
Convolutional neural network
Time series modeling
AR model identification
ARMA model identification
Deep neural network
Physical system modeling

ABSTRACT

We propose model order selection methods for autoregressive (AR) and autoregressive moving average (ARMA) time-series modeling based on ImageNet classifications with a 2-dimensional convolutional neural network (2-D CNN). We designed two models for two realistic scenarios: (1) a general model which emulates the scenario that validation and test datasets do not necessarily have the same dynamics as the training data, (2) a specific model which emulates the opposite scenario—the validation and test datasets share the dynamics of the training data. The results were compared to those of both Akaike Information criterion (AIC) and Bayesian Information criterion (BIC). Using simulation examples, we trained 2-D CNN-based Inception-v3 and ResNet50-v2 models for either AR or ARMA order selection for each of the two scenarios. The proposed ResNet50-v2 to use both time-frequency and the original time series data outperformed AIC and BIC for all scenarios. For the general model, the average of relative error reduction (ARER) when compared to the BIC method in the clean and three noisy environments was 19.07% (±14.22%) for the AR order for an AR process, and 5.67% (±2.83%) for the ARMA order for an ARMA process. The ARERs significantly improved to 73.92% (±30.95%) and 65.58% (±38.61%) for the AR and ARMA models, respectively, for the specific model scenario.

1. Introduction

In this paper, we propose parametric model order selection methods for autoregressive (AR) and autoregressive moving average (ARMA) time-series models using ImageNet classifications with a 2-dimensional convolutional neural network (2-D CNN). The AR and ARMA models are very well-known statistical methods for the analysis of stochastic processes in many diverse fields such as spectral estimation, time series forecasting and prediction, and biomedical engineering [1–3]. In the physical realm, a variety of natural signals such as speech, electrocardiogram (ECG), and seismic signals are formulated by an underlying AR structure since any time-series signal can be modeled by an AR process in the real world [1,6].

These models enable accurate statistical analysis for a better understanding of a physical system and to predict the next observed values in a time series [8]. One of the most popular methods is to compute the power spectral density based on an AR model [4]. The ARMA model is another popular parametric approach, which uses the input and output signal to model the dynamics of physiological systems via either transfer function analysis or impulse response functions (IRF). For example, measurements of heart rate (HR) and instantaneous lung volume (ILV) fluctuations were used to estimate a linear impulse response function. Similarly, renal blood flow and arterial blood pressure data were used to estimate renal autoregulatory mechanisms [2, 5, 32, 33]. In addition, for short-term prediction and forecasting, autoregressive integrative moving average (ARIMA), one variant of the ARMA model, provides more accurate results when compared to some of the popular machine learning methods such as the multi-layer perceptron, support vector machine, and long short-term memory (LSTM) [3].

To estimate model parameters, selecting the correct model order for either an AR or an ARMA model is of utmost important [6], as the performance of AR and ARMA models is critically affected by the model order selection [7,8]. Model order identification is a crucial step in the process of estimating accurate AR/ARMA parameters [9]. There are many techniques for AR and ARMA order selection, including the well-known Akaike Information Criterion (AIC) [10], Bayesian Information Criterion (BIC) [11], minimum description length (MDL) [12], cumulant-based determination [13], minimum Eigen value criterion (MRV) [14], final prediction error [15], and neural networks-based order selection methods [16–21]. Recently, the genetic algorithm ARMA (GA-ARMA) [6,22] and the minimum of kurtosis methods [23] were devel-

* Corresponding author.
E-mail address: ki.chon@uconn.edu (K.H. Chon).

https://doi.org/10.1016/j.sigpro.2021.108026
0165-1684/

oped for ARMA model order estimation. The GA-ARMA uses a genetic algorithm for ARMA model order selection and it is touted as solving the local minima issue. The kurtosis method uses a minimization of kurtosis criterion to identify the optimal model order.

In most of the aforementioned model order selection methods, the order is estimated by considering all possible combinations of the initially selected ARMA (p, q) or AR (p) parameters. For example, an order range (p) of AR and a pair order range (p, q) of ARMA orders must be defined prior to model order identification. The model order coefficients of the system are estimated based on the assumed values of p and q. Finally, for each of the model order combinations a specific criterion, which usually entails minimizing the loss function, is used to find the optimal model order [22]. Since these approaches require calculation of the coefficients of either AR or ARMA models for every possible combination, the computation time becomes quite expensive [22]. The higher computational time makes it more difficult to apply either AR or ARMA modeling to real-world applications, which often require real-time results. Further complicating AR and ARMA model order estimation is that often data are corrupted with various noise sources which affect the accuracy of the parameter estimation. Therefore, one needs to develop not only the pre-trained order selection models for accurate model order determination but also account for noise contamination.

Recently, the convolutional neural network (CNN), a deep learning approach, has reported good performance for classification tasks in a variety of applications such as visual recognition, object detection, natural language processing, speech recognition, and medical image analysis [24]. The 2-D CNN-based ImageNet architectures such as AlexNet, Inception, and ResNet were introduced and competed in classification of 1,000 labels [25–29]. The ImageNet-based models train informative features using large datasets by determining weight matrices for target labels for each input. For practical applications, the pre-trained matrices can detect correlations between new input data and these large datasets, and predict a correct label for the input data by considering common patterns among them as a supervised learning approach. These processes enable robust applications for the real world. There has not been much literature on using deep learning for model order estimation, however. It has been shown that when multiple features derived from time-frequency analysis [30,31] (e.g., partial autocorrelation, autocorrelation functions) are used as input data to a CNN, better performance was achieved when compared to using a single feature (e.g. time series itself without any preprocessing or data transformation) for model order identification [30,31]. In another study [38], the authors used the original time series data as the input to a CNN for ARMA order identification but their best accuracy was less than 21%.

In this paper, we propose supervised model order selection methods using a 2-D CNN-based ImageNet with transformation of the time series into time-frequency features as well as the original time series as the input data to the network for AR and ARMA time-series modeling. In this study, we designed two models for each objective: (1) a general model for any time series in which both validation and testing datasets' coefficients are entirely different from those of the training data, and (2) a specific model in which both validation and testing datasets' coefficients are subsets of the training data. The specific model was also evaluated with three types of additive Gaussian white noise (20 dB, 10 dB, and 0 dB).

This paper is organized as follows: in Section II, our proposed methods are explained. Section III describes the experimental design to train the model order selection. Section IV and V describe the results and the relevant discussion, respectively. Finally, conclusions are provided in Section VI.

2. Proposed methods

Our method consists of three steps: time series modeling, data preprocessing, and model order selection. Each of these steps is detailed below.

2.1. Time series modeling

The AR (p) and ARMA (p,q) time-series signals are formulated as follows:

$$y(n) = \sum_{i=1}^{p} \varphi(i) y(n-i) + e(n) \tag{1}$$

$$y(n) = \sum_{i=1}^{p} \varphi(i) y(n-i) + \sum_{j=0}^{q} \theta(j) x(n-j) \; + e(n) \tag{2}$$

In Eqs. (1) and (2), p is the order of the autoregressive (AR) filter and q is the order of the moving average (MA) filter. The two parameters, φ(i) and θ(j), represent the coefficients of the AR and MA terms, respectively. The model orders p and q and the coefficients are unknown. $y(n)$ is the output signal of the time series and $x(n)$ is the input signal, where n is the total number of data points. The input $x(n)$, which is the MA portion, is an independent and identically distributed Gaussian. The $e(n)$ term is the residual error. For simulations, we added various noise levels of additive white Gaussian noise (AWGN). The signal-noise-ratio (SNR) was formulated as follows:

$$SNR_{dB} = 10log_{10}\left(\frac{P_y}{P_e}\right) \tag{3}$$

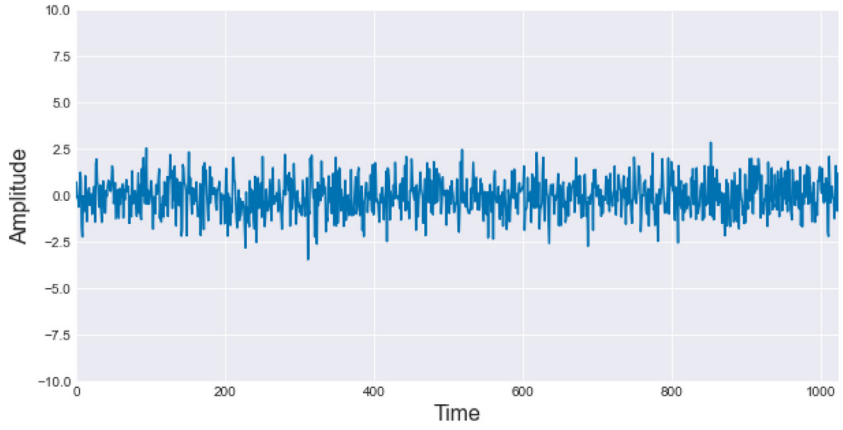
In (3), P_y is the variance of the time-series signal $y(n)$ and P_e is the variance of the AWGN. For simulations, we generated AR and ARMA time series with p varying from 1-9 and q varying in the 0-9 range ($1 \le p \le p_{max}$, $0 \le q \le q_{max}$), with each signal containing 1,024 points. The maximum orders were purposely chosen higher than the true model orders since they are unknown for real-life data. Representative examples of simulated data with an AR and an ARMA model are shown Fig. 1:

2.2. Data preprocessing

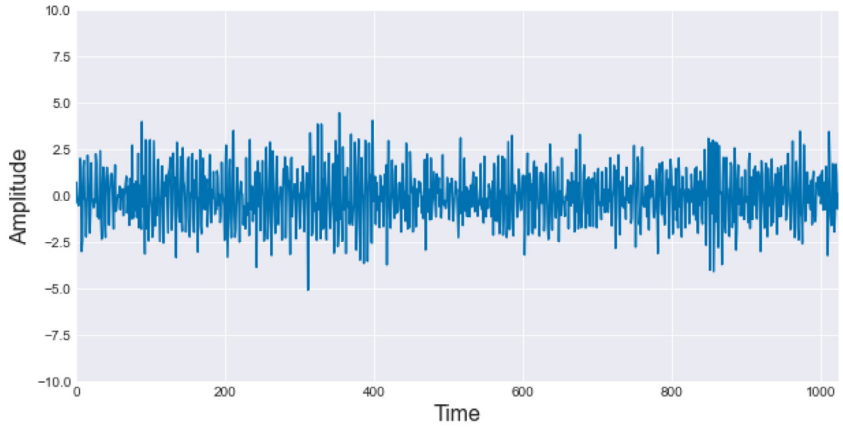
In this paper, we use both the original time-series signals and their log mel-scaled spectrograms as the input data to CNN architecture, as shown in Fig. 2. We use this approach since prior studies have shown that use of both the original signals and their derived features provides better performance than does using only the original time series data [30,31]. To compute the spectrogram, the discrete short-time Fourier transform (STFT) is used. The STFT converts a time-series signal into the time-frequency domain by computing discrete Fourier transforms (DFT) over short overlapping windows [36]. For STFT, a real-valued time-series signal y that contains 1,024 points is up-sampled to $\bar{y}$ that consists of 3,072 points using reflection padding which mirrors the signal on the first and last sample, respectively. The discrete STFT χ of real-valued signal $\bar{y}$ is formulated as follows [36]:

$$\chi(n,\ k) := \sum_{\mathfrak{l}=0}^{N-1} \bar{y}(\mathfrak{l} + nH) w(\mathfrak{l}) \exp\left(-\frac{2\pi i k \mathfrak{l}}{N}\right) \tag{4}$$

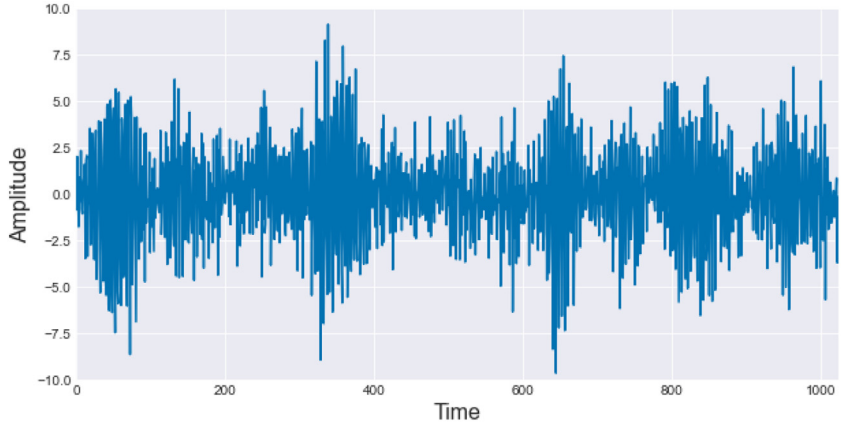
The STFT χ is given by $k \in$ [0:K], where K = N/2 is the frequency index bounded by the Nyquist frequency. The n and k of $\chi(n,\ k)$ are the k^{th} Fourier coefficient for the n^{th} time frame [36]. We assumed a hop size H = 128 and a window length of N = 2,048. The window $w(\mathfrak{l})$ for $\mathfrak{l} \in$ [0:N −1] is set to the Hanning



(a) Input signal $x(n)$



(b) Output signal $y(n)$ for AR(5)



(c) Output signal $y(n)$ for ARMA(5, 5)

Fig. 1. Input signal $x(n)$ and corresponding output $y(n)$ for AR(5) and ARMA(5, 5) models.

window [36]. The spectrogram $\mathcal{Y}(n,\ k)$, a two-dimensional matrix of the magnitude of the STFT, is formulated as below:

$$\mathcal{Y}(n,\ k) := |\chi(n,\ k)|^2 \tag{5}$$

For the mel-scaled spectrogram, we obtained the mel scale and a triangular overlapping filterbank matrix using Slaney's definition [37]. The mel scaling from Hertz to mel is linear below 1 kHz and logarithmic above 1 kHz. The Slaney-style mel scale is computed by (6).

$$Hz\_to\_mel = 1000 + \log_2\left(\frac{f}{2000}\right) * \frac{27.0}{\log_2 64} \tag{6}$$

In (6), f is Hertz with $f \in$ [0:F] where $F = Fs/\ 2$. The sampling rate Fs is 48,000 Hz. The Slaney-style filter bank $M(k, m)$[37] with $k \in$ [0:K] and mel scale $m \in$ [0:M=127] uses 30 linearly-spaced filters and 98 log-spaced filters. The details for the Slaney-style filter bank are described in [37]. The mel-scaled spectrogram $\overline{\mathcal{Y}(n,\ m)}$ is achieved by implementing a dot product of the spectrogram $\mathcal{Y}(n,\ k)$ and the filter bank $M(k, m)$ shown in Eq. (7). The log scale is calculated using power-to-dB conversion.

$$\overline{\mathcal{Y}(n,\ m)} = \mathcal{Y}(n,\ k) \cdot M(k, m) \tag{7}$$

For data preprocessing, time-series signals and their log mel-scaled spectrograms were subdivided as shown in Fig. 2. Each

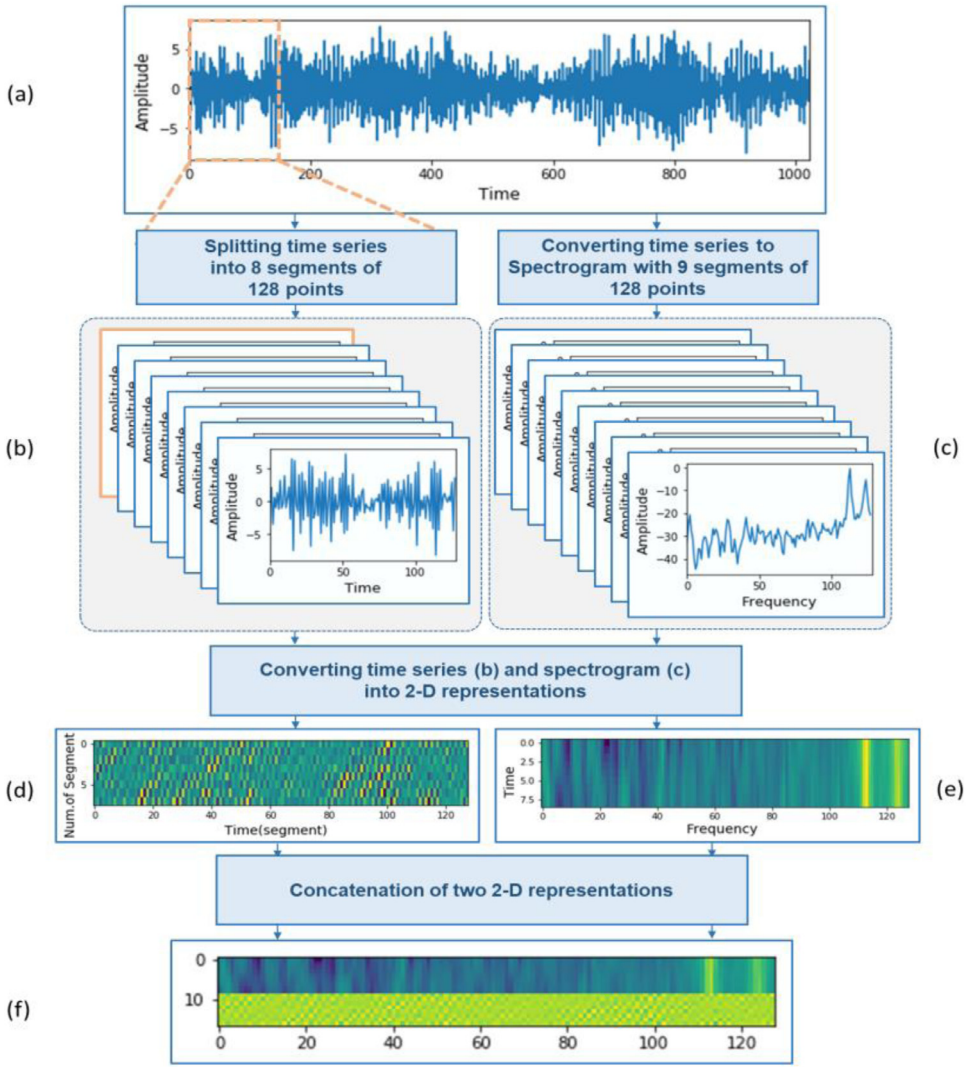


Fig. 2. Data pre-processing for ARMA identification.

time-series signal, consisting of 1,024 points (Fig. 2(a)), was split into 8 segments of 128 points (Fig. 2(b)). The log-scaled spectrogram features (Fig. 2(c)), 9 segments of 128 points (lengths of time frame 9 and mel-scaled range from 0 to 127), were extracted from the time-series signal (Fig. 2(a)). For 2-dimensional CNN, the 2-D representations of both time-series segments (Fig. 2(b)) and mel-scaled spectrograms were expressed as Fig. 2(d) and (e). Finally, both Fig. 2(d) and (e) were combined into one vector of 17 $\times$ 128 (Fig. 2(f)) as an input vector for the model order selection. The combined original time series data and spectrogram, which is denoted as $\delta(w,\ m)$ in Eq. (8), is used as the model training data for the CNN architecture.

$$\delta(w,\ m)\ =\ \overline{\mathcal{Y}(n,\ m)}\ +\!+\ \mathrm{u}(l,\ m) \tag{8}$$

In Eq. (8), $u(l,\ \mathrm{m})$ is the original signal which is split into l segments of m points, shown Fig. 2. The $\overline{\mathcal{Y}(\mathrm{n},\ \mathrm{m})}$ is the mel-scaled spectrogram and "++" denotes combined signals. The spectrogram $\delta(\mathrm{w},\ \mathrm{m})$ is combined based on m, thus w is the sum of n and l. This preprocessing technique enables one to train the network model more efficiently by using more diverse information. Note that when the input data are not large enough, the spectrogram may not provide sufficient feature dynamics, hence, including the original time series prevents this.

2.3. Order selection methods

For the model order selection, we selected two popular ImageNet classification models: Google Inception-v3 [26] and ResNet50-v2 [28] based on 2-D CNN. Inception- and ResNet-based models consist of deep convolutional layers to obtain significant patterns derived from large datasets. The CNN model is formed by many multiple layers with convolutional and pooling operations. These operations are applied to each layer which generates a variety of CNN filters with stride and padding parameters. The CNN model captures both low-level and high-level feature representations between input data and output target by building deep layers which create many features with various filters. These combinations enable capturing of significant spatial and temporal nonlinear correlations between the data and the target.

The Inception-v3 is built by symmetric and asymmetric blocks of 42 layers developed from the first GoogleNet (Inception-v1) [27], using three inception modules with 24 million parameters. Each inception module consists of different convolutional filters to generate a variety of features, and the output feature maps are concatenated into a vector to form the input to the next stage [26], as shown in Fig. 3. The structure enables estimating more informative features using these various feature matrices, which are concatenated. The details for the Inception structure are shown in [26].

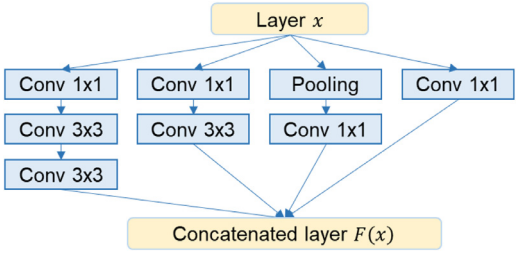


Fig. 3. An Inception module made by Inception-v3.

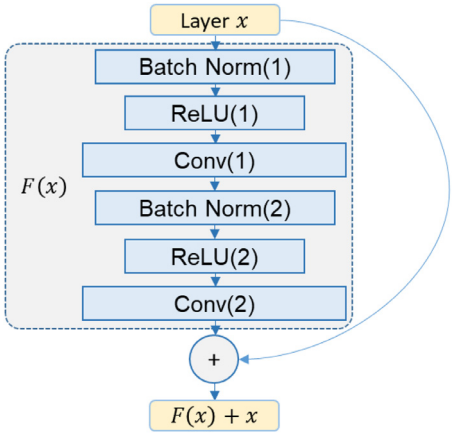


Fig. 4. Residual module made by ResNet-v2.

The ResNet50-v2 [28] is composed of multiple residual module-v2s with 26 million parameters, revised from the first residual module [29]. The residual module-v2 contains batch normalization, ReLU activation, and convolutional layers to form $F(x)$, as shown in Fig. 4. $F(x) + x$ is denoted as the residual module, and it is realized by mapping an identity skip connection from the layer x to $F(x)$. The skip connection assumes identity mapping to prevent the CNN from having vanishing gradients.

The overall structure, with identity mapping, achieves higher performance with increased depth of the CNN layers, which in turn produces results that are better than generic CNN structures that use only $F(x)$. The details for ResNet-v2 are provided in [26]. Both ImageNet models as described include the average and max pooling, batch normalization, dropouts, and fully connected layers as well as convolutional layers. The training process for each model is described as follows:

Algorithm: ImageNet model training process.

Data_X = datasets of the pre-processed signal $\delta(w, m)$
if type(Data_X) == AR:
Label_y = the ranges of the AR orders of Data_X
else if type(Data_X) == ARMA:
Label_y = The multiplication between the ranges of the AR and MA orders of Data_X
for epoch = 0:total_epochs
Upsampled_X = data_upsampling(Data_X)
for batch = 0:(total_size(Data_X)/batch_size)
*start_point = batch*batch_size*
*end_point = (batch+1)*batch_size*
batch_X = Upsampled_X(start_point:end_point,:)
batch_y = Label_y(start_point:end_point)
Training_ImageNet(input: batch_X, output: batch_Y)

Each input vector $\delta(w, m)$ was up-sampled from 17 × 128 to 112 × 128 to cover a large convolutional network structure. The large networks include a lot of padding, stride, and pooling operations, and they induce feature map vanishing when input size is too small. The size of the output layer, *Label_y* is based on the initial choice of model orders. For our case, we purposely chose the output size, y, of the ImageNet to be overdetermined as: 9 for AR, 10 for MA, and 90 for ARMA since these model orders ranged from 1 to 9 for AR and 0 to 9 for MA. When trained, the ImageNet estimates the model orders from the input data and the output layer, and it contains the probabilities associated with each model order. For example, there are 9 probability numbers for an AR model order selection of 9, 10 probability numbers for an MA order selection of 10, and 90 probability numbers for an ARMA (9, 10) model. There is a probability associated with each output layer, and the label index of the output layer with the highest probability is chosen as the estimated AR/ARMA order.

Finally the ImageNet models train the up-sampled $\delta(w, m)$ to predict output *Label_y*. The predicted label was determined via Softmax classifier. For the model order training process, both ImageNet models were initialized using the Xavier initializer. The loss function was Softmax cross entropy. The Adam optimizer for ResNet50-v2 and the RMSprop optimizer for inception-v3 were selected, with a learning rate of 0.001. The epoch was 25, which denotes that the entire training dataset was trained 25 times using the batch size of 512. These optimal values for optimizer and learning rate were determined via trial and error to find the best hyperparameters.

3. Experiment

This section describes the datasets for training, validation, and testing for model order selection. We designed a set of experiments in order to evaluate the general and specific AR/ARMA models including noise contamination of various levels of AWGN. Each of these steps is detailed below.

3.1. Dataset

We generated AR and ARMA datasets for the general and specific models for the scenarios shown in Table 1.

As shown in Table 1, we generated AR and ARMA time-series data sets for (1) the general model for any time series, and (2) the specific model without and with various levels of AWGN. The time series of AR/ARMA models were synthetically generated using Eqs. (1) and (2). For the general model, 362,160 datasets were generated for each of the AR and ARMA signals using a total of 362,160 coefficients. The MA portion $x(n)$ was created using the independent and identically distributed Gaussian distribution. For training, 360,000 datasets were used. The test and validation data sets had 1,080 data each and they had entirely different coefficients than those used in the training data. For both the general and specific models, only the output signal $y(n)$ of the time series was used as the input vector to the two different models of the ImageNet.

For the specific model, we generated AR signals of 271,800 datasets and ARMA signals of 271,800 datasets. The AR and ARMA model datasets were generated using 2,718 coefficients, respectively. For each of the 2,718 coefficients, the MA portion x(n) was generated 100 times using independent and identically distributed Gaussian distribution, thus resulting in 271,800 datasets. Of the 271,800 datasets, 270,000 were used for training, 900 were used for validation, and 900 were used for testing, for both AR and ARMA models. The test and validation data included the same coefficients as those of the training data, but these coefficients were different between test and validation datasets for both AR and ARMA models. The AR/ARMA model coefficients were generated randomly while retaining the system's stability and invertibility. In addition, three different levels of GWN were added to both AR and ARMA models to achieve SNR of 20 dB, 10 dB, and 0 dB.

Table 1
Datasets for general and specific models for each scenario.

Scenario /model	Training	Validation	Test (*noise)	Number of coefficients
(1)AR model	360,000	1,080	1,080	362,160
(1)ARMA model	360,000	1,080	1,080	362,160
(2)AR model	270,000	900	900(*4)	2,718
(2)ARMA model	270,000	900	900(*4)	2,718

3.2. Model training

We trained an AR model for the AR process and separate AR, MA, and ARMA models for the ARMA process using each ImageNet model. The output size was 9 for the AR model, 10 for MA and 90 for ARMA. The input size of the models was fixed to 112 × 128, as described in Section II. We also trained both ImageNets using only the original time-series signal, $u(l, m)$, in Eq. (8) to compare this approach to our proposed input consisting of both time-frequency and the original time series data. The $u(l, m)$ was up-sampled to 112 × 128. The hyper-parameters of the ImageNet using $u(l, m)$ were set with the same parameters as that of the case with the combined input consisting of both time-frequency and the original time series data. The models were trained using Python 3.7 with Tensorflow 1.18.3 version.

The best models were selected depending on the accuracies of the validation datasets and the chosen model's performance was evaluated using the test datasets. For the scenario when AWGN was added to the specific data sets, the training data were based on only the uncontaminated data. This was done to simulate the realistic scenario where we do not know how much the data have been corrupted by various noise sources.

4. Results

In this section, we evaluate the accuracy of each method in correctly estimating the model orders. For proposed methods, each of the best order selection model is chosen based on the highest accuracy in the validation dataset post training. We also compare the models made with ImageNet to the two widely used traditional approaches to estimating the model order determination methods: the Akaike information criterion (AIC) [10] and Bayesian information criterion (BIC) [11].

Figs. 5 (a) and (b) shows the model order selection accuracies using Inception-v3 and ResNet-v2 with both time-frequency and the original time series data, and only the original time series data, respectively, on the validation and test datasets that are not contaminated by any noise levels. As shown in Figs. 5 (a) and (b), the accuracies for AR and ARMA model order determination were better with the combined time-frequency and the original time series than using only the original time series data for all cases considered. These results suggest that the proposed multiple features extraction using both time-frequency and the original data provide more accurate model order determination.

Fig. 6 illustrates in detail the accuracies for training, validation, and test at each epoch using both models using the proposed time-frequency and the original time series. One epoch indicates an entire training dataset that is used at a given time, thus Fig. 6 shows how the performance of the various model identifications is changed for each epoch. For the AR time-series signals, the best accuracies of 78.24% and 80.09% were obtained by the Inception-v3 and ResNet50-v2 models, respectively, using test sets. In the specific modeling case, their accuracies improved to 95.56% and 95.89%.

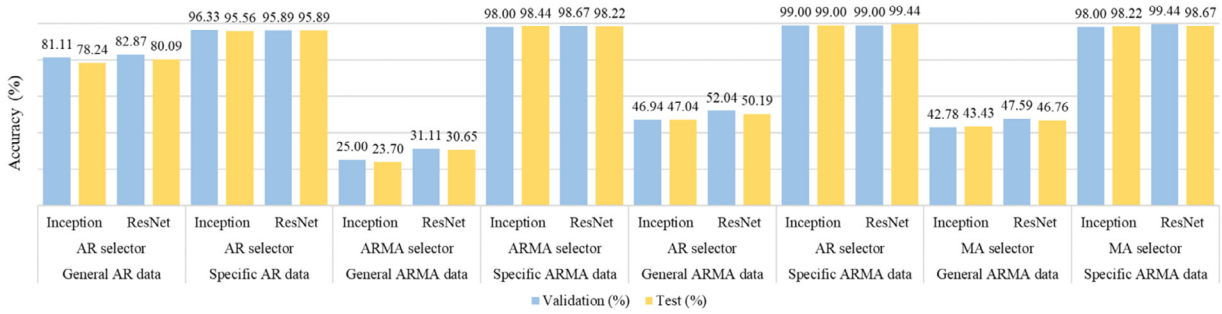
For the ARMA time series, we examined the accuracy in two ways. The first criterion was the number of combined AR and MA orders that were correctly determined. The second criterion was the number of separate AR and MA model orders that were correctly determined. The accuracies of the general ARMA model order selection using time-frequency and time series data were 23.70% for the Inception-v3 and 30.65% for ResNet50-v2 models, respectively, using test datasets, per the first criterion (determination of combined AR and MA orders correctly). For the specific model, the accuracies for the ARMA order selection for the Inception-v3 and ResNet50-v2 models using both time-frequency and time series data were 98.44% and 98.22%, respectively, using the test datasets, per the first criterion.

Figs. 5 (a) and 6 show the performance using separate AR (47.04% for Inception-v3, 50.19% for ResNet50-v2) and MA (43.43% for Inception-v3, 47.76% for ResNet50-v2) model order selection for the general time series involving ARMA processes. The accuracies improved to 99.00% (for Inception-v3) and 99.44% (for ResNet50-v2) for AR and 98.22% (for Inception-v3) and 98.67% (for ResNet50-v2) for MA for the specific model using the second criterion (AR and MA models were separately counted).

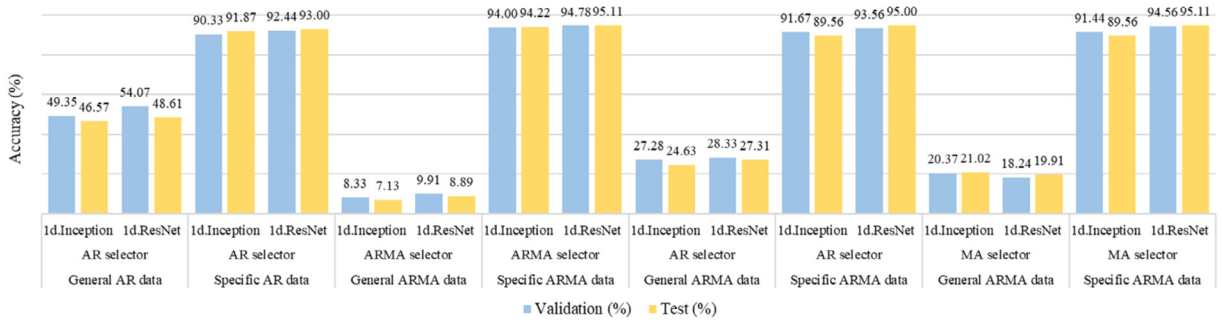
Tables 2 and 3 show the results for the proposed general and specific models, respectively, in determining accurate model orders for the clean and noisy signals with AWGN at 20 dB, 10 dB, and 0 dB SNR levels. Also shown in these tables are the comparison of ResNet50-v2 and Inception-v3 using both time-frequency and the original time series data as well as only the original time-series data to both AIC and BIC methods. Table 2 shows the result of estimating AR model order when the output label is only the AR model as well as the combined ARMA model orders when the output label is an ARMA model. Table 3 shows the accuracies of each of the AR and MA models that were correctly determined for a given ARMA model.

Both the proposed general and specific models for AR and ARMA model order determination outperformed AIC and BIC for nearly all cases. ResNet50-v2 in general had slightly better performance than did Inception-v3, thus, henceforth, the former method will be mainly described for the results in Tables 2 and 3. In Table 2, the ResNet50-v2-based general models with both time-frequency and time series data achieved accuracies of 79.11%, 73.33%, 49.33%, and 22.78% in clean, 20 dB, 10 dB, and 0 dB conditions for the AR order for a given AR process. For the ARMA process, we obtained the best test accuracies of 30.11%, 24.44%, 14.11%, and 4.00% using the ResNet50-v2 for the clean data and the three sequentially decreasing SNR levels, respectively. The performance of AR and ARMA model order selection using ResNet50-v2 improved with the specific models; the specific models obtained the best accuracies of 95.89% (clean data), 93.78% (SNR = 20 dB), 76.67% (SNR= 10 dB), and 29.33% (SNR = 0 dB) for the AR order in a given AR process, and accuracies of 95.89% (clean data), 93.78% (SNR=20 dB), 76.67% (SNR=10 dB), and 29.33% (SNR = 0 dB) for the ARMA order given an ARMA process. In addition, for most cases, the proposed approach to using both time-frequency and the original time series data provided more accurate model order determination than using only the time series itself.

Table 3 provides the results of separate AR and MA model order selection for a given ARMA time-series data. In Table 3, the mark [*] next to the model name indicates the results are either for the AR or MA parts of the ARMA models. Other general and specific models that do not include a mark [*] are individual



(a) The validation and test accuracies for AR, MA, and ARMA order selections (Inception-v3 vs. ResNet50-v2 using both time-frequency and time series data)



(b) The validation and test accuracies for AR, MA, and ARMA order selections (Inception-v3 vs. ResNet50-v2 using only time series data)

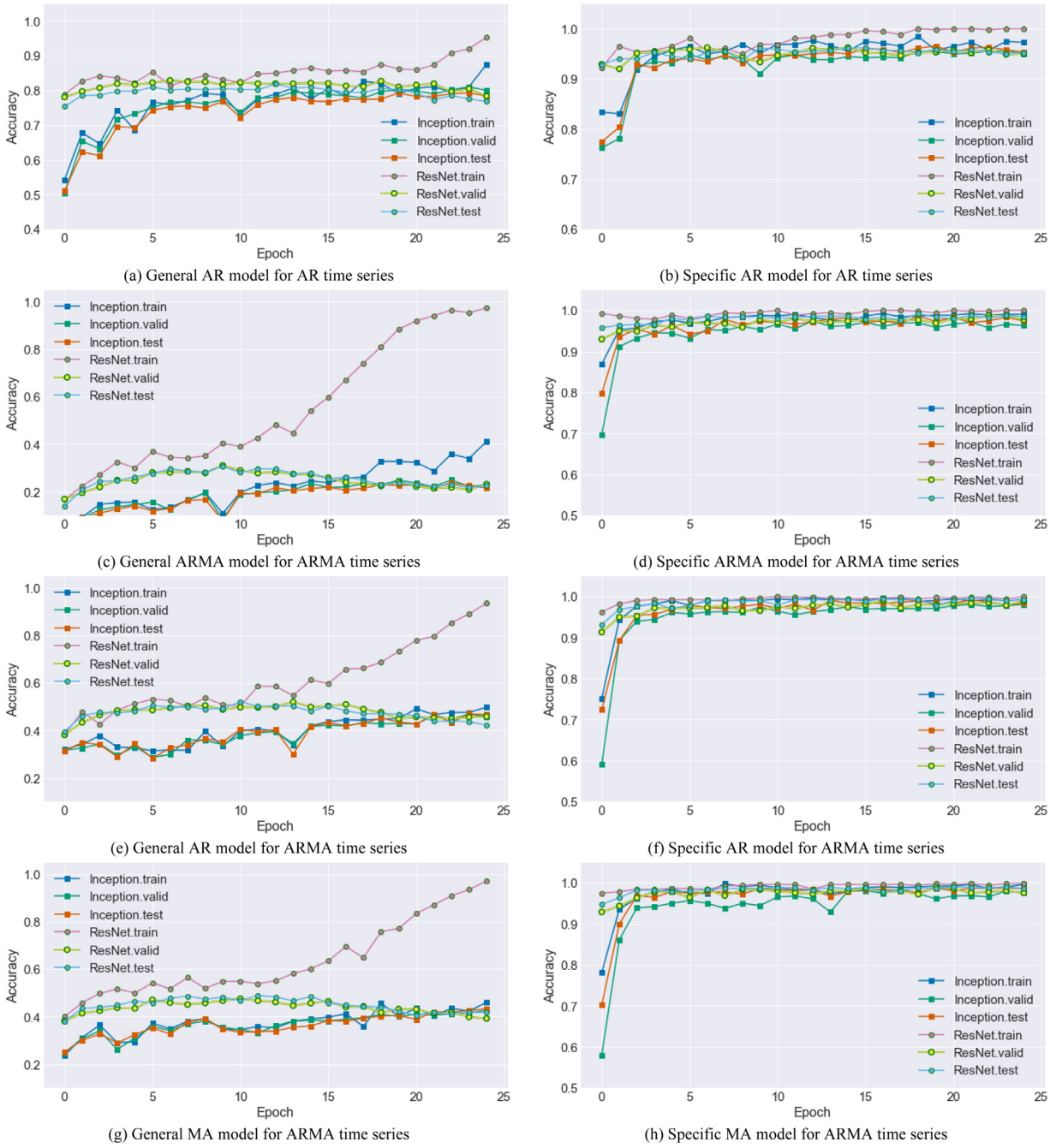
Fig. 5. The performance comparison for AR, MA, and ARMA order selections (clean signals).

AR and MA models that were trained for only the AR or MA order, respectively. The accuracies obtained by the AR and MA parts of ARMA models were similar to the performance of the individual AR and MA models. The ResNet50-v2-based general models (those that do not include a mark [*]) using time-frequency and time series data obtained accuracies of 52.89% (clean data), 50.56% (SNR = 20 dB), 39.67% (SNR = 10 dB), and 24.22% (SNR = 0 dB) for the AR order, 49.00% (clean data), 43.89% (SNR = 20 dB), 29.11% (SNR = 10 dB), and 12.33% (SNR = 0 dB) for the MA order selections. The accuracies improved using the specific ARMA model to 98.78% (clean data), 94.33% (SNR = 20 dB), 69.78% (SNR = 10 dB), and 28.67% (SNR = 0 dB) for the AR order; 98.67% (clean data), 93.11% (SNR = 20 dB), 62.22% (SNR = 10 dB), and 19.33% (SNR = 0 dB) for the MA order selections. In addition, for most cases, the proposed approach to using both time-frequency and the original time series data provided more accurate model order determination than using only the time series itself. The execution times for each model in Tables 2 and 3 are the average computational times for the four test data sets which consist of the clean and three different SNR levels. The computational time for preprocessing to convert time series into a time-frequency plot took 0.004 seconds for a given dataset. For these tests we used Intel(R) Xeon(R) E-2246G CPU @3.60GHz, and 32GB memory in Windows 10.

Table 2
AR and ARMA order selection accuracy for each AR and ARMA process (Specific AR and ARMA Time series datasets).

Order selector	Model	Clean (%)	20 dB (%)	10 dB (%)	0 dB (%)	Ave. Execution Time (sec)
AR	AIC	64.22	53.67	29.56	18.44	468.90
	BIC	69.33	62.11	41.89	21.22	468.72
	Inception-v3(general)	76.00	70.22	**51.67**	**27.89**	**18.05**
	ResNet50-v2(general)	**79.11**	**73.33**	49.33	22.78	34.84
	Inception-v3(specific)	95.67	**94.44**	**81.67**	**35.22**	**16.99**
	ResNet50-v2(specific)	**95.89**	93.78	76.67	29.33	32.41
	1d.Inception-v3(general)	54.07	48.61	**47.59**	35.46	17.22
	1d.ResNet50-v2(general)	49.35	46.57	44.91	**37.04**	32.62
	1d.Inception-v3(specific)	91.78	89.00	**76.33**	**33.78**	13.12
	1d.ResNet50-v2(specific)	93.00	90.67	76.11	31.33	28.95
ARMA	AIC	17.89	15.11	8.33	3.33	32553.77
	BIC	22.78	19.78	10.00	1.22	32515.41
	Inception-v3(general)	19.67	18.00	11.22	**4.22**	**18.11**
	ResNet50-v2(general)	**30.11**	**24.44**	**14.11**	4.00	33.06
	Inception-v3(specific)	**98.44**	93.44	**62.89**	**14.78**	**17.22**
	ResNet50-v2(specific)	98.22	**94.67**	61.67	7.89	32.33
	1d.Inception-v3(general)	8.89	8.52	5.19	3.24	16.01
	1d.ResNet50-v2(general)	7.13	5.93	4.07	1.94	34.41
	1d.Inception-v3(specific)	94.22	93.67	75.78	31.33	13.46
	1d.ResNet50-v2(specific)	95.11	93.78	**81.56**	**31.33**	28.46

Note: "1d." denotes ImageNet models using only the original time series data



(a) General AR model for AR time series

(b) Specific AR model for AR time series

(c) General ARMA model for ARMA time series

(d) Specific ARMA model for ARMA time series

(e) General AR model for ARMA time series

(f) Specific AR model for ARMA time series

(g) General MA model for ARMA time series

(h) Specific MA model for ARMA time series

Fig. 6. The performance of training, validating, and testing each epoch by proposed models for AR, MA, and ARMA order selection using the proposed both time-frequency and time series features.

5. Discussion

We proposed two different models of ImageNet-based deep learning, the Inception-v3 and ResNet50-v2, for approximating AR and ARMA model orders. We also used a preprocessing technique which used both the time series and its spectrogram to ensure diversity and enhanced feature dynamics for better accuracy of the model order determination. Traditional model order determinations are largely based on least-squares approaches to minimize the associated cost functions. Some of the well-known and widely used traditional model order methods are the AIC and BIC. Given the advances in deep learning methods, in this work we examined if some of the recently developed and accurate ImageNet-based methods can more accurately determine the model orders when compared to AIC and BIC. Hence, we performed simulation examples involving various AR and ARMA models with and without noise sources. Moreover, we examined how the proposed deep learning methods would fare when the testing datasets' coefficients are related to those of the training data, or are not. The latter case represents realistic scenarios.

We found that for all cases considered involving both AR and ARMA models, the two ImageNet models, in particular the ResNet50-v2 using time-frequency and time series data, provided better accuracy in determining correct model orders when compared to either the AIC or BIC. When the testing and validation data did not contain coefficients associated with the training data, which we called the general model, as expected, the accuracy in determining the correct model order suffered. However, the per-

Table 3
AR and MA order selection accuracy for ARMA processes (specific ARMA time series datasets).

Order selector	Model	Clean (%)	20 dB (%)	10 dB (%)	0 dB (%)	Ave. Execution Time (sec)
AR	AIC*	31.78	32.11	26.00	18.56	32553.77
	BIC*	36.67	36.33	27.67	12.11	32515.41
	Inception-v3(general)*	40.89	42.11	34.22	**25.89**	**18.11**
	ResNet50-v2(general)*	**51.67**	**46.67**	**38.56**	24.11	33.06
	Inception-v3(general)	47.22	44.56	36.89	**25.44**	18.30
	ResNet50-v2(general)	**52.89**	**50.56**	**39.67**	24.22	33.04
	1d.Inception-v3(general)*	28.24	27.04	26.11	16.02	16.01
	1d.ResNet50-v2(general)*	28.33	28.15	23.80	16.94	34.41
	1d.Inception-v3(general)	24.63	23.52	21.30	15.93	16.19
	1d.ResNet50-v2(general)	27.31	28.15	23.24	18.89	34.52
	Inception-v3(specific)*	**98.89**	**96.11**	69.56	24.11	**17.22**
	ResNet50-v2(specific)*	98.78	94.33	69.78	28.67	32.33
	Inception-v3(specific)	**99.44**	94.33	**72.78**	28.78	17.41
	ResNet50-v2(specific)	99.00	**95.22**	71.78	**31.67**	32.23
	1d.Inception-v3(specific)*	24.26	22.50	15.93	14.63	13.46
	1d.ResNet50-v2(specific)*	20.28	18.98	15.83	14.07	28.46
	1d.Inception-v3(specific)	89.56	88.67	74.56	34.44	13.28
	1d.ResNet50-v2(specific)	95.00	93.22	**84.11**	**36.22**	28.54
MA	AIC*	30.56	29.78	20.33	12.89	32553.77
	BIC*	39.00	35.22	19.89	8.89	32515.41
	Inception-v3(general)*	39.67	35.11	27.89	**13.33**	**18.11**
	ResNet50-v2(general)*	**49.00**	**43.89**	**29.11**	12.33	33.06
	Inception-v3(general)	47.22	44.56	36.89	**25.44**	18.30
	ResNet50-v2(general)	**52.89**	**50.56**	**39.67**	24.22	33.04
	1d.Inception-v3(general)*	24.26	22.50	15.93	14.63	16.01
	1d.ResNet50-v2(general)*	20.28	18.98	15.83	14.07	34.41
	1d.Inception-v3(general)	21.02	22.13	17.31	13.80	16.06
	1d.ResNet50-v2(general)	19.91	19.54	17.31	13.98	34.59
	Inception-v3(specific)*	98.44	93.44	62.89	14.78	**17.22**
	ResNet50-v2(specific)*	98.22	**94.67**	61.67	7.89	32.33
	Inception-v3(specific)	98.22	90.78	63.11	23.56	17.39
	ResNet50-v2(specific)	**98.67**	93.11	62.22	19.33	32.60
	1d.Inception-v3(specific)*	95.33	94.56	79.89	39.78	13.46
	1d.ResNet50-v2(specific)*	95.89	94.89	84.44	39.67	28.46
	1d.Inception-v3(specific)	89.56	88.22	71.89	30.89	13.26
	1d.ResNet50-v2(specific)	95.11	92.78	**82.11**	**36.44**	28.77

Note: the results for AR and MA order selections made by ARMA models contain a mark [*] next the model name

formance of the ResNet50-v2 as well as the Inception-v3 was better than that of either the AIC or BIC. Certainly, when the validation and testing datasets contained coefficients from the training data, which we called the specific model, the model order determination accuracies with both ResNet50-v2 and Inception-v3 were quite accurate ($>$ 96%) for most cases when the data were not corrupted with AGWN. However, the AIC and BIC were not able to provide more than 39% accuracy. As expected, for both the general and specific model scenarios, both ImageNet-based methods' performance degraded commensurate with decreasing SNR levels, but again, their accuracies were better than were those of both AIC and BIC. We also compared ImageNets performance using both time-frequency and time series data against only the time series data. Only in two cases of the specific scenario, 10 dB and 0 dB, the input consisting of only the time series provided better results than combined time-frequency and time series data. Otherwise, the ImageNet models using proposed time-frequency and the original time series data outperformed the case where the input data consisted of only the original time series. Our results are consistent with previous reports which showed that multiple features derived from time-frequency analysis as the input of the CNN provided better performance than using only the time-domain feature [33. 34].

In Tables 2 and 3, the ResNet50-v2 model using time-frequency and time series features was found to be better than all methods, and BIC was better than AIC, thus, Fig. 7 reports only the percentage of relative error reduction (RER) between ResNet50 using the multiple features and BIC: $(Error_{ResNet} - Error_{BIC})/Error_{BIC}$ *100 (%). For the AR order selection for an AR process, the general model's RERs were 31.89%, 29.61%, 12.80%, and 1.98% for the clean, 20 dB, 10 dB, and 0 dB signals, respectively. For the general ARMA models, when the requirement is that both AR and MA terms are correctly determined, the RERs for the four conditions (clean data and 3 SNR levels) were 9.49%, 5.81%, 4.57%, and 2.81%. When only the AR terms were correctly determined from the ARMA process, the RERs were found to be 25.61%, 22.34%, 16.59%, and 13.79% for the four conditions; when only the MA terms were correctly determined from the ARMA process, the RERs were 16.39%, 13.38%, 11.51%, and 3.78% for the four conditions. Certainly, requiring that both AR and MA parameters in combination be accurate is a more stringent criterion than having either only the AR or only the MA model order be accurate; hence, we see better performance with the latter. These results suggest that when only the output signal $y(n)$ is available but with *a priori* knowledge that other inputs affect the output $y(n)$, it is better to obtain separate AR and MA models than the combined ARMA model.

Unlike the general models, the specific models can be applicable for cases when there exist sufficient training data or the experimental conditions generated a diverse set of data. When such conditions have been met, we showed via simulation examples that one obtains highly accurate model order determination results, as shown in Tables 2 and 3. Again, the results are for ResNet50-v2 and for the clean and three levels of SNR (20, 10, and 0 dB). We found RERs of 95.89%, 93.78%, 76.67%, and 29.33% for the AR order for an AR process; 97.98%, 91.82%, 58.77%, and 13.73% for both AR and MA simultaneously for an ARMA process; 99.12%, 91.09%, 62.36%, and 18.97% for the AR order estimation for an ARMA process; and 97.82%, 89.36%, 52.84%, and 11.46% for the MA order estimation for an ARMA process. Note that for both the general and specific model simulation cases, the training data were based on

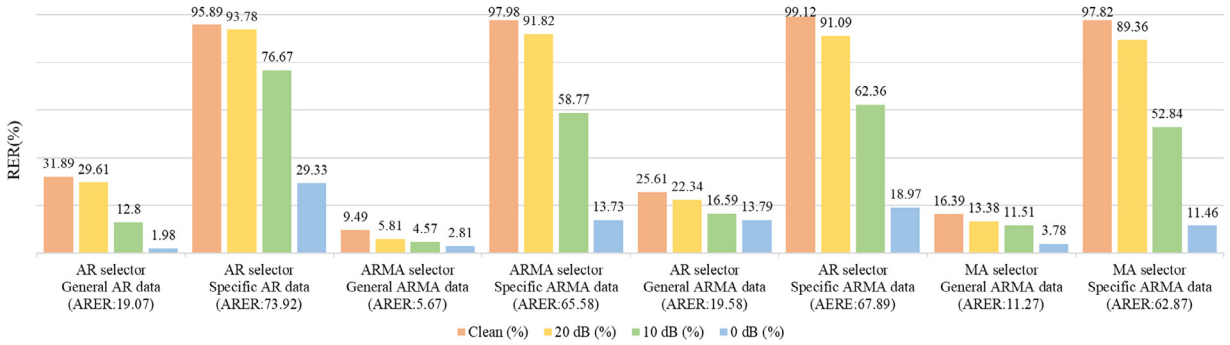


Fig. 7. The performance of relative error reduction between ResNet50-v2 and BIC in specific AR and ARMA processes.

only the clean data. The results reported above for both scenarios were based on testing data from the noise-contaminated data. Thus, these proposed methods can potentially be used to obtain accurate model order selection when the dataset is representative of the overall dynamics of the system. The specific model may require extensive training data to learn the dynamics of the system. To this end, a data augmentation approach using the generalized adversarial neural network (GAN) [34,35] can potentially be used for generating more training data for deep learning. However, we found that the general models that do not train with the dynamics of the system still provide better accuracies when compared to traditional methods. Especially, the general AR model obtained an accuracy of 80% for an AR process.

The ResNet50-v2 and Inception-v3 methods with the preprocessing provided faster execution time when compared to either AIC or BIC, leaving aside the extensive training time of the proposed approaches. Once the networks had been trained as the supervised models, the execution time was ~13-28 (for AR) or ~1,000-1,900 (for ARMA) times faster than for AIC and BIC, as shown in Tables 3 and IV. The faster execution time can lead to potential real-time application of these AR and ARMA models to various physical systems, provided that training has been done *a priori*.

The purpose of determining an accurate ARMA model order is that it can be applied to real databases. However, since we do not know the true model order for any physiological system, we limited our results to synthetically generated data with various SNR conditions to mimic real-life scenarios (noise contamination of the data) as best as we can, to examine how accurately we can determine the true model order. Note that we know the true model order since the data are synthetically generated with *a priori* determined model order. From the simulation examples, as the results do provide good confidence in the accuracy of model order determination, we can then extrapolate that our approach will most likely provide very close approximation of a real-life system's model order.

6. Conclusion and future work

In this paper, we proposed supervised model order selection methods for AR and ARMA time series using 2-D CNN-based Inception-v3 and ResNet50-v2 deep learning models and a preprocessing approach which combines both the time series and its spectrogram for both ImageNet models. To demonstrate the performance of these CNN-based methods, we designed two models—one general and one specific. The general model refers to when the validation and testing data are blind to the coefficients of the training data, whereas in the specific model the validation and testing data share dynamics with the training data. The general model simulates real life scenarios where only the output signal $y(n)$ is available, from which one needs to estimate either the AR or ARMA model orders. If the output signals truly represent the system (e.g. without significant effects from other input perturbations), both the ResNet50-v2 and Inception-v3 provided good accuracies and they were both better than either the AIC or the BIC. If the output signals are affected by noise sources (e.g. Gaussian white noise), then the deep learning methods' performance degraded, as expected. However, their performance was still far better than that of either AIC or BIC. Moreover, these deep learning methods were more tolerant of AWGN than were AIC and BIC. The results of the specific model suggest that if the training data is comprehensive, there is a good chance that accurate model orders can be obtained even when only the output data is available for a system that is perturbed by other input sources. As shown in simulation examples, this is apparently not the case for either the AIC or BIC, as their accuracy approached only ~39% at best for clean data, and this value fell precipitously with decreasing SNR levels.

While the training time is expensive for these deep learning approaches, the testing time is significantly faster than that of either the AIC or BIC. Hence, once training is completed, the model order determination for any physical system can potentially be computed in real time. Given the superior performance of the deep learning methods over the traditional approaches for model order determination of AR and ARMA time series using simulation examples, future steps are to further investigate their application to various physiological systems, including renal and cardiovascular systems, as the results may uncover additional important dynamics that have been masked with the traditional model order selection methods.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

CRediT authorship contribution statement

Jihye Moon: Data curation, Formal analysis, Software, Validation, Writing - original draft. **Md Billal Hossain:** Methodology, Writing - original draft, Investigation. **Ki H. Chon:** Conceptualization, Methodology, Project administration, Writing - review & editing.

References

[1] M. Aboy, O. Márquez, J. Mcnames, R. Hornero, T. Trong, B. Goldstein, Adaptive modeling and spectral estimation of nonstationary biomedical signals based on kalman filtering, IEEE Trans. Bio-med. Eng. 52 (2005) 1485–1489.
[2] S. Lu, K. Ju, K.H. Chon, A new algorithm for linear and nonlinear ARMA model parameter estimation using affine geometry [and application to blood flow/pressure data], IEEE Trans. Biomed. Eng. 48 (Oct. (10)) (2001) 1116–1124.

[3] S. Makridakis, E. Spiliotis, V. Assimakopoulos, Statistical and machine learning forecasting methods: concerns and ways forward, PLOS ONE 13 (Mar. (3)) (2018) e0194889.
[4] D.G. Manolakis, V.K. Ingle, S.M. Kogon, Statistical and adaptive signal processing, Spectral Estimation, Signal Modeling, Adaptive Filtering and Array Processing, McGraw-Hill, New York, 2000.
[5] K.H. Chon, R.J. Cohen, Linear and nonlinear ARMA model parameter estimation using an artificial neural network, IEEE Trans. Biomed. Eng. 44 (Mar. (3)) (1997) 168–174.
[6] Z.S. Abo-Hammour, O.M.K. Alsmadi, A.M. Al-Smadi, M.I. Zaqout, M.S. Saraireh, ARMA model order and parameter estimation using genetic algorithms, Math. Comput. Modell. Dynam. Syst. 18 (2) (2012) 201–221.
[7] Y. Zhang, B. Liu, X. Ji, D. Huang, Classification of EEG signals based on autoregressive model and wavelet packet decomposition, Neural Process. Lett. 45 (2016).
[8] L. Fenga, Bootstrap order determination for ARMA models: a comparison between different model selection criteria, J. Probab. Stat. (2017).
[9] G.B. Giannakis, J.M. Mendel, Cumulant-based order determination of non–Gaussian ARMA models, IEEE Trans. Acoust. Speech Signal Process. 38 (8) (1990) 1411–1423.
[10] H. Akaike, Statistical predictor identification, Ann. Znst. Stat. Math. 22 (1970) 203–217.
[11] G. Schwarz, Estimating the dimension of a model, Ann. Stat. 6 (1978) 461–464.
[12] J.M. Mendel, Tutorial on higher order statistics (spectra), Proc. IEEE 79 (1991) 278–305.
[13] E.J. Hannan, The estimation of the order of an ARMA process, Ann. Stat. 8 (1980) 1071–1081.
[14] G. Liang, D.M. Wilkes, J.A. Cadzow, ARMA model order estimation based on the eigenvalues of the covariance matrix, IEEE Trans. Signal Process. 41 (1993) 3003–3009.
[15] H. Akaike, Fitting autoregressive models for prediction, Ann. Znst. Stat. Math. 21 (1969) 243–247.
[16] J. Lee, W. Jhee, A two-stage neural network approach for ARMA model identification with ESACF, Decision Support Syst. 11 (5) (1994) 447–461.
[17] K. Lee, S. Oh, An intelligent approach to time series identification by a neural network-driven decision tree classifier, Decision Support Syst. 17 (1996) 183–197.
[18] W. Jhee, K. Lee, J. Lee, A neural network approach for the identification of the Box-Jenkins model, Netw.: Comput. Neural Syst. 3 (3) (1992) 323–339.
[19] K. Lee, J. Yang, S. Park, Neural network-based time series modeling: ARMA model identification via ESACF approach, in: [Proceedings] 1991 *IEEE International Joint Conference on Neural Networks*, 1, Singapore, 1991, pp. 232–236.
[20] T. Chenoweth, R. Hubata, Robert, R.S. Louis, Automatic ARMA identification using neural networks and the extended sample autocorrelation function: a reevaluation, Decision Support Syst. 29 (2000) 21–30.
[21] M.B. Hossain, J. Moon, K.H. Chon, Estimation of ARMA model order via artificial neural network for modeling physiological systems, IEEE Access 8 (2020) 186813–186820.
[22] S. Rolf, J. Sprave, W. Urfer, Model identification and parameter estimation of ARMA models by means of evolutionary algorithms, in: Proceedings of the IEEE/IAFE 1997 Computational Intelligence for Financial Engineering (CIFEr), New York City, NY, USA, 1997, pp. 237–243.
[23] H. Bellahsene, A. Taleb-ahmed, ARMA order model detection using minimum of Kurtosis: application on seismic data, Arab. J. Geosci. (2018).
[24] L. Liu, W. Ouyang, X. Wang, et al., Deep learning for generic object detection: a survey, Int. J. Comput. Vis. 128 (2020) 261–318.
[25] A. Krizhevsky, I. Sutskever, G. Hinton, ImageNet classification with deep convolutional neural networks, Neural Inf. Process. Syst. 25 (2012).
[26] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception architecture for computer vision, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 2818–2826.
[27] C. Szegedy, et al., Going deeper with convolutions, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, 2015, pp. 1–9.
[28] K. He, X. Zhang, S. Ren, J. Sun, Identity Mappings in Deep Residual Networks, 2016 cite arxiv:1603.05027Comment: ECCVcamera-ready.
[29] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 770–778.
[30] P. Khanarsa, K. Sinapiromsaran, Automatic SARIMA order identification convolutional neural network, Int. J. Mach. Learn. Comput. 10 (5) (2020) 662–668.
[31] P. Khanarsa, A. Luangsodsai, K. Sinapiromsaran, Self-identification deep learning ARIMA, J. Phys.: Conf. Series 1564 (2020) 012004.
[32] C. Liu, S.C.H. Hoi, P. Zhao, J. Sun, Online ARIMA algorithms for time series prediction, in: AAAI-16: Proceedings of Thirtieth AAAI Conference on Artificial Intelligence, 12-17, Phoenix, Arizona USA, 2016, pp. 1867–1873.
[33] K.H. Chon, D. Hoyer, A. Armoundas, N. Holstein-Rathlou, D.J. Marsh, Robust nonlinear autoregressive moving average model parameter estimation using stochastic recurrent artificial neural networks, Ann. Biomed. Eng. 27 (1999) 538–547.
[34] M. Frid-Adar, E. Klang, M. Amitai, J. Goldberger, H. Greenspan, Synthetic data augmentation using GAN for improved liver lesion classification, in: 2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018), Washington, DC, 2018, pp. 289–293.
[35] M. Frid-Adar, I. Diamant, E. Klang, M. Amitai, J. Goldberger, H. Greenspan, *"GAN-based synthetic medical image augmentation for increased CNN performance in liver lesion classification, Neurocomputing,"* Volume 321, pp. 321-331, 2018.
[36] M. Müller, "Fundamentals of music processing," 10.1007/978-3-319-21945-5, 2015.
[37] M. Slaney, Auditory Toolbox: A MATLAB Toolbox for Auditory Modeling Work. Technical Report, version 2, Interval Research Corporation, 1998.
[38] W. Tang (2018), "Model identification for ARMA time series through convolutional neural networks", https://arxiv.org/pdf/1804.04299v1.pdf (unpublished paper)