NLP (COMP5413) Assignment 1:

Non Linear Regression With Deep Learning

Khushal Paresh Thaker
Dept. of Computer Scinece
Lakehead University
Thunder Bay, Canada
kparesh@lakeheadu.ca
1106937

Abstract—***Pricing of a house is influence by various factors. Manually predicting the value of the house might have significant errors. With deep learning growing, methods from it can be used to predict the prices. The paper summarizes the method used to load the California housing data set, plot each feature of the dataset on separate sub plots and predict the median house value by using a non linear regression model with a 1D convolution based neural network.***

Index Terms-- Convolution neural network, non linear regression model, natural language processing, max pooling layer

I. INTRODUCTION

The advent of technology has led to machines learning new ways to perform tasks that human can perform, but quicker. Recently, the domain of machine learning has gained new insights in training model and predicting values which is quite difficult and error prone when performed manually. Real estate has gained a lot of attention off lately, being a vital part of any country's economy. With increasing population, the need for houses increases. To meet the demands, the prices of houses have to be accurate so as to attract potential buyers. The concept of machine learning and NLP can be applied to this domain in order to predict cost of a house making humans decision making easy and trust worthy. Price of a house depends on a lot of factors, both internal and external.

In order for predicting the house value, a dataset is required. Therefore, California housing dataset is used for the model provided in the paper. The model used is a non linear regression model with a 1D Convolution based neural network. Non linear regression model is a model that presents a relationship between a dependent variable and an independent variable from a data which is non linear. A non linear regression model is applied because it is more suitable than linear, as the correlation coefficient between the independent and dependent variables shows a linear tendency. The California housing dataset contains 20640 records and 10 attributes with both numeric and non numeric features.

II. BACKGROUND

Before CNN was introduced, deep learning architecture such as Autoencoder was used wherein the dimension reduction occurred efficiently[5]. Multi layer perceptron(MLP) performs dimension reduction in information processing. The only difference between autoencoder and MLP is that AE recreates the input while MLP uses certain inputs to predict the target output. Time delay neural network(TDNN), the motivation behind the development of CNN, reduces computation by having the weights in temporal dimension. As CNN has replaced matrix multiplication with convolutions, the weights have reduced and so the efficiency increased.

A Convolution neural network contains shifting convolutional and pooling layers with a convolution filter to perform feature extraction[1]. CNN model typically contains small filters on input data. The pooling layer reduces the computational cost of the learning process and also reduces overfitting[4]. CNN incorporates the concept of parameter sharing wherein, only one set of parameters are enough for it to be learned instead of learning at every location[5]. Fewer parameters mean less time to compute and hence increases efficiency. The final layer will be the multi layer perceptron which basically converts the input(extracted features) into the output as shown in fig 1. The advantage of CNN is that it can be combined into various deep learning architectures where the input of another convolutional layer is the output of the current CNN.

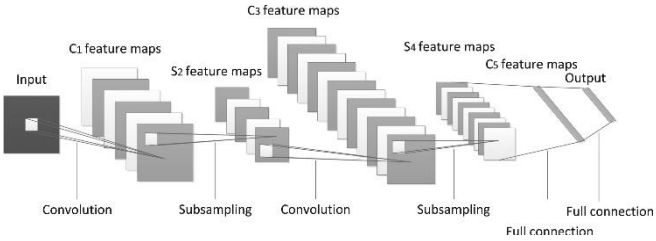


Fig 1: Schematic CNN

California housing dataset was used to predict the median house value using the ensemble learning based prediction model[3] where ensemble models like random tree are used and compared with others. [4] uses a CNN model with self-adaptability to predict the stock price.

III. Proposed Model

The proposed model is a multi layer, 1D convolution network to perform regression on California housing dataset with PyTorch in Google Colab. A modified dataset of California housing is downloaded and uploaded onto the drive so that it can be fetched on to the code. The model is split 70:30 for training and testing respectively using the utility 'model_selection' from sklearn library. The model is then processed to remove any incomplete entries. The column 'median_house_value' is being predicted by using the other columns from the dataset. The class defined for initialization method will be a sub class to the 'torch.nn.Module' class which is the base class for all neural network classes.

In the initialization method, we define the input layer and store the parameters for it. Kernel size, which is the number of filters in the convolutional layer, is defined along with the max pooling layer. Max pooling layer is used to reduce the total number of layers. There is also a average pooling layer, which isn't the best for this model and hence max pooling layer is used. Flatten layer is defined, which is used to convert the data into a 1D array so that it can be passed as an input to the next layer. An activation function, ReLU (Rectified Linear Unit) is used to output the input directly if positive or output zero, if negative[2].

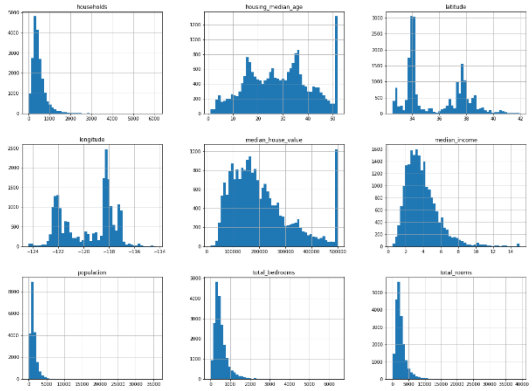


Fig 2: Histogram Representation

Each feature of the dataset is unique and can be represented in many ways. One such way is a histogram plot shown in Fig 2. To optimize the model, Stochastic Gradient Descent (SGD) is used wherein, it takes the learning rate and the parameter that has to be optimized as its parameter to perform updates. The model makes use of L1Loss function to reduce the loss and R2Score, which is the coefficient of determination, is the performance measure to determine the closeness of data to the regression line. Batch size impacts the efficiency of training and also the noise level of the gradient estimate. For this model, the batch size is maintained at 64 after comparing with that of 128 size.

Fig 3 represents each feature of the California housing dataset namely, longitude, latitude, housing_median_age, total_rooms, total_bedroom, population, household, median_income and median_house_value on separate subplots.

The table 1 represents the various characterstics of the california housing dataset. This table is procured by using the pandas dataframe 'describe()' functionality, which when run provides the count, mean value, standard deviation, minimum, percintile value and the maximum value of each feature. It is useful way of having key statistics to go about the model proposal.

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_income	median_house_value
count	20433.000000	20433.000000	20433.000000	20433.000000	20433.000000	20433.000000	20433.000000	20433.000000	20433.000000
mean	-119.570689	35.633221	28.633094	2636.504233	537.870553	1424.946949	499.433465	3.871162	206864.413155
std	2.003578	2.136348	12.591805	2185.269567	421.385070	1133.208490	382.299226	1.899291	115435.667099
min	-124.350000	32.540000	1.000000	2.000000	1.000000	3.000000	1.000000	0.499900	14999.000000
25%	-121.800000	33.930000	18.000000	1450.000000	296.000000	787.000000	280.000000	2.563700	119500.000000
50%	-118.490000	34.260000	29.000000	2127.000000	435.000000	1166.000000	409.000000	3.536500	179700.000000
75%	-118.010000	37.720000	37.000000	3143.000000	647.000000	1722.000000	604.000000	4.744000	264700.000000
max	-114.310000	41.950000	52.000000	39320.000000	6445.000000	35682.000000	6082.000000	15.000100	500001.000000

Table 1: Characteristics of the california housing dataset

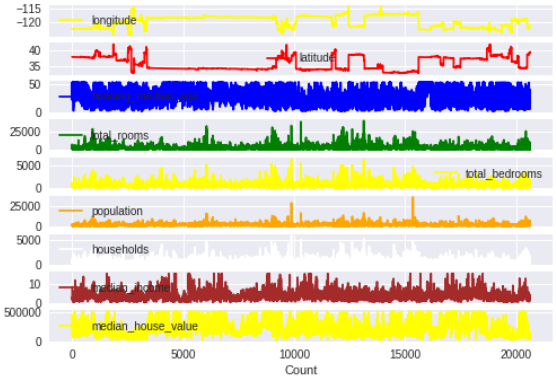


Fig 3: Subplot of each feature

A method is created to allow concurrent running of the batches through the model. The average L1 Loss function is procured by first, getting the prediction of the training set from the model, calculate the loss and then get the R^2 score. It is important to clear the errors while training so that they don't accumulate.

The main part of the model lies in the architecture of CNN where it has two layers which are run through the ReLU layer. ReLU is preferred over other activation function such as tanh

or sigmoid because the functions tend to saturate i.e., largest value tends to 1 and smallest value tends to -1 for tanh and 0 for sigmoid. This drawback is called as the vanishing gradient problem. ReLU, being a non linear activation function saturates only uni-directionally and hence, the vanishing gradient problem reduces drastically.

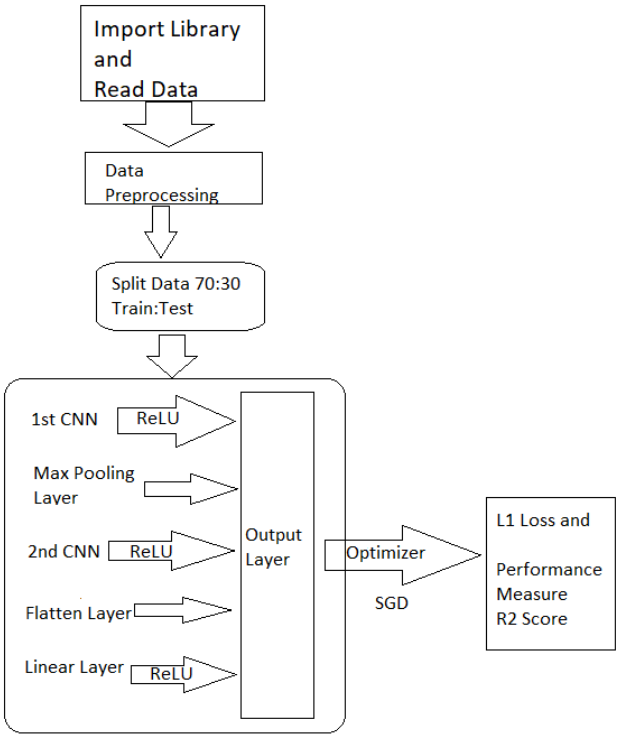


Fig 4: Typical Flowchart of the model

To have the model trained, the epochs are defined to 10. For the defined epoch, the model goes through the batches and gets the average loss. The flow diagram of the model is presented in the fig 4. L1 Loss is used because in some cases when there are outliers present in a model, L2 Loss tends to adjust the model to fit the outliers which decreases the accuracy. L1 Loss is highly resistant against outliers when compared to L2 Loss function.

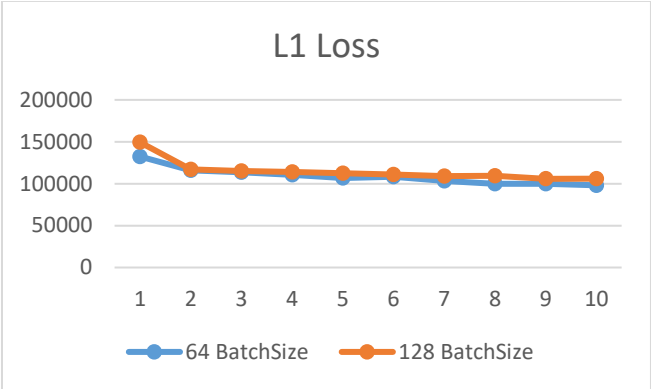
IV. EXPERIMENTAL ANALYSIS

The given model was split 70:30 for training and testing respectively and is compared with itself when run with two different batch sizes, 64 and 128 respectively. Model is run through 10 epochs and tends to perform better with 64 as its batch size generates 95903.6697368421 as its average Loss and -0.187834070677232 as the average R^2 Score, where as with 128 as the batch size it generated the R^2 score of -0.625353261509786 and the L1 Loss as 112648.476230053 as shown in the table 2.

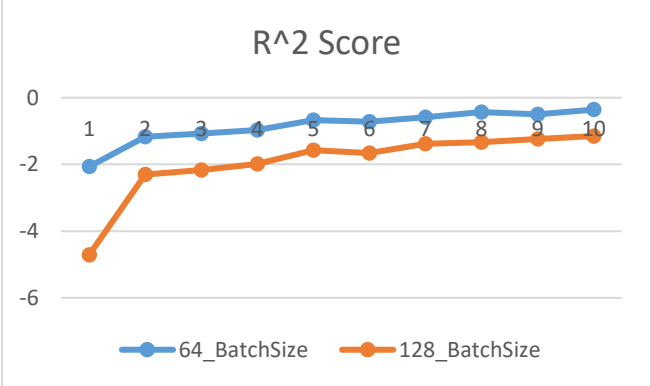
Batch Size	Avg. L1 Loss	Avg. R^2 Score
64	95903.6697368421	-0.187834070677232
128	112648.476230053	-0.625353261509786

Table 2: Average Loss and R^2 Score

Graph 1 depicts the L1 Loss values for different batch size 64 and 128 respectively when ran for 10 epochs. Comparatively, when run with 64 batch size, it provides a better size. Once trained, the Loss value keeps decreasing with the every epoch. Similarly, Graph 2 compares the R^2 Score of the model with a 64 and 128 batch through 10 epochs. It is noticeable that R^2 score increases with every run and is the most for batch size 64 eventhough there was a major increase of the score from 1 to 2 for 128 batch size. There is a gradual increase and decrease in the case of R2 score and L1 Loss because with every iteration of the epoch, system understands better and predicts efficiently.



Graph 1: L1 Loss for the model using different batch size



Graph 1: R^2 Score for the model using different batch size

When surveyed previous papers on similar model, [4] proposed a model which was trained and tested with stock market dataset. This was validated with the previous model and seem to have got an average ROGUE SU recall score of 0.13023. Another model proposed by [3], uses an ensemble learning based approach to predict house prices. This model used the same dataset (California housing) and got a mean squared error value of 41811.422310.

CONCLUSION

The proposed model is a 1D Convolution based neural network with non linear regression approach to predict the median house value of California housing dataset. The model was tested with 30% dataset. The performance measure used is R^2 score. The average score was -0.187834070677232 with an average L1 Loss value as 95903.6697368421 when run through 10 epochs with 64 as the batch size, comparatively better than the model when run with 128 as the batch size, The code is run in google colab using python. The earlier models where compared to, in which CNN was used as well to predict stock price.

REFERENCES

[1] *Le Zhang, P.N Suganthan, A Survey of randomized algorithms for traininig neural networks, Information Sciences, Volumes 364-365, 2016*

[2] *Agarap, Abien Fred. "Deep learning using rectified linear units (relu)." arXiv preprint arXiv:1803.08375 (2018).*

[3] *Cao, Buyang, and Bowen Yang. "Research on Ensemble Learning-based Housing Price Prediction Model." Big Geospatial Data and Data Science 1, no. 1 (2018): 1-8.*

[4] *L. Sayavong, Z. Wu and S. Chalita, "Research on Stock Price Prediction Method Based on Convolutional Neural Network," 2019 International Conference on Virtual Reality and Intelligent Systems (ICVRIS), Jishou, China, 2019, pp. 173-176.*

[5] *Liu, Weibo, Zidong Wang, Xiaohui Liu, Nianyin Zeng, Yurong Liu, and Fuad E. Alsaadi. "A survey of deep neural network architectures and their applications." Neurocomputing 234 (2017): 11-26.*