Knowledge and Information Systems (2020) 62:4541–4568
https://doi.org/10.1007/s10115-020-01495-8

REGULAR PAPER

Deep end-to-end learning for price prediction of second-hand items

Ahmed Fathalla[1,2] · Ahmad Salah[1,3] · Kenli Li[1] · Keqin Li[1,4] · Piccialli Francesco[5]

Received: 12 July 2019 / Accepted: 11 July 2020 / Published online: 24 July 2020
© Springer-Verlag London Ltd., part of Springer Nature 2020

Abstract
Recent years have witnessed the rapid development of online shopping and ecommerce websites, e.g., eBay and OLX. Online shopping markets offer millions of products for sale each day. These products are categorized into many product categories. It is crucial for sellers to correctly estimate the price of the second-hand item. State-of-the-art methods can predict the price of only one item category. In addition, none of the existing methods utilized the price range of a given second-hand item in the prediction task, as there are several advertisements for the same product at different prices. In this vein, as the first contribution, we propose a deep model architecture for predicting the price of a second-hand item based on the image and textual description of the item for different sets of item types. This proposed method utilizes a deep neural network involving long short-term memory (LSTM) and convolutional neural network architectures for price prediction. The proposed model achieved a better mean absolute error accuracy score in comparison with the support vector machine baseline model. In addition, the second contribution includes twofold. First, we propose forecasting the minimum and maximum prices of the second-hand item. The models used for the forecasting task utilize linear regression, LSTM, and seasonal autoregressive integrated moving average methods. Second, we propose utilizing the model of the first contribution in predicting the item quality score. Then, the item quality score and the forecasted minimum and maximum prices are combined to provide the item's final predicted price. Using a dataset crawled from a website for second-hand items, the proposed method of combining the predicted second-hand item quality score with the forecasted minimum and maximum price outperforms the other models in all of the used accuracy metrics with a significant performance gap.

Keywords LSTM · ARIMA · SARIMA · Linear regression · Time series analysis · Price prediction · Second-hand items

Ahmed Fathalla, Ahmad Salah and Kenli Li have contributed equally to this work

✉ Ahmed Fathalla
fathalla@hnu.edu.cn; a.fathalla@science.suez.edu.eg

✉ Kenli Li
lkl@hnu.edu.cn

Extended author information available on the last page of the article

1 Introduction

With the rapid development of Internet technologies, people are increasingly engaging in online shopping. Online shopping has a vital role in our daily lives due to the associated low cost, high convenience, ease of use, and other such advantages. Consequently, many types of retail websites, such as OLX and eBay, are available in the online market.

In particular, the past decades have seen rapid growth in second-hand consumption across many global markets as a result of the booming collection of used and unwanted products. Pricing is not only a science but also an art that requires statistical and experimental formulas to create a profile for both the brand and product in the market. One of the main challenges faced by retailers is pricing.

Research scholars have focused extensively on the problem of price prediction and price forecasting of different commodities by using diverse forecasting methods; the relevant applications include oil price forecasting [12], electricity price forecasting [32], stock price forecasting [53], coal price prediction [5], house price prediction [33], Bitcoin price forecasting [36], car price prediction [44], price prediction of ecommerce items [56], and price prediction of second-hand cars [43]. However, the price prediction of second-hand ecommerce items has not been widely addressed.

Several methods can be used to address the aforementioned problems, and these methods can be divided into two categories: statistical models and machine learning models. The methods can also be classified based on the utilized approach, i.e., machine learning (ML) and time series. In addition, the methods can be classified based on the data type, i.e., tabular and non-tabular data, text, image, product reviews, and trends.

Remarkably few studies have addressed the price prediction of ecommerce products through time. Therefore, only limited effort has been made in the context of time series price prediction of ecommerce products. Yang et al. [60] used the autoregressive moving average and kernel-based extreme learning machine methods for forecasting electricity prices using tabular data. Carta et al. [11] exploited the advantages of time series, reputation, and sentiment analysis to perform forecasting of product prices by using the autoregressive integrated moving average (ARIMA) model. Tseng et al. [56] used news textual data to propose a signal, sentiment, autoregressive and moving average (SSA–ARMA) model for price forecasting of ecommerce products, based on time series and sentiment analysis.

Noor et al. [41] used a multiple linear regression model to predict prices of new and second-hand vehicles, for which the dataset is in a tabular format. Yang et al. [59] proposed a model for predicting vehicle prices based only on product images by using a custom CNN architecture. [29,53] used sentiment analysis and machine learning for predicting stock prices. Kalaiselvi et al. [28] developed pricing analytics for smartphone products by using a multilayer feed forward neural network. Ahmed et al. [3] used a dataset of tabular data and images to address house price prediction using support vector regressor (SVR) and neural network (NN) models.

The price prediction of second-hand items has not been widely addressed. Only a few studies have addressed the price prediction of used products in a specific domain, specifically, the price prediction of second-hand cars [43].

To the best of the authors' knowledge, state-of-the-art methods have limited work for predicting the prices of second-hand products based on the ML methods. In addition, a method to predict second-hand product prices by using statistical-based approaches and time series models has not been established yet. Moreover, ML-based methods address only a certain product, while no effort has been made for developing a generic model that can

predict the price for a set of different product types. Furthermore, most of the existing second-hand price prediction methods used the textual attribute of products and do not focus on the visual features, i.e., images. However, the price prediction models of second-hand products should rely on product images beside textual data.

Compared to traditional ML techniques, the use of deep learning (DL) methods to extract considerably more information details is impressive due to their proved success at tackling complex learning problems along with high efficiency in automatically extracting features and reducing the number of handcrafted features [14,15,63].

Deep learning architectures have demonstrated state-of-the-art performance in processing text, images, speech and audio on various natural language processing (NLP) and computer vision tasks, which include language modeling [23], speech recognition [7,13], computer vision [21,24], sentence classification [30], machine translation [58], and prediction [16].

In this study, we combine recurrent neural network (RNN), more specifically the LSTM variation, and CNN architectures in an end-to-end learning model to predict the prices of second-hand products. The product's textual and visual data are consumed by an LSTM model and a CNN model, respectively. Then, the outputs of these two models are merged to generate the final prediction. In addition, we address the problem of second-hand product price forecasting using machine learning, deep learning, and statistical-based models. The major contributions of this study are as follows:

- We addressed the price prediction of second-hand products by considering a set of product features, i.e., visual and textual product features by using end-to-end learning.
- This paper is the first to propose a model that can predict the prices of different types of second-hand items.
- This paper is the first to propose a time series-based model for price range forecasting and forecasting the specific price for a second-hand product, which represents a twofold model. First, we developed a system for forecasting the price range of products for a specific product category. Second, we utilized the price prediction model to predict the quality of a product, which, in turn, reflects the cost of the product. Finally, by projecting the product quality (i.e., price prediction) on the forecasted price range, the product price in the subsequent time steps could be forecast.

The remaining paper is organized as follows. Section 2 describes the related work. Section 3 describes the research methodology. Section 4 presents the experimental results. Finally, the conclusions of the study are presented in Sect. 5.

2 Related work

Several researchers have focused on the problem of product price prediction and forecasting. The proposed methods have been applied to different applications such as oil price forecasting, electricity price forecasting, stock price forecasting, house price prediction, Bitcoin price forecasting, car price prediction, the price prediction of ecommerce items, and predictions of the end price of online auctions. Several methods are used to address the aforementioned problems, and these methods can be divided into two main categories: statistical models and artificial intelligence models.

Nevertheless, applications of traditional statistical techniques are limited to more objective factors, which results in many statistical analysis methods experiencing difficulties in achieving satisfactory results [56]. Moreover, statistical models are generally linear fore-

casting techniques, which may not perform satisfactorily when the data frequency is high [32].

Most existing price prediction models for selling second-hand items focus only on selling items in a specific domain. For instance, several researchers have investigated the prediction of car prices [41,43]. In [22,49], the authors proposed the prediction of end prices for online auctions of laptop and PDA categories, respectively. The researchers attempted to realize the best deal based on individual profit maximization.

In the following subsections, we discuss the relevant work pertaining to price prediction based on different types of data and time dependent/independent products.

2.1 Price prediction based on tabular data

Pal et al. [43] used a Kaggle dataset to perform price prediction of second-hand cars; linear regression and random forest regression models were used to solve the regression problem, and the random forest achieved better results with an R^2 accuracy of 95.82% on the training data and 83.63% on the test data. Unfortunately, in this study, the authors did not consider the price–time dependency of the used cars, and no visual features were used. Furthermore, the proposed model was applicable to only one category of products.

The Mercari price suggestion challenge [37], which is a Kaggle competition, aims at suggesting selling prices of second-hand products based on a set of tabular features, namely item description, name, item condition, category name, brand name, and shipping. Ali et al. [6] addressed the aforementioned problem using multilayer perception (MLP) of two hidden layers after applying many preprocessing and feature extraction techniques. The proposed model achieved a root mean squared logarithmic error score of 0.5001.

2.2 Price prediction based on text

Price prediction models that use sentiment analysis [22,40,48,56] address two independent domains, namely feature extraction and price prediction. However, current prediction models have some limitations. First, feature extraction is performed manually by searching for a specific set of custom features (e.g., car model, volume of cylinder, mileage in kilometers, year of manufacture, paint color, manual/automatic transmission, and price [43,48]) or by performing sentiment analysis, which is often used in opinion mining by classifying text into positive, negative, and neutral [29,56]. Second, the price prediction models are established and their performance is determined with respect to only a specific set of features. Such models must be rebuilt and updated with the parameters fine-tuned manually; in some cases, the model itself must be changed to fit a new group of features, which creates a gap between products that exhibit a tendency of dynamically variable features and price prediction models.

Stock prices prediction is a special type of time series predictions in which sentiment analysis plays a key role due to the high correlation between stock market prices and economy news. Vanstone et al. [57] evaluated the inclusion of sentiment variables of news articles and Twitter sentiment, which resulted in an improvement in the accuracy of prediction of stock prices. Shastri et al. [53] claimed that stock price prediction depends on the sentiment analysis of news headlines and other historical stock data.

Many studies that use the two aforementioned prediction stages include stock price prediction [29,53], where the naïve Bayes technique is used to determine the news polarities to be either positive or negative at the first step. While [29] used the k-nearest neighbor (KNN), [53] used a neural network for the second step that involved predicting the prices of stocks

for a particular day. However, such sentiment analysis methods are not applicable for product features, as product textual features cannot be classified as positive, negative, or neutral for determining the product price. Specifically, the product price depends on a set of measurable features, which reflect product quality.

Kalaiselvi et al. [28] formulated pricing analytics for smartphone products according to some features such as the specifications, name, product features, and reviews. To determine the quality of a smartphone, the authors performed sentimental analysis on the reviews collected for each smartphone, which reflected the demand and quality of the product. Subsequently, a multilayer feedforward neural network was used to predict the prices. Although the dataset consisted of the historical data of smartphones over the past few years, the proposed model did not exploit the time factor in the variation of prices through time.

Pudaruth et al. [48] predicted the prices of used cars by using different machine learning methods, in which the data were collected from newspapers by extracting a set of features for each car, i.e., car model, volume of the cylinder, mileage in kilometers, and year of manufacture. Considering the large number of product types, applying the method proposed in [48] for extracting features is challenging and time-consuming, which is not desirable as the data are manually collected and organized.

2.3 Price prediction based on visual features

Visual features play key roles in the purchasing process, which is possibly the most crucial factor in the decision making of a buyer [19]. Typically, a buyer will look at product pictures to obtain a general idea of the overall characteristics of the product before making his/her purchase decision. However, price prediction based on visual features has not been sufficiently addressed.

To demonstrate the impact of images in the prediction process, Di et al. [19] discussed the role of images in ecommerce by clarifying the impact of images across various dimensions. The authors stated that the "watch" behavior encoded complex signals in which the image played a key role compared to other selling variables. Ahmed et al. [3] examined the impact of adding visual features of houses and demonstrated that the house price prediction accuracy was strongly correlated to the number of visual features. Furthermore, the addition of visual features improved the R^2 value by a factor of 3 compared with that pertaining to text-only features. You et al. [62] tested the visual features, which are a reflection of a real estate property, for estimating the real estate price. Poursaeed et al. [47] confirmed that the visual aspects of a house were the key elements in its market value.

Yang et al. [59] proposed a model for predicting prices based only on product images by using two separate datasets pertaining to bikes and cars; the authors developed a custom CNN architecture, PriceNet, which is an extension of the SqueezeNet architecture. The model resulted in an R^2 accuracy of 0.98 for both the bike and car datasets. However, the authors built two independent models for each dataset, and they did not consider either the time factor in price prediction or textual attributes of other products.

Only a few researchers employed the visual feature in the price prediction task; however, most of them addressed the problem of house price prediction [3,33,47,62]. In addition, only one study addressed the vehicle price prediction using visual data [59]. As a result, a research gap exists pertaining to the problem of ecommerce price prediction based on visual features as the main feature or one of the features used for price prediction.

2.4 Time series analysis for price forecasting

Sensitive products such as oil, energy, gold, and stocks, when compared with other commodities, have high volatility and are affected by many factors, e.g., political factors, influence of neighboring markets, textual data collected from social media, and other economic factors [32]. Additionally, product prices change over time. For example, the number of cars registered between 2003 and 2013 witnessed a spectacular increase of 234% [48]. Therefore, price prediction models that do not consider the time factor for price prediction are not effective and should not be relied on for predicting product prices in the future.

Tseng et al. [56] proposed the signal, sentiment, autoregressive and moving average (SSA–ARMA) model for price forecasting of ecommerce products, which is based on the time series and sentiment analysis of textual news; in this model, the time–price dependency is represented by the factors influencing the news, including the sentiment expression of comments and electronic news from a commodity website. Although the time–price dependency was considered in this work, price prediction was not performed for subsequent time steps. Moreover, the proposed model did not consider other primary features, i.e., visual features of the sold items.

In a similar manner, in [11], the ARIMA time series model was utilized to forecast ecommerce Amazon products by combining the information of the Amazon products with the information of the product manufacturers. The product information included the main features, product price, category, and Google reviews of the product. The manufacturer's information included the Google trends of the manufacturer and name. This approach depended heavily on Google trends and reviews to predict the product price. Unfortunately, this information is not available for most products, especially products manufactured by worldwide manufacturers. Thus, predicting the product price based on a product's description, images, and category is a more generic approach.

To the best of the authors' knowledge, none of the existing models utilize the minimum and maximum price of the item to improve the task of price prediction. In addition, none of the existing models can predict the price of several product types simultaneously, as all the models can perform predictions for only one product type. Furthermore, most price prediction models rely on only a limited number of features or online product reviews/descriptions. The state-of-the-art methods do not consider other important features such as different product category prices, time factor, nor product visual features.

3 Methodology

In the following, we use the term price forecasting to describe the task of determining the price of a product based on the time variable only; otherwise, we use the term price prediction for determining the product price based on several variables, whether the time variable is included or not. In this section, we describe the methodology of the proposed models for price prediction of second-hand items (Sect. 3.1) and price forecasting of second-hand items (Sect. 3.2).

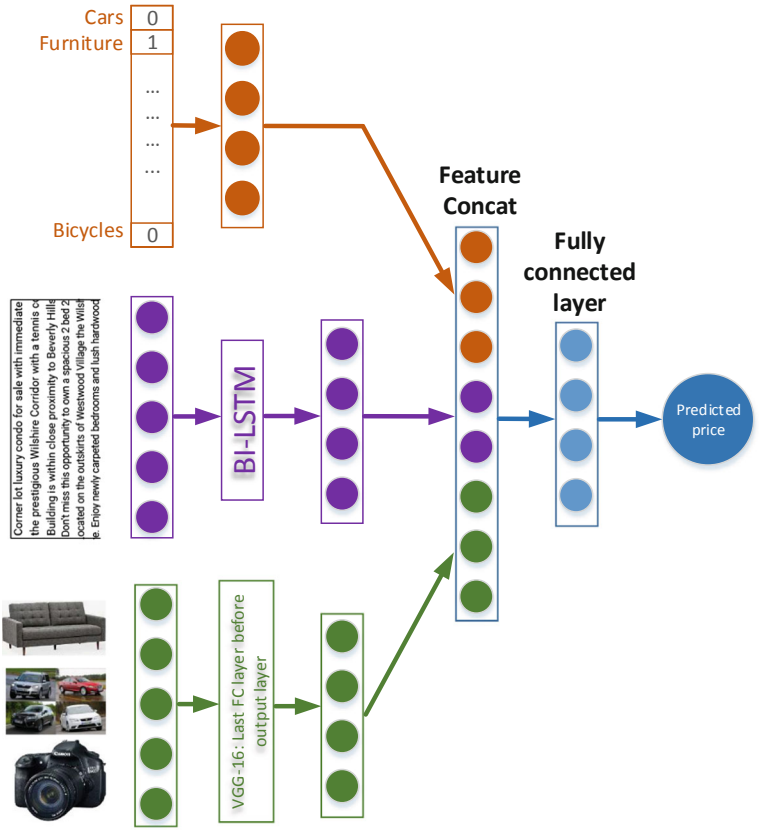


Fig. 1 Deep learning model for price prediction

3.1 Price prediction of second-hand items

3.1.1 Dataset

The dataset for second-hand items consists of advertisements for the sold items, each of which includes a list of product attributes, namely title, description, type (e.g., car, instrument, furniture, etc.), image, and price. While processing the textual data (title and description), we aligned the textual features together into one attribute named product_description. The dataset consists of 107,000 observations for 24 different product types, where the item prices range from 1$ to 20,000$.

3.1.2 Proposed price prediction model

In this section, we provide a detailed description of the proposed model. As shown in Fig. 1, the model consists of three neural network branches, namely one-hot-encoder, a bidirectional LSTM (BI-LSTM) [51], and CNN, for processing the product_type, textual features, and visual features, respectively. The proposed model consists of four main components:

1. Input layer: The input layer consists of three input vectors for the three network branches.

2. Network branches:
   (a) Product_type.
   (b) Item_description.
   (c) Item_image.
3. Merging layer: Generated by concatenating the outputs of the above three network branches.
4. Output layer: Features of the merging layer are used for predicting the final price.

3.1.3 Network branch for handling the item_description of second-hand items

To extract the textual feature from the item_description of the second-hand item in the price prediction problem, the textual data should be processed in several steps before being input to the deep learning model (i.e., LSTM). An LSTM model has the capability to automatically extract features from sequential data (text of item_description). The steps include preprocessing of the textual data, word embedding, which transforms words into numerical value feature vectors, and feeding of the data to the LSTM model for extracting the textual features. Finally, the LSTM model output passes the extracted features to the merging layer, as shown in Fig. 1.

– Preprocessing:

  Text preprocessing is the basic step in the pipeline of an NLP system, and it influences the model prediction performance. We utilized preprocessing techniques to prepare the textual data to fit the input layer of the textual network branch. The preprocessing techniques include tokenization to split sentences into words, removal of punctuation and exclamation marks, conversion of words into lower case, removal of stop words, and application of stemming and lemmatizing to reduce the number of unique words by removing inflections by dropping unnecessary characters.
– Word embedding:

  We utilized the word embedding technique of the item_description textual data of the second-hand item to obtain a better representation of the textual data. The main goal of word embedding is to obtain more efficient word representations. Word embedding is generally realized using word occurrence statistics obtained using a variety of techniques, which are trained on a large corpus of words by mapping discrete words into dense vectors [38,39]. Generally, two methods are employed to obtain the word embedding. The first method is to learn word embedding during the training process, in which word embedding vectors are initialized randomly and updated while minimizing the model objective function. An alternative method is to use a pretrained word embedding trained on much larger training data such as Glove [46] or word2vec [39].

  The textual training data (i.e., product_description) consist of the input sentence S, which consists of N words derived from vocabulary list V, and $S = \{w_1, \ldots, w_N\}$. Each word w_i is represented by a real-valued embedding vector $e_i \in \mathbb{R}^d$, where d is the size of the embedding vector, and it is considered a hyper-parameter. An embedding matrix $E \in \mathbb{R}^{|V| \times d}$ is formed by aligning the embedding of all corpus words in V, where $|V|$ represents the number of words in V. The embedding matrix E is a parameter to be learned or initialized from a pretrained word embedding, as mentioned previously. The

words of the input sentence are fetched using the embedding matrix and fed into the next layer as real-valued vectors $E_S = \{e_1, \ldots, e_N\}$.

– BI-LSTM:

We utilize an LSTM model [25] for handling the sequential data of the product_description. Specifically, BI-LSTM is employed, which allows network training in both time directions simultaneously and obtaining access to both the future and past contexts [51]. The output of the BI-LSTM is passed to a fully connected layer. Then, the output of this fully connected layer is concatenated with the extracted visual features and the one-hot-encoding of the product_type in the merging layer, as shown in Fig. 1.

3.1.4 Network branch for handling the image of second-hand items

The neural network branch for processing images is trained using transfer learning. Specifically, we use the pretrained VGG-16 [54], which has recently become the preferred model for extracting features from images [59]. After loading the VGG-16 model architecture and weights, we removed the networks' output layers, so that the output is a vector of size 4,096. The remaining layers are set as nontrainable, and a fully connected layer of 1,024 nodes is introduced. Finally, the output is passed to the merging layer to be concatenated with the output of the textual features and the one-hot-encoder network branches.

3.2 Time series price range forecasting of second-hand items

A time series is a sequence of data points recorded in time order. The effective forecasting of a time series can help realize better usage of the information for analysis and decision making. In this subsection, we describe the data used for price forecasting of second-hand products. Furthermore, we propose three methods for price forecasting, namely machine learning, deep learning, and statistical methods.

3.2.1 Dataset preparation

Our dataset includes four features: date, product description, image, and price. Considering the same product at different time points, the only feature that can be affected during such time points is the price of the second-hand product. Therefore, the process of forecasting a second-hand product as a time series analysis problem should include only the price feature, and as a result, the proposed time series is a univariate time series.

In this context, we propose handling the task of forecasting the price range of each product type separately. In other words, we propose building a model for each product type for forecasting the price range of this product type. The data of each product type are divided by the date feature (i.e., time points). For instance, if the collected data of a certain product type are spread over 210 time points (e.g., weeks), and the number of sold items of this product at each time point is 40, then the total number of sold items of this product type is $210 \times 40 = 8,400$ items.

Thus, the dataset is transformed into two separate datasets, as shown in Fig. 2. The first dataset includes the maximum prices of a product type at each time point; this dataset has two features, namely the date of the time point and the highest price at that time point. Similarly, the second dataset includes the minimum prices of a product type at each time point. We refer to these two datasets as *min-price-time-series* and *max-price-time-series*

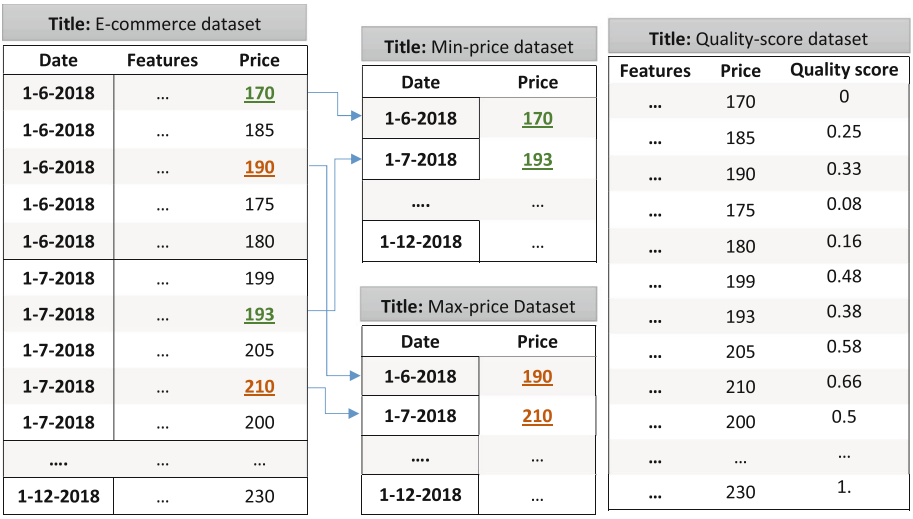


Fig. 2 Method for transforming the second-hand ecommerce dataset of a specific product type into min price, max price, and quality score datasets

datasets, respectively. Thus, we have two different time series problems, as illustrated in Fig. 2.

Using these two time series datasets, we can forecast the price range for any product. Intuitively, the predicted price should be a scalar. Thus, we propose using the description and image of the product to define the quality level of the predicted second-hand item. For the same product and for the same time point, the advertisement with the highest price is assigned to have a quality score of one, and the advertisement with the lowest price is assigned a quality score of zero. Thus, item features (item textual and visual features) are used to predict the product quality (Quality_score), as described in Sect. 3.2.2. Then, the obtained quality score is combined with the forecasted price range to produce the final predicted item's price.

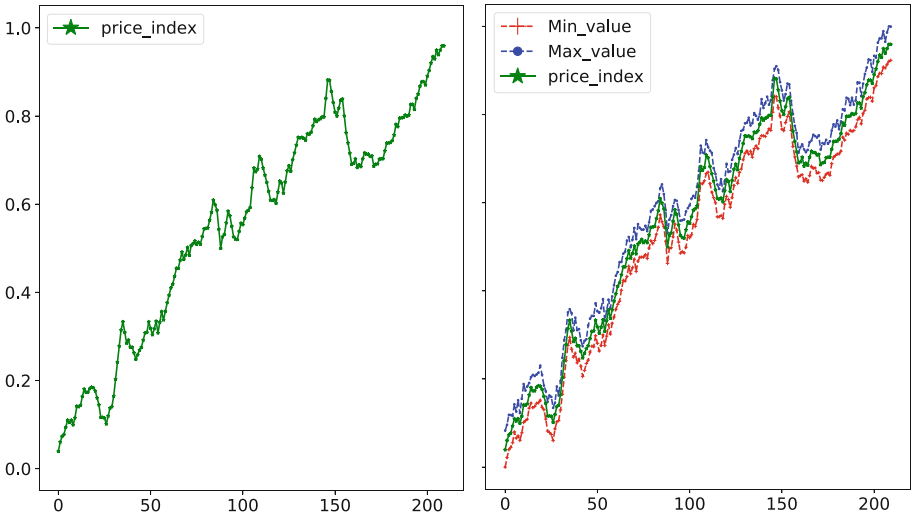
Moreover, we utilized an historical (i.e., 210 historical time points) economic price index[1] for different product types as a reference for price values during the 210 time points, as shown in Fig. 3. The price of any new product is the same at a given time point, which is the official price announced by the producer because all new products have the same level of quality. However, a second-hand product at a certain time point likely has several prices, as each second-hand item has a different quality.

3.2.2 Quality score model

We utilize the proposed price prediction model described in Sect. 3.1 to obtain an estimate of the quality of an item, as the price corresponds to the quality of an item. The Quality_score (q_score) is represented by a number between 0 and 1, which reflects the item quality; a higher q_score value corresponds to higher quality. The q_score of an item can be determined using its corresponding price by using Eq. 1.

$$q\_score_i = \frac{item\_price_i - min(pt_{list})}{max(pt_{list}) - min(pt_{list})} \tag{1}$$

[1] https://fred.stlouisfed.org/series/IQ41000



(a) A price_index **(b)** Max_price and Min_price

Fig. 3 The price_index, Max_price, and Min_price for car parts product type at different time points

where q_score_i is the quality_score assigned to an $item_i$, $item\_price_i$ is the predicted price of $item_i$, and pt_{list} is the product_type price list that $item_i$ belongs to.

3.2.3 The proposed time series forecasting models

To forecast the minimum and maximum prices of second-hand ecommerce products, we propose two different models: one to forecast the minimum price from the *min-price-time-series* dataset and another model to forecast the maximum price from the *max-price-time-series*. We call these two models as the *Min* and *Max* models/problems. For each forecasting problem (Min and Max), we used three different methods, namely machine learning baseline (linear regression), deep learning (stacked LSTM-DNN), and statistical models (ARIMA/SARIMA).

This is because we need to know the best method, of these three methods, suitable for the current data at hand. The *min-price-time-series* and *max-price-time-series* datasets consist of only two columns: date (time point) and price value, which implies that the series corresponds to a univariate time series forecasting problem. For both LSTM-DNN and linear regression models, we used one lag feature to convert the time series into a supervised machine learning problem.

Based on the data nature and the accuracy of the forecasting models, the forecasted minimum price might be greater than the forecasted maximum price. For instance, if the minimum and maximum prices are very close and the forecasting model accuracy is not high, then there is a probability that the forecasted minimum price is greater than the forecasted maximum price.

Therefore, we propose an alternative for this scenario. At any time point, there are three pieces of information, namely the minimum price, the maximum price, and the range (the maximum price minus the minimum price). Having any two pieces of information, the third piece of information can be calculated. In this vein, we propose using two models for fore-

casting the minimum price and the absolute value of the range, instead of forecasting the maximum price. Then, the predicted maximum price is calculated by adding the absolute value of the predicted range value to the forecast minimum price. Thus, the maximum price at any time point is always greater than or equal to the forecasted minimum price, as the predicted range value is absolute. To conclude, we have the following two proposed methods for forecasting the price range.

1. The proposed min–max model: For a given time point, we propose building a model to forecast the second-hand item's minimum price and another model to forecast the maximum price.
2. The proposed min-range model: For a given time point, we propose building a model to forecast the minimum price and another model to forecast the price range. Then, the second-hand item's maximum price is forecast by adding the absolute forecasted range value to the forecasted minimum price.

Linear regression baseline:
Our baseline for price forecasting is linear regression using one lag feature. Moreover, in case the minimum and maximum time series are nonstationary, we propose adding a trend component as a new independent variable to the two linear regression by creating a vector of consecutive integers from 1 to T, where T is the total number of time points.

LSTM-DNN:
The most effective deep-learning-based method for time sequence models is the RNN. Recently, LSTM for time series modeling demonstrated superior performance because of its end-to-end modeling, ability of feature extraction, easy incorporation of exogenous variables and ability to model the interactions of nonlinear features [8]. Hence, the proposed time series model to forecast the price of second-hand items consists of one LSTM layer connected to a neural network of fully connected layer(s). The final neural network layer is connected to an output layer of a single neuron of linear activation function to determine the forecasted price (i.e., final output).

The motivation behind connecting the LSTM layer to fully connected layers is to include a recurrent layer that can learn and model the sequential relations in the time series data alongside the additional hidden layers that are used to recombine the learned representations from previous layers and develop new representations at high levels of abstraction.

To achieve the best results using the proposed LSTM-DNN model, we performed hyperparameter tuning to obtain the optimal set of parameter values by training and evaluating the model of possible combinations of the individual parameter values. The mean absolute error (MAE), described in Eq. 3, is used as an objective function to evaluate the model performance.

ARIMA/SARIMA:
Finally, as we propose time series forecasting, we considered using statistical-based methods as well. The statistical-based methods for forecasting performance is significant when the time series data have a stationary and cyclical pattern. In this context, we utilized the ARIMA model to handle the univariate price forecasting problem. The ARIMA model is more efficient than other statistical models such as exponential smoothing or simple linear regression. The ARIMA forecasting technique consists of three parameters: p represents the number of lags used in the AR model, d represents the number of differencing steps to be applied on the series, and q represents the number of error terms of the MA model. Moreover, we propose using SARIMA, instead of ARIMA, in case the time series data have a seasonality component, as SARIMA can handle seasonal data better than the ARIMA. The SARIMA model has four additional parameters (i.e., P, D, Q, and S).

3.2.4 The proposed integrated price prediction model

The integration of the three aforementioned models is done in two steps. First, price range forecasting is performed by forecasting the minimum and maximum prices at specific subsequent time steps. Second, the q_score of an item is predicted using the quality score model. The resulting q_score value is projected on the forecasted price range to predict the price of $item_i$ for a number of time steps ahead, this number is represented as st. Eq. 2 describes the calculation of the forecasting method.

$$item\_price_{i,st} = min_{st} + (max_{st} - min_{st}) * q\_score_i \tag{2}$$

where min_{st} and max_{st} represent the output of the *Min* and *Max* models with st time steps ahead, respectively, and q_score_i is the predicted q_score of $item_i$.

4 Experimental results

4.1 Dataset

The first stage of any price prediction model is to collect the data. The type of data used in this work is second-hand ecommerce data crawled from an ecommerce website. Web crawling is a process of scraping data from a website. The periodicity of the collected observations is weekly.

In this work, we used Selenium Webdriver[2] framework, which is a browser automation tool for extracting data from websites. The collected data of each product consist of different product attributes, namely title, description, type (e.g., cars, bicycles, furniture, etc.), image, and price.

To prepare the dataset to be used as an input to the model, two main steps are performed: the removal of extreme outliers that have extremely high price values and the application of log transformation to the price values to be similar to a normal distribution; subsequently, the prices are normalized in the range of [0, 1] as scaling the output variables is a critical step in using neural network models [10]. The reported results correspond to the normalized prices. To evaluate the proposed models, the dataset was split into training and testing sets with a ratio of 80% and 20%, respectively, by taking into consideration a balanced ratio of the different product types.

4.2 Experimental setup

To implement the proposed deep learning model, the Keras [18] library together with a mathematical language library, TensorFlow [1], was employed. The proposed framework was developed using the Python programming language. Additionally, to implement the ARIMA/SARIMA model, we used the statsmodel library [52], which is a Python module that provides functions and classes for the estimation of many statistical models. Furthermore, the list of all other libraries used is as follows: Pandas [35], Numpy [42], Matplotlib [26], Scikit-Learn (SVR, Train_Test_Split, GridSearch, OneHotEncoder, TfidfVectorizer and PCA) [45] and NLTK (PorterStemmer and stopwords) [9]. The experiments were performed on a computer running 64-bit Linux OS with two 2.3 GHz Intel 8-core processors and four P100 NVIDIA GPUs.

[2] https://www.seleniumhq.org/docs/03_webdriver.jsp

4.3 Accuracy metrics

Two different metrics were used to evaluate and compare the performance of the proposed price prediction models (i.e., the first contribution), namely MAE and coefficient of determination R^2. MAE is the average absolute difference between the actual prices and the predicted prices, and it is expressed as in Eq. 3. The second metric is R^2, which is a popular metric for price prediction that measures the relative closeness of the predicted values to the actual values. The resulting value is a value between 0 and 1, where a higher value represents a higher accuracy of the prediction model. The R^2 value is calculated using Eq. 4.

For price range forecasting, we used the widely used time series accuracy metrics. We considered MAE (Eq. 3) as proposed in [4,17] and the symmetric mean absolute percentage error (sMAPE) metric [34] to avoid the issues affecting MAPE, as suggested in [32]. sMAPE accuracy metric is described in Eq. 5.

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}\left|y_i - \hat{y_i}\right| \tag{3}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y_i})^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2} \tag{4}$$

$$\text{sMAPE} = \frac{100}{N}\sum_{i=1}^{N}\frac{\left|y_i - \hat{y}\right|}{(|y_i| + \left|\hat{y_i}\right|)/2} \tag{5}$$

where N is the number of observations, y_i denotes the true values, $\hat{y_i}$ represents the predicted values, and $\bar{y}$ denotes the mean true values.

Finally, the integrated price prediction model, which utilizes the quality score prediction and price range forecasting models, we used the MAE and sMAPE accuracy metrics. In addition, we propose using a new accuracy metric measuring the percentage of the predicted prices that belong to the true price range in Eq. 6. For instance, assume that we used a model to predict the prices of 30 items at the same time point, and the minimum and maximum prices at this time point were 120 and 150, respectively. If the model predicted the prices of only 18 second-hand items in the range of [120, 150], then the in-range score for this model equals 60%. The higher the in-range score, the better the performance of the model.

$$in\_range\,score = \frac{\sum_{tp=1}^{M}\sum_{i=1}^{K}\begin{cases}1, & if\ pred\_item\_price_{tp,i} \in [min_{tp}, max_{tp}]\\ 0, & otherwise\end{cases}}{M \times K} \times 100\% \tag{6}$$

where M is the number of time points, and K is the number of advertisements per time point tp, $pred\_item\_price_{tp,i}$ is the predicted price for item i sold at time point tp, and min_{tp} and max_{tp} are the minimum and maximum prices for time point tp, respectively. For simplicity, we consider that all the time points have K advertisements.

4.4 Second-hand price prediction

4.4.1 Baseline

We constructed a support vector machine (SVM) to perform the regression tasks. For visual features, we used the VGG-16 model for extracting the visual features using the last fully

Table 1 Hyper-parameter values of price prediction model

Parameter	Value	
#neurons in network branches	Product type	[24-128]
	Item description	[500-256-64]
	Item image	[VGG16-1024]
Batch size	64	
N	500	
d	128	

connected layer before the output layer, and PCA [2] was used to reduce the feature dimensionality to 100 features. For the textual features, we applied TF-IDF [50] to the product description. In addition, we used PCA to reduce the feature dimensionality to 100 features. Subsequently, we merged both the textual and visual features alongside the one-hot-encoding of the product type to be the input matrix for the SVM model. Finally, we added the one-hot-vector of the product types to the input matrix.

We performed hyper-parameter tuning of the SVM model with respect to three hyper-parameters, C, tolerance and epsilon, and ran a grid search on a large scale of the parameters. Specifically, we used LinearSVR, while the other hyper-parameters were set to the default parameter values provided by the LinearSVR implementation. The best performance converged after using C less than 1 with a wide range of epsilon and tolerances; thus, we selected the following values: C = 0.01, tolerance = 0.00001, and epsilon= 0.1.

4.4.2 BI-LSTM-CNN model hyper-parameters

The number of neurons in the model branch layers, batch size, and values of N and d for the proposed method are reported in Table 1.

The batch normalization [27] technique was employed throughout the architecture to reduce the internal covariate shift between the training and validation minibatches by normalizing the nonlinear ReLU activation to have a zero mean and unit variance. To prevent model overfitting, we employed dropout regularization [55] on the fully connected and LSTM layers, with rates ranging from 0.4 to 0.6. Subsequently, we employed L1 and L2 weight regularization with a regularization factor of 0.001.

The word embedding matrix was initialized randomly due to the worse accuracy results obtained by using the pretrained word embedding (i.e., Glove and word2vec). The reason behind this accuracy is that the item description corpora have several misspelled words and many item parts that are not found in the pretrained word embedding trained using different background corpora. Consequently, the word embedding matrix is initialized randomly and trained during the main task of price prediction.

The weight initialization of the model layers follows a normal distribution. The activation functions of all the hidden layers are ReLU, except for LSTM and the output layers, which have tanh and sigmoid activation functions, respectively. Additionally, the model is optimized using the Adam [31] optimizer with default learning and decay rate settings. We applied the learning rate scheduler during model training to monitor the validation loss metrics. The scheduler reduces the learning rate by a factor of 0.15 when no significant improvement is noted in the validation metrics for 2 epochs. This aspect is beneficial during model training once learning stagnates.

Table 2 LSTM-DNN model architecture

	Type	#nodes	Activation
1st hidden layer	LSTM	32	tanh
2nd hidden layer	FC layer	32	ReLU
3rd hidden layer	FC layer	8	ReLU
Output layer	FC layer	1	Linear

4.4.3 BI-LSTM-CNN model loss function

The proposed deep learning model for the price prediction task is trained by minimizing the objective function mentioned in Eq. 7; that is, given a training set $\{(X_i, Y_i)\}_{i=1}^{N}$ of N observations, the model is trained using the following loss function:

$$\min_{w} \frac{1}{N} \sum_{j=1}^{N} \{ \left\| Y_j - F\left(X_j, w\right) \right\|_1 \} \tag{7}$$

where w denotes the network weights, $F : \mathbb{R}^n \rightarrow \mathbb{R}^1$ represents the neural network map, and n is the length of the input vector. The mean absolute error is used instead of the more traditional mean square error because product prices of different product types have different price ranges. Thus, the Euclidean norm places a large penalty on the large errors generated when using the model.

4.5 Tuning for second-hand price forecasting

In this section, we first describe the hyper-parameter tuning of LSTM-DNN. Next, the ARIMA model parameter values (i.e., p, d and q) are selected based on the ACF and PACF plots. The input data of linear regression and LSTM-DNN are split into a ratio of 0.8 and 0.2 for the training and testing tasks, respectively. Considering space limitations, details regarding the time series experiments of only one product_type (i.e., Min and Max forecasting models of a specific product_type) are clarified; results of experiments for the other product_types exhibited the same trends.

Linear regression baseline:
We trained the linear-regression model using one lag feature. Subsequently, we performed forecasting up to 30 time steps ahead. In addition, we used a trend component as an independent variable as a response to the nonstationary data of the minimum and maximum time series.

LSTM-DNN:
Similar to number of lag features applied in the linear-regression model, we used one lag feature for price forecasting. First, we tuned the DL model with respect to a set of hyper-parameters, namely number of hidden layers, number of hidden nodes, activation functions, and batch sizes. Furthermore, the early stopping method is used to stop training to avoid overfitting when the generalization error increases [61]. In addition, we used the Adam optimizer as an optimization algorithm to train the deep neural network. The model architecture that corresponded to the best test results is presented in Table 2.

ARIMA/SARIMA:
The ARIMA model provides better results when the time series is stationary. A stationary time series is one with statistical properties (i.e., mean, variance, autocorrelation, etc.) are

Table 3 SARIMA parameters

Parameter	Value
P	0
D	0
Q	1
S	12

that are constant over time. Therefore, the first step when using an ARIMA model is to verify the stationarity of the data, which is determined by applying the augmented Dickey–Fuller (ADF) test [20]. The ADF test results in a probability value of 0.722, which indicates the nonstationarity of the series. The general method to convert a nonstationary series to a stationary series uses a series difference.

To obtain the ARIMA parameters (e.g., p, d, and q), we used plots of the autocorrelation function (ACF) and partial autocorrelation function (PACF). Figure 4 shows the ACF of the original time series and the first two differencing orders. The series reaches stationarity within two orders of differencing. However, the ACF for the 2nd differencing indicates that the lag moves to the far negative zone quite rapidly, which suggests that the series might have been overdifferenced. Consequently, we set the order of differencing as 1 even though the series is not perfectly stationary.

Following these steps, the parameter q is obtained by checking the ACF plot at $d = 1$. The first lag is significant as it is above the blue area, and the second lag is not significant. Therefore, $q = 1$ is the value of the MA term. Finally, the last parameter of ARIMA model is p, which is obtained using the PACF plot. Figure 5 shows the PACF plot at $d = 1$. In the right figure, the first lag is significant, as it is above the blue area, while the second lag is not significant. Consequently, $p = 1$ is the value of the PA term.

To check time series seasonality, the time series data are decomposed to check for the existence of the seasonal component. After decomposing the *min-price-time-series* and *max-price-time-series* datasets of all the product types of the dataset, we found that all the product type data have a seasonal component.

We explained the utilized process of decomposing the time series for checking for the existence of the seasonal component through decomposing one example (i.e., *min-price-time-series* of car parts product type). Figure 6 is obtained by decomposing the time series using the Python function *seasonal_decompose* from the *statsmodels* library. Figure 6 shows that the minimum price time series of the car parts product type has a seasonal component for one product type. The maximum price time series data show the same behavior. Therefore, SARIMA was utilized, which has seasonal parameters (P, D, Q, and S). We performed a grid search to determine the optimal seasonal parameters within a range of values (e.g., 0, 1, and 2) for P, D, and Q parameters and for parameter S the search space included 6, 12, and 24. The optimal seasonal parameters are listed in Table 3. These seasonal parameters are set through a grid search; in other words, several seasonal parameters combinations of values are tested and the combination of parameter values with the highest accuracy metrics scores are reported.

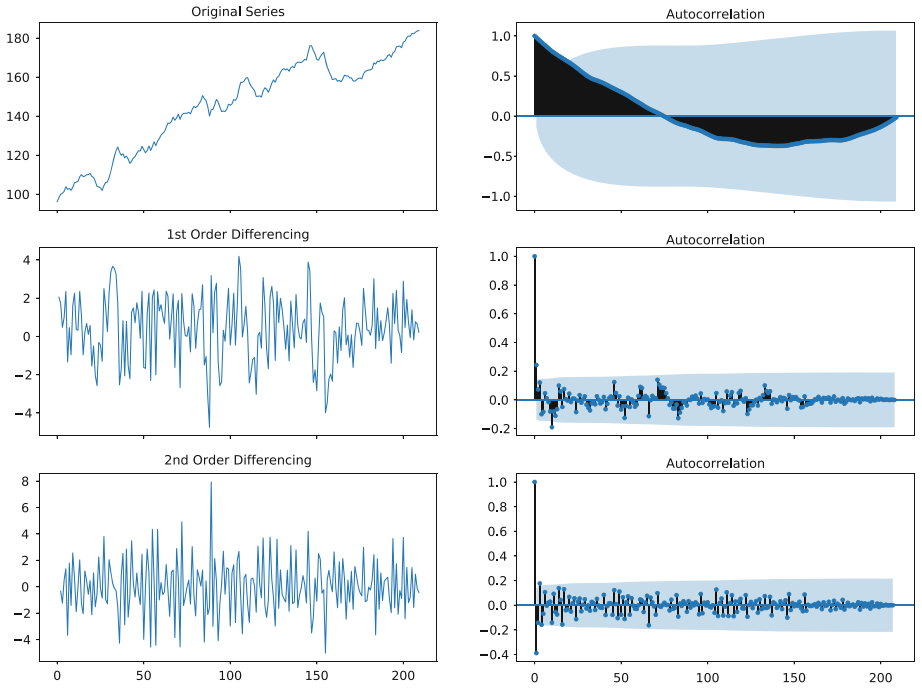


Fig. 4 ACF for the original series and the first two differencing orders

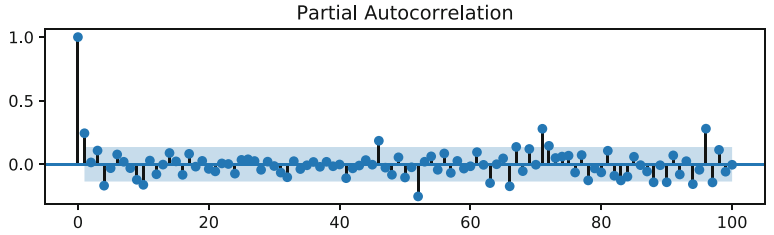


Fig. 5 PACF for the first differencing order

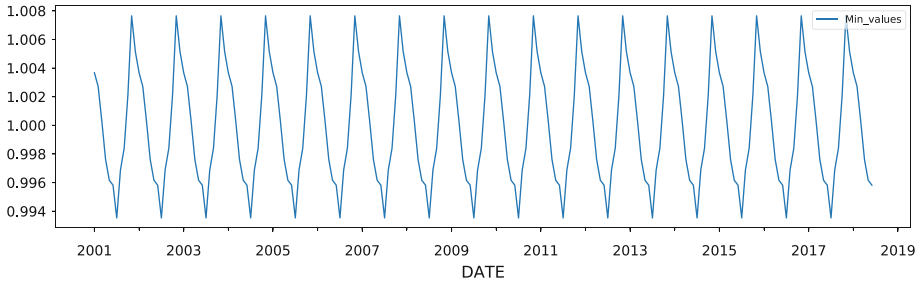


Fig. 6 Time series seasonality

Table 4 Price prediction results

Model	Results	
	R^2	MAE
SVR	66.43	0.0929
BI-LSTM-CNN	77.14	0.0702

4.6 Results and discussion

4.6.1 Results of second-hand price prediction

Table 4 presents the results for the proposed BI-LSTM-CNN price prediction model and the baseline SVR model. The proposed BI-LSTM-CNN model achieved an R^2 accuracy metric of 77.14, which demonstrates its superiority over the SVR baseline model. The reason behind this finding is the ability of deep learning models to obtain better representation of the data, thereby achieving higher accuracy results for the prediction task. The proposed BI-LSTM-CNN model converged and attained the best results on the test set after 10 epochs. The training time and model size are 7 hours and 327 MB, respectively. The SVR model is trained in 10.8 seconds, and the model size is 3 KB.

4.6.2 Results of minimum and maximum price forecasting

To evaluate the foretasted minimum and maximum prices of different product types, we utilized the data of the car parts and furniture product types. Using these aforementioned time series data, Fig. 7 depicts the true minimum prices of the time points and the forecasted minimum prices for the same time points of the test data set, which includes 42 time points, for three different algorithms. The figure shows that the forecasted minimum prices are very close to the true prices at the same time points. The forecasting of the maximum and range time series data has similar performance to the minimum prices forecasting.

The proposed method predicted the minimum and maximum prices separately through two different predictive models. Hence, one condition should be met; the predicted minimum price should always be less than the predicted maximum price. To verify this condition, Fig. 8a depicts the min–max model's predicted prices of the training and test sets for both the minimum and maximum prices time series (i.e., *min-price-time-series* and *max-price-time-series* datasets) of the furniture product type. Similarly, Fig. 8b depicts the predicted prices of the min-range model. At any given time point, the forecasted maximum price in Fig. 8 is always greater than the forecasted minimum price.

The MAE and sMAPE results of the single-step-ahead price forecasting models (i.e., *Min* and *Max* models) are reported in Table 5. Figure 9 shows the MAE metric for error of 30-time-step-ahead forecasting. The forecasting results demonstrate the efficiency of the LSTM in handling the time series data, which is a result of its ability to obtain relevant information from the past input data and potential to model complex nonlinear patterns, which is suitable for time series forecasting. In contrast, the ARIMA/SARIMA model is as a linear forecasting model and not capable of modeling other types of nonlinearity in a time series [64].

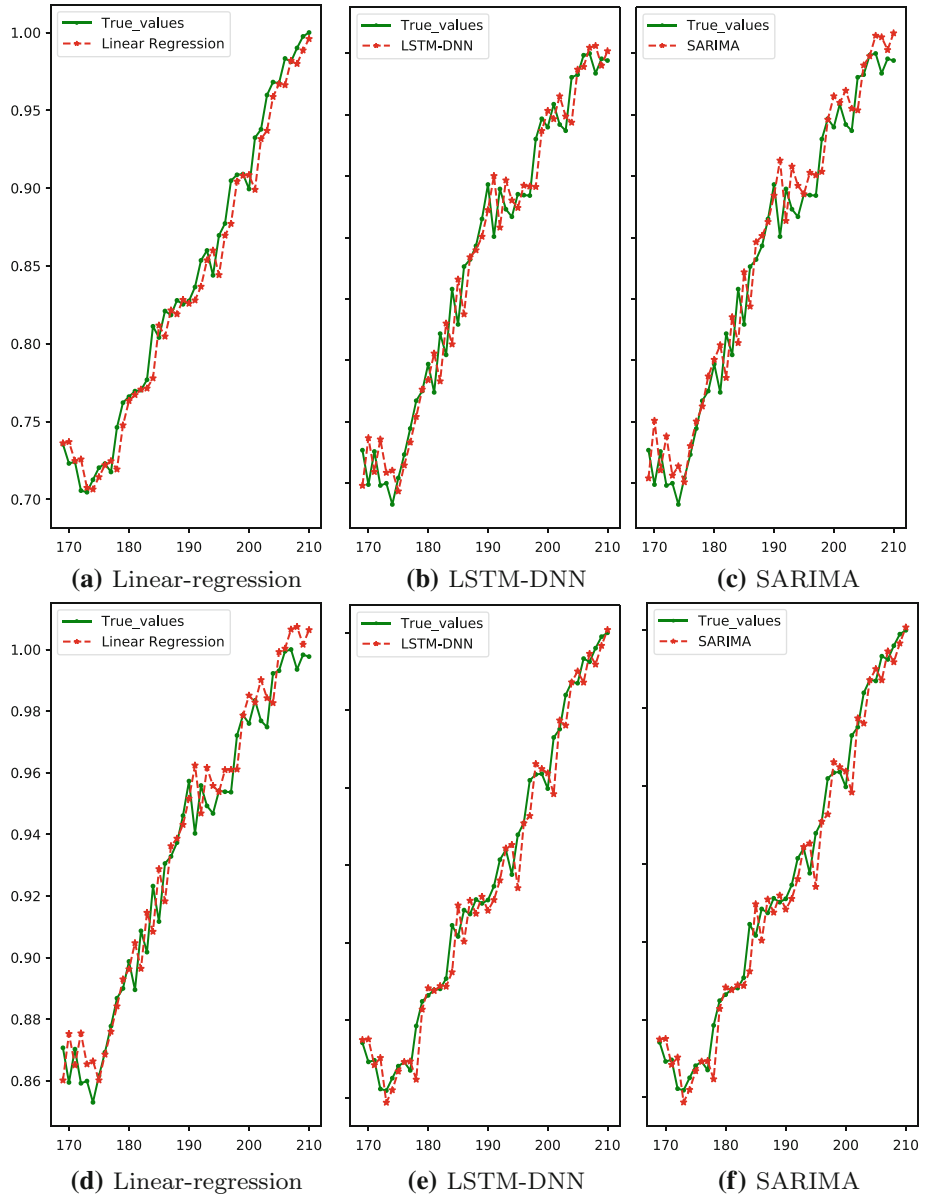
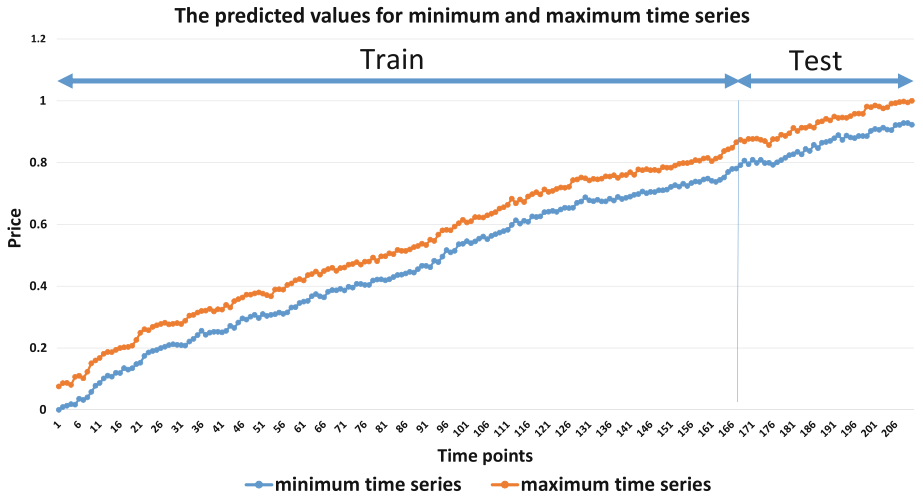


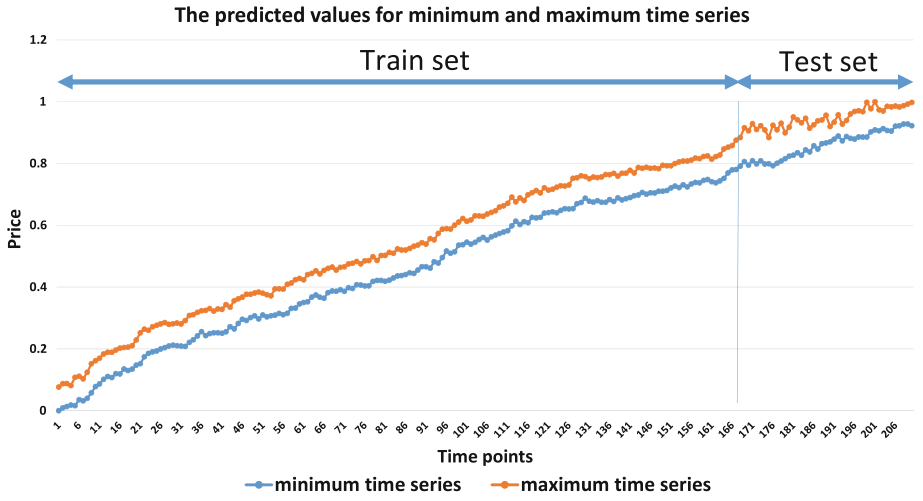
Fig. 7 Single-time-step-ahead forecasting for the minimum prices of the furniture product type (**a**–**c**) and car parts product type (**d**–**f**)

4.6.3 Results of the integrated price prediction model

The integration of the price forecasting models (i.e., *Min* and *Max* models) and the Quality_score model is shown in Fig. 10. First, the price range forecasting is performed using the *Min* and *Max* models, with MAE scores as shown in the figure. Second, the quality score of the second-hand item is predicted based on its features. Finally, the forecasted price is



(a) Price prediction of the proposed min-max model



(b) Price prediction of the proposed min-range model

Fig. 8 The forecasted prices for minimum versus maximum time series

calculated by projecting the Quality_score on the forecasted price range, which has an MAE score of 0.0702.

The integrated price prediction model has several components, (i.e., the *Min* and *Max* and the quality score models). In addition, the integrated proposed model has two version (i.e., the proposed min–max model and the proposed min-range model). Table 6 lists the MAE, sMAPE, and in-range accuracy scores of the linear regression, LSTM, and the proposed model (with its two versions). The dataset of the reported results in Table 6 belongs to one product type, i.e., furniture product type. This product type has 11,760 advertisements spread over 210 time points (weeks). We selected this product type because it has the largest number

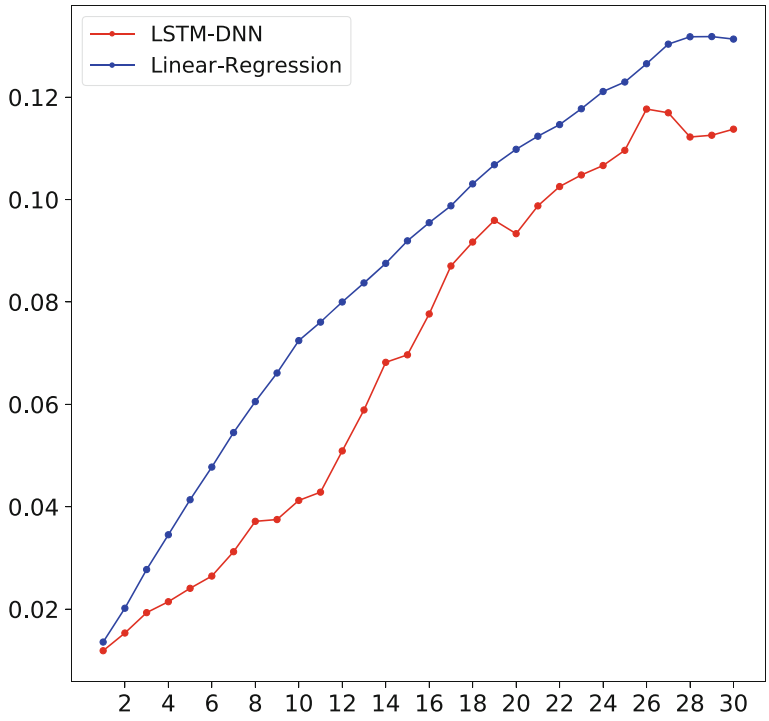


Fig. 9 Mean absolute error for 30-time-step-ahead forecasting

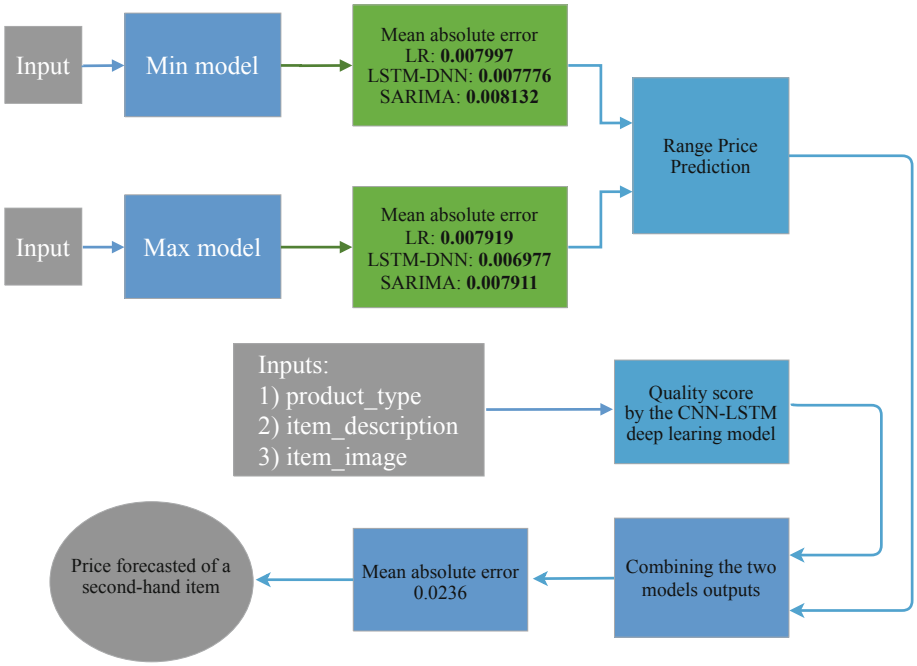


Fig. 10 Integration of the forecasting models

Table 5 Error for single-step-ahead forecasting

Prod. Type	Model	*Min* Model		*Max* Model	
		MAE	sMAPE	MAE	sMAPE
Car parts	Linear regression	0.0100	1.1992	0.0167	2.0056
	LSTM-DNN	0.0089	1.0827	0.0160	1.1536
	SARIMA	0.0106	1.2733	0.0168	2.0259
Furniture	Linear regression	0.0080	0.8600	0.0079	0.8534
	LSTM-DNN	0.0078	0.8416	0.0070	0.7572
	SARIMA	0.0081	0.8745	0.0079	0.8526

Table 6 Accuracy comparison of different methods

	MAE	sMAPE	In-range (%)
Linear Regression	0.1032	10.967	55.23%
LSTM	0.0497	5.540	43.07%
The proposed min–max model	**0.0236**	**2.661**	**80.95**%
The proposed min-range model	0.0247	2.778	64.54

of observations (i.e., advertisements). The listed scores show that the proposed methods (i.e., min–max and min-range) have the best performance for the three accuracy metrics in comparison with the baseline methods.

On the one hand, the proposed method makes use of three pieces of information, namely the predicted item quality score, the forecasted minimum price, and the forecasted maximum price. On the other hand, the other methods report only the predicted item price which is equivalent to the item quality score of the proposed method. Thus, the proposed method uses two additional pieces of information. We linked this better performance to the proposed method in comparison with the other method for this aforementioned discussion.

5 Conclusion

This work has two main contributions. The first contribution is to predict the price of second-hand items based on textual and visual features for different product types, as state-of-the-art ecommerce price prediction methods do not focus on visual features. The proposed model highlights the feasibility of combining images and textual data to make a prediction. Additionally, the proposed method allows a single model to be applicable to a dataset of different product types, in contrast to other price prediction models that use different prediction models to handle datasets of different categories of products. We utilized a hybrid CNN-LSTM model for the task of price prediction which achieved a better performance in comparison with the baseline model. The MAE scores of the proposed price prediction model and the SVM baseline model are 0.07 and 0.09, respectively.

The second contribution is to improve the predicted price of the first contribution. Thus, we proposed forecasting the minimum and maximum prices of the second-hand item, i.e., the price range of the product type this item belongs to. Then, we combined the item quality score, which is equivalent to the predicted price of the first contribution, with the forecasted

minimum and maximum prices to produce the final predicted price. We utilized three different models for the forecasting task, namely LSTM, linear regression, and SARIMA. The MAE scores of the proposed price prediction model and the two baseline methods (i.e., linear regression and LSTM) are 0.02, 0.05, and 0.10, respectively.

As future work, we will focus more on extracting features from the input images that can be used for evaluating the quality of the item to be sold, in turn helping in predicting the optimal price for an item. In addition, it would be interesting to explore the possibility of considering multiple images for each item, in addition to providing a clearer, summarized, and informative item description.

Acknowledgements This research is partly supported by the National Key R&D Program of China (Grants: SQ2018YFB020061) and the NSFC (61860206011, 61625202, 61602170, and 61750110531).

References

1. Abadi M, Barham P, Chen J, Chen Z, Davis A, Dean J, Devin M, Ghemawat S, Irving G, Isard M et al (2016) Tensorflow: A system for large-scale machine learning. In: 12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16), pp 265–283
2. Abdi H, Williams LJ (2010) Principal component analysis. WIREs Comput Stat 2(4):433–459. https://doi.org/10.1002/wics.101
3. Ahmed E, Moustafa M (2016) House price estimation from visual and textual features. arXiv preprint arXiv:1609.08399
4. Alamaniotis M, Bargiotas D, Bourbakis NG, Tsoukalas LH (2015) Genetic optimal regression of relevance vector machines for electricity pricing signal forecasting in smart grids. IEEE Trans Smart Grid 6(6):2997–3005
5. Alameer Z, Fathalla A, Li K, Ye H, Jianhua Z (2020) Multistep-ahead forecasting of coal prices using a hybrid deep learning model. Resour Policy 65:101588
6. Ali AAS, Seker H, Farnie S, Elliott J (2018) Extensive data exploration for automatic price suggestion using item description: case study for the kaggle mercari challenge. In: Proceedings of the 2nd international conference on advances in artificial intelligence. ACM, pp 41–45
7. Amodei D, Ananthanarayanan S, Anubhai R, Bai J, Battenberg E, Case C, Casper J, Catanzaro B, Cheng Q, Chen G, et al (2016) Deep speech 2: End-to-end speech recognition in english and mandarin. In: International conference on machine learning, pp 173–182
8. Assaad M, Boné R, Cardot H (2008) A new boosting algorithm for improved time-series forecasting with recurrent neural networks. Inf Fusion 9(1):41–55
9. Bird S, Klein E, Loper E (2009) Natural language processing with Python: analyzing text with the natural language toolkit. O'Reilly Media, Inc
10. Bishop CM et al (1995) Neural networks for pattern recognition. Oxford University Press, Oxford
11. Carta S, Medda A, Pili A, Reforgiato Recupero D, Saia R (2019) Forecasting e-commerce products prices by combining an autoregressive integrated moving average (ARIMA) model and google trends data. Future Int 11(1):5
12. Cen Z, Wang J (2019) Crude oil price prediction model with long short term memory deep learning based on prior knowledge data transfer. Energy 169:160–171
13. Chan W, Jaitly N, Le QV, Vinyals O, Shazeer NM (2018) Speech recognition with attention-based recurrent neural networks. US Patent 9,990,918
14. Chen C, Li K, Teo SG, Chen G, Zou X, Yang X, Vijay RC, Feng J, Zeng Z (2018) Exploiting spatio-temporal correlations with multiple 3d convolutional neural networks for citywide vehicle flow prediction. In: 2018 IEEE international conference on data mining (ICDM). IEEE, pp 893–898
15. Chen J, Li K, Bilal K, Li K, Philip SY et al (2018) A bi-layered parallel training architecture for large-scale convolutional neural networks. IEEE Trans Parallel Distrib Syst 30(5):965–976
16. Chen J, Li K, Tang Z, Bilal K, Li K (2016) A parallel patient treatment time prediction algorithm and its applications in hospital queuing-recommendation in a big data environment. IEEE Access 4:1767–1783
17. Chitsaz H, Zamani-Dehkordi P, Zareipour H, Parikh PP (2018) Electricity price forecasting for operational scheduling of behind-the-meter storage systems. IEEE Trans Smart Grid 9(6):6612–6622
18. Chollet F et al (2015) Keras. https://keras.io

19. Di W, Sundaresan N, Piramuthu R, Bhardwaj A (2014) Is a picture really worth a thousand words? On the role of images in e-commerce. In: Proceedings of the 7th ACM international conference on Web search and data mining. ACM, pp 633–642
20. Dickey DA, Fuller WA (1979) Distribution of the estimators for autoregressive time series with a unit root. J Am Stat Assoc 74(366a):427–431
21. Duan M, Li K, Ouyang A, Win KN, Li K, Tian Q (2020) Egroupnet: a feature-enhanced network for age estimation with novel age group schemes. ACM Trans Multimed Comput Commun Appl 16(2):1–23
22. Ghani R (2005) Price prediction and insurance for online auctions. In: Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining. ACM, pp 411–418
23. Goldberg Y (2016) A primer on neural network models for natural language processing. J Artif Intell Res 57:345–420
24. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 770–778
25. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780
26. Hunter JD (2007) Matplotlib: a 2d graphics environment. Comput Sci Eng 9(3):90
27. Ioffe S, Szegedy C (2015) Batch normalization: accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167
28. Kalaiselvi N, Aravind K, Balaguru S, Vijayaragul V (2017) Retail price analytics using backpropogation neural network and sentimental analysis. In: 2017 fourth international conference on signal processing, communication and networking (ICSCN). IEEE, pp 1–6
29. Khedr AE, Yaseen N et al (2017) Predicting stock market behavior using data mining technique and news sentiment analysis. Int J Intell Syst Appl 9(7):22
30. Kim Y (2014) Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882
31. Kingma DP, Ba J (2014) Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980
32. Lago J, De Ridder F, Vrancx P, De Schutter B (2018) Forecasting day-ahead electricity prices in europe: the importance of considering market integration. Appl Energy 211:890–903
33. Law S, Paige B, Russell C (2018) Take a look around: using street view and satellite images to estimate house prices. arXiv preprint arXiv:1807.07155
34. Makridakis S (1993) Accuracy measures: theoretical and practical concerns. Int J Forecast 9(4):527–529
35. McKinney W, et al (2010) Data structures for statistical computing in python. In: Proceedings of the 9th python in science conference, vol 445. Austin, TX, pp 51–56
36. McNally S, Roche J, Caton S (2018) Predicting the price of bitcoin using machine learning. In: 2018 26th Euromicro international conference on parallel, distributed and network-based processing (PDP). IEEE, pp 339–343
37. Mercari price suggestion challenge (2018). https://www.kaggle.com/c/mercari-price-suggestion-challenge
38. Mikolov T, Chen K, Corrado G, Dean J (2013) Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781
39. Mikolov T, Sutskever I, Chen K, Corrado GS, Dean J (2013) Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems, pp 3111–3119
40. Nicholson D, Paranjpe R (2013) A novel method for predicting the end-price of eBay auctions. stanford
41. Noor K, Jan S (2017) Vehicle price prediction system using machine learning techniques. Int J Comput Appl 167(9):27–31
42. Oliphant TE (2006) A guide to NumPy, vol 1. Trelgol Publishing, New York
43. Pal N, Arora P, Kohli P, Sundararaman D, Palakurthy SS (2018) How much is my car worth? A methodology for predicting used cars' prices using random forest. In: Future of information and communication conference. Springer, pp 413–422
44. Pant DR, Neupane P, Poudel A, Pokhrel AK, Lama BK (2018) Recurrent neural network based bitcoin price prediction by twitter sentiment analysis. In: 2018 IEEE 3rd international conference on computing, communication and security (ICCCS). IEEE, pp. 128–132
45. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V et al (2011) Scikit-learn: machine learning in python. J Mach Learn Res 12:2825–2830
46. Pennington J, Socher R, Manning C (2014) Glove: global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp 1532–1543
47. Poursaeed O, Matera T, Belongie S (2018) Vision-based real estate price estimation. Mach Vis Appl 29(4):667–676
48. Pudaruth S (2014) Predicting the price of used cars using machine learning techniques. Int J Inf Comput Technol 4(7):753–764
49. Raykhel I, Ventura D (2009) Real-time automatic price prediction for eBay online trading. In: Twenty-first IAAI conference

50. Robertson S (2004) Understanding inverse document frequency: on theoretical arguments for IDF. J Document 60(5):503–520
51. Schuster M, Paliwal KK (1997) Bidirectional recurrent neural networks. IEEE Trans Signal Process 45(11):2673–2681
52. Seabold S, Perktold J (2010) Statsmodels: econometric and statistical modeling with python. In: 9th Python in science conference
53. Shastri M, Roy S, Mittal M (2019) Stock price prediction using artificial neural model: an application of big data
54. Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556
55. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R (2014) Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res 15(1):1929–1958
56. Tseng KK, Lin RFY, Zhou H, Kurniajaya KJ, Li Q (2018) Price prediction of e-commerce products through internet sentiment analysis. Electron Commerce Res 18(1):65–88
57. Vanstone BJ, Gepp A, Harris G (2018) The effect of sentiment on stock price prediction. In: International conference on industrial, engineering and other applications of applied intelligent systems. Springer, pp 551–559
58. Vaswani A, Bengio S, Brevdo E, Chollet F, Gomez AN, Gouws S, Jones L, Kaiser Ł, Kalchbrenner N, Parmar N et al (2018) Tensor2tensor for neural machine translation. arXiv preprint arXiv:1803.07416
59. Yang RR, Chen S, Chou E (2018) AI blue book: vehicle price prediction using visual features. arXiv preprint arXiv:1803.11227
60. Yang Z, Ce L, Lian L (2017) Electricity price forecasting by a hybrid model, combining wavelet transform, ARMA and kernel-based extreme learning machine methods. Appl Energy 190:291–305
61. Yao Y, Rosasco L, Caponnetto A (2007) On early stopping in gradient descent learning. Constr Approx 26(2):289–315
62. You Q, Pang R, Cao L, Luo J (2017) Image-based appraisal of real estate properties. IEEE Trans Multimed 19(12):2751–2759
63. Zeng D, Liu K, Lai S, Zhou G, Zhao J, et al (2014) Relation classification via convolutional deep neural network
64. Zhang GP (2003) Time series forecasting using a hybrid arima and neural network model. Neurocomputing 50:159–175

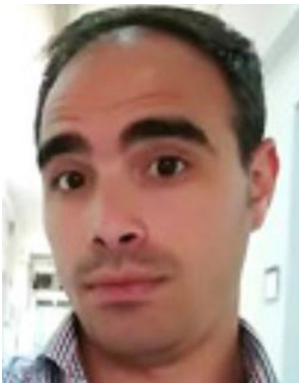
Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Ahmed Fathalla received the M.S. degree of Engineering in computer science and technology in 2015, Hunan University, Changsha, China. His research interests are mainly in machine learning.

Ahmad Salah received the PhD degree in computer science from Hunan University, China, in 2014. He received the masters degree in CS from Ain-shams University, Cairo, Egypt. He is currently an associate professor of Computer Science and Technology with Zagazig University, Egypt. He has published 21 papers in peer-reviewed journals, such as the IEEE Transactions on Parallel and Distributed Systems and IEEE-ACM Transactions on Computational Biology Bioinformatics, and ACM Transactions on Parallel Computing. His current research interests are parallel computing, computational biology, and machine learning.

Kenli Li received the PhD degree in computer science from Huazhong University of Science and Technology, China, in 2003. He was a visiting scholar at the University of Illinois at Urbana-Champaign from 2004 to 2005. He is currently a full professor of computer science and technology at Hunan University and deputy director of the National Supercomputing Center in Changsha. His major research areas include parallel computing, high-performance computing, and grid and cloud computing. He has published more than 130 research papers in international conferences and journals, such as IEEE Transactions on Computers, IEEE Transactions on Parallel and Distributed Systems, Journal of Parallel and Distributed Computing, ICPP, CCGrid. He is an outstanding member of CCF. He is a member of the IEEE and serves on the editorial board of the IEEE Transactions on Computers.

Keqin Li is a SUNY Distinguished Professor of computer science with the State University of New York. He is also a Distinguished Professor at Hunan University, China. His current research interests include cloud computing, fog computing and mobile edge computing, energy-efficient computing and communication, embedded systems and cyber-physical systems, heterogeneous computing systems, big data computing, high-performance computing, CPU-GPU hybrid and cooperative computing, computer architectures and systems, computer networking, machine learning, intelligent and soft computing. He has published over 690 journal articles, book chapters, and refereed conference papers and has received several best paper awards. He currently serves or has served on the editorial boards of the IEEE Transactions on Parallel and Distributed Systems, the IEEE Transactions on Computers, the IEEE Transactions on Cloud Computing, the IEEE Transactions on Services Computing, and the IEEE Transactions on Sustainable Computing. He is a Fellow of IEEE.



Piccialli Francesco is an Assistant Professor in the University of Naples FEDERICO II Department of Electrical Engineering and Information Technology. His research interests are focused on the Internet of Things (IoT) and Internet of Everything (IoE) paradigms. He is also focusing on research on data mining and data analytics techniques applied on data coming from IoT world. Consequently, topics of his research are certainly those of the smart environments, location-based and context-aware services and applications, which finalize their applications within the smart city framework.

Affiliations

Ahmed Fathalla[1,2] · Ahmad Salah[1,3] · Kenli Li[1] · Keqin Li[1,4] · Piccialli Francesco[5]

Ahmad Salah
ahmad@hnu.edu.cn; ahmad@zu.edu.eg

Keqin Li
lik@newpaltz.edu

Piccialli Francesco
francesco.piccialli@unina.it

1 College of Computer Science and Electrical Engineering, Hunan University, and the National Supercomputing Center in Changsha, Changsha, Hunan, China

2 Department of Mathematics, Faculty of Science, Suez Canal University, Ismailia, Egypt

3 Faculty of Computers and Informatics, Zagazig University, Sharkia, Egypt

4 State University of New York, New York, USA

5 University of Naples Federico II, Naples, Italy