ID2223 Project

Group members: Salman Niazi and Shadi Issa

Project Description

The goal of the project is to predict the solar radiation level near the Earth's surface. The dataset consists of 2 million labeled data samples stored in individual files in CSV format. A data sample looks like the following.

lev	p	T	q	lwhr
0	19.231	-80.0	0.0	0.122
1	57.692	-80.0	0.0	0.451
2	96.154	-70.874	0.029	-1.229
3	134.615	-51.083	0.262	-2.732
4	173.077	-36.489	0.977	-3.429
5	211.538	-25.816	2.211	-3.574
6	250.0	-17.87	3.756	-3.536
7	288.462	-10.404	5.431	-3.802
8	326.923	-6.608	4.226	-2.198
9	365.385	-2.388	8.776	-4.203
10	403.846	1.264	10.375	-3.567
11	442.308	4.462	11.895	-3.146
12	480.769	7.318	13.347	-2.829
13	519.231	9.903	14.733	-2.598
14	557.692	12.261	16.054	-2.397
15	596.154	14.421	15.778	-1.997
16	634.615	16.428	18.499	-2.239
17	673.077	18.304	19.648	-2.006
18	711.538	20.054	20.738	-1.908
19	750.0	20.443	20.147	-0.906
20	788.462	23.377	22.793	-1.692
21	826.923	24.749	23.775	-1.582
22	865.385	23.968	24.695	-0.116
23	903.846	27.491	25.599	-1.538
24	942.308	26.804	26.464	-0.053
25	980.769	29.988	27.285	-1.448

The first two columns (**lev**, **p**) are static, that is, the values of these columns in all the data samples are the same. Therefore, the first two columns does not add contain any useful information that can be used to predict the final labels. The column **T** represents the temperature, **q** represents the pressure and the column **lwhr** represents the radiation. The **lwhr** is the label column. Each

CSV file contains exactly 26 data points representing temperature, pressure and solar radiation at the Earth's surface at 26 different heights. For example, in the above data the 26th data point shows that the temperature near the Earth surface is 29.988, the pressure is 27.285 and the solar radiation is -1.448.

Feature Normalization

The input features and the label values vary quite a lot that causes the gradient to fluctuate. All the input features and labels are normalized using min-max scaling. Which is defined as

$X_{norm} = (X - X_{min}) / (X_{max} - X_{min})$

Feature	Min	Max
T	-80.0	29.988
q	0.0	27.285
lwhr	-9.94	6.69

Initial Solution

According to the universal approximation theorem, Feedforward neural networks with a single hidden layer can approximate continuous functions. In our problem we are trying to replace an analytical model with a statistical model to enhance the performance; hence a feedforward neural network with a single hidden layer is ought to be a suitable solution.

Feedforward Neural Network Model

Figure 1 shows the model of our feedforward network. It is composed of a single hidden layer with 20 neurons and an output layer of a single linear neuron. Neurons within the hidden layer use a sigmoid activation function. The weights of the hidden neurons has the structure 26×1, while the weights of the output neuron has the structure 1×26 to produce the output *lwhr*. Both hidden and output layer have a bias that is initialized to zero.

Evaluation

The model was implemented using Tensorflow running in a VMware virtual machine which is install on a 2 core Macbook. The following are the values for different parameters obtained after hyper-parameter optimization.

- Training data set size 1,400,000 (70%).

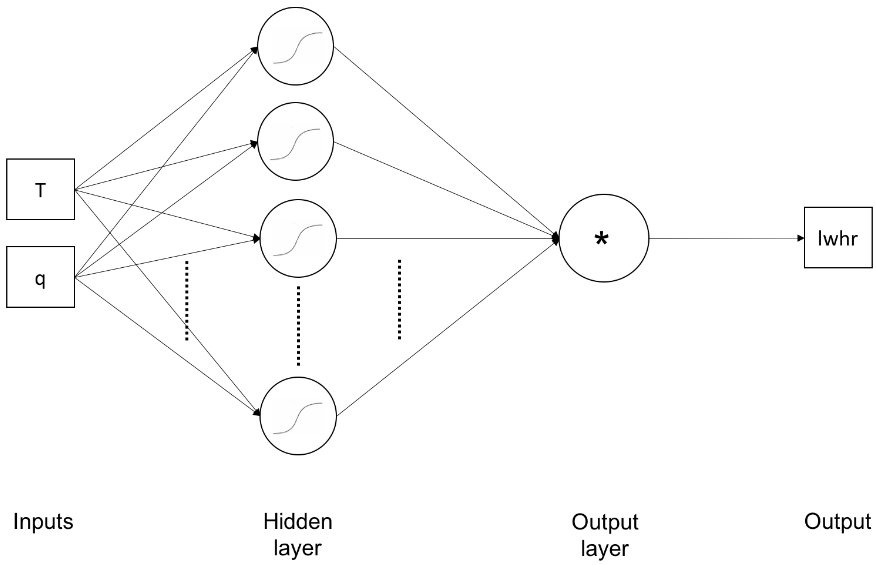

Figure 1: Architecture of feedforward neural network

- Test data set size 600,000 (30%).
- Learning Rate 0.05
- Max number of Epochs 14000
- Batch Size 100
- Weights were randomly initialized such that the random numbers had *mean=0.1* and *stddev=0.3*
- Bias are initialized to zeros

Mean Square Error of the FFN Model

Proposed Solution

The goal of the project is to predict the solar radiation near the Earth's surface using the temperature and pressure values. This is a *regression* problem with 26 output labels. The problem can be solved using machine learning techniques, such as, *multivariate regression* and deep learning techniques, such as, *feed forward neural networks* and *convolution neural networks*. I have chosen convolution neural networks *(CNN)* for this problem as CNN are suitable for problems when there is a corelation between the input features. In out case the input feature are corelated by height, that is the samples are collection at 26 different distances from the earth surface and the measurements of the features and the labels gradually change.

Convolution Neural Network Model

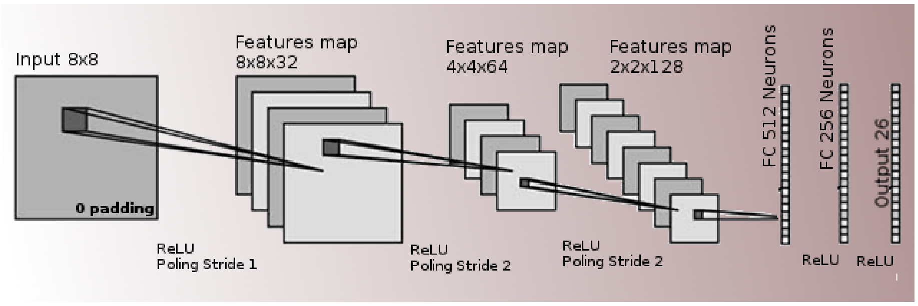

Figure 2: Convolution Neural Network Model

The model of the convolution neural network model is shown in Figure 2. It consists of three convolution layers, two fully connected layers and an output layer.

- **Layer 1 (Convolution):** The first layer is a convolution layer that uses *2x2* filter with *32* features. The layer is followed by a ReLU normalization layer and max pooling layer with a stride of *1*.
- **Layer 2 (Convolution):** The second layer is a convolution layer that uses *2x2* filter with *64* features. The layer is followed by a ReLU normalization layer and max pooling layer with a stride of *2*.
- **Layer 3 (Convolution):** The third layer is a convolution layer that uses *2x2* filter with *128* features. The layer is followed by a ReLU normalization layer and max pooling layer with a stride of *2*.
- **Layer 4 (Fully Connected Layer):** The fourth layer is fully connected layer with 512 neurons using ReLU activation function.
- **Layer 5 (Fully Connected Layer):** The fourth layer is fully connected layer with 256 neurons using ReLU activation function.
- **Layer 6 (Output Layer):** The last layer is 26 neuron output layer.

Input

The convolutin neural network expects an input matrix of size $m\ x\ n$ size. We could combine the **T** and **q** columns to form a 26x2 matrix as shown below.

This 26x2 input matrix did not produce very promissing results, as pooling can not shink the width of the input matrix. The minimum mean square error achieved uisng this input format was 0.3.

The input can be morphed into 8 x 8 matrix with zero padding as there are only 52 input features

1	1
2	2
3	3
4	4
5	5
.	.
.	.
.	.

Figure 3: 26x2 Input Matrix

1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48
49	50	51	52	0	0	0	0
0	0	0	0	0	0	0	0

Figure 4: 8 x 8 Input Matrix

Regularization

For regularization *dropout* is used in the last fully connected layer. The dropout value was set to *0.95*, that is, during each training epoch 5% of the neurons in the last fully connected layer are randomly set inactive. This enables all neurons to equally learn the model.

Model Complexity

Layer	Size	Memory	Weights	Bias
Input	8x8x1	64	0	0
CONV	8x8x32	8x8x32 = 2048	2x2x1 x 32 = 128	32
POOL	8x8x32	8x8x32 = 2048	0	0
CONV	8x8x64	8x8x64 = 4096	2x2x1 * 64 = 256	64
POOL	4x4x64	4x4x64 = 512	0	0
CONV	4x4x128	4x4x128 = 2048	2x2x1 * 128 = 512	128
POOL	2x2x128	2x2x128 = 512	0	0
FC	1x512	512	2x2x128x512 = 262144	512
FC	1x256	256	512x256 = 131072	256
OUT	1x26	26	26x256 = 6656	26

Total memory = 413908 x 4 bytes (*float32*) x 2 (back propagation) = 3311264 = 3.1 Megabytes

Evaluation

The model was implemented using Tensorflow running in a docker instance. The docker instance was run on HP ProLiant DL360p Gen8 with 32 cores and 256 GB of RAM. Following are the values for different parameters values obtained

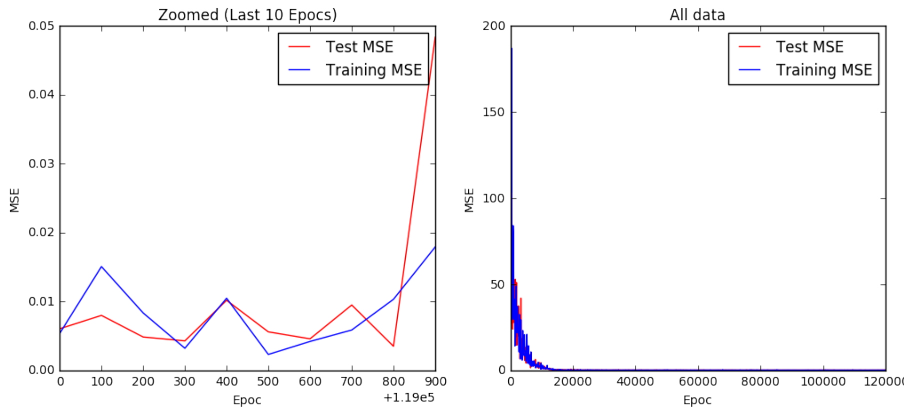

Figure 5: Mean Square Error of the CNN Model

Figure 5 shows how the mean square error (MSE) of the model drops as the training progresses. The x-axis of the graphs show the elapsed epochs and the y-axis of the graphs show the MSE. The graph on the left is a zoomed version of the graph on the right. The zoomed version of the graphs show last ten epochs. From the graphs it is clear that MSE drop to 0.0009 after 2800 epochs.