Predicting Future Property Sales Using LSTM Time Series Model

Rozina M,
B00857003
Affiliation Author
Msc Artificial Intelligence
Ulster University

Predicting Future Property Sales Using LSTM Time Series Model

Rozina M,
Affiliation Author
Msc Artificial Intelligence
Ulster University

ABSTRACT: House price forecasting is a prominent issue, and independent researchers are actively adopting computational modeling or deep learning algorithms to conduct relevant investigations. Nevertheless, just because a few studies may not take into account all of the factors that influence home values, forecast findings are often not accurate. As a result, for home prediction, we offer an end-to-end simultaneous self-attention approach. We employ satellite maps to assess the conditions surrounding residences, and we input information about public amenities such as playgrounds, universities, and metro rapid transport hubs to depict the availability of services. We leverage attention processes that are often used in picture, voice, and interpretation tasks to determine key qualities that potential home buyers examine.

1. INTRODUCTION

When provided financial data, the algorithm may apply weights intelligently. Our suggested framework varies from self-attention approaches in that it takes into account the effect of two separate characteristics in order to properly understand relationship connecting features and improve accurate channel. Experiments are carried out to illustrate the precision. Real sales prices in property purchase data from 2007 to 2019, external users data from the Kaggle platform is taken, and satellite images crawled to use the Google Maps programming interface are among the scientific results.

These samples are used to train our prediction scenario and correlate it to the analysis of building deep learning-based techniques including Intense Gradient Enhancing as well as Light Gradient Enhanced system, deep learning, and several focus models. Property businesses and realtor experts must calculate number of times properties will be purchased in the coming year in condition for market forces in the property market to be harmonized. Accommodation (housing) is a basic requirement, together with physiological requirements such as oxygen, water supplies, and feeding. As a result, home may well be a speculative investment, as well as a social and psychological capital.

The assessment results gathered in our investigation will assist us to better understand the evolution of the economic growth of the country and therefore will plan for the future predictions for numerous businesses associated to the real estate sector. Furthermore, the goal of this study is to carry out a literature review in order to aid future research. We want to figure out what their underlying relationships are as well as keep improving learning algorithm.

Keywords: LSTM, House Price prediction, time series analysis, forecasting, deep learning models, neural networks, data estimation methodology

2. LSTM AND TIME SERIES ANALYSIS FOR HOUSE PRICE PREDICTION

Different communities of purchasers may be open to different aspects of a home. Nuclear group of families, for instance, may emphasize parks and nearby institutions for their kids. As a result, if the critical aspects of a property detected by a long short - term memory fit the demands of a potential buyer, the property can be suggested to the buyer. As a result, our ultimate study objective is to increase forecast accuracy as well as look into the factors that influence price outcomes. In our created LSTM model, we use an embedding and input large datasets.

These facts, as well as the approaches presented, not only enhance the accuracy of property price predictions, but highlight the elements that influence home pricing We not only anticipate housing values accurately, but we also pinpoint the factors that influence them.

As a result, we create a framework based on awareness that can understand the interplay between showcases and summarizes important characteristics for home buyers. We also devise unique approaches for extracting information. Characteristics from large datasets to aid model improvement performance.

We present a combined self-attention technique that evaluates two-hop links amongst distinct qualities to detect their underlying linkages and enhance model training effectiveness.

The ability to accurately estimate property sales is critical for utilizing the available resources in the property market. Nevertheless, predicting how many properties will be sold next year is extremely difficult for property corporations or property specialists.

Let's discuss the crucial terminologies from the following table:

Terminologies Used	**Meanings**
LSTM	Long short-term memory (LSTM) is a type of RNN and helps to learn order dependency in sequencing prediction tasks.
Time Series Forecasting	When you generate scientific forecasts given historical time stamped info, you're doing time series predicting. It entails developing models based on previous data and applying them to reach conclusions and guide future business choices.
Data Estimation	To design studies, evaluate data, and draw conclusions, data estimation seems to be a data processing approach that combines outcome measures, statistical significance, and specificity planning, with conceptual analysis.
Many-to-many sequence	The output is generated anytime each entry is received, because unlike one-to-one paradigm. That really is, a many-to-many sequence can identify each character in quite an input pattern.
MSE	The mean squared error, or MSE, is indeed a metric for determining how near a best - fit line is just to pieces of data.

Table 1: An outline of LSTM based terminologies that we will be using

3. METHODOLOGY

A data is divided into Training and Validation in estimate purposes. The volume of training plus test data is by far the most significant issue that arises in estimating techniques. Metadata should be used to build convolution neural networks even more than feasible. The testing set uses the data that was not utilized inside the training phase. The actual output results are calculated to the results produced by sending the test information in the network. Its main goal is to see if the ability to describe the collected sample is adequate.

Throughout the research, 70 percent training, and then 30 percent testing as well as 80 percent training set, and 20 percent test division of the given dataset is commonly used to determine training dataset. Out of these, 70% of such a given dataset was applied for learning as well as 30% for testing throughout this study.

A Long Short-Term Memory or the popularly called LSTM framework related to time sequences was used in this section of the research. This component was developed in Python and executed with KERAS libraries, a deep learning package being used create the LSTM network.

Recurrent networks, as is well documented, are a deep learning technique that produces superior results when the quantity of large datasets is excessive. Nevertheless, as mentioned underneath the literature title, there seem to be scenarios where a simple LSTM structure is learned with minimal data, although in a very tiny number of cases.

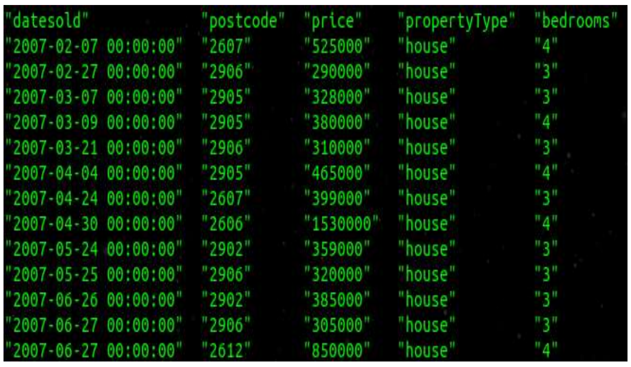
These resulting error measurements were obtained with varied epoch values even during learning algorithm. We will also be looking at the graphs on how well LSTM has worked.

4. IMPLEMENTATION OF LSTM

(a)Problem Statement – Many-to-many sequencing Prediction Problem

First, let's define the problem. We are given multivariate time series dataset of property sales for 2007-2019 periods. The data contains sales prices for houses and units with 1,2,3,4,5 bedrooms.

Here are some rows from this dataset



```
"datesold"               "postcode"  "price"    "propertyType"  "bedrooms"
"2007-02-07 00:00:00"   "2607"      "525000"   "house"         "4"
"2007-02-27 00:00:00"   "2906"      "290000"   "house"         "3"
"2007-03-07 00:00:00"   "2905"      "328000"   "house"         "3"
"2007-03-09 00:00:00"   "2905"      "380000"   "house"         "4"
"2007-03-21 00:00:00"   "2906"      "310000"   "house"         "3"
"2007-04-04 00:00:00"   "2905"      "465000"   "house"         "4"
"2007-04-24 00:00:00"   "2607"      "399000"   "house"         "3"
"2007-04-30 00:00:00"   "2606"      "1530000"  "house"         "4"
"2007-05-24 00:00:00"   "2902"      "359000"   "house"         "3"
"2007-05-25 00:00:00"   "2906"      "320000"   "house"         "3"
"2007-06-26 00:00:00"   "2902"      "385000"   "house"         "3"
"2007-06-27 00:00:00"   "2906"      "305000"   "house"         "3"
"2007-06-27 00:00:00"   "2612"      "850000"   "house"         "4"
```

Figure 1: Head of sales price dataset

1. Datesold-Date on which this property was sold
2. Postcode-4 digit postcode of the suburb where the property was sold
3. Price -Price for which the property was sold

Property type and bedroom parameters are self-explanatory. Dataset consists of 28000 observations. Our aim is to predict the prices of 3 future sales using information about previous sold property (20 previous time steps). Before building a prediction model, let's visualize data

As part of the dataset, the details like postcode, price for which the property was sold, number of bedrooms with each individual property and the property type, whether it's a house or unit were provided. More information here House Property Sales Time Series | Kaggle

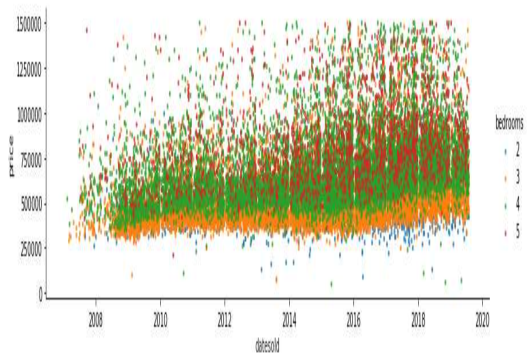


Figure 2: total number of bedrooms(defined by color), price(y-axis),datesold(x-axis)

(b)LSTM Neural Network Architecture

To accomplish the task defined above, Recurrent Neural Networks are used. This type of networks are well-fitted for solving vanishing gradient problems.Also they are able to handle long term dependency problems.

Recently, LSTMs have been successfully applied for a number of many-to-many type sequence prediction problems, which include areas like Speech Recognition, Machine Translation, Language modeling, Stock prediction, Game Learning.

Particular LSTM model used for price prediction has following architecture

- ✓ LSTM Layer 1. Receives input of shape (timesteps,5) (previous window of observations) and produces a sequence of 32-dimensional vectors
- ✓ LSTM Layer 2 Returns a single vector of dimension 64
- ✓ Layer 3. Dense (fully connected) layer with output dimension equal to 16(size of future predictions) and ReLu activation
- ✓ Layer 4. Dense (fully connected) layer with output dimension equal to 3(size of future predictions) and linear activation

The usual Network architecture of LSTM looks like just the following one:

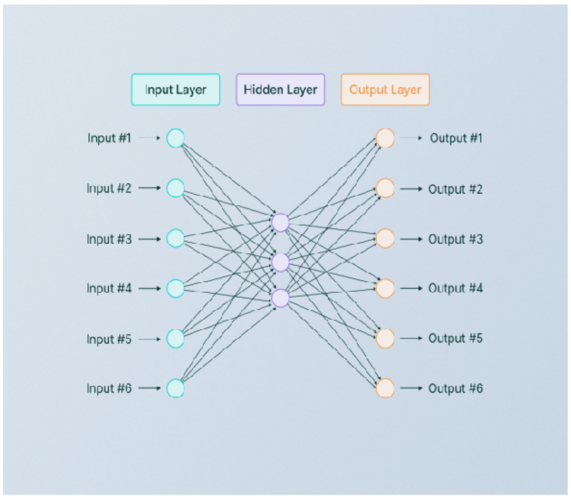


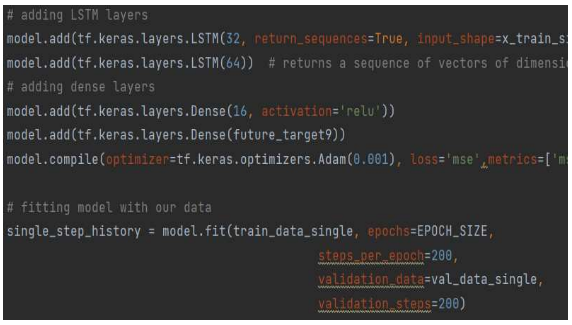
Figure 3: LSTM network

(c) Evaluation Metrics

Evaluation metrics used for this problem is Mean Squared Error.

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2$$

This helps calculate the metrics using the sold house price prediction and the actual price in the listing.



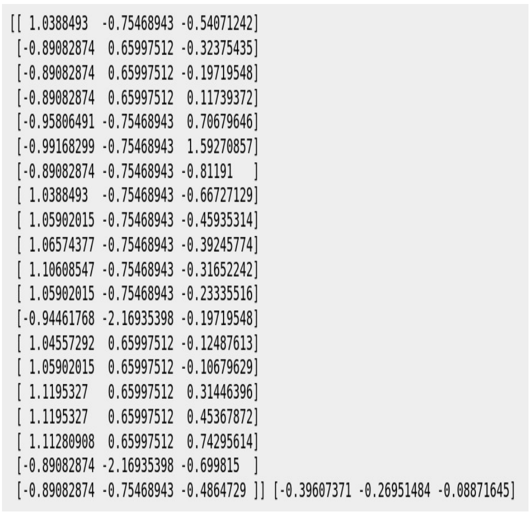
```
# adding LSTM layers
model.add(tf.keras.layers.LSTM(32, return_sequences=True, input_shape=x_train_s
model.add(tf.keras.layers.LSTM(64))  # returns a sequence of vectors of dimensi
# adding dense layers
model.add(tf.keras.layers.Dense(16, activation='relu'))
model.add(tf.keras.layers.Dense(future_target9))
model.compile(optimizer=tf.keras.optimizers.Adam(0.001), loss='mse',metrics=['m

# fitting model with our data
single_step_history = model.fit(train_data_single, epochs=EPOCH_SIZE,
                                steps_per_epoch=200,
                                validation_data=val_data_single,
                                validation_steps=200)
```

Figure 4: MSE analysis

(d) Performance Analysis Of The LSTM Model

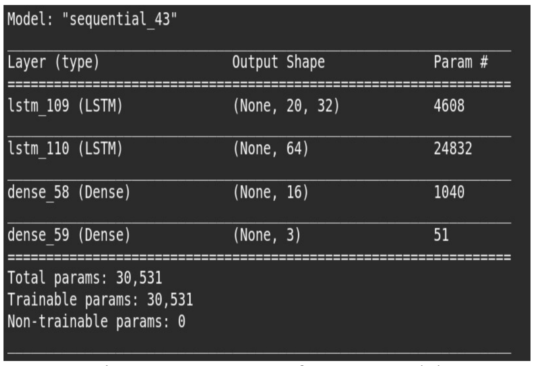
All data is divided into two sets.80 % was used for training and 20 % was used for testing. Before going into network data is normalized with the mean and standard deviation (obtained from training data), because it helps with model convergence and stability.



```
[[ 1.0388493  -0.75468943 -0.54071242]
 [-0.89082874  0.65997512 -0.32375435]
 [-0.89082874  0.65997512 -0.19719548]
 [-0.89082874  0.65997512  0.11739372]
 [-0.95806491 -0.75468943  0.70679646]
 [-0.99168299 -0.75468943  1.59270857]
 [-0.89082874 -0.75468943 -0.81191   ]
 [ 1.0388493  -0.75468943 -0.66727129]
 [ 1.05902015 -0.75468943 -0.45935314]
 [ 1.06574377 -0.75468943 -0.39245774]
 [ 1.10608547 -0.75468943 -0.31652242]
 [ 1.05902015 -0.75468943 -0.23335516]
 [-0.94461768 -2.16935398 -0.19719548]
 [ 1.04557292  0.65997512 -0.12487613]
 [ 1.05902015  0.65997512 -0.10679629]
 [ 1.1195327   0.65997512  0.31446396]
 [ 1.1195327   0.65997512  0.45367872]
 [ 1.11280908  0.65997512  0.74295614]
 [-0.89082874 -2.16935398 -0.699815  ]
 [-0.89082874 -0.75468943 -0.4864729 ]] [-0.39607371 -0.26951484 -0.08871645]
```

Figure 5: An example of normalized input window and target predictions (in the lower right corner)

Data with different mean/variance structure can cause serious network stability issues.



```
Model: "sequential_43"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_109 (LSTM)              (None, 20, 32)            4608
_________________________________________________________________
lstm_110 (LSTM)              (None, 64)                24832
_________________________________________________________________
dense_58 (Dense)             (None, 16)                1040
_________________________________________________________________
dense_59 (Dense)             (None, 3)                 51
=================================================================
Total params: 30,531
Trainable params: 30,531
Non-trainable params: 0
_________________________________________________________________
```

Figure 6: Summary of LSTM Model

Here are some of the prediction examples:

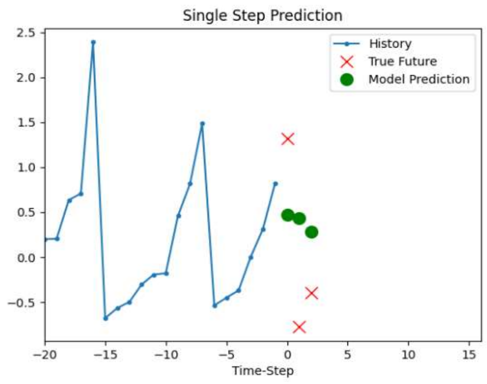


Figure 7: prediction 1

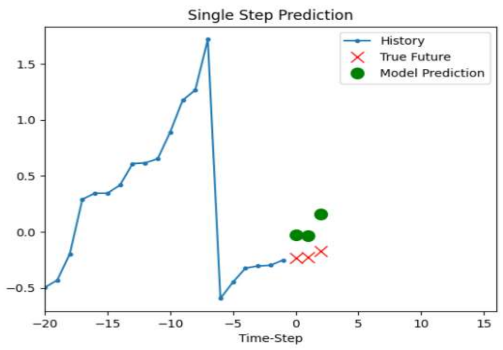


Figure 8: prediction 2

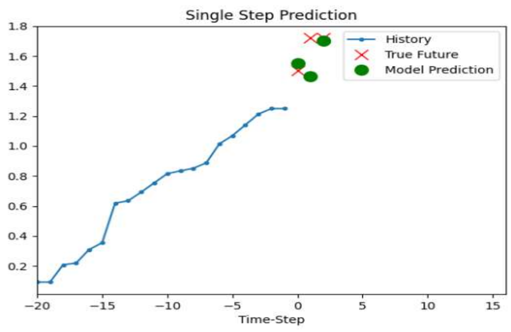


Figure 9: prediction 3

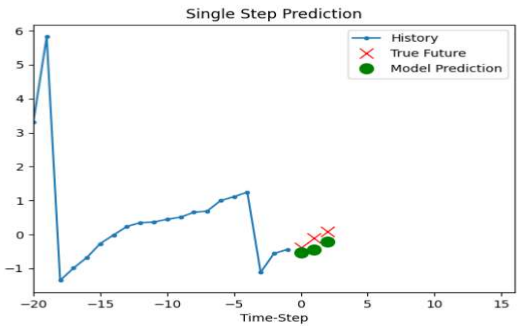


Figure 10: prediction 4

Plot of training loss is predicted below. As we can see, it is very noisy, although it is converging.

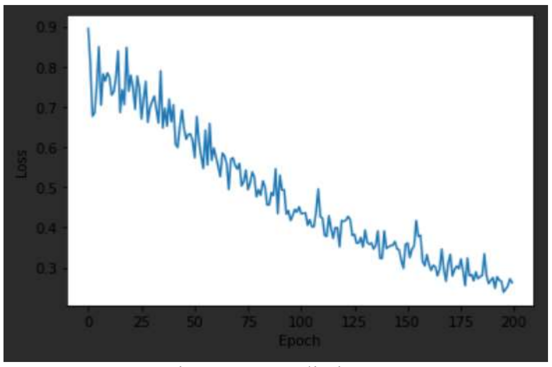
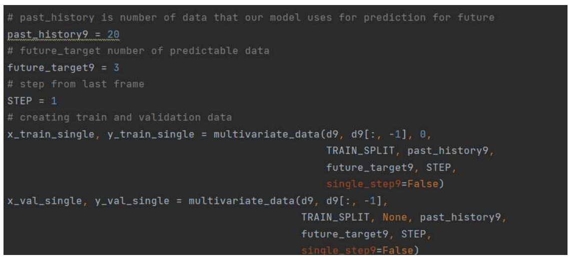


Figure 11: prediction 5

5. CODE SNIPPETS WITH RESULTS

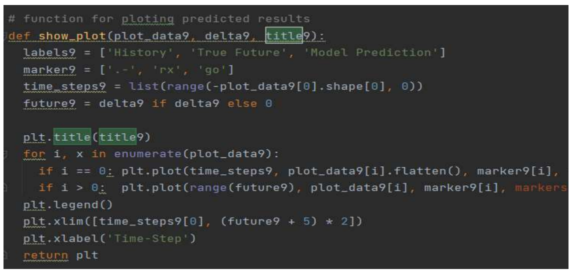
Past history for modeling the data is shown below



```
# past_history is number of data that our model uses for prediction for future
past_history9 = 20
# future_target number of predictable data
future_target9 = 3
# step from last frame
STEP = 1
# creating train and validation data
x_train_single, y_train_single = multivariate_data(d9, d9[:, -1], 0,
                                                   TRAIN_SPLIT, past_history9,
                                                   future_target9, STEP,
                                                   single_step9=False)
x_val_single, y_val_single = multivariate_data(d9, d9[:, -1],
                                               TRAIN_SPLIT, None, past_history9,
                                               future_target9, STEP,
                                               single_step9=False)
```

Figure 12: past history

Plotting predicted results is shown below

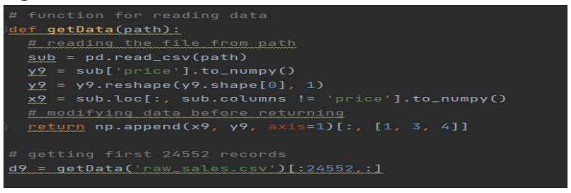


```
# function for ploting predicted results
def show_plot(plot_data9, delta9, title9):
  labels9 = ['History', 'True Future', 'Model Prediction']
  marker9 = ['.-', 'rx', 'go']
  time_steps9 = list(range(-plot_data9[0].shape[0], 0))
  future9 = delta9 if delta9 else 0

  plt.title(title9)
  for i, x in enumerate(plot_data9):
    if i == 0: plt.plot(time_steps9, plot_data9[i].flatten(), marker9[i],
    if i > 0:  plt.plot(range(future9), plot_data9[i], marker9[i], markers
  plt.legend()
  plt.xlim([time_steps9[0], (future9 + 5) * 2])
  plt.xlabel('Time-Step')
  return plt
```

Figure 13: predicted results

Reading the dataset is shown in the below figure

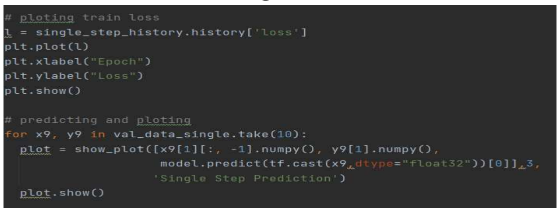


```
# function for reading data
def getData(path):
  # reading the file from path
  sub = pd.read_csv(path)
  y9 = sub['price'].to_numpy()
  y9 = y9.reshape(y9.shape[0], 1)
  x9 = sub.loc[:, sub.columns != 'price'].to_numpy()
  # modifying data before returning
  return np.append(x9, y9, axis=1)[:, [1, 3, 4]]

# getting first 24552 records
d9 = getData('raw_sales.csv')[:24552,:]
```

Figure 14: dataset loading

Plot of training loss is in code below.

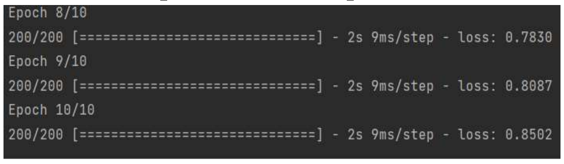


```
# ploting train loss
l = single_step_history.history['loss']
plt.plot(l)
plt.xlabel("Epoch")
plt.ylabel("Loss")
plt.show()

# predicting and ploting
for x9, y9 in val_data_single.take(10):
  plot = show_plot([x9[1][:, -1].numpy(), y9[1].numpy(),
                    model.predict(tf.cast(x9,dtype="float32"))[0]],3,
                   'Single Step Prediction')
  plot.show()
```

Figure 15: training loss

200 epochs in the output screen below



```
Epoch 8/10
200/200 [==============================] - 2s 9ms/step - loss: 0.7830
Epoch 9/10
200/200 [==============================] - 2s 9ms/step - loss: 0.8087
Epoch 10/10
200/200 [==============================] - 2s 9ms/step - loss: 0.8502
```

Figure 16: 200 epochs

6. CONCLUSION

We construct a consistent model that provides housing forecasting in this work. We offer an awareness technique to dynamically allocate weights as per distinct attributes or examples in this system, which imports large datasets to enrich home data. Given the time constraints, the amount of sales throughout this research was approximated on a nationwide but instead of city level and also as monthly statistics. Rather than evaluating the entire nation due to time constraints throughout this study, house sales might be estimated to every city or part of the country employing categorization criteria like as its first or super cheap sales, or pricing to residents or immigrants.

7. REFERENCES

1. M. Jaderberg, K. Kavukcuoglu, K. Simonyan, A. Zisserman, ''Spatial transformer networks,'' in Proc. Advance. Neural Info. Syst., 2015, pp. 2017–2025.
2. R. Gupta, A. Kabundi, and S. M. Miller, ''Forecasting the US real house price index: Structural and non-structural models with and without fundamentals,'' Econ. Model., vol. 28, no. 4, pp. 2013–2021, Jul. 2011.
3. https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9395585
4. https://bme.vgtu.lt/index.php/JBEM/article/view/10190
5. https://www.kaggle.com/datasets/htagholdings/property-sales?select=raw_sales.csv