**REPORT**

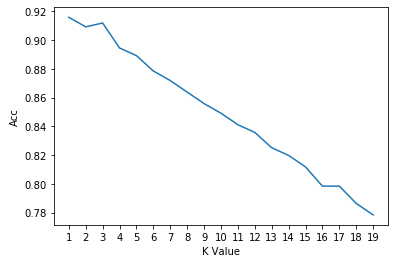
**User Identification Model:** This model is used to identify the user and not if the signature is valid or not. There are 100 users and hence the model predicts the user id based on input only genuine signatures are used to train the model. These Genuine signature have been split into training and testing data.

Training data shape:1747

Test data shape: 749

**K-Nearest Neighbor**

After training the model for various k values, these are the results recorded.



As observed in the graph, for k value 1 and 3 the accuracy is highest with values being 91.58% and 91.18% respectively.

**Neural Network**

**Train data set :1747 values**

**Validation Set : 249 values**

**Test Set : 500 values**

Model was trained for various hidden layers and other parameters. Model Specification can be found below:

**2 Hidden Layer:**

**1st hidden layer : 350 nodes , activation function relu.**

**2nd hidden layer: 150 nodes, activation function relu.**

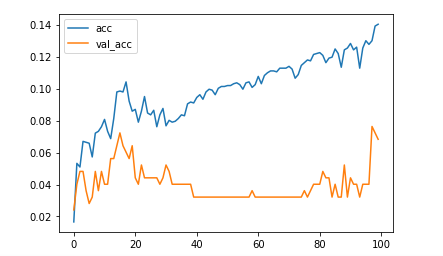
**Output layer: 100 nodes, activation function softmax.**

**Learning rate :** 0.0005 **, optimizer :** adam**, loss :** SparseCategoricalCrossentropy **,epoch :**100

Training acc : 14%

Validation acc: 7.5%

Test Acc : 6.4%



**3 Hidden Layers:**

**1st hidden layer : 350 nodes , activation function relu.**

**2nd hidden layer: 250 nodes, activation function relu.**

**3rd hidden layer:150nodes, activation function relu.**

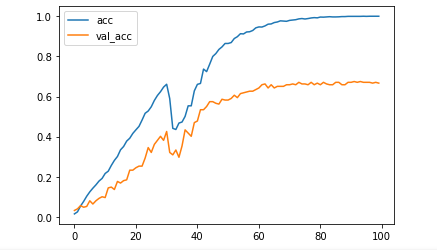
**Output layer: 100 nodes, activation function softmax.**

**Learning rate :** 0.0005 **, optimizer :** adam**, loss :** SparseCategoricalCrossentropy **,epoch :**100

Training acc : 100%

Validation acc: 67%

Test Acc : 66.2%



**4 Hidden Layers:**

**1st hidden layer : 350 nodes , activation function relu.**

**2nd hidden layer: 250 nodes, activation function relu.**

**3rd hidden layer:150 nodes, activation function relu.**

**4th hidden layer:125 nodes, activation function relu.**

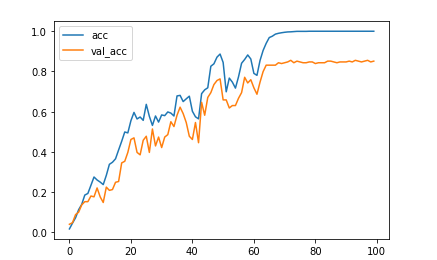
**Output layer: 100 nodes, activation function softmax.**

**Learning rate :** 0.0005 **, optimizer :** adam**, loss :** SparseCategoricalCrossentropy **,epoch :**100

Training acc : 100%

Validation acc: 85.5%

Test Acc : 85%



**4 Hidden Layers:**

**1st hidden layer : 350 nodes , activation function relu.**

**2nd hidden layer: 250 nodes, activation function relu.**

**3rd hidden layer:200 nodes, activation function relu.**

**4th hidden layer:150 nodes, activation function relu.**

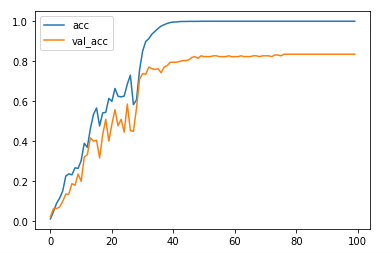
**Output layer: 100 nodes, activation function softmax.**

**Learning rate :** 0.0005 **, optimizer :** adam**, loss :** SparseCategoricalCrossentropy **,epoch :**100

Training acc : 100%

Validation acc: 83.5%

Test Acc : 81.2%



**5 Hidden Layers:**

**1st hidden layer : 350 nodes , activation function relu.**

**2nd hidden layer: 250 nodes, activation function relu.**

**3rd hidden layer:200 nodes, activation function relu.**

**4th hidden layer:150 nodes, activation function relu.**

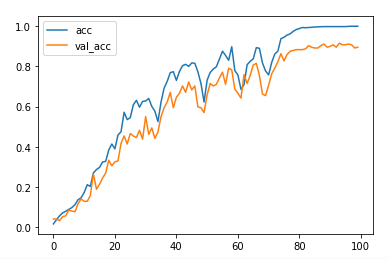
**5th hidden layer:125 nodes, activation function relu.**

**Output layer: 100 nodes, activation function softmax.**

**Learning rate :** 0.0005 **, optimizer :** adam**, loss :** SparseCategoricalCrossentropy **,epoch :**100

Training acc : 100%

Validation acc: 89.5%

Test Acc : 87%

**6 Hidden Layers:**

**1st hidden layer: 400 nodes, activation function relu.**

**2nd hidden layer: 350 nodes, activation function relu.**

**3rd hidden layer:250 nodes, activation function relu.**

**4th hidden layer:200 nodes, activation function relu.**

**5th hidden layer:150 nodes, activation function relu.**

**6th hidden layer:125 nodes, activation function relu.**

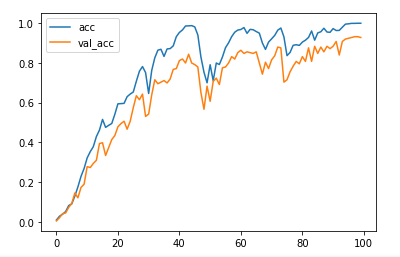
**Output layer: 100 nodes, activation function softmax.**

**Learning rate :** 0.0005 **, optimizer :** adam**, loss :** SparseCategoricalCrossentropy **,epoch :**100

Training acc : 99.6%

Validation acc: 92.3%

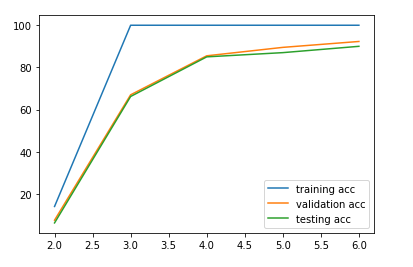
Test Acc : 90%



Summary:

User Accuracy Model

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | Specification | Training Acc | Validation Acc | Test Acc |
| KNN | 1 neighbor | - | - | 91.58 |
| KNN | 3 neighbor | - | - | 91.18 |
| DNN | 2 HL 350,150 | 14% | 7.5% | 6.4% |
| DNN | 3HL 350,250,150 | 100% | 67% | 66.2% |
| DNN | 4HL 350,250,150, 125 | 100% | 85.5% | 85% |
| DNN | 5HL 350, 250, 200, 150,125 | 100% | 89.5% | 87% |
| DNN | 6HL 400, 350 , 250, 200, 150,125 | 99.6% | 92.3% | 90% |



Fake – Genuine Model:

5 Hidden Layer model:

**1st hidden layer: 350 nodes, activation function relu.**

**2nd hidden layer:250 nodes, activation function relu.**

**3rd hidden layer:200 nodes, activation function relu.**

**4th hidden layer:100 nodes, activation function relu.**

**5th hidden layer:50 nodes, activation function relu.**

**Output layer: 100 nodes, activation function softmax.**

**Learning rate :** 0.0005 **, optimizer :** adam**, loss :** binary\_crossentropy **,epoch :**100

Acc : 83-85%

FAR : 6-8%

FFR : 8-10%

tn, fp, fn, tp

484 91 65 542

**DATA SET**

We have used MCYT dataset, which consists of 5000 dynamic signatures, collected from 100 users. Out of 5000 signatures, 2500 are genuine, and the rest 2500 are skilled forgeries. The data set provides the following discrete-time series data of each signature: i) x-axis position, ii) y-axis position, iii) pressure applied, iv) azimuth angle of the pen and v) pen inclination. The sampling rate at which the data has been captured is 100Hz. Using the given data, we have extracted additional features such as the time taken for signing, number of pen ups, average pressure, the ratio of the signature, velocity in x and y direction.

**DATA PRE-PROCESSING**

Since everyone has a unique signature, all the signatures have a different number of data points which cannot be directly used for training the model. All the signatures must have the same number of data points.

All of the data was imported to Python environment where several preprocessing techniques were used. To make the signature of equal length, either the signs could be up-sampled to the signature having the highest data points or down-sampled to the signature having the least number of data points. Both the methods have their merits and demerits. We ended up using both of the techniques. As we would be losing points while down sampling, we need to take care that the signature doesn’t lose its characteristics.

We first used the Ramer–Douglas–Peucker (RDP) algorithm on all the signature to only keep the most significant data points in each signature while also maintain the shape and characteristics of the signature. The length of each signature after this preprocessing will still be varying but the number of points is reduced by a significant amount. Now to make all the signature of equal length we selected the length of signature with highest number of points after RDP. All the signatures were then up sampled to that particular number of points. This allowed us to make all the signature of equal length while maintaining its characteristics and at the same time reducing the data. All the data was normalized so that all the signature were in the same range.