APPLICATION OF 1D CONVOLUTIONAL NEURAL NETWORKS USING NON-LINEAR REGRESSION MODEL

NAGA TEJA GUTTIKONDA
MSc Computer Science
Lakehead University
nguttiko@lakeheadu.ca

***Abstract*—During the last decade the convolutional neural networks has become very famous for developing the various applications in the areas pf Computer vision and Natural Processing .2D CNN's has millions of parameters and has the capability to learn very complex things where they are trained with massive databases. But still 1D CNN is preferred when the data available is less or confined to the application .1D CNN has achieved a best state of performance for several application-oriented things. In this paper our goal is to predict the median house value using various other parameters such as longitude, latitude, total number of rooms, total number of bedrooms, population, number of households, median income where we should minimize the loss using non linear regression model solution. We use the one convolutional layer followed by one max pooling layer with batch size of 64. The data is trained for 150 epochs and loss is minimized from around 70,000 to 44,200 and R2 score reaches 0.7.**

***Index Terms*—regression, loss minimization, CNN, pooling.**

I. Introduction

Artificial neurons which are used in traditional ANNs which are the first order models of biological neurons. Biological level is mainly performed at the level of cells. Basically, each neuron processes the electrical signals based on three main operations 1) Reception of other outputs 2) Integration of output processed 3) Final signal activation at axon In 1940's McCulloch-Pitts proposed the first "artificial neuron model" further which has been used in various multi-perceptron models. During 1994 LENETS is the first convolutional neural network which has been proposed in the field of Deep learning. It consists of three layers namely convolution, pooling, nonlinearity, and it uses multilayer neural network for final classification. LENET avoids the cost of computation by use of sparse connection matrix.

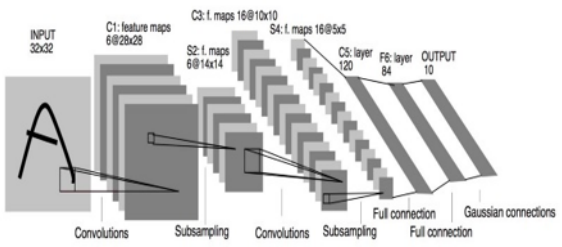

Fig. 1. Architecture of LENET

There was a period of incubation between 1998 – 2010 where the increasing power in the resources is not identified by most of the people and there is impeccable amount of rise in data during this period with the increase in use of mobile devices. All sudden the ALEXNET was proposed by Alex krizhesky during 2015 where it is used to learn more complex objects. It started using RELU for nonlinearity for the real-world applications, and dropout layer to remove selected neurons.

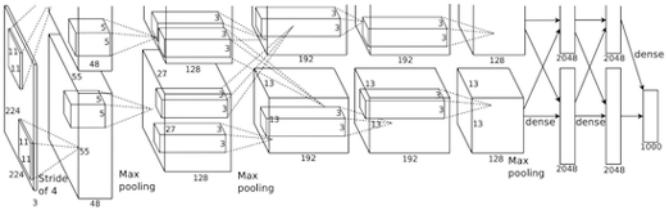

Fig. 2. Architecture of ALEXNET

During the 2015 revolution the RESNET has been developed which feeds the output two convolutional layers successively and bypass the input to the next layers of the network. By bypassing almost 1000 layers are trained for the very first time. Nonlinear regression is a very popular technique

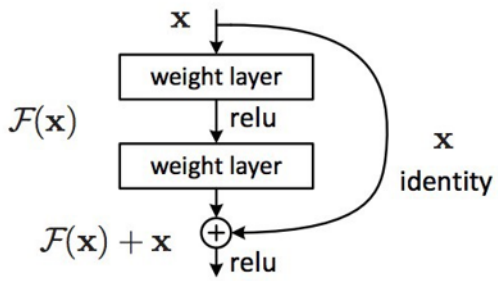

Fig. 3. Architecture of RESNET

in Mathematics as well as Engineering. It refers to the models that are nonlinear in parameters. It concerns the predictions of responses, parameter estimates It varies regularly based on parameterization. Least square estimator is used to find the

closest point among all feasible points in the solution space. In NLR the curvature array developed by watts is used to access the curvature and nonlinearity. Gauss Newton method which is based on the iterative local approximation is used to solve the problem of nonlinear least squares. The parameter which makes the prediction biased in the nonlinear models is termed as Intrinsic Nonlinearity.

II. Literature survey

Forward propagation is used where the input of each feature is the accumulation result of (l -1) where the previous signal is combined with their proper filters which are passed to the activation function .Backpropagation is used to calculate the error at the output layer and the purpose of doing this is to minimize the error for the improve the learning rate Subsampling is performed to reduce the size of final feature vector .It can be done at various ways here we use the average of its corresponding block to match the performance.Prediction is done by gathering the features of last subsampling layer .Two different methods namely GPR and SVR are used which are used to feed the features.SVM performs the LR in a feature space by the usage of intensive loss where it is based on idea of deriving the estimate with unknown relationship with various vector of observations as input to the system. Flatness of function is improved by mapping data from the actual feature space to high dimensional vector.Cross validation is used with three folds to compute the best parameter values.Due to large amount of data available the layers are minimized to three.There is significant gain in accuracy from 21.3

III. BASIC ARCHITECTURE OF 1D CNN:

Configuration of 1D CNN is formed by using following parameters:
1) Number of hidden CNN and NLP Layers
2) Size of kernel in each layer
3) Factor of subsampling in each layer
4) The choice of selecting activation functions

Various terminologies used in 1D CNN:
1) Convolution layer: It uses filter to perform convolution operations instead of matrix multiplication for scanning the input I with respect to its dimensions

2) Pooling Layer: It is particularly applied after the layer of convolution to perform down sampling

3) Fully connected layer: This is the type of layer which receives the flattened input where each input is connected to all the neurons present.

Commonly used activation functions:
1) Rectified Linear Unit (reLU): It is used to introduce the non linearities into the network because the real-world problems are not at all linear in nature.
2) SoftMax: It is seen as generalized logistic function where vector of scores is taken as input to give the output in the range of 0-1

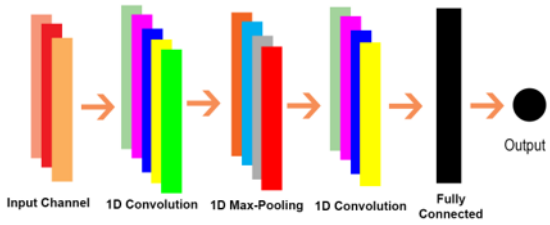

Fig. 4. Basic Architecture of 1D CNN

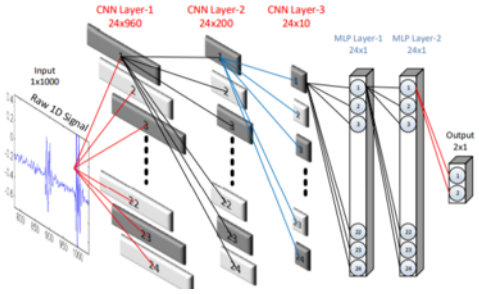

Fig. 5. Example of 1D CNN Configuration with 3 CNN layers and 2 MLP layers

IV. Implementation

First the data is read by using pd.readcsv then the following functions are called

3.1) Modelloss function : It takes the model , dataset and optimizer as input to estimate the average loss . For each input the loss is calculated and during the training it uses the optimizer to minimize the average loss

3.2) CnnRegressor function : Here it defines the one convolutional layer, one max pooling layer and one fully connected layer. Feed function takes the input and self layer where reLU operation is carried out inside to introduce the nonlinearity

3.3) Analysis : The given data is analyzed and represent in form of graph for better understanding

3.4) Training: Here we use SGD and Adam optimizers to minimize the loss. The application is trained for 150 epochs to obtain the optimal solution

3.5) Testing : Data is tested with the help of CUDA GPU and by calling the avgloss function inside to caluclate the average loss and R2 score values

```
Proposed network
class CnnRegressor(torch.nn.Module):
  def __init__(self, batch_size, inputs,
      outputs):
    super(CnnRegressor, self).__init__()
    self.batch_size = batch_size
```

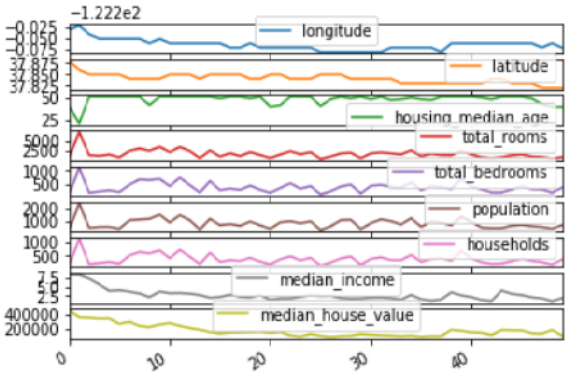

Fig. 6. Graphical Representation of data

```
  self.inputs = inputs
  self.outputs = outputs
  self.input_layer = Conv1d(inputs,
      batch_size, 1)
  self.max_pooling_layer = MaxPool1d(1)
  self.conv_layer = Conv1d(batch_size, 128,
      1)
  self.flatten_layer = Flatten()
  self.linear_layer = Linear(128, 64)
  self.output_layer = Linear(64, outputs)
def feed(self, input):
  input = input.reshape((self.batch_size,
      self.inputs, 1))
  output = relu(self.input_layer(input))
  output = self.max_pooling_layer(output)
  output = relu(self.conv_layer(output))
  output = self.flatten_layer(output)
  output = self.linear_layer(output)
  output = self.output_layer(output)
  return output
```

The main goal of our application is to reduce the loss value which can be done by following ways 1) By adding the dropout layer 2) By trying various optimizers 3) By changing the structure of network 4) By performing data shuffling 5) By trying different batch size 6) By checking with different epochs during training

V. Results and Discussions

TABLE I
Comparision of Obtained Results

Optimizer	Epochs	Batch size	Average Loss
SGD	10	32	78,000
SGD	100	64	70,000
AdaMax	10	32	68,000
AdaMax	500	64	60,000
AdaMax	300	64	58,000
Adam	10	32	54,000
Adam	300	64	55,000
AdaDelta	500	64	62,000
SGD , Adam	500	32	50,800
SGD , Adam	200	32	53,700
SGD , Adam	100	64	49,200
SGD , Adam	150	64	44,200

Here the network is tested with various optimizers for minimizing the loss . We use various optimizer such as SGD, Adam, AdaMax, RMSProp, Adadelta .Number of epochs and Batch size is changed continuously along with the use of various optimizers to reach the optimal solution.The optimal solution is obtained by using both SGD and Adam optimizers with 150 epochs and batch size of 64 to get the average loss value of 44,200 and R2 score of 0.7

VI. Conclusion and Future Works

By usage of non linear regression model with 1 dimensional CNN the loss has been minimized using the given criteria.It can be made further efficient by changing the network structure and training with more parameters

Future work includes developing the model which makes the loss decrease further with better network structure

VII. References

References

[1] Hsin Hsiung Huang,“Nonlinear Regression Analysis”, ”Statistics: Nonlinear Regression”, International Encyclopedia of Education, 3rded. Elsevier
[2] Serkan Kiranyaz, Onur Avci , Osama Abdeljaber , Onur avci, Turker ince , Moncef gabbouj, Daniel.J.Inman, “1D Convolutional neural networks and applications”.
[3] A. Cichoki, R. Unbehauen, Neural Networks for Optimization and Signal Processing, 3rd ed., 1994
[4] J.P. Resop, A Comparison of Artificial Neural Networks and Statistical Regression with Biological Resources Applications, 2006
[5] Hamilton, D. C. and Watts, D. G. (1985). A quadratic design criterion for precise estima- tion in nonlinear regression models Techno- metrics, 27, 241-250.