

**Applied Machine Learning**

**ELEC 8900**

**Course Project**

**Winter 2020**

**Project Progress Report**

**Convolutional Networks with Noisy Labels**

|  |  |
| --- | --- |
| **Name** | **Student ID** |
| **Jagadeesh Sankar Parati** | **105138525** |
| **Monica Neelisetty** | **110005725** |
| **Rajasekhar Reddy Sangam** | **105122180** |

**Index**

**Contents Page No’s**

1. Introduction to Machine learning 1

1.1Convolution Neural Networks 1

1.1.1 Architecture of the Convolution Neural Network 2

2. Introduction 3

2.1 Noisy Labels 3

2.1.1 Label Flips 3

2.1.2 Outlier Flips 4

3. Requirements 5

3.1 Anaconda with Jupyter notebook 5

3.2 Training Data 6

3.3 Testing Data 6

3.4 Tensorflow 6

3.5 Keras 6

4. Algorithm 7

5. Development Procedure 7

6. Modeling 10

6.1 Base structure 10

6.2 Equations and working 11

6.3 Learning Noise Distribution 12

7. Timeline 12

8. Conclusion 13

9. References 14

**Figures** **Page No’s**

1. Architecture of Convolution Network 2
2. Label Flips 3
3. Outlier Flips 4
4. Anaconda 5
5. Jyupter Notebook 5
6. Base structure 10
7. Learning Noise Distribution 11
8. Timeline 13

**ABSTRACT**

As, it is a known fact that labeling of images manually may not be accurate and effective task. It is very difficult to label every image from the large dataset. Not only this we can also collect large number of data from internet, but the labelling of images may not be accurate. This type of incorrect labelling arises to two different, types of noise. They are

Label flips: A different label of the same class is assigned.

Outlier Flips: Image is labelled with a completely different class.

So, we use Convolution Neural Networks in order to achieve the better results in the labelling of such noisy data. We have different approach, as per the noise. For label flip, a linear layer of noise is added between the Soft Max layer and cost layer. For Outlier noise, we create an additional class of outlier, which helps us to use the previously described noise model and can also reduce the outlier noise by adjusting the alpha and the hyper parameter.

We have used python as our programming language to implement this project. Later, on we have divided the data into 10 folds and did the cross-validation loop. Also, calculate the f-measure from the confusion matrix which is obtained from each cross-validation loop.

So, finally through this project we did, convolution training for label flips and Outliers. This will make the deep learning model robust and accurate.

**Introduction to Machine Learning:**

Machine learning is the study of computer model without depending on explicit instruction, only relying on patterns. It automatically builds a mathematical model based on the sample data provided called as training data based on the prediction. From the email filtering to the advance computer vision the usage of machine learning is increasing tremendously. Machine learning is the subset of artificial intelligence. It is closely related to the computer statistics. There are different approaches to predict and form the algorithm:

1. Supervised Learning

2. Unsupervised Learning

3. Reinforcement Learning

4. Semi-supervised Learning

5. Self Learning

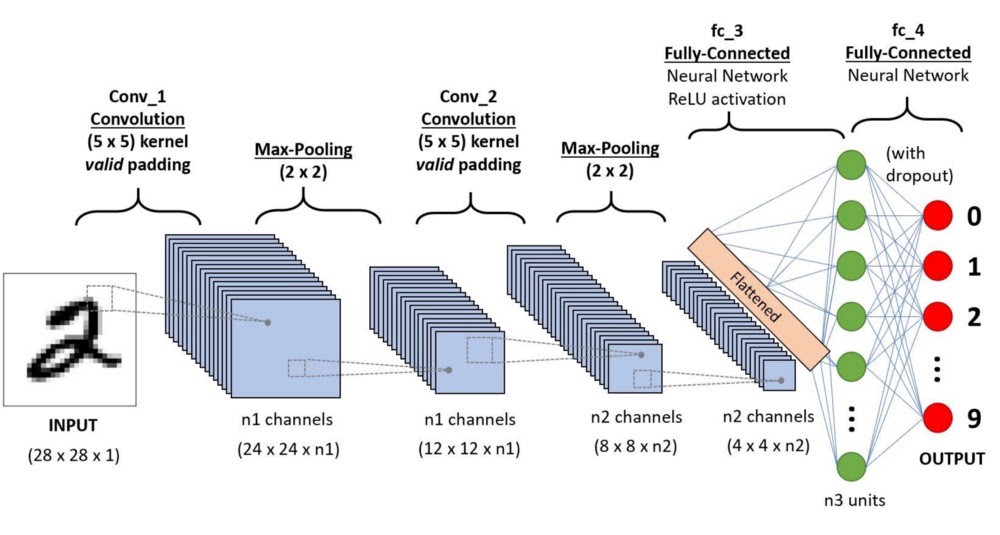
6. Featured Learning.

There are also few more learning methods. In this project the model is based on the supervised learning. Deep learning is the key concept [1]. The pre-requirements of this project is Convolution Neural Network.

**1.1 Convolutional Neural Networks.**

A Convolution Neural Network is a specialized artificial neural network which uses perceptron, a machine learning unit algorithm. A convolutional neural network has an input layer, output layer and various hidden layers, that are able to extract the features.

**1.1.1 Architecture of the Convolution Neural Network:**



**Figure 1. Architecture of Convolution Network**

The network starts with two convolution and max-pooling networks, that are followed with two fully connected layers. The convolution and pooling layers are used to recognize and extract the features of the image, whereas in the fully connected layers, the images are classified into pre-defined classes.

The first part in a convolutional neural network filters and extracts the features of an image and then it pools the extracted features to reduce the size of the data. At last, it adds an activation function, so that the network is a non-linear function. So, all these steps are summed to convolution + pooling + activation layers.

Machine Learning and Convolution Neural Networks go hand-in-hand. Machine Learning mostly relies on function of Convolution Neural Networks for processing and analysing of the images, time series data and languages. CNN, being a powerful tool for image and data analysis, it has been used widely from image recognition to drug discovery.

1. **Introduction:**

As we know, labeling manually may not be accurate and effective. It is very difficult to label every value for large datasets. In addition, we can collect larger data from social networks and on interne, But the labels may not be accurate. This is called as Noisy labels

**2.1 Noisy Labels**

It is a known fact that, labelling of large number of data manually may not produce accurate and effective results. Instead of that, our data may have some noisy labels, that contain some freely available data which may or may not be accurate. This noise in training data leads to the degradation in performance. Especially noise in labels is more when compared to noise in the input feature. They are classified into two types on noise on labels.

**2.1.1 Label Flips:**

Label Flips are where a different label of the same class is assigned instead of the correct label within the same dataset. In this, the image is assigned with a different label of the same dataset, instead of the correct label within the same dataset.



**Figure 2. Label Flips**

In the above image, the label flips consist of only three classes, but sometimes the labels are confused among the classes.

**2.1.2 Outlier Flips:**

The image is labeled with a completely different class which is not considered in the dataset. In this, the image is labeled with a completely different class which does not belongs to that dataset.



**Figure 3. Outlier Flips**

The above image shows the example of outliers, in which the images are unrelated to the class, but posses one of the three labels.

A simple approach to handle these noisy labels is data preprocessing stage, where labels suspected are removed or corrected. However, the difficulty of this is to distinguish the informative hard samples from the harmful mislabeled noise. The effects of these labeled are well studied in common classifiers and robust variants has been proposed.

Contrary to expectations a single robust model is a standard Convnet for both the noisy labels. This technique is named as a Novel Modification technique, in which an extra linear layer of noise is added to network in order to correct the noisy labels in the data. This linear layer will predict the true labels for both i.e., the label flip as well as the outlier noise using Back Propagation.

So, we use a different approach called Convolutional Neural Network to achieve better results in labeling of such noisy data. In this project, we analyze how the discriminatively trained Convolutional Network behave when applied to such noisy data. We are using a technique called Novel Modification where an extra layer of noise is added to the network to correct the noisy labels in the data.

In this project, we are proposing two approaches as per the type of noise. For the Label Flips a linear layer of noise is been added between the Soft max layer and cost layers. In which the Linear Noise layer will get trained as per the noisy label data and gets the weights updated to minimize the cost function and helps to reduce both Label Flip as well as the outlier noise in this case.

**3 Requirements:**

1. Anaconda with Jupyter notebook
2. Training data
3. Testing data
4. TensorFlow
5. Keras

**3.1 Anaconda with Jupyter notebook:**

Anaconda is an open source application for python programing. In this project, we are using python as a programming language. Anaconda is a graphical user interface. Install anaconda and open it, we can see it will run on with root base. In root base we can see spider, Jupyter notebook, pycharm and few more running applications. We can use any of those environments.

Figure 4. Anaconda

We need few more libraries for our programing, so install TensorFlow . Click on TensorFlow and then install it. On the same kernel also install keras and restart the application. Now open the anaconda terminal and run it on the new environment TensorFlow as root base. Now Open Jupyter notebook to run the python code.

Figure 5. Jupyter Notebook

**3.2 Training Data:**

In machine learning, we require data to train and build the model. The data which we use to train the data is called training data. The data also need labeling. The training data is the set of data which helps the program or machine to understand the model. The program learns the model using this training data and helps us to provide the results. This training data can be further divide into training and validation sets for evaluating the model using loss, accuracy and other kind of metrics.

**3.3 Testing Data:**

Testing data is the data which helps to check the model efficiency. This testing data is independent to the training data. We will use a different data to test whether the given set of data gets the expected results. We find the efficiency of the model using confusion Matrix and other different metrics.

**3.4 Tensorflow:**

It is an open source software library for python. It include the all math library and neural network for machine learning. It works as base to many other working libraries of python.

**3.5 Keras:**

It works on the tensor flow. It consists libraries of deep neural networks and makes a fast run. The file are necessary for this project as it supports tensorflow’s core libraries.

**4 Algorithm:**

Scale the Data

Input

2 hidden Dense layers with Regularizer

CNN with Input layer

CNN with Input layer

Split the Train Data to train and validation data

Cost Layer with Sparse Cross Entropy and optimizer Adam

Soft Max Layer

Linear Noise Layer

Batch Normalization

Obtain the predicted values and performance metrics to Excel

Evaluate the model with Test Data

**5 Development Procedure:**

* Import all required libraries like Pandas, Numpy, Sequential, Dense, Activation, Layer Normalization and other required libraries from Keras.
* Import the train and test data using built in function read\_excel from Pandas.
* Preprocess the data by scaling the data using a built in function MinMaxScaler() which is imported from sklearn.preprocessing library. This function scales the data between 0 to 1 when the data is positive and -1 to 1 when there are negative values in the data.
* Create a FOR loop to count the number of columns in the label data to process into the Model individually.
* Create a Sequential Model with input layer has 42 inputs as the train data has 42 features in it.
* Add hidden Dense layers with activation function as ‘Relu’ with the last hidden layer with a Regularizer to penalize the weights if they are high.
* Soft Max Layer will be added to obtain the output probabilities of classes of each label.
* The Linear Noise layer is to be added after the Soft Max layer which helps us to learn the noise the label data. This is done by adding Gaussian Noise layer followed by a Dense layer with Activation as ‘Linear’.
* Compile the model using the loss function as Categorical Cross Entropy which has to be minimized through back propagation by adjusting the weights of the model . As the output is not one hot encoded values we are using Sparse Categorical Cross Entropy function with the optimizer as ‘Adam’.
* To optimize the above model check points like Early Stopping and Model Checkpoints are added. Early Stopping with the validation loss minimum will help the model to check the minimum validation loss and helps the model to stop moving ahead if the validation los increases as per the patience value given.
* Model Checkpoints helps us to the save the best model with best validation accuracy and helps us to use that model to predict the testing labels.[4]



* The train data is split into two part as train data and validation data using train\_test\_split function which helps us to evaluate the model using the validation accuracy an loss .
* Now after training the model with the train data and validating it with the validation data which is split from the train data the model is used to predict the test labels.
* Using the predicted test labels we find the accuracy and f measure of each fold and all the predicted values as well as the performance values will be written to excel sheets using function called Excel Writer imported from Pandas.
* This model will go on running until all the 25 labels get their predictions and performance measures like accuracy and f measure values as

Accuracy = (True Positive + True Negative) /(Positive + Negative)

F Measure = (2 \* Precision \* Recall) / (Precision + Recall)

where as

Precision = True Positive/(True Positive + False Positive)

Recall = True Positive/(True Positive + False Negative)

* Thus the model will be trained using labels with different kind of noisy levels and helps us to get the predicted labels as well as the accuracy metrics such as accuracy and f Measure.
* For the B datasets, the same procedure would be followed with a change in the input layers of the models as there are 22 Features in the B Sample data.

**6 Modeling:**

**6.1 Base structure:**

Let us start modeling the labeled flip noise. As we know the base is always convolution neural network which consist of convolution layer, polling layer and fully connected layer. The output from neural network is known as the softmax layer. We will add an extra layer of noise. We can use gaussian or linear noise layer. For better results we added both the noise layers. The output from the noise model is called as cross entropy cost.

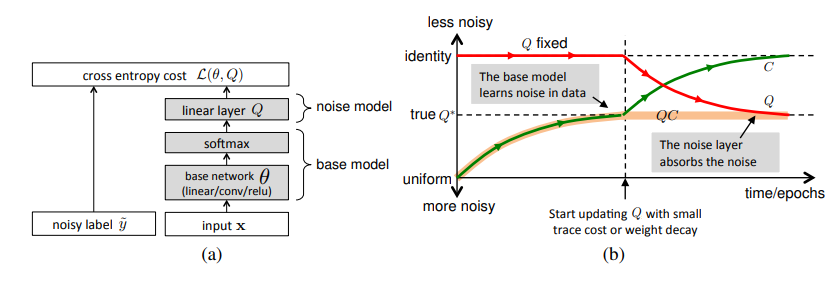
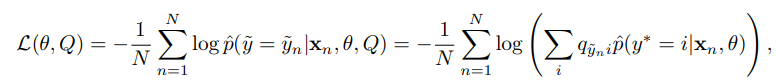


Figure 6. Base Structure

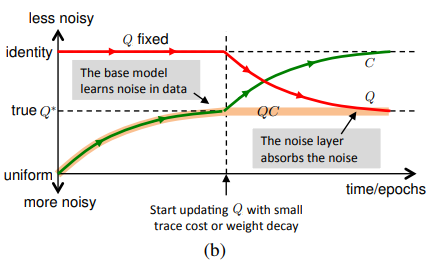
**6.2 Equations and working:**

As initially we need data to train the model, consider the training data is (xn, y∗ n ). The classification model uses the probability matrix of Q for prediction. In this model there are two parts: the base model parameterized by θ and the noise model parameterized by Q.



Q is the noise distribution which became the weight matrix of the linear noise layer, Due to this change the probability from the base model increases and there are better chance of matches the noisy labels.

In the below image the noise matrix in green is while the base model trained, while red is set to the identity. Later we start regularizing Q based on the requirement capturing the noise property and leaving the model clean. This is how the noise can be minimized in labeled flip noise.

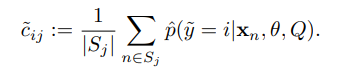


**Figure 7.Learning Noise distribution**

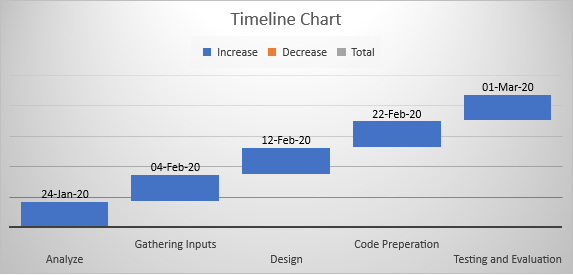
**6.3 Learning Noise Distribution:**

Previously we changed the noise model setting Q = Q∗ to optimise the accuracy of the true labels. While working we face that even the true noise distribution(Q\*) is unknown. At that time rely on the noise data itself. As noise layer is constrained linier layer the weights of Q must be updated along with the other network. We will do this by back propagation through the cross-entropy loss.

Simply minimizing the loss will not give the desired output, we need train the process as QC = C˜ → Q∗, where C˜ is the confusion matrix of the combined model and Q∗ is the true noise distribution of data. The confusion matrix (C˜ ) is defined by:



**7. Timeline:**



**8. Conclusion:**

Due to the availability of large data, Convolution Network model is used to achieve excellent recognition results. But the data available is not accurately labeled. By considering all the labels are having high level noisy data and training convolution network model identify the true labels of the images. The training and process varies based on the types of noise. In this project we are going to do convolution training for Label Flips and Outliers. This will make the deep learning model robust and accurate.

**9. References:**

[1] “Machine learning,” Wikipedia, 21-Mar-2020. [Online]. Available: https://en.wikipedia.org/wiki/Machine\_learning. [Accessed: 25-Mar-2020].

[2] “Convolutional Neural Networks,” DeepAI, 17-May-2019. [Online]. Available: https://deepai.org/machine-learning-glossary-and terms/ convolutional neural-network. [Accessed: 29-Mar-2020].

[3] Sainbayar, Bruna, Joan, Manohar, Bourdev, Lubomir, Fergus, and Rob, “Training Convolutional Networks with Noisy Labels,” *arXiv.org*, 10-Apr-2015. [Online]. Available: https://arxiv.org/abs/1406.2080. [Accessed: 05-Feb-2020].