Nonlinear Regression model for predicting house prices in California

[1]Komal Barge
Computer Science
Lakehead University
Thunder Bay, Canada
bargek@lakeheadu.ca

***Abstract*—In this machine learning paper, I analyzed the real estate property prices in California. The information on the real ´estate listings was extracted from Luís Torgo's page (University of Porto). I have predicted the prices of real estate properties based on features such as geographical location, living area, and number of rooms, households, population, income etc. I have performed nonlinear regression using deep convolutional neural network(DCNN) which has given the accuracy of around 84% in training and 77% in testing. This model architecture is an inspiration of AlexNet DCNN design with an additional convolutional layer. I will present the details of the model design, experimental analysis and performance metrics in this paper.**

***Index Terms*—Convolutional layer, Neural Network, regression, deep**

I. Introduction

Over the last few years, Artificial Neural Networks has become essential and one of the most dominant tools as it facilitates in handling massive amounts of data. Convolutional Neural Network (CNN), a class of deep neural networks is designed to recognize, analyze and classify images enabling deep learning for computer vision. CNN can be utilized for classification as well as regression learning methods. CNN is broadly classified into main parts: a convolution/pooling layer and a fully connected (FC) layer. In image classification, the former is responsible for breaking up the image into tiny chunks and analyzing them and the results of these are fed as an input to a FC layer in predicting the image. Whereas, in regression problems, the layers act in same way but results in a real value at the end instead of giving probabilities like classification models.

Regression models have been out there for several years and have proved to be very useful in modeling real-world problems and providing useful forecasts, in both science and industrial environments and industry. At the same time, neural networks and deep learning are that in popularity, and can compute complicated problems and provide results that mimic the human brain's process of learning. A standard CNN relies on breaking up the data into smaller intricate details and analyze them in isolation helping in making an informed decision about the instance in its entirety. Convolution layer acts as a filter passing over the data matrix, creating a feature map aiding in feature extraction. This is then down sampled in the pooling layer, extracting only the most important and relevant information. Fully connected layer (Linear layer) takes the output of the previous layers, flattens them and turns them into a single vector, applies weights to predict the final real value.

II. Literature Review

Prices of real estate properties is critically linked with our economy [1]. Despite this, we do not have accurate measures of housing prices based on the vast amount of data available. A property's appraised value is important in many real estate markets such as sales, loans, and its marketability. Usually, property prices estimates are often concluded by professional appraisers. The disadvantage of this method is that the appraiser is likely to be influenced due to vested interest from the mortgage broker, lender, buyer, or seller. Therefore, an automated prediction system can serve as an unbiased third-party source that may be less biased. An automated price prediction system may be useful for buyers of real estate properties to identify under/overpriced properties currently on the market.

For first time buyers with very little experience, this can be helpful and recommend purchasing bid approaches for buying properties. Nissan Pow et al [1] analyzed the real estate property prices in Montreal. They used and evaluated different regression methods such as linear regression, Support Vector Regression (SVR), k-Nearest Neighbours (kNN), and Regression Tree/Random Forest Regression. They predicted the asking price with an error of 0.0985 using an ensemble of kNN and Random Forest methods. They also present the details of the analysis, and the testing and validation results for the different algorithms [1].

III. Proposed Model

Regression problem analysis helps in modelling the relationship between one dependent variable (which you are trying to estimate) and one or more independent variables (the model input). Regression analysis may show if an important relationship exists between the independent variables and the dependent variable, and the strength of the impact - by how much you can anticipate the dependent variable to change when the independent variables move. The simplest, linear regression equation as shown in Fig. 1.

$$\gamma = \beta_1 + \beta_2 X_2 + \beta_3 X_3 + \cdots + \beta_k X_k + \varepsilon$$

Fig. 1. Linear Regression equation.

y - dependent variable—the target value the regression model to predict.

X2,3..k - independent variables—one or more values that the model takes as an input, using them to estimate the dependent variables.

[beta]1,2,3..k - Coefficients—these are weights that state how crucial each of the variables is for predicting the dependent variable.

[error]Error—the distance between the value estimated by the model and the actual dependent variable y. Statistical methods can be used to estimate and reduce the size of the error term, to improve the predictive power of the model.

A. Regression model using DCNN

Regression analysis have several types such as linear, polynomial, logistic, stepwise etc. Neural networks are reducible to regression models — For example, this very simple neural network is equivalent to a logistic regression‘, with input neurons, hidden neurons and output neurons. It takes numerous input parameters, multiplies them by their weights, and runs them through batch normalization, ReLU activation and max pooling layer as shown in figure Fig. 2.

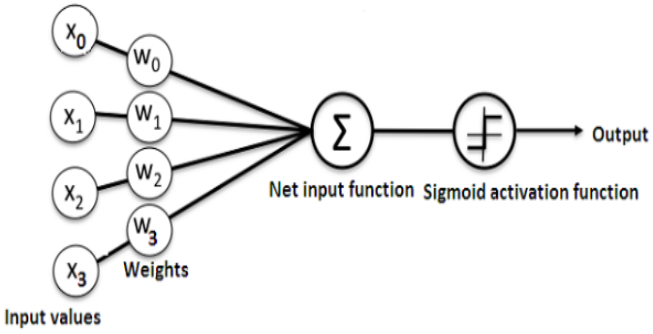

Fig. 2. Simplified Convolutional Neural Network.

B. Inspiration of the model

There are various DCNN architectures available such as AlexNet, DenseNet, ResNet, GoogleNet etc. For this nonlinear model, I picked up the network layer architecture idea from AlexNet, which consists of 5 convolutional and 3 fully connected layers [4]. In the proposed model, I have added 6 one dimensional(1D) convolutional layers followed up with 1D batch normalization, ReLU activation, max pooling layers similar to the AlexNet and after flattening the output of the 6th convolutional layer, I have added 3 fully connected linear layers and an output layer which gives an estimated value by taking 32 input parameters. First convolutional layer is of 8x128 input with a kernel size of 1 and with a stride of 1. The second convolutional layer takes input as the (response-normalized and pooled) output of the first convolutional layer. The third, fourth, fifth and sixth convolutional layers are connected to one another with intervening pooling, ReLU as an activation and normalization layers and has a size of 128x128 connected to the (normalized, pooled) outputs. Proposed model is shown in figure Fig. 3.

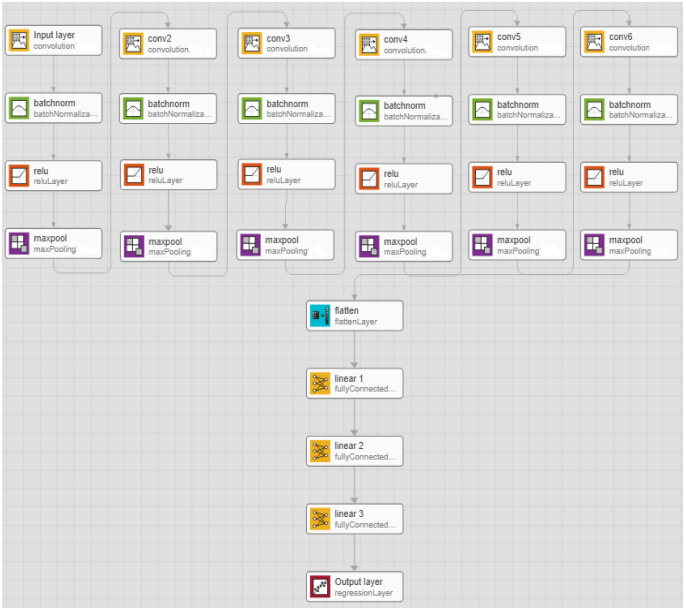

Fig. 3. Proposed Nonlinear Regression Convolutional Neural Network.

C. Details about dataset and model linearity

The California housing dataset consists of 8 features that are longitude, latitude, housing median age, total number of rooms, total number of bedrooms, population, number of households, and median income which are fed as an input parameter to the first convolutional network. When this neural network is learned, gradient descent will be performed until it reaches the optimal weights for the model to find coefficients that are better and fit the data. As I have used ReLU function, it helps the network to converge very quickly. Although, it looks like a linear function, ReLU has a derivative function and allows for backpropagation. Because of this activation function, this solution model becomes nonlinear regression model.

D. Optimization function

While training the regression model, We have used Adaptive Moment Estimation (Adam) as an optimization function. It computes adaptive learning rates for each parameter. Adam performs well and compares favorably with other adaptive learning-method algorithms because it converges very quickly and the Model’s learning speed is quiet Fast and efficient and it also corrects any problem faced by other optimization techniques such as vanishing learning rate, slow convergence or high variance in parameter updates that lead to fluctuating loss function [2]. Optimization algorithms helps in minimizing

an objective function which is simply a mathematical function dependent on the Model's internal learnable parameters which are used in computing the target values(Y) from the set of inputs(X) used in the model. Internal parameters such as weights(w) and bias(b) values in neural network are learned and updated in a direction of optimizing loss function. By using the combination of these learning parameters, it has performed better.

IV. Experimental Analysis

In this experimental model, Firstly, the housing dataset has been split into 70:30 ratio. This dataset is having some incomplete entries which I have removed before splitting the dataset. While designing the model, I tried to tune many hyperparameters as well as changed the layers and modified them accordingly.

A. Initial model design

In the very first experiment, I used 1 convolutional, 1 FC and Stochastic Gradient Descent (SGD) as a loss function, this model design performed poorly on both training and testing sets [3]. In other words, the model was underfitting the dataset. Even, after changing the hyperparameters in convolutional layers or increasing the number of layers, did not help in improving the performance. After adding the layers this design, it provided accuracy around 20%. After addition of 5 more convolutional layers, it gave the accuracy of around 35% whereas, further addition of layers did not improve the accuracy.

B. Vanishing/exploding gradient issue

After altering the batches per epoch that I have sent as 64, the accuracy does not vary significantly. But, one thing that I noticed is if you increase the batches per epoch, the model will surely learn better in less number of epochs. In later experiment, I kept those few convolutional layers but changed loss function from SGD to Adam. That's when the model achieved better performance i.e. around 50% in less number of epochs. For this dataset, SGD was trying to vanish the gradient whereas, Adam provided fast convergence and Adam outperformed SGD technique.

C. Number of trainable parameters

After changing the learning rate parameters by 0.01 or 0.1, accuracy did not vary much. But, after using 1e-6 as a learning parameter, model started exploding the gradient. So, I kept the learning parameter constant to 0.001.

D. Additional layers

Application of Batch Normalization layer after convolutional layer and keeping the dimensions of input and output features constant to 128 in all the convolutional layers were helpful in achieving the accuracy of around 79% in testing.

E. Kernel size

Variations in kernel size was not giving any exceptional performance, So, kept kernel size constant to 1 with stride rate as 1.

F. Number of epochs and Inference time

Number of epochs alterations gave some better insights but going after 300 epochs is not providing any better results. Increasing the number of epochs were taking lot of inference time because of the many convolutional layers.

V. Conclusion

This price prediction experimental analysis has been done by using a California based housing dataset with 20641 instances and 8 important attributes. In this paper, I have proposed a nonlinear regression neural network model which performs better with r-squared value of around 0.77 and L1loss of around 38000. In this neural network, Adam optimization based model has outperformed SGD based model design by 10%. By keeping the convolutional input-output features dimensions same across most of the layer and using batch normalization layer has given far better results for r-squared value.

VI. Appendix

The dataset used for this project can be downloaded from this link:
https://github.com/ageron/handson-ml/tree/master/datasets/housing

The github link for the code can be found at this below link:
https://github.com/komalbarge45/NaturalLanguageProcessing/blob/master

References

[1] Nissan Pow, Emil Janulewicz, Liu (Dave) Liu, "Applied Machine Learning Project 4 Prediction of real estate property prices in Montreal"

[2] Neural Network Optimization concepts : https://missinglink.ai/guides/neural-network-concepts/hyperparameters-optimization-methods-and-real-world-model-management/

[3] Underfitting and Overfitting problem : https://www.deeplearning-academy.com/p/ai-wiki-overfitting-underfitting

[4] Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks", Advances in neural information processing systems, 1097-1105