CSE-574 Project 2

Mitul Modi
UB Person No : 50288649
University at Buffalo, The State University of New York
mitulraj@buffalo.edu

November 1, 2018

Abstract

This project report represents different machine learning approaches to solve hand writing comparison task. Problem is solved by three approaches - Linear Regression, Logistic Regression and Neural Network. Performance and parameter tuning for each approach are compared.

Data Set

For this project, "AND" images samples extracted from CEDAR Letter dataset is used. We have two types of datasets. One is generated by extracting features by human observation, and other generated by GSC algorithm. Human Observed dataset has 9 features for each image while GSC generated dataset has 512 features. Then pairs of images are created with features of both images in pair. While making pairs, two different datasets are created with different approaches. In first approach, features of both images are just concatenated. So after concatenation, we have 1024 features for each pair of GSC dataset and 18 features for each pair of Human Observed Dataset. In another scheme, corresponding features of both images are subtracted and absolute value of subtraction is used as a feature of a pair. This way four different datasets are created.

1. Human Observed dataset with feature concatenation
2. Human Observed dataset with feature subtraction
3. GSC generated dataset with feature concatenation
4. GSC generated dataset with feature subtraction

If both images in a pair are from same writer or from different writer using features of both images. If both images are pair from same writer, we assign label 1 to that pair. If both images are from different writer, it is assigned label 0.

Data Partitioning

Main goal of data partitioning is to divide data into three sets - Training, Validation and Test sets. But for this dataset, two more issues needs to addressed.

First is related to class balance. For both datasets - Human observed and GSC generated, negative class(pairs of different writers) is dominating by a huge margin. It mean there are large number of different writer pairs than same writer pairs. Models trained with imbalanced data like this tend to be biased towards dominating class. To handle this, same amount of records for both class are picked while partitioning data.

Second issue is when we deploy model in production environment, it is very unlikely that same writer's data is used to train model will also be used for prediction. So we need to partition our data such that all images from a single writer will be present in only one of three dataset. If an image from a writer A is present in Training dataset then no image from writer A will be present in validation or testing dataset and vice versa. This is referred as unseen writer partitioning.

Evaluation Metrics

Different models can be evaluated best with different metrics. Like Linear Regression models are best evaluated on E_{RMS} while Logistic Regression and Neural Network are best evaluated on Accuracy of prediction. As this is binary classification problem, Precision and Recall are also useful metrics to evaluate model. Precision indicates out of all positive labels predicted by model, what fraction of them is correct. Recall indicates what fraction of total positive labels was predicted as positives correctly. Our task is to identify same hand writing which can be used for fraud detection in forensics. For this it is necessary that maximum number of actual positives identified as positives, as missing a fraud can be very costly. It means Recall should be given higher importance than Precision for this problem. Here, values of all above criteria are represented for each model.

Experiments and Results

I have implemented models using three different algorithms - Linear Regression, Logistic Regression and Neural Networks with each of four datasets. Performance of each experiment is discussed below.

GSC generated Dataset

Table 1 and 2 shows performance of all models trained using GSC generated datasets with subtracted features and concatenated features respectively. It is interesting to note that for linear regression and logistic regression, models performed significantly better with subtracted features compared to concatenated features. However, Neural Network performed equally good for both datasets. It is evident that Neural Network performs better than other two models.

Algorithm	E_{RMS} Tr	E_{RMS} Val	E_{RMS} Test	Acc Tr	Acc Val	Acc Test	Precision	Recall
Linear Regression with Basis Func	0.4995	0.505	0.51	52.08	49.1	52	0	N/A
Linear Regression	0.3747	0.3666	0.3595	81.2584	82.1761	83.3556	82.75	81.69
Logistic Regression	0.3728	0.3626	0.3536	80.4316	81.7791	82.5676	84	81
Neural Network	0.3349	0.3937	0.404	85.51	84.49	82.565	85.71	83.15

Table 1: Results for GSC generated Dataset with subtracted features

Algorithm	E_{RMS} Tr	E_{RMS} Val	E_{RMS} Test	Acc Tr	Acc Val	Acc Test	Precision	Recall
Linear Regression with Basis Func	0.503	0.499	0.501	51.88	49.1	51.573	0	N/A
Linear Regression	0.4797	0.4842	0.4889	61.8933	60.9981	59.6693	60.5	59.1
Logistic Regression	0.4802	0.4865	0.4892	61.6892	59.5464	59.4325	64	58
Neural Network	0.1044	0.4796	0.3737	95.9	86.69	86.03	87.33	84.31

Table 2: Results for GSC generated Dataset with concatenated features

Linear Regression

For linear regression I tried experimenting with Basis Functions and without basis functions. With basis functions, model was performing very poor. It was prediction only negative labels which resulted in 0 precision. Even changing parameters like no of basis functions, learning rate and regularization coefficient, performance didn't change much, with E_{RMS} value being constant around 0.49. Without basis functions model performed reasonably well for subtracted features with E_{RMS} of 0.35 and 81 recall. However, it performed poorly on concatenated features. Figures 1 and 2 shows how values of E_{RMS} and Accuracy changed with each epoch. Till epoch 500, both models were performing similarly but for concatenated features, it performance remained constant afterwards.

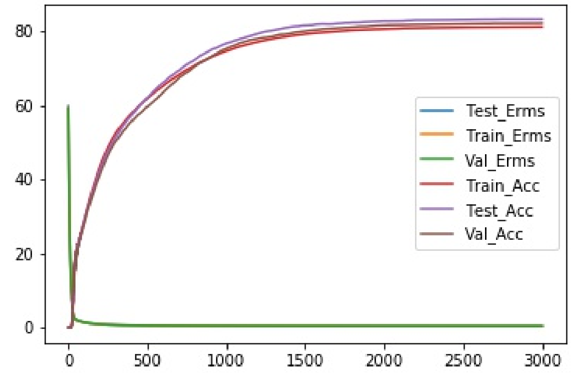

Fig 1 : Linear Regression without Basis Functions

on GSC dataset subtracted features

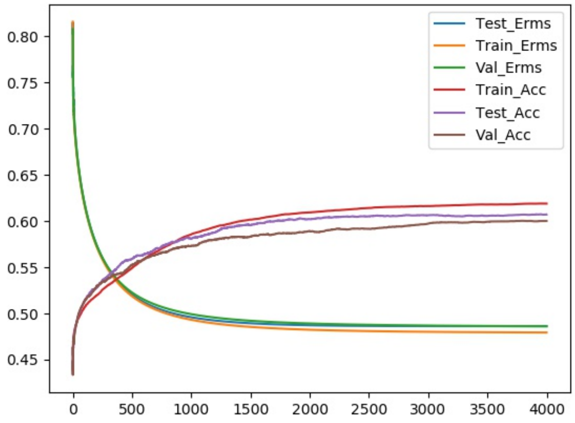

Fig 2 : Linear Regression without Basis Functions

on GSC dataset concatenated features

Logistic Regression

Logistic Regression is a classification algorithm, which gives probabilities of each class. From table 1 and 2, results of logistic regression are very similar to linear regression. Like linear regression model performed much better with features subtracted data. Figures 3 and 4 shows how values of E_{RMS} and Accuracy changed with each epoch.

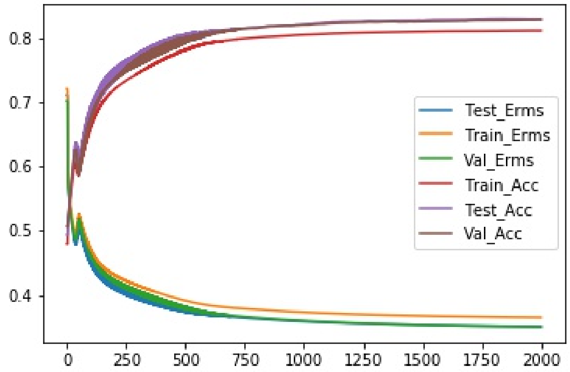

Fig 3 : Logistic Regression on GSC dataset subtracted features

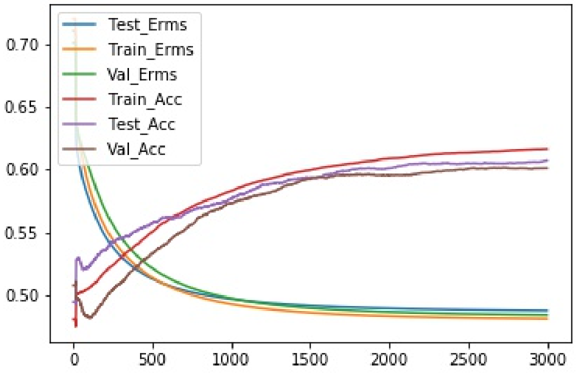

Fig 4 : Logistic Regression on GSC dataset concatenated features

Neural Network

Initially I tried with 1 and 2 layers neural network. Even after trying different values of parameters performance was not improving, so I tried with 4 layer neural network which gave far better results, which are shown in table 1 and 2. Figures 5 and 6 shows how values of accuracy and E_{RMS} changes with each epoch. From figures it can be observed that values of loss and accuracy improved significantly during initial stages, but after epochs 35, it remained almost constant. In addition to table results, figures 5 and 6 also shows neural network performed almost similar for both datasets. So it can be concluded that Neural Network is very robust model compared to Linear regression and Logistic regression.

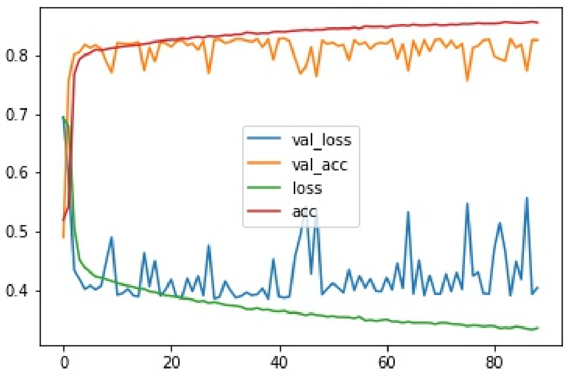

Fig 5 : Neural Network on GSC dataset subtracted features

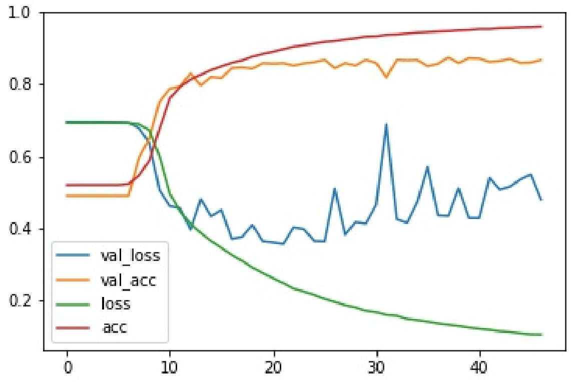

Fig 6 : Neural Network on GSC dataset concatenated features

Human Observed Dataset

Table 3 and 4 shows performance of all models trained using Human observed datasets with concatenated features and subtracted features respectively. Tables shows that all models performed poorly when trained with Human Observed Datasets, with highest accuracy and recall being just 56 and 54 respectively.

Algorithm	E_{RMS} Tr	E_{RMS} Val	E_{RMS} Test	Acc Tr	Acc Val	Acc Test	Precision	Recall
Linear Regression with Basis Func	0.4999	0.5	0.501	50.51	41.899	41.83	1	0.43
Linear Regression	0.4948	0.5027	0.5042	57.6095	55.6122	51.3966	62.67	40.86
Logistic Regression	0.6964	0.6627	0.6771	50.1587	55.3921	53.012	60.0	44.11
Neural Network	0.5632	0.8183	0.6839	69.37	51.98	53.2163	46.67	54.68

Table 3: Results for Human Observed Dataset with concatenated features

Algorithm	E_{RMS} Tr	E_{RMS} Val	E_{RMS} Test	Acc Tr	Acc Val	Acc Test	Precision	Recall
Linear Regression with Basis Func	0.4999	0.5	0.501	50.51	41.899	41.83	1	0.43
Linear Regression	0.4983	0.500	0.5049	54.5958	51.8518	44.6927	53.33	38.46
Logistic Regression	0.7365	0.7153	0.6995	44.8412	47.5728	49.6932	41.33	34.44
Neural Network	0.6123	0.7509	0.6571	66.69	49.52	56.8181	32	42.1

Table 4: Results for Human Observed Dataset with subtracted features

Linear Regression

Like with GSC datasets, with Human Observed dataset also I tried using Basis functions with linear regression. Here also it performed poorly predicting all values positively which resulted in precision of 1 but only 0.43 recall. However, unlike with GSC dataset, model didn't improved even without basis functions. I tried tuning parameters by changing learning rate and regularization coefficient in range of 0.001 to 0.5, but performance didn't improve.

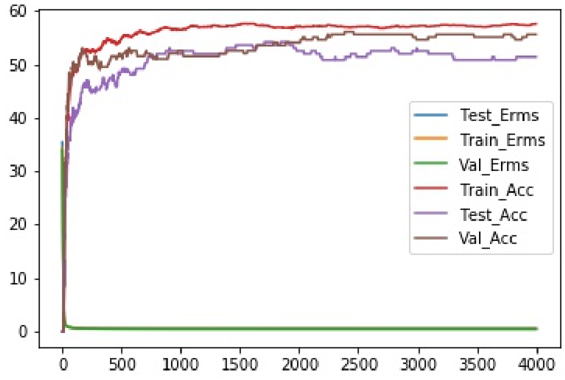

Fig 7 : Linear Regression without Basis Functions

on Human observed dataset concatenated features

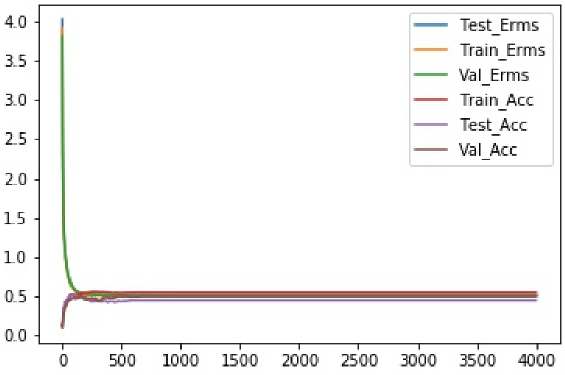

Fig 8 : Linear Regression without Basis Functions

on Human observed dataset subtracted features

Logistic Regression

For Human Observed dataset logistic regression also performed poorly on both concatenated features and subtracted features with highest accuracy and recall being 53 and 44 respectively.

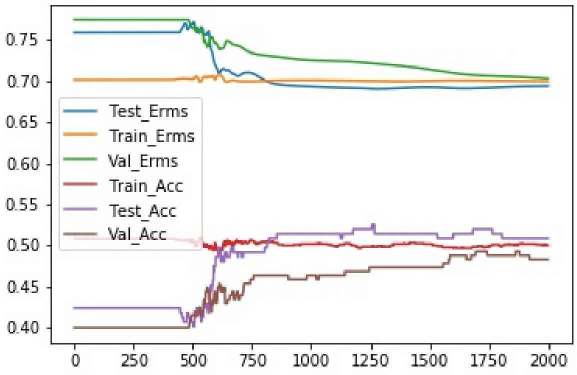

Fig 9 :

Logistic Regression on Human observed dataset concatenated features

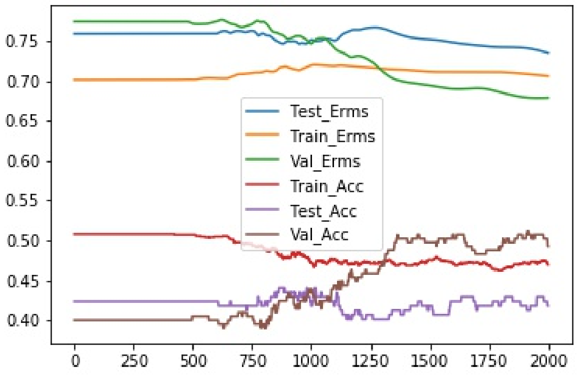

Fig 10 :

Logistic Regression on Human observed dataset subtracted features

Neural Network

As seen from tables 3 4, neural network model performed similar for both concatenated and subtracted features just like GSC dataset. So for this dataset also Neural Network performance doesn't change much according to method of features merge. However, performance of both model for Human Observed dataset is very poor compared to GSC dataset.

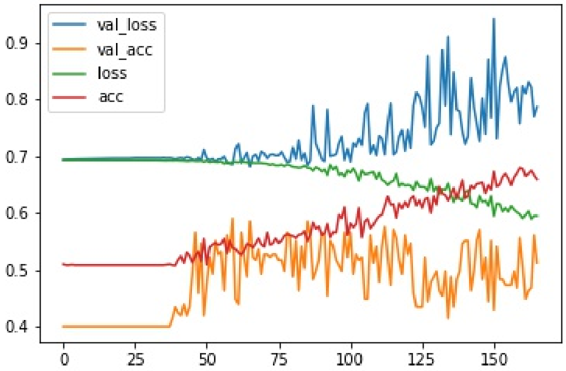

Fig 11 :

Neural Network on Human observed dataset concatenated features

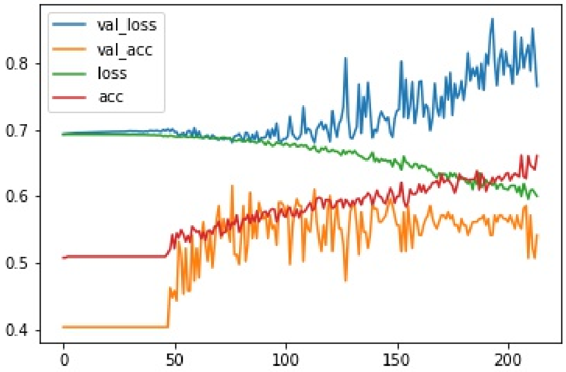

Fig 12 :

Neural Network on Human observed dataset subtracted features

1 Conclusion

we have for both datasets. After partitioning and achieving class balance, GSC generated data had 10 times number of training records than Human Observed Data. This difference of training data is affecting performance of models which can be clearly observed in results. Performance of models trained using GSC generated(best performance of 86% Test Accuracy) dataset are performing significantly better compared to Human Observed dataset(Best performance of 57% Test Accuracy).

2 Learning Outcomes

I would like to summarize key points I learnt by working on this project:

- First observation is importance of weight initialization. I used numpy rand function to generate initial weights for linear and logistic regression. However, sometimes too higher values causes model diverging or converging very slow. When I multiplied random values with fraction like 0.1, algorithm converged smoothly.

- Learning Rate - Sometimes, values off Loss increases exponentially at each epoch, and eventually it will cause math error. To solve this learning rate optimization was important. For some models very small learning rate is needed.

- Robustness of Neural Network - Neural Network is robust against what method we choose to merge features of both images compared to linear and logistic regression.

- Amount of Data matters - Comparing performance of models based on Human observed data and GSC data, it is evident that very less data makes very hard to achieve good model performance.

- Class Imbalance in data - When we have more amount of data of one class compared to other class, model will generalize poorly. So it is important to balance amount of data for both classes.