House Price Estimation Using a Combined Deep Learning Model

Sirish Prabakar, Dane Jew

House Price Estimation Using a Combined Deep Learning Model

Abstract

Deep Learning models are sometimes required to solve and predict complex real world problems. This is because the data or feature used to generate the analysis can come in many diverse forms. A model may be asked to analyze numeric, text, visual, and even audio data. It is necessary at times to create a model that can handle multiple types of data. To that end we will take a look at an example of this in the prediction of home prices based on both numeric data, such as number of bedrooms, and image data, such as photos of the house.

This project will attempt to create a composite model using Neural Networks and Convolution NN to predict house prices based on numeric and image data.

Problem Statement

The goal of this project is to create a home price prediction system using a composite model to use both numeric data and images. The composite model will consist of a Convolution Neural Network and a Fully Connected Neural Network.

Methodology

In order to create a model that will accept both numeric and image data, it must combine two separate models for each type of data. The first model, a Fully Connected Neural Network, will be used to analyze the numeric data. The numeric data includes the number of bedrooms, number of bathrooms, the area (or size of the property), and the Zip Code. (See Figure 1.)

The other model is a Convolution Neural Network used to analyze a collection of images in jpeg (jpg) format. Each photo is prefixed by a number which corresponds with the numeric data used for the other model. The images are name #_bathroom.jpg, #_bedroom.jpg, #_frontal.jpg, and #_kitchen.jpg. The number prefix is used when all four photos for each record are merged into one photo. (See Figure 2)

Figure 1.
Home Features Dataset

	Bedrooms	Bathrooms	area	zipcode	price
0	4	4.0	4053	85255	869500
1	4	3.0	3343	36372	865200
2	3	4.0	3923	85266	889000
6	3	4.0	2544	85262	799000
10	5	5.0	4829	85266	519200

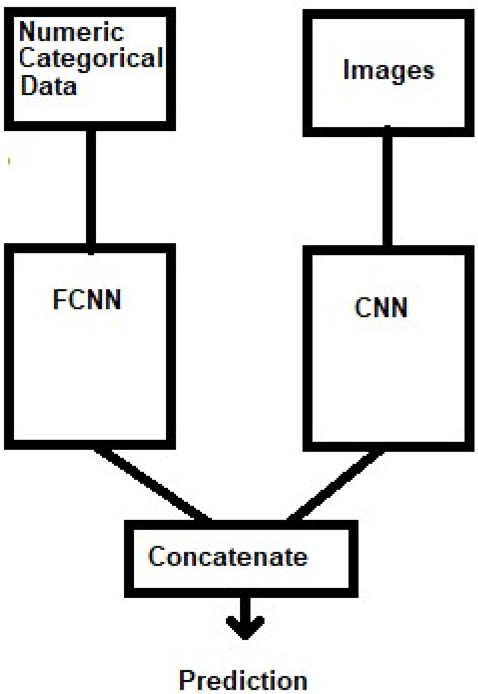
Figure 2.
Image Dataset

	bathroom_img	bedroom_img	frontal_img	kitchen_img
0	1_bathroom.jpg	1_bedroom.jpg	1_frontal.jpg	1_kitchen.jpg
1	2_bathroom.jpg	2_bedroom.jpg	2_frontal.jpg	2_kitchen.jpg
2	3_bathroom.jpg	3_bedroom.jpg	3_frontal.jpg	3_kitchen.jpg
3	4_bathroom.jpg	4_bedroom.jpg	4_frontal.jpg	4_kitchen.jpg
4	5_bathroom.jpg	5_bedroom.jpg	5_frontal.jpg	5_kitchen.jpg

Figure 3
Merged Image

Both the FCNN and the CNN models will be merged using the Keras Functional API which provides a robust and flexible tool set to create complex models.

Figure 4.
Home Price Prediction Model



Experimental Results and Analysis

The model was generated but there was great difficulty in generating the correct output. During training the data loss and error does not seem to converge.

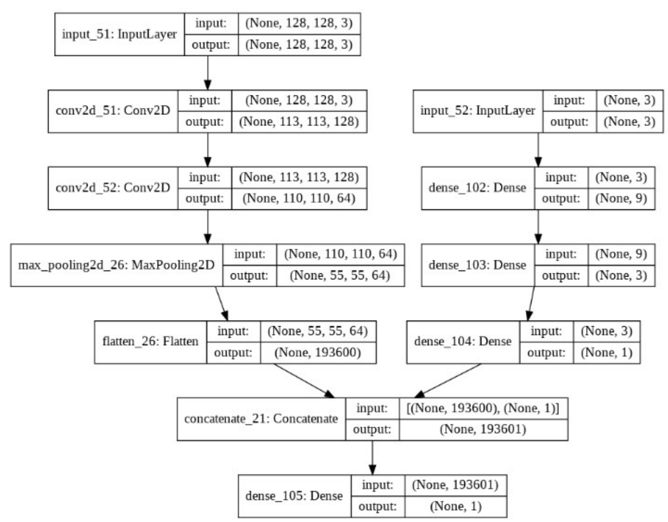
Figure 5

Model Summary

```
Model: "model_26"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_51 (InputLayer)           (None, 128, 128, 3)  0
__________________________________________________________________________________________________
conv2d_51 (Conv2D)              (None, 113, 113, 128 98432       input_51[0][0]
__________________________________________________________________________________________________
input_52 (InputLayer)           (None, 3)            0
__________________________________________________________________________________________________
conv2d_52 (Conv2D)              (None, 110, 110, 64) 131136      conv2d_51[0][0]
__________________________________________________________________________________________________
dense_102 (Dense)               (None, 9)            36          input_52[0][0]
__________________________________________________________________________________________________
max_pooling2d_26 (MaxPooling2D) (None, 55, 55, 64)   0           conv2d_52[0][0]
__________________________________________________________________________________________________
dense_103 (Dense)               (None, 3)            30          dense_102[0][0]
__________________________________________________________________________________________________
flatten_26 (Flatten)            (None, 193600)       0           max_pooling2d_26[0][0]
__________________________________________________________________________________________________
dense_104 (Dense)               (None, 1)            4           dense_103[0][0]
__________________________________________________________________________________________________
concatenate_21 (Concatenate)    (None, 193601)       0           flatten_26[0][0]
                                                                 dense_104[0][0]
__________________________________________________________________________________________________
dense_105 (Dense)               (None, 1)            193602      concatenate_21[0][0]
==================================================================================================
Total params: 423,240
Trainable params: 423,240
Non-trainable params: 0
__________________________________________________________________________________________________
None
```

Figure 6

Composite Model



Task Division and Project Reflection

The work breakdown is as follow in the table below:

Coding		
	Work Pct	
Dane Jew	50%	
Sirish Prabakar	50%	

Documentation		
	Work Pct	
Dane Jew	50%	
Sirish Prabakar	50%	

Challenges

One of the challenges is preprocessing the data, especially associating the numeric/categorical data with the image data. This get more tricky with the split between training and test data. Fortunately the train_test_split tool provided by SKLearn can be leveraged to preserve the order of both datasets, thus preserving the association of rows from both sets.

Google Colab also presents a challenge when reading a file from the file system. It is tricky for the notebook to read files form the google drive which has a complex file system or github which requires a generated link. Uploading a file from the local file system is the easiest way to read data, but it is time consuming depending on the size of the data.

Due to time constraints there was little time to perform any parameter tuning.

Conclusion

The composite model has promise but due to it's complexity it is difficult to design and maintain. The ability to classify or predict based on various types of input is very valuable. It is hope in the future when there is more time to be able to explore this further.