Bilkent University

CS 464 - Introduction to Machine Learning

Sign Language Detection

Project Report

Instructor : Ayşegül Dündar
Teaching Assistant: İlayda Beyreli Kokundu

21703955 Aral Saleyhan

21702401 Ipek Tüfekcioğlu

21704302 Ekin Doğaç Öztürk

21801347 Akmuhammet Ashyralyyev

1. INTRODUCTION

The goal of this project is to evaluate the results of various models on the MNIST Sign Language dataset. The dataset is the MNIST dataset in its original form. It includes csv files that hold the values of pixels as a characteristic of sign language pictures. The images were created in a grayscale format of 28x28 pixels. As a result, each image has 784 features. Despite the fact that American Sign Language has 26 letters, the dataset excludes J and Z letters. Samples with J or Z labels are likewise excluded from the Test and Train pictures. As a result of this problem, the dataset will be altered in order to remove the J and Z labels. There are 27455 training samples available. On the other hand, there are 7172 test cases for test purposes. The training sample will be used to select validation sets. The validation sets were selected to be one-fifth the size of the training sets. Convolutional Neural Network (CNN) and Neural Network were chosen as the models. After the models have been implemented, the performance will be evaluated to determine which model provides the best prediction. Each model's performance will be compared using a variety of performance criteria. Accuracy and learning time were chosen as the metrics. Furthermore, because the dataset contains 24 distinct labels, the confusion matrix will be used to assess the model's performance in terms of class prediction accuracy. Regularization approaches will be employed in order to increase the accuracy of the models on test data.

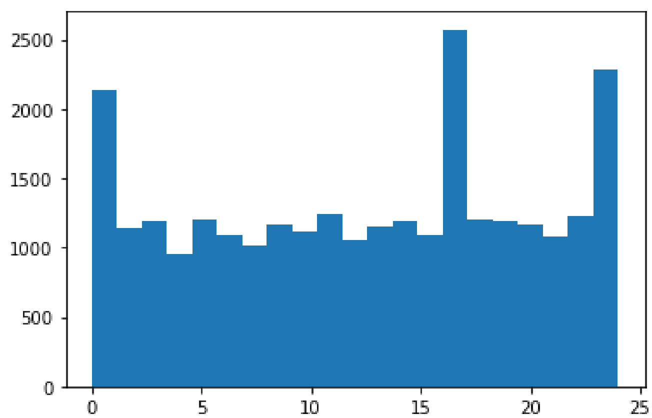


Figure1:Histogram of the dataset

2. PROBLEM DEFINITION

Nearly 600 000 individuals in the United States are deaf across all age categories. Approximately 6,000,000 individuals (2.2%) say they have "a lot of difficulties" hearing with. Over 28,000,000 people (or 10% of the population) say they have "a little difficulty." Over 35,000,000 individuals (13%) say they have some level of hearing problem which is a very high percentage of the population of the United States. [1] Therefore, understanding sign language is very crucial. In order to solve this problem. Therefore, the American sign language detection project matters. The solution for this is to obtain the best result implementing different models with different parameters (activation functions).

3. METHODS

Two methods are implemented.

Convolutional Neural Network (CNN)

CNN stands for "Convolutional Neural Network", and it is another model that we explored and implemented. [2] CNN is a type of neural network that is commonly utilized in image analysis. It has convolutional layers and pooling layers in addition to the Neural Network. These layers are necessary for recognizing the patterns (features) defined in convolution kernels, and the pooling layers minimize data dimensionality, resulting in fewer calculations. We did not utilize predetermined kernels in the development of this model, instead creating a model to discover the optimal ones together with the weights in the receptive field. The fully connected layers, receptive field, and weights are all based on Neural Network logic. This model is trained using gradient descent, an optimization algorithm for determining the best output, in this case weights. The default step size (predefined in TensorFlow) is 0.001 and the iteration number is 50. The rest of the statistical data in this model is as follows: This model has two convolutional filters, two max-pooling filters, one flattening layer, and one hidden layer.

Artificial Neural Network (ANNs)

The primary goal of ANNs is to tune weights (parameters) over time to achieve the best prediction accuracy [3]. Non-linearity is provided by activation functions in neurons. There may be several levels, which are depicted as follows:

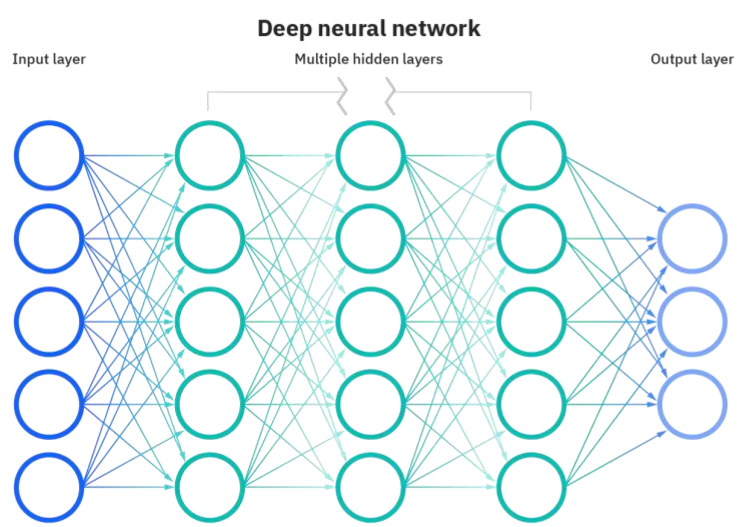


Figure 1 - Deep Neural Network schematic [3]

There are input, output, and hidden layers, as shown in Figure 1. If only a single neuron is examined:

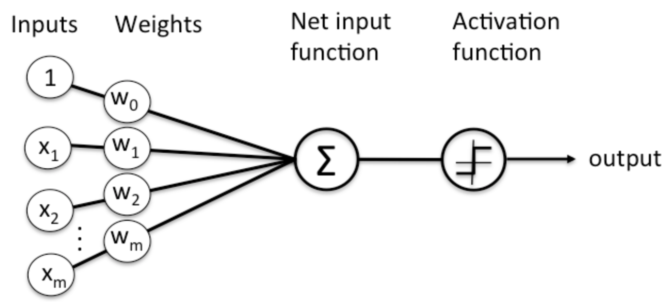


Figure 2: Neuron Schematic [4]

The equation can be stated as follows when it comes to the analytical examination:

$$output = \varphi(\sum_{i=1}^{m} x_i w_i)$$

where $\varphi(.)$ is activation function

The Adam optimizer was utilized, as well as the MSE loss function. The Backpropagation Algorithm was utilized during the learning step. In the hidden layers for the activation function, the relu, parametric relu and sigmoid functions were used as an intermediate layer in order to do classification, and SoftMax was used in the last layer.

Adam Optimizer

In both models, for optimization, Adam optimizer was used. It is a combination of Adaptive Gradient Algorithm (AdaGrad) and Root mean square (RMSProp). It calculates the exponential moving average of the gradient and it also calculates the squared gradient []. The average of the first moment and the average of the second moment of the gradient are used. The moving averages are as follows:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$$
$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$$

In those models' parameters selected as
learning rate: 0.001
Beta1 = 0.9
Beta2 = 0.999
Epsilon = 10^{-7}

Early Stopping

It is a regularization method. As the model converges, training is stopped. The convergence stage is decided by evaluating the models' performance on the validation dataset. If the performance decreases or does not change, training is stopped. The number of epochs before stopping is chosen as 3 (by trial and error) for models used in this project.

Dropout

It is a regularization method. It drops out the neurons with probability 1-p at each stage. As the neurons are removed the weights that are connected to those neurons are removed at a particular stage. Therefore, the complexity decreases. Additionally, overfitting is prevented. In this project p value is selected as 0.2 (trial and error).

4. RESULTS

The filters that have been used in CNN were learnt by the models. Since we couldn't give meaning to them, we didn't discuss the effect of the CNN models. You can find visualized kernel in Appendix 1.

4.1. CNN Model without Dropouts

The detailed output of CNN Model without dropouts that has been obtained from training can be seen in Appendix 2.

4.1.1 Summary of CNN Model without Dropouts

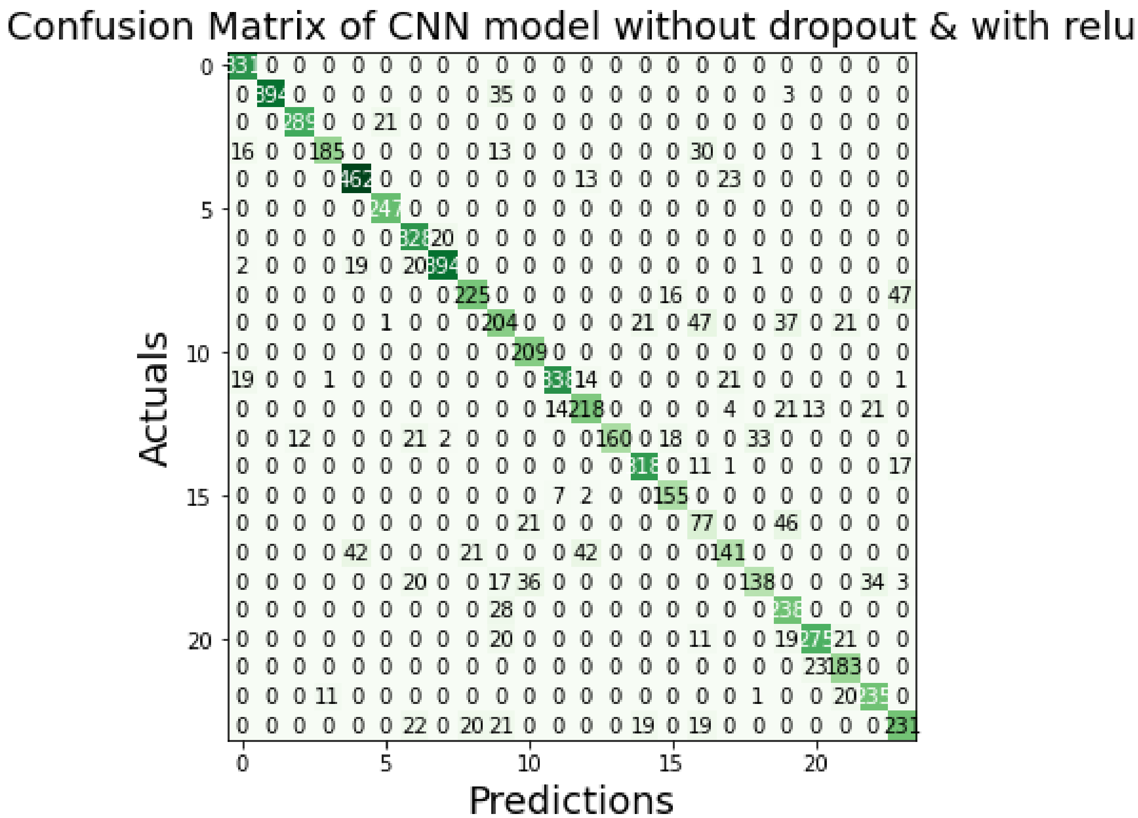
Layer (type)	Output Shape	Param #
First filters (Conv2D)	(None, 26, 26, 3)	30
Batch_normalization_1 (BatchNormalization)	(None, 26, 26, 3)	12
MaxPool1(MaxPooling2D)	(None, 13, 13, 3)	0
Second filters (Conv2D)	(None, 11, 11, 3)	84
Batch_normalization_2 (BatchNormalization)	(None, 13, 13, 3)	12
MaxPool2 (MaxPooling2D)	(None, 5, 5, 3)	0
Flatten of Convs Output (Flatten)	(None, 75)	0
Hidden_Layer_1 (Dense)	(None, 1024)	77824
Hidden_Layer_2 (Dense)	(None, 512)	524800
Output_Layer (Dense)	(None, 24)	12312

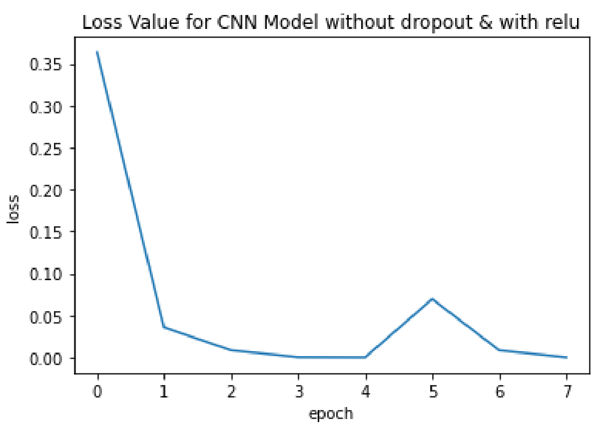
Total params: 615,074
Trainable params: 615,062

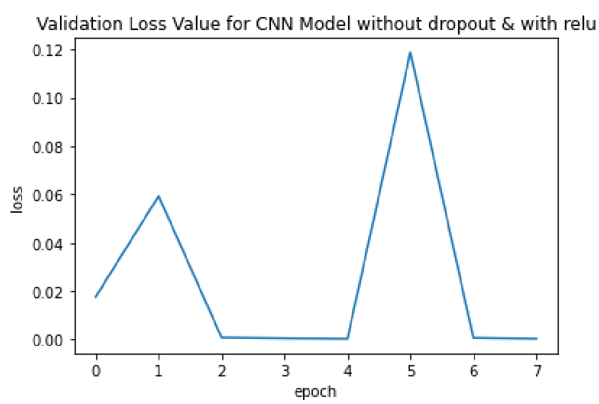
Non-trainable params: 12

	ReLU	Parametric ReLU	Sigmoid
Epoch Number	8	17	24
Train Time (secs)	101.51	230.45	329.69
Train time per Epoch	12.70	13.56	13.74

	ReLU	Parametric ReLU	Sigmoid
Test Accuracy	0.83	0.82	0.84



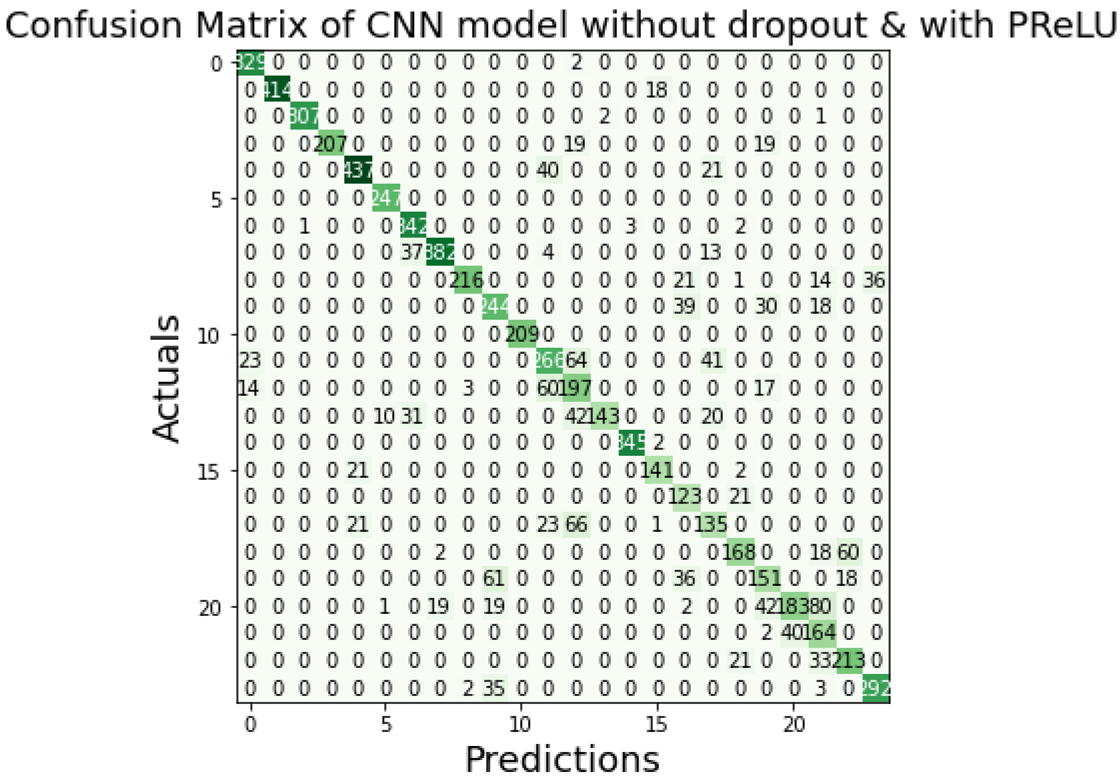


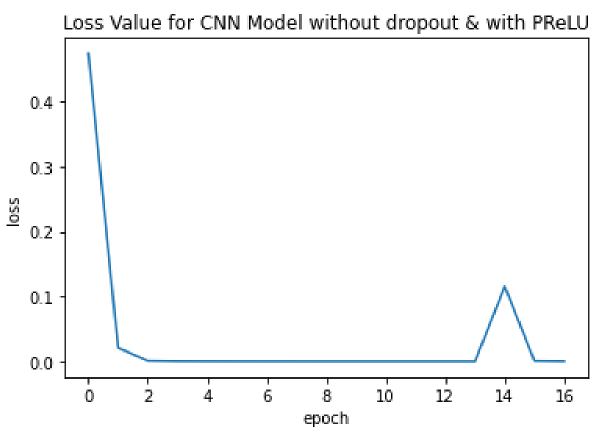


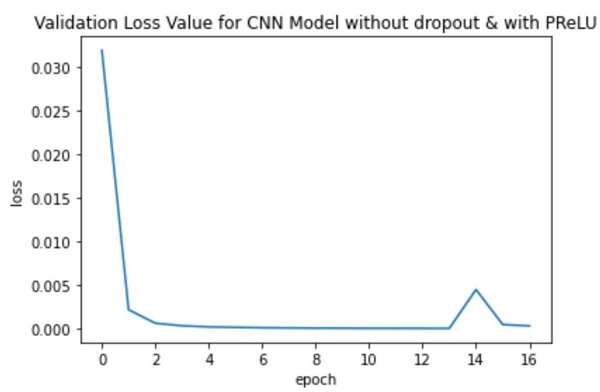
	precision	recall	f1-score	support
a	0.90	1.00	0.95	331
b	1.00	0.91	0.95	432
c	0.96	0.93	0.95	310
d	0.94	0.76	0.84	245

e	0.88	0.93	0.90	498
f	0.92	1.00	0.96	247
g	0.80	0.94	0.86	348
h	0.95	0.90	0.92	436
i	0.85	0.78	0.81	288
k	0.60	0.62	0.61	331
l	0.79	1.00	0.88	209
m	0.94	0.86	0.90	394
n	0.75	0.75	0.75	291
o	1.00	0.65	0.79	246
p	0.89	0.92	0.90	347
q	0.82	0.95	0.88	164
r	0.39	0.53	0.45	144
s	0.74	0.57	0.65	246
t	0.80	0.56	0.66	248
u	0.65	0.89	0.76	266
v	0.88	0.79	0.84	346
w	0.75	0.89	0.81	206
x	0.81	0.88	0.84	267
y	0.77	0.70	0.73	332

accuracy: 0.83 7172
macro avg 0.82 0.82 0.82 7172
weighted avg 0.84 0.83 0.83 7172



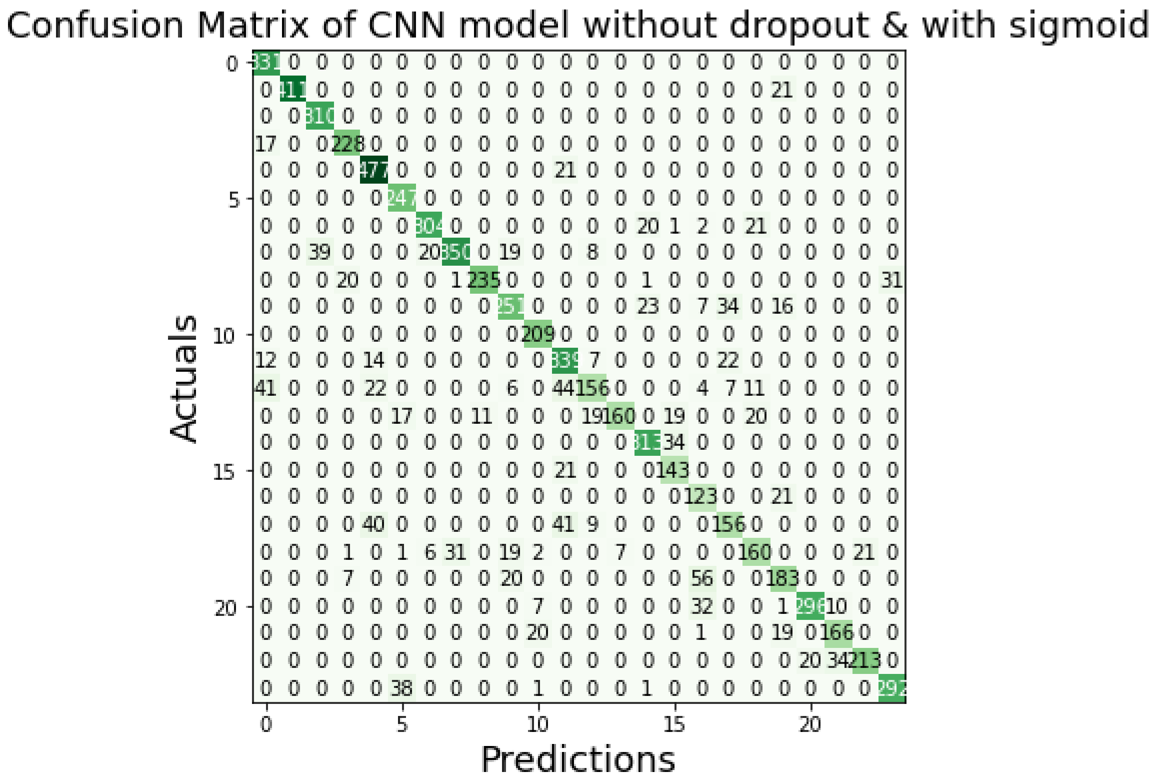


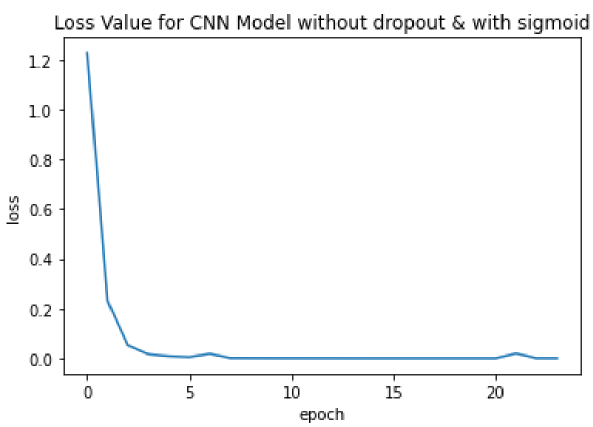


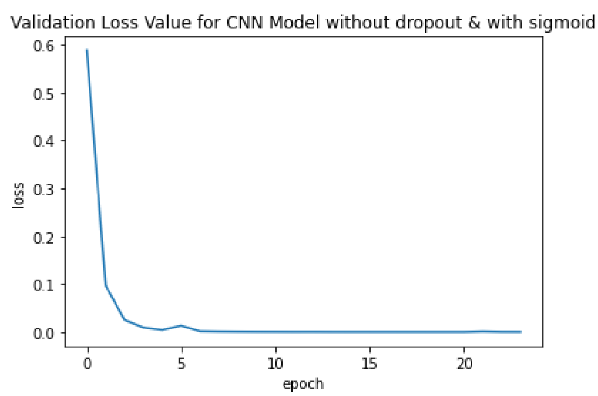
	precision	recall	f1-score	support
a	0.90	0.99	0.94	331
b	1.00	0.96	0.98	432
c	1.00	0.99	0.99	310
d	1.00	0.84	0.92	245
e	0.91	0.88	0.89	498

f	0.96	1.00	0.98	247
g	0.83	0.98	0.90	348
h	0.95	0.88	0.91	436
i	0.98	0.75	0.85	288
k	0.68	0.74	0.71	331
l	1.00	1.00	1.00	209
m	0.68	0.68	0.68	394
n	0.51	0.68	0.58	291
o	0.99	0.58	0.73	246
p	0.99	0.99	0.99	347
q	0.87	0.86	0.87	164
r	0.56	0.85	0.67	144
s	0.59	0.55	0.67	246
t	0.78	0.68	0.73	248
u	0.58	0.57	0.57	266
v	0.82	0.53	0.64	346
w	0.50	0.80	0.61	206
x	0.73	0.80	0.76	267
y	0.89	0.88	0.88	332

accuracy 0.82 7172
macro avg 0.82 0.81 0.81 7172
weighted avg 0.83 0.82 0.82 7172







	precision	recall	f1-score	support
a	0.83	1.00	0.90	331
b	1.00	0.95	0.98	432
c	0.89	1.00	0.94	310
d	0.89	0.93	0.91	245
e	0.86	0.96	0.91	498
f	0.82	1.00	0.90	247

g	0.92	0.87	0.90	348
h	0.92	0.80	0.86	436
i	0.96	0.82	0.88	288
k	0.80	0.76	0.78	331
l	0.87	1.00	0.93	209
m	0.73	0.86	0.79	394
n	0.78	0.54	0.64	291
o	0.96	0.65	0.77	246
p	0.87	0.90	0.89	347
q	0.73	0.87	0.79	164
r	0.55	0.85	0.67	144
s	0.71	0.63	0.67	246
t	0.75	0.65	0.70	248
u	0.70	0.69	0.69	266
v	0.94	0.86	0.89	346
w	0.79	0.81	0.80	206
x	0.91	0.80	0.85	267
y	0.90	0.88	0.89	332

accuracy 0.84 7172
macro avg 0.84 0.84 0.83 7172
weighted avg 0.85 0.84 0.84 7172

4.2. CNN Model with Dropouts

The detailed output of CNN Model with dropouts that has been obtained from training can be seen in Appendix 3

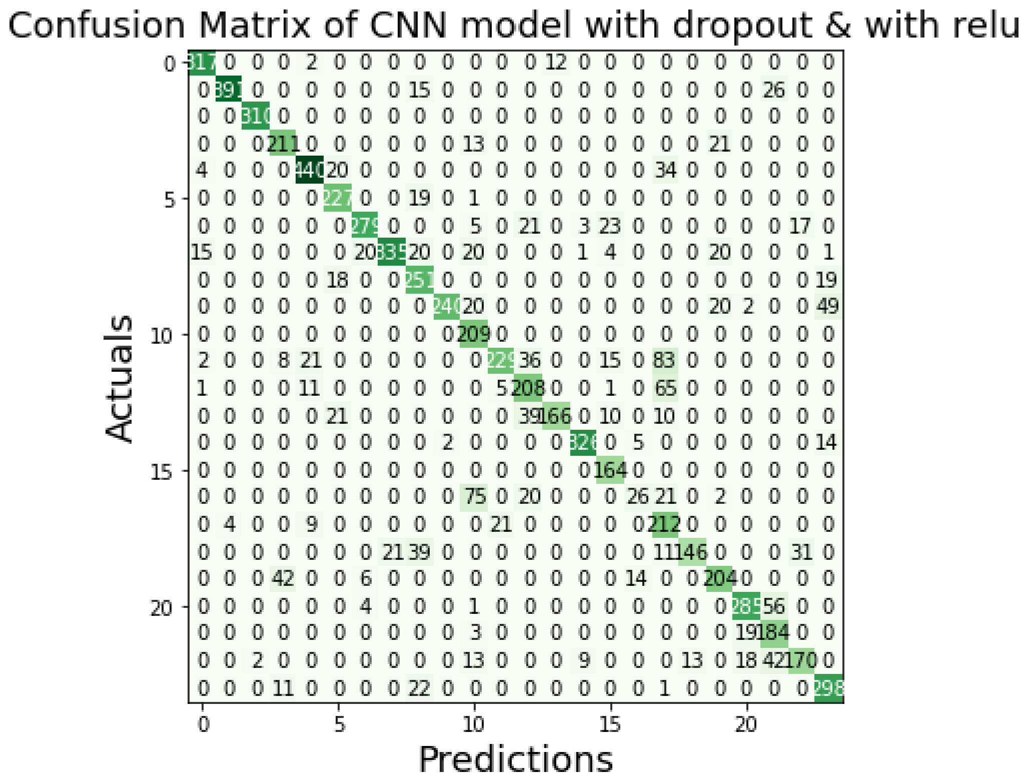
4.2.1. Summary of CNN Model with Dropouts

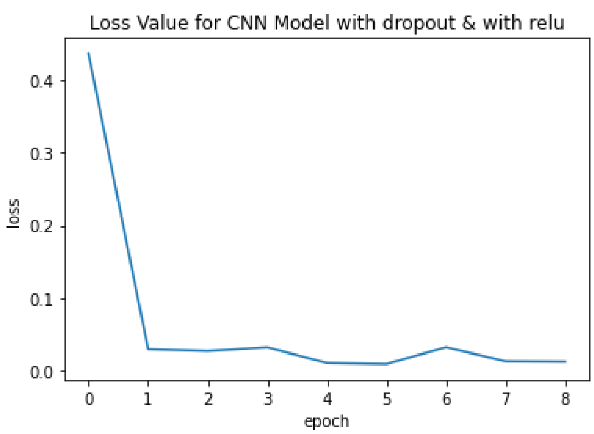
Layer (type)	Output Shape	Param #
First_filters (Conv2D)	(None, 26, 26, 3)	30
Batch_normalization_1 (BatchNormalization)	(None, 26, 26, 3)	12
MaxPool1(MaxPooling2D)	(None, 13, 13, 3)	0
Secon_filters (Conv2D)	(None, 11, 11, 3)	84
Batch_normalization_2 (BatchNormalization)	(None, 13, 13, 3)	12
MaxPool2 (MaxPooling2D)	(None, 5, 5, 3)	0
Flatten_of_Convs_Output (Flatten)	(None, 75)	0
Hidden_Layer_1 (Dense)	(None, 1024)	77824
Dropout_1 (Dropout)	(None, 1024)	0
Hidden_Layer_2 (Dense)	(None, 512)	524800
Dropoit_2 (Dropout)	(None, 512)	0
Output_Layer (Dense)	(None, 24)	12312

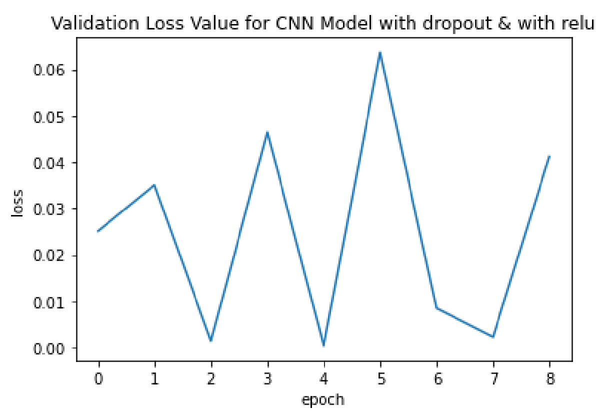
Total params: 615,074
Trainable params: 615,062
Non-trainable params: 12

	ReLU	**Parametric ReLU**	**Sigmoid**
Epoch Number	9	10	20
Train Time (secs)	117,02	141,10	312.08
Train time per Epoch	13.00	14.11	15.60

	ReLU	**Parametric ReLU**	**Sigmoid**
Test Accuracy	0.81	0.86	0.88



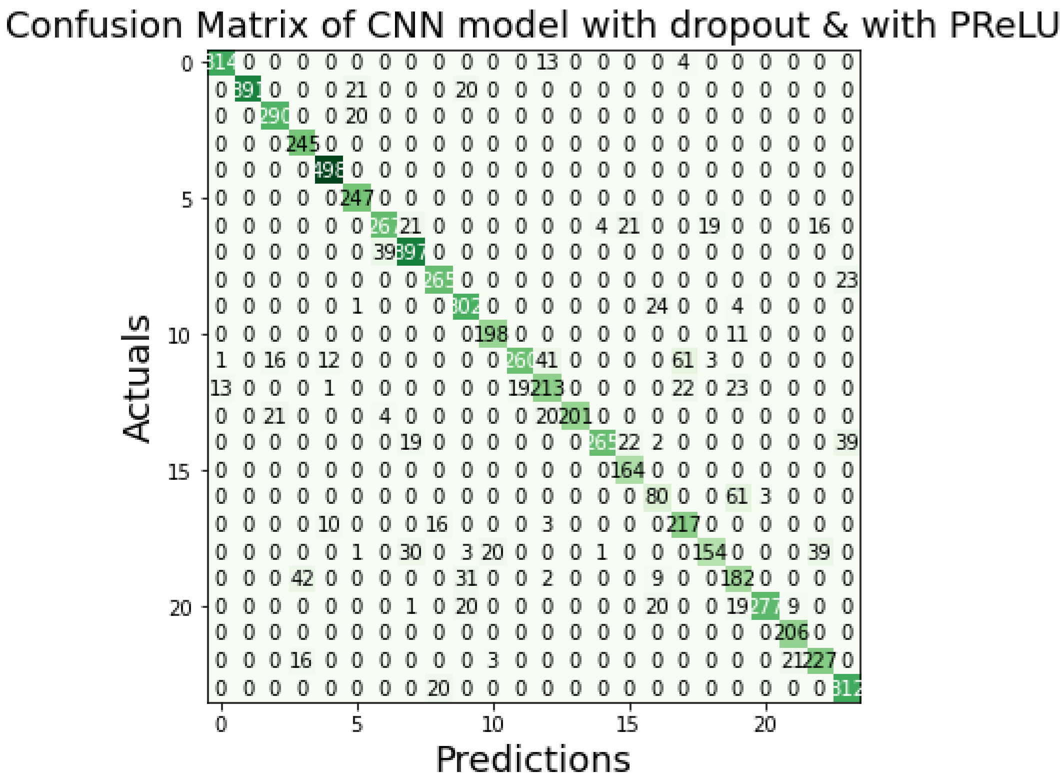


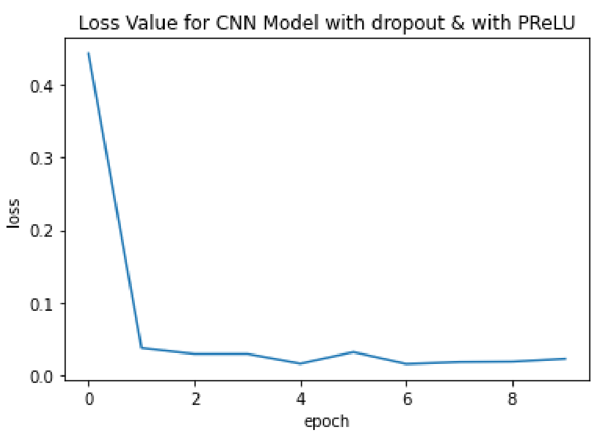


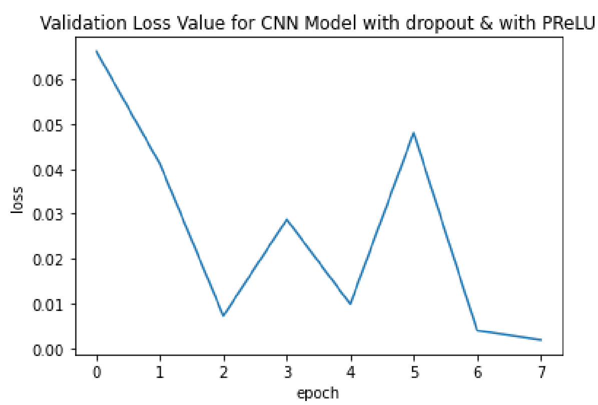
	precision	recall	f1-score	support
a	0.94	0.96	0.95	331
b	0.99	0.91	0.95	432
c	0.99	1.00	1.00	310
d	0.78	0.86	0.82	245
e	0.91	0.88	0.90	498
f	0.79	0.92	0.85	247

g	0.90	0.80	0.85	348
h	0.94	0.77	0.85	436
i	0.69	0.87	0.77	288
k	0.99	0.73	0.84	331
l	0.58	1.00	0.73	209
m	0.90	0.58	0.71	394
n	0.64	0.71	0.68	291
o	0.93	0.67	0.78	246
p	0.96	0.94	0.95	347
q	0.76	1.00	0.86	164
r	0.58	0.18	0.28	144
s	0.49	0.86	0.62	246
t	0.92	0.59	0.72	248
u	0.76	0.77	0.77	266
v	0.88	0.82	0.85	346
w	0.60	0.89	0.72	206
x	0.78	0.64	0.70	267
y	0.78	0.90	0.84	332

accuracy 0.81 7172
macro avg 0.81 0.80 0.79 7172
weighted avg 0.84 0.81 0.81 7172



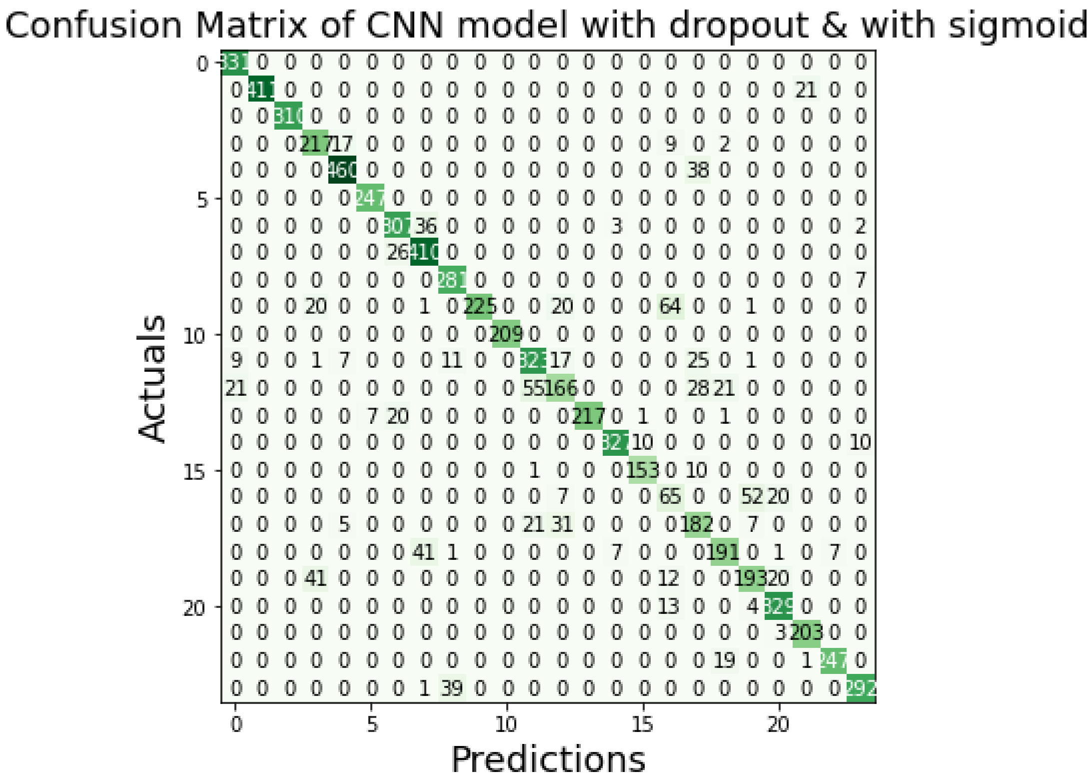


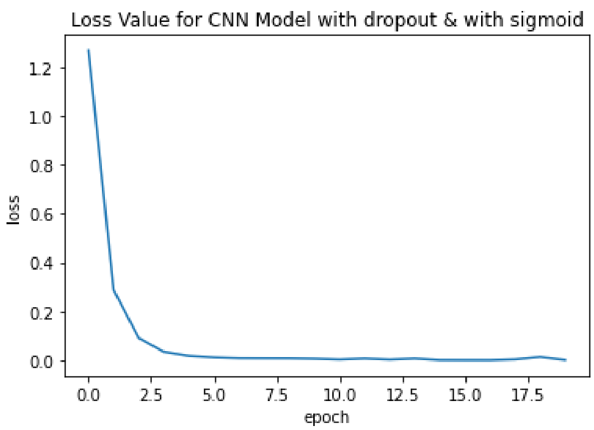


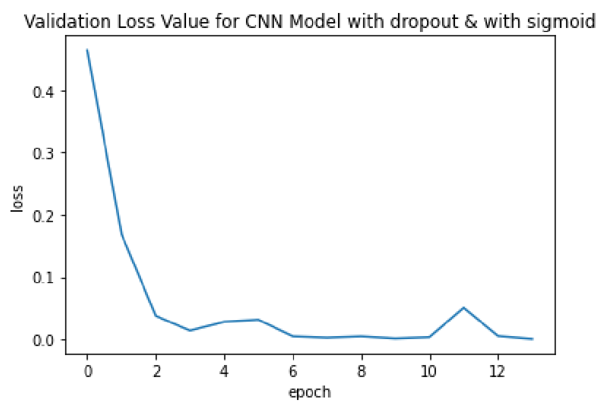
	precision	recall	f1-score	support
a	0.96	0.95	0.95	331
b	1.00	0.91	0.95	432
c	0.89	0.94	0.91	310
d	0.81	1.00	0.89	245
e	0.96	1.00	0.98	498
f	0.85	1.00	0.92	247

g	0.86	0.77	0.81	348
h	0.85	0.91	0.88	436
i	0.88	0.92	0.90	288
k	0.80	0.91	0.85	331
l	0.90	0.95	0.92	209
m	0.93	0.66	0.77	394
n	0.73	0.73	0.73	291
o	1.00	0.82	0.90	246
p	0.98	0.76	0.86	347
q	0.79	1.00	0.88	164
r	0.59	0.56	0.57	144
s	0.71	0.88	0.79	246
t	0.88	0.62	0.73	248
u	0.61	0.68	0.64	266
v	0.99	0.80	0.88	346
w	0.87	1.00	0.93	206
x	0.80	0.85	0.83	267
y	0.83	0.94	0.88	332

accuracy 0.86 7172
macro avg 0.85 0.86 0.85 7172
weighted avg 0.87 0.86 0.86 7172







	precision	recall	f1-score	support
0	0.92	1.00	0.96	331
1	1.00	0.95	0.98	432
2	1.00	1.00	1.00	310
3	0.78	0.89	0.83	245
4	0.94	0.92	0.93	498
5	0.97	1.00	0.99	247

6	0.87	0.88	0.88	348
7	0.84	0.94	0.89	436
8	0.85	0.98	0.91	288
9	1.00	0.68	0.81	331
10	1.00	1.00	1.00	209
11	0.81	0.82	0.81	394
12	0.69	0.57	0.62	291
13	1.00	0.88	0.94	246
14	0.97	0.94	0.96	347
15	0.93	0.93	0.93	164
16	0.40	0.45	0.42	144
17	0.64	0.74	0.69	246
18	0.82	0.77	0.79	248
19	0.75	0.73	0.74	266
20	0.88	0.95	0.92	346
21	0.90	0.99	0.94	206
22	0.97	0.93	0.95	267
23	0.94	0.88	0.91	332

accuracy 0.88 7172
macro avg 0.87 0.87 0.87 7172
weighted avg 0.88 0.88 0.88 7172

4.3. ANN Model without Dropouts

The detailed output of ANN Model without dropouts that has been obtained from training can be seen in Appendix 4

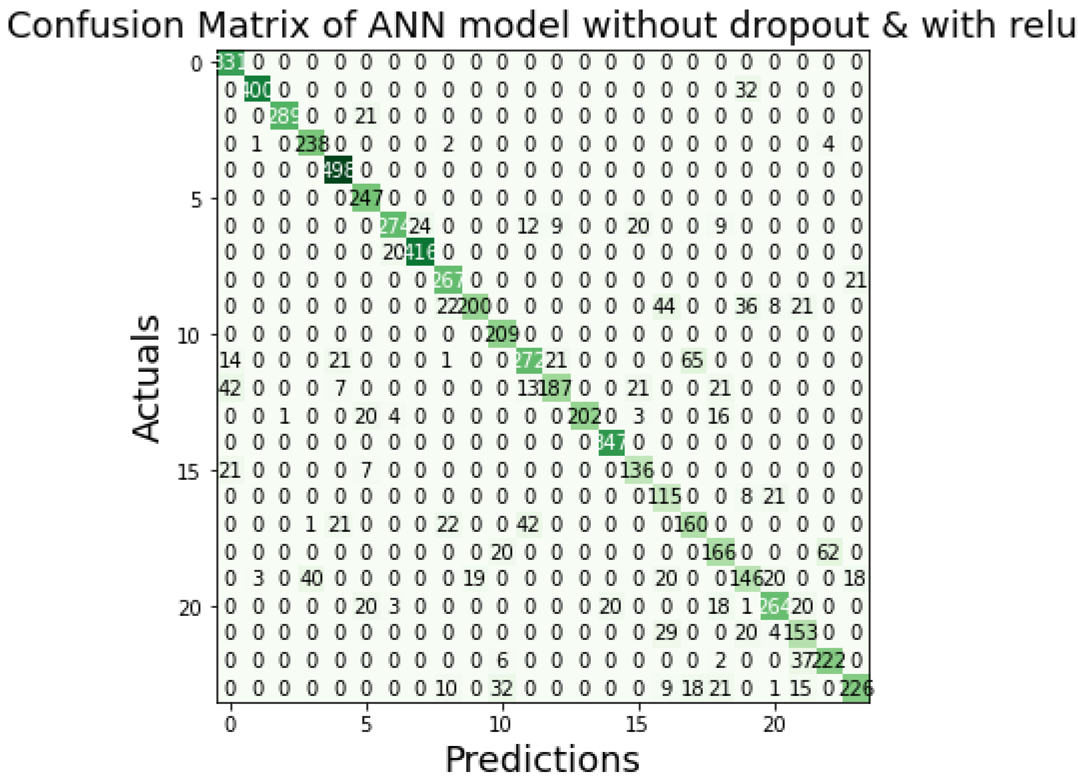
4.3.1. Summary of ANN Model without Dropouts

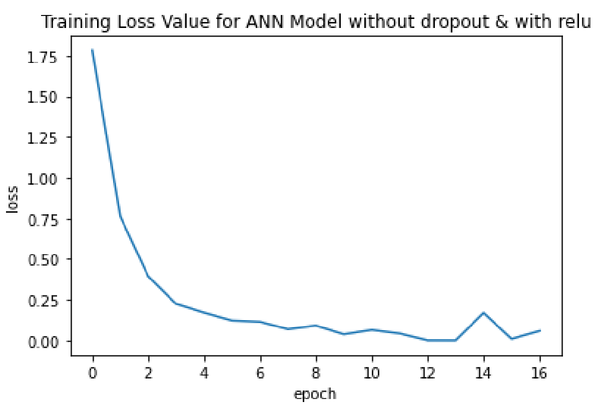
Layer (type)	Output Shape	Param #
Hidden_Layer_1 (Dense)	(None, 1024)	803840
Hidden_Layer_2 (Dense)	(None, 512)	524800
Output_Layer (Dense)	(None, 24)	12312

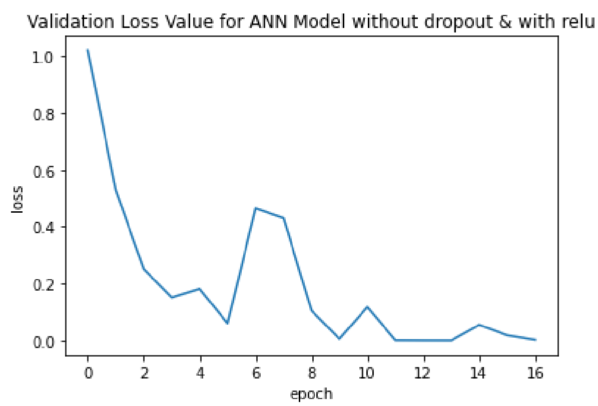
Total params: 1,340,952
Trainable params: 1,340,952
Non-trainable params: 0

	ReLU	**Parametric ReLU**	**Sigmoid**
Epoch Number	17	15	13
Train Time (secs)	167.58	153.41	124.04
Duration of Each Epoch (secs)	9.86	10.23	9.54

	ReLU	**Parametric ReLU**	**Sigmoid**
Test Accuracy	0.83	0.84	0.81



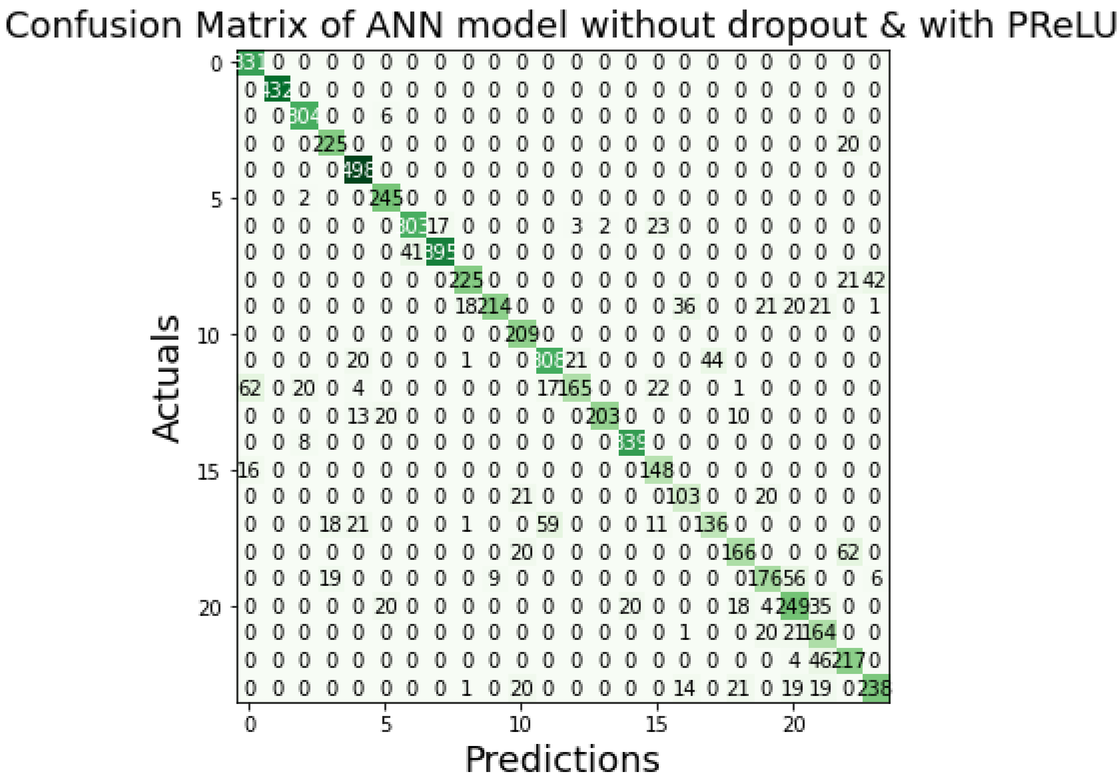


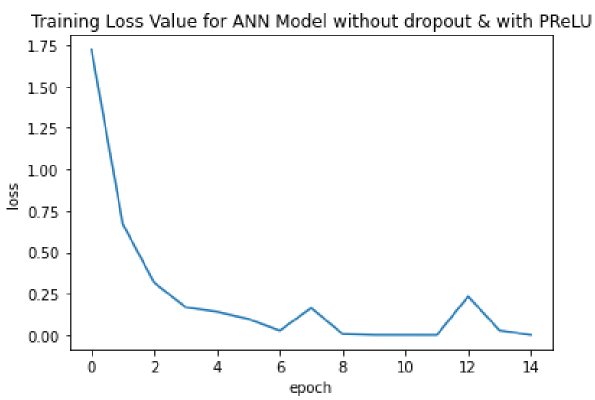


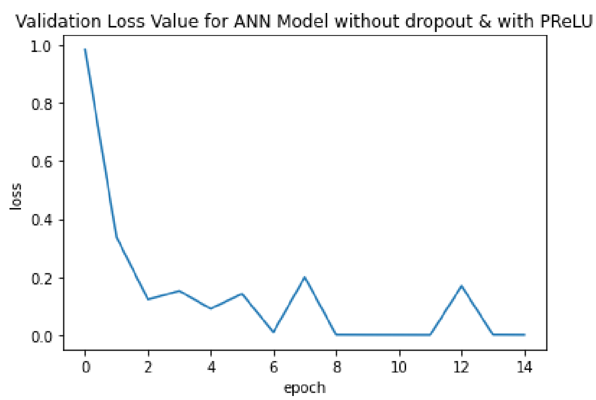
	precision	recall	f1-score	support
a	0.81	1.00	0.90	331
b	0.99	0.93	0.96	432
c	1.00	0.93	0.96	310
d	0.85	0.97	0.91	245
e	0.91	1.00	0.95	498
f	0.78	1.00	0.88	247

g	0.91	0.79	0.84	348
h	0.95	0.95	0.95	436
i	0.82	0.93	0.87	288
k	0.91	0.60	0.73	331
l	0.78	1.00	0.88	209
m	0.80	0.69	0.74	394
n	0.86	0.64	0.74	291
o	1.00	0.82	0.90	246
p	0.95	1.00	0.97	347
q	0.76	0.83	0.79	164
r	0.53	0.80	0.64	144
s	0.66	0.65	0.65	246
t	0.66	0.67	0.66	248
u	0.60	0.55	0.57	266
v	0.83	0.76	0.80	346
w	0.62	0.74	0.68	206
x	0.77	0.83	0.80	267
y	0.85	0.68	0.76	332

accuracy: 0.83 7172
macro avg: 0.82 0.82 0.81 7172
weighted avg: 0.84 0.83 0.83 7172



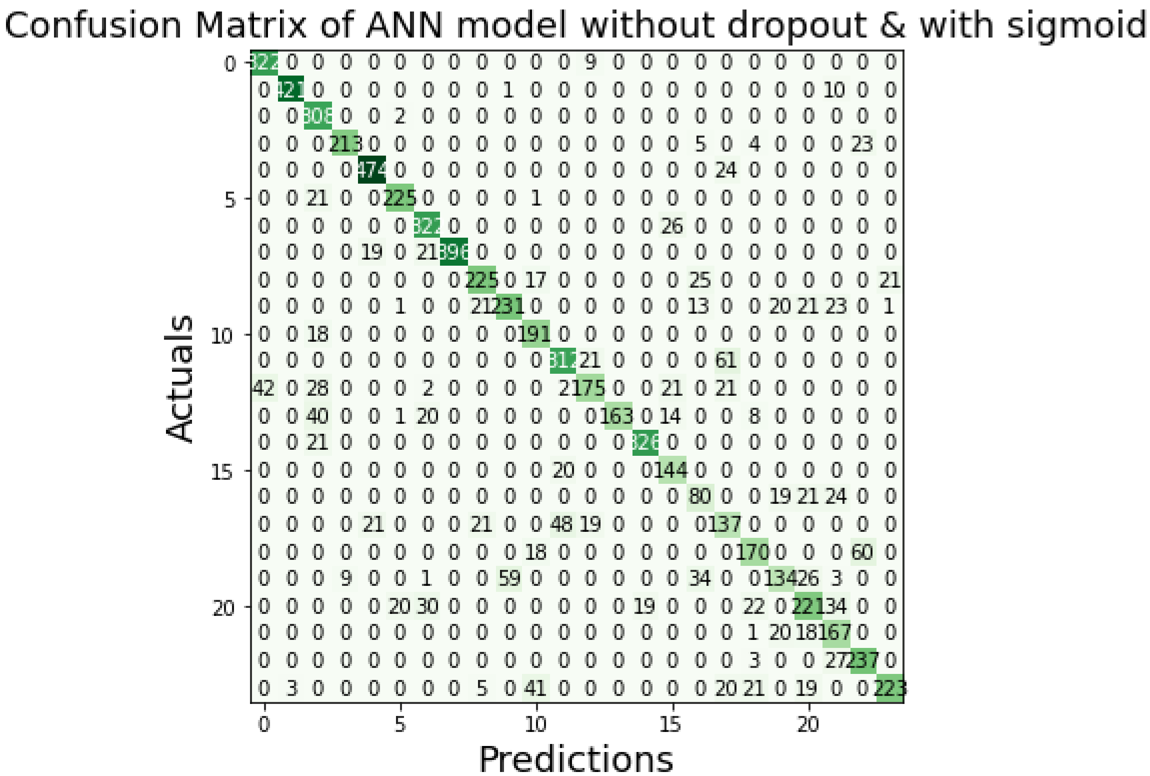


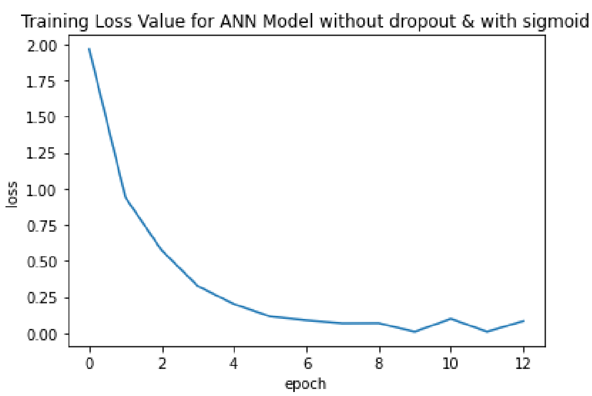


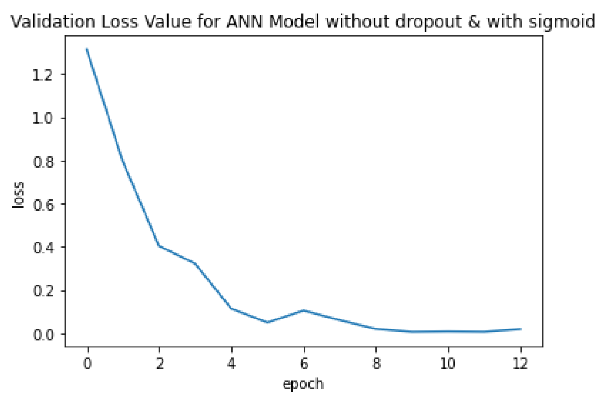
	precision	recall	f1-score	support
a	0.81	1.00	0.89	331
b	1.00	1.00	1.00	432
c	0.91	0.98	0.94	310
d	0.86	0.92	0.89	245
e	0.90	1.00	0.94	498
f	0.84	0.99	0.91	247

g	0.88	0.87	0.88	348
h	0.96	0.91	0.93	436
i	0.91	0.78	0.84	288
k	0.96	0.65	0.77	331
l	0.77	1.00	0.87	209
m	0.80	0.78	0.79	394
n	0.87	0.57	0.69	291
o	0.99	0.83	0.90	246
p	0.94	0.98	0.96	347
q	0.73	0.90	0.80	164
r	0.67	0.72	0.69	144
s	0.76	0.55	0.64	246
t	0.77	0.67	0.72	248
u	0.73	0.66	0.69	266
v	0.67	0.72	0.70	346
w	0.58	0.80	0.67	206
x	0.68	0.81	0.74	267
y	0.83	0.72	0.77	332

accuracy:			0.84	7172
macro avg:	0.83	0.82	0.82	7172
weighted avg:	0.84	0.84	0.83	7172







	precision	recall	f1-score	support
a	0.88	0.97	0.93	331
b	0.99	0.97	0.98	432
c	0.71	0.99	0.83	310
d	0.96	0.87	0.91	245
e	0.92	0.95	0.94	498
f	0.90	0.91	0.91	247

g	0.81	0.93	0.87	348
h	1.00	0.91	0.95	436
i	0.83	0.78	0.80	288
k	0.79	0.70	0.74	331
l	0.71	0.91	0.80	209
m	0.82	0.79	0.80	394
n	0.78	0.60	0.68	291
o	1.00	0.66	0.80	246
p	0.94	0.94	0.94	347
q	0.70	0.88	0.78	164
r	0.51	0.56	0.53	144
s	0.52	0.56	0.54	246
t	0.74	0.69	0.71	248
u	0.69	0.50	0.58	266
v	0.68	0.64	0.66	346
w	0.58	0.81	0.68	206
x	0.74	0.89	0.81	267
y	0.91	0.67	0.77	332

accuracy:			0.81	7172
macro avg:	0.80	0.80	0.79	7172
weighted avg:	0.82	0.81	0.81	7172

4.4. ANN Model with Dropouts

The detailed output of ANN Model with dropouts that has been obtained from training can be seen in Appendix 5

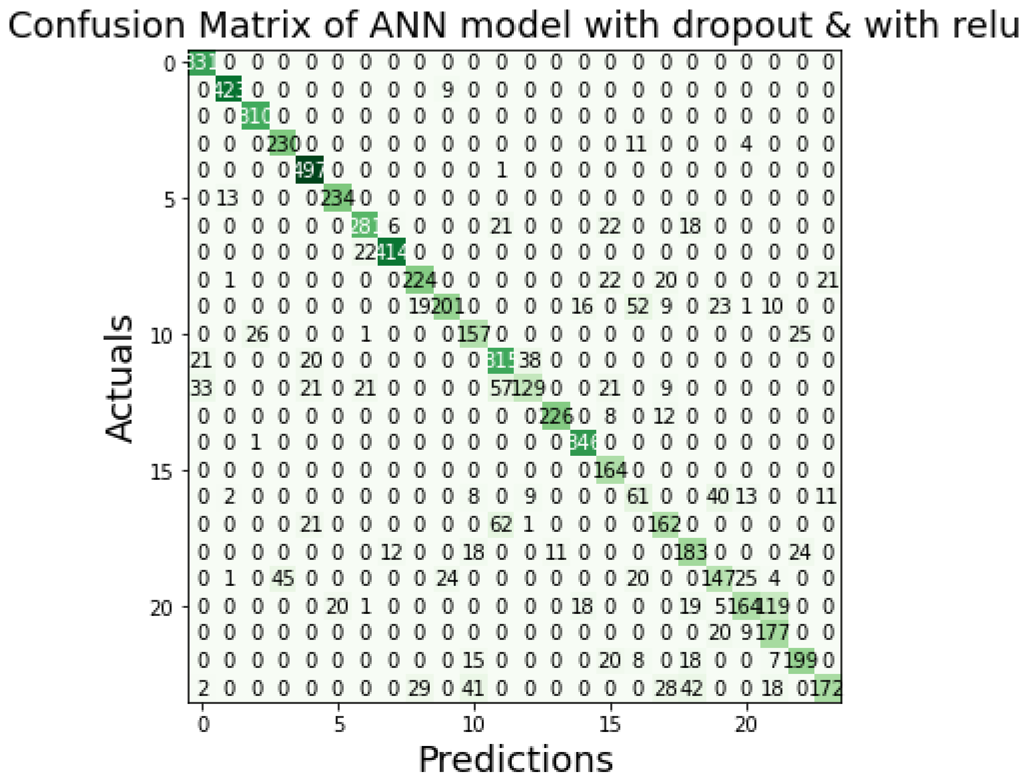
4.4.1. Summary of ANN Model with Dropouts

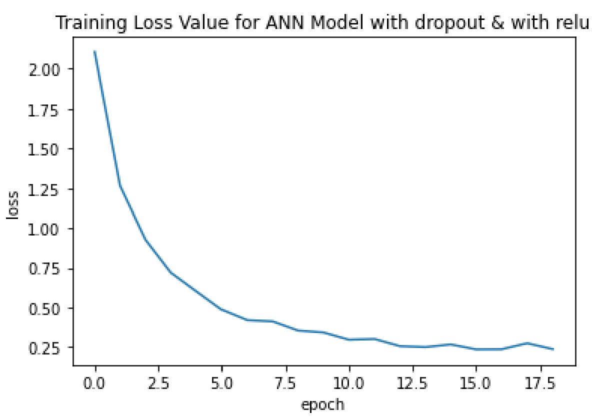
Layer (type)	Output Shape	Param #
Hidden_Layer_1 (Dense)	(None, 1024)	803840
Dropout_Layer_1 (Dropout)	(None, 1024)	0
Hidden_Layer_2 (Dense)	(None, 512)	524800
Dropout_Layer_2 (Dropout)	(None, 512)	0
Output_Layer (Dense)	(None, 24)	12312

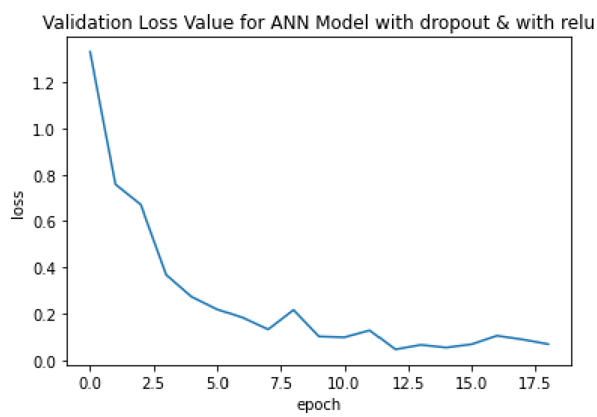
Total params: 1,340,952
Trainable params: 1,340,952
Non-trainable params: 0

	ReLU	**Parametric ReLU**	**Sigmoid**
Epoch Number	19	17	19
Train Time (secs)	185.10	179.41	188.11
Duration of Each Epoch (secs)	9.74	10.55	9.90

	ReLU	**Parametric ReLU**	**Sigmoid**
Test Accuracy	0.80	0.81	0.82



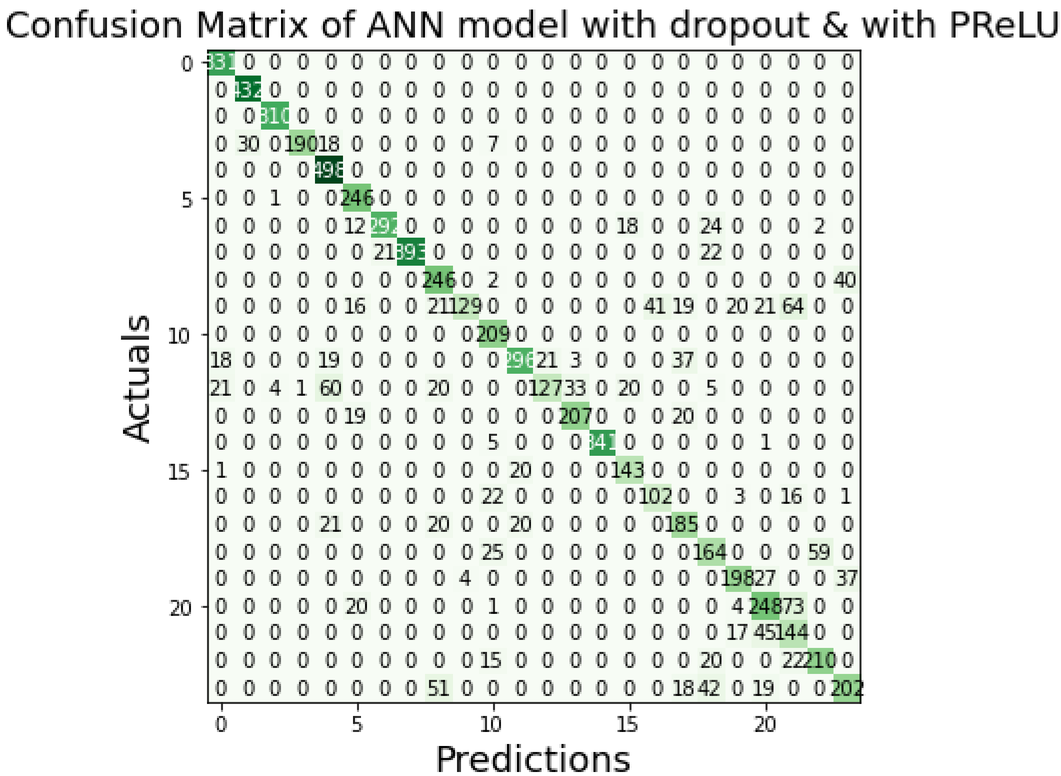


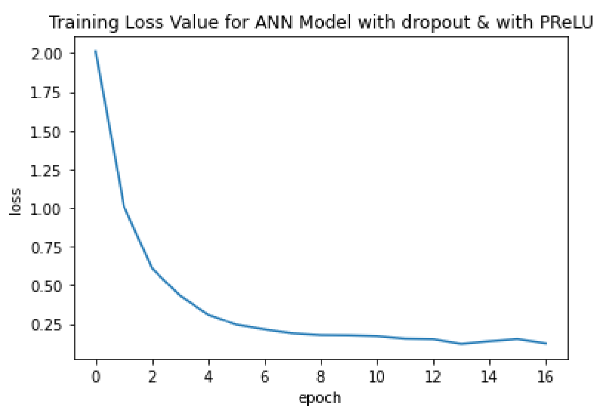


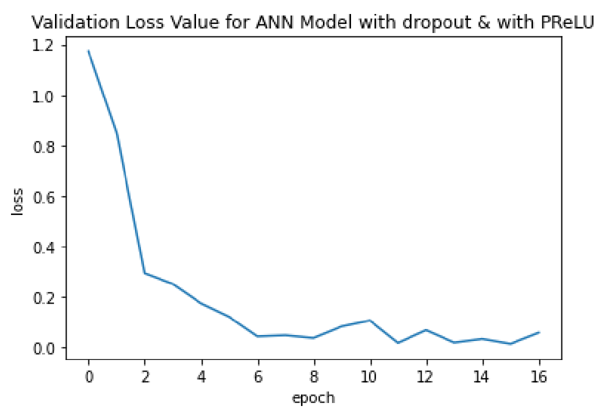
	precision	recall	f1-score	support
a	0.86	1.00	0.92	331
b	0.96	0.98	0.97	432
c	0.92	1.00	0.96	310
d	0.84	0.94	0.88	245
e	0.89	1.00	0.94	498
f	0.92	0.95	0.93	247

g	0.86	0.81	0.83	348
h	0.96	0.95	0.95	436
i	0.82	0.78	0.80	288
k	0.86	0.61	0.71	331
l	0.66	0.75	0.70	209
m	0.69	0.80	0.74	394
n	0.73	0.44	0.55	291
o	0.95	0.92	0.94	246
p	0.91	1.00	0.95	347
q	0.64	1.00	0.78	164
r	0.40	0.42	0.41	144
s	0.68	0.66	0.67	246
t	0.65	0.74	0.69	248
u	0.63	0.55	0.59	266
v	0.76	0.47	0.58	346
w	0.53	0.86	0.65	206
x	0.80	0.75	0.77	267
y	0.84	0.52	0.64	332

accuracy:			0.80	7172
macro avg:	0.78	0.79	0.77	7172
weighted avg:	0.81	0.80	0.80	7172



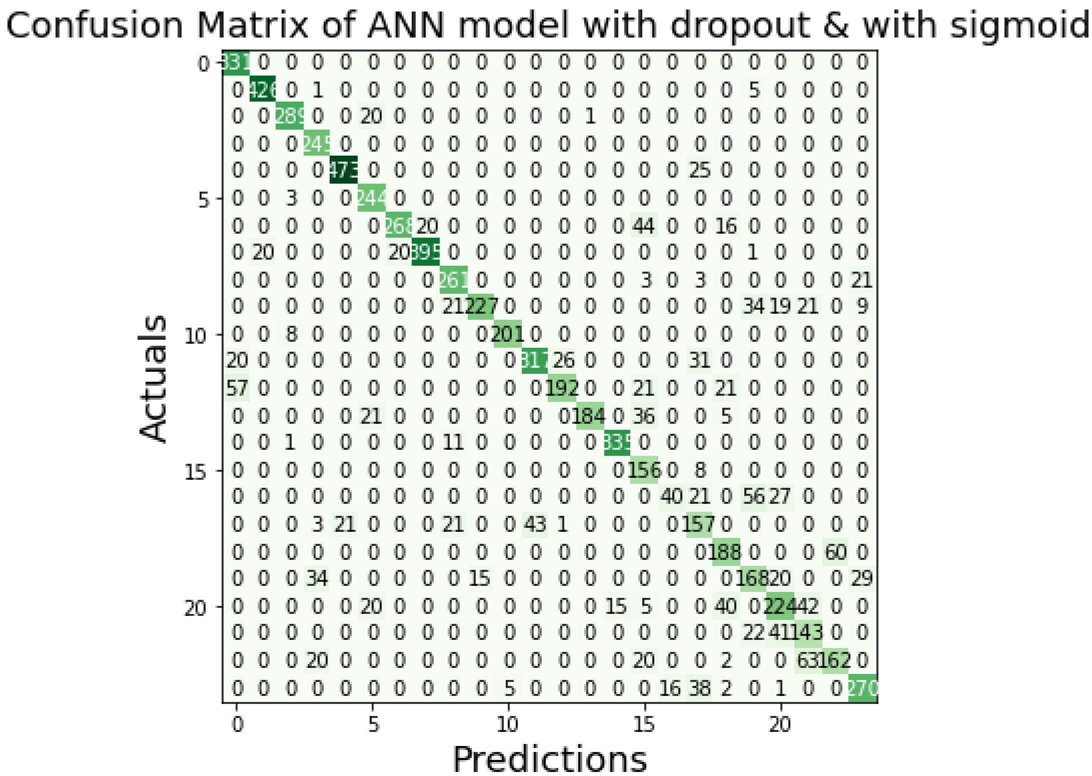


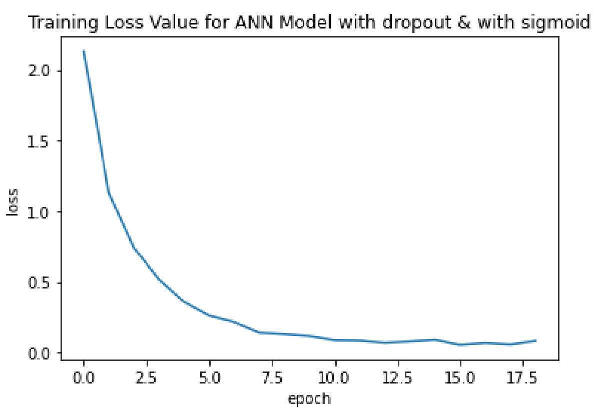


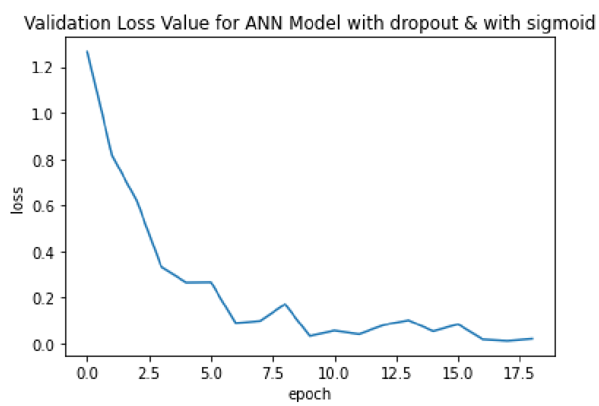
	precision	recall	f1-score	support
a	0.89	1.00	0.94	331
b	0.94	1.00	0.97	432
c	0.98	1.00	0.99	310
d	0.99	0.78	0.87	245
e	0.81	1.00	0.89	498
f	0.79	1.00	0.88	247

g	0.93	0.84	0.88	348
h	0.93	0.84	0.88	436
i	0.69	0.85	0.76	288
k	0.97	0.39	0.56	331
l	0.73	1.00	0.84	209
m	0.88	0.75	0.81	394
n	0.86	0.44	0.58	291
o	0.85	0.84	0.85	246
p	1.00	0.98	0.99	347
q	0.79	0.87	0.83	164
r	0.71	0.71	0.71	144
s	0.66	0.75	0.70	246
t	0.59	0.66	0.62	248
u	0.82	0.74	0.78	266
v	0.69	0.72	0.70	346
w	0.45	0.70	0.55	206
x	0.77	0.79	0.78	267
y	0.72	0.61	0.66	332

accuracy: 0.81 7172
macro avg: 0.81 0.80 0.80 7172
weighted avg: 0.83 0.81 0.81 7172







	precision	recall	f1-score	support
a	0.81	1.00	0.90	331
b	0.96	0.99	0.97	432
c	0.96	0.93	0.95	310
d	0.81	1.00	0.89	245
e	0.96	0.95	0.95	498
f	0.80	0.99	0.88	247

g	0.93	0.77	0.84	348
h	0.95	0.91	0.93	436
i	0.83	0.91	0.87	288
k	0.94	0.69	0.79	331
l	0.98	0.96	0.97	209
m	0.88	0.80	0.84	394
n	0.88	0.66	0.75	291
o	0.99	0.75	0.85	246
p	0.96	0.97	0.96	347
q	0.55	0.95	0.69	164
r	0.71	0.28	0.40	144
s	0.55	0.64	0.59	246
t	0.69	0.76	0.72	248
u	0.59	0.63	0.61	266
v	0.67	0.65	0.66	346
w	0.53	0.69	0.60	206
x	0.73	0.61	0.66	267
y	0.82	0.81	0.82	332

accuracy:			0.81	7172
macro avg:	0.81	0.80	0.80	7172
weighted avg:	0.83	0.81	0.81	7172

The table that contains only f1-scores is given in Appendix 6. This table is used for the comparisons of models in terms of performance.

5. DISCUSSION

Before discussing the results that we obtained from training, we will discuss the outcomes of our early models. In the early stage of the project, we used a single activation function, which was sigmoid. We had a single hidden layer which had 10 neurons. The results were around 60% of accuracy and this was not sufficient. Then, we trained our model with the relu activation function, and it gave us better accuracy, which was above

75%. Later, we decided to investigate the outcomes from models that will be using different activation functions. Also, we include dropouts in between layers in order to increase the accuracy for both CNN and ANN and add additional layers. We chose the neurons as 1024 and 512 for layers since the first layer should have been greater than the input size, which was 576 (24x24), and additionally, computers are good in calculations in size of power of 2. However, the dropouts caused the models to overfit, and we decided to use regularization which also failed to solve the problem. The reason that we found was due to the many iterations, which in our case was 50, and this caused the model to memorize the images. We used an early stop with n=3 and trained our model with/without dropouts and found that the early stop prevents the overfitting. However, the outcomes of using or not using dropouts were different and we decided to investigate the effect of dropouts. Hence, the final goal of the project became the investigation of CNN and ANN models with 3 different activation functions and the effect of using dropouts to the outcome. In overall, we investigated 12 different models and compared the results in terms of training time, storage, accuracies, precision and recall.

The discussion will be focused on 2 parts which are comparison between activation functions (sigmoid, prelu, relu) and comparison between models (ANN vs CNN). Also, the contribution of dropouts in CNN and ANN will be discussed.

In order to compare the execution time correctly, we have focused on the train time per epoch, which is being calculated by Train Time / Epoch Number. This variable was required since the execution time was mainly determined by the epoch number. Since all parameters are initialized randomly, the epoch numbers as well as other parameters each run. Hence, all discussions about time will be referred to time per epoch.

Dropouts in TensorFlow are implemented with additional matrices that are randomly initialized (which will decide the neurons to be killed). These matrices are being multiplied with weights so the output of the layer will contain the weights that have not been killed and zeros for those that have been eliminated. The ratio of the neurons that will be eliminated should be specified by the user, which in our case is 0.2.

In order to evaluate the performance of models, we cannot only use accuracy since it would not provide us accurate information and sometimes even provides us with faulty results of our model. It is good to look for precision and recall which will give us more information about the label predictions. Moreover, f1-score gives us the harmonic mean of both. If both are high, then recall and precision is high. If both are low, then the f1-score is also low. Finally, if one of them is low while the other is high, it will lead to the medium f1-score. So, we will focus on f1-score for each model as well as accuracy.

CNN:

The time for CNN model without dropouts with relu, prelu and sigmoid are 12.70, 13.56, 13.75, respectively. The time for CNN models with dropouts with the same activation functions are 13.00, 14.11, 15.60, respectively. As it is seen, the dropouts increase the execution time for all CNN models. This is because the model randomly selects the weights to be eliminated and multiplies the matrix with the weights being taken from the hidden layers. This causes the additional computations which increases the execution time. Among them, the relu function is the fastest one as it is expected. Since PReLu includes the negative values to the calculation, the higher execution time compared to the ReLu was expected. On the other hand, sigmoid function requires calculating the exponential function of the natural number, which is highly costly in terms of CPU, and this makes its model the slowest one.

The accuracies of CNN model for activation functions without Dropouts are 0.83, 0.82, 0.84 and with Dropouts are 0.81, 0.86, 0.88, respectively. The sigmoid function for with/without dropout models is the highest in terms of accuracy. However, in terms of f1-score, CNN model with dropouts and with PReLu activation functions has greater scores in general. In CNN without dropouts, the recall and precision are generally higher in sigmoid than PReLu. In the ReLu function, the precision and recall are similar (the numbers are quite close) to PReLu in general. These results display the relation between f1-score and accuracy, which the order is the same. On the other hand, CNN model with the dropout has the highest accuracy in sigmoid which is also the same result in terms of f1-score (after considering in general). The ReLu function is the lowest one in general since both recall and precision is medium, resulting in medium f1-score. Overall, in CNN model, the sigmoid function has the best performance and ReLu has the lowest one. For the larger output size, the sigmoid would have performed the worst case since the outcomes will diverge to 1 and there will be very small difference in between larger output making the prediction hard. Since the weights are not diverging to the larger values, the mentioned problem did not occur. On the contrary, it had the best performance.

ANN:

The execution time for ANN models without dropouts with samek activation functions are 9.86, 10.23 and 9.54. On the other hand, the times for models with dropouts are 9.74, 10.55, 9.90. Models for PReLu and sigmoid functions, the execution time increases as dropouts are involved. However, the Relu function performs differently than others as it is seen. Slight decrease in execution time of the ReLu model was not seen often in our earlier training, and this phenomenon might have been occured by initial values of parameters and matrices for dropout masks that have been initialized randomly.

The accuracies of ANN model relu, prelu, sigmoid without dropout are 0.83, 0.84, 0.81, respectively. For the model with dropouts, the accuracies are 0.80, 0.81, 0.82. In terms of accuracy, the dropouts seem to have a negative effect. On the other hand, in comparison between relu and prelu functions in ANN without dropout models, there is a slight difference that relu has greater value. However, it also demonstrates the fact that performance cannot be evaluated only from accuracy even though the difference is negligible. The effect of dropouts seems to favor the prelu function in f1-score. However, in this case, the accuracy shows us that the best model is sigmoid, which is not in terms of both precision and recall. Hence, it can be said that PReLu is the best model among models having dropout while ReLu is the best among models that don't have dropouts.

CNN vs ANN:

As it can be seen from the number of trainable parameters, CNN has 615,074. This number shows that apart from the input data, there is such several variables required to train the model (excluding local variables). On the other hand, ANN has 1,340,952 trainable parameters. The difference originates from the input layer, in which CNN reduces to 75 while there are 1024 inputs in ANN, and this reflects the number of neurons. As a result, the ANN requires more storage space than CNN. On the other hand, in general, ANN models are faster than CNN models. Since CNN has more parameters, it is expected to learn slower but since CNN's initial computations are expensive, it causes models to be slower than ANN's models. Finally, in terms of performance in predictions, CNN models with dropouts' outcomes is the best compared to others. It is also expected since CNN model is generally good at image classification problems then ANN since convolution layers extract the pattern/feature of images that will be used for the classification. Also, dropouts help to reduce the noise and increase the performance in CNN models unlike in ANN.

In order to choose the best model for our project, we had to consider them in terms of performance, speed and storage. Since our dataset is not so large and the existence of early stops, the total training time is not so large (which is 6 minutes at most). Also, the computers' storages are advanced and become cheap nowadays, moreover, our 15 million parameters are not so huge data for the computer, consequently, performance is considered to be the main criteria. CNN model with dropouts and with sigmoid activation function has the best prediction performance, and we conclude that in overall, it is also the best one for our project and dataset.

6. CONCLUSION

Our project has evolved from investigating kNN, CNN and ANN to investigating ANN and CNN with different activation functions and effects of dropouts. We eliminate the kNN model since we concluded that the image processing based on similarity would not be able to perform well for new data and since we do not have uniformly distributed data (3 letters' occurrence is higher), the model will tend to interpret wrongly. During the training we concluded that dropouts increase the execution times and prediction performance generally. On the other hand, sigmoid had the best prediction and slowest performance among CNN models. Also, CNN model is better than ANN in terms of accuracy, recall, precision and f1-score. Hence, we concluded that in our project, CNN model with dropouts and with sigmoid activation function is the best model.

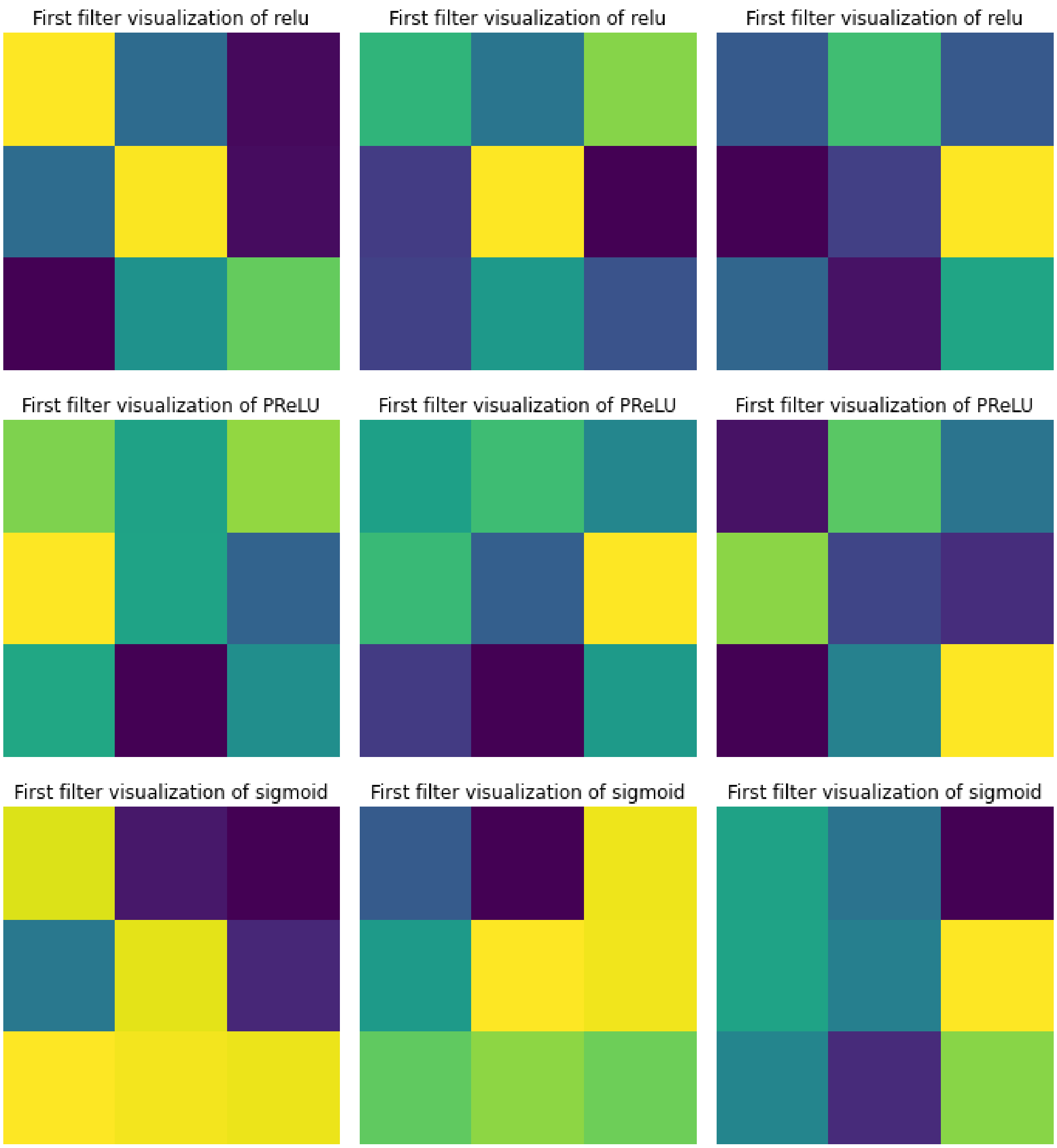
The work allocation during project is done in the following way:
Aral Saleyhan: Finding the topic, code CNN model and create tables, plots matrices etc. from outcomes, placing the results section in final report
Ipek Tüfekcioğlu: Finding the topic, code ANN model and create tables, plots matrices etc. from outcomes, placing the results section in final report
Ekin Doğaç Öztürk: Research on models and activation functions, code ANN model and analyzing the outcomes, arranging the results section in final report
Akmuhammet Ashyralyyev: code CNN model, Training results analysis and interpretation, discussion part of final report, generalize and prepare the scripts.

Our other friend Berke Ceran withdrew from class, so we didn't put his name into this report nor mentioned the work that he had done.

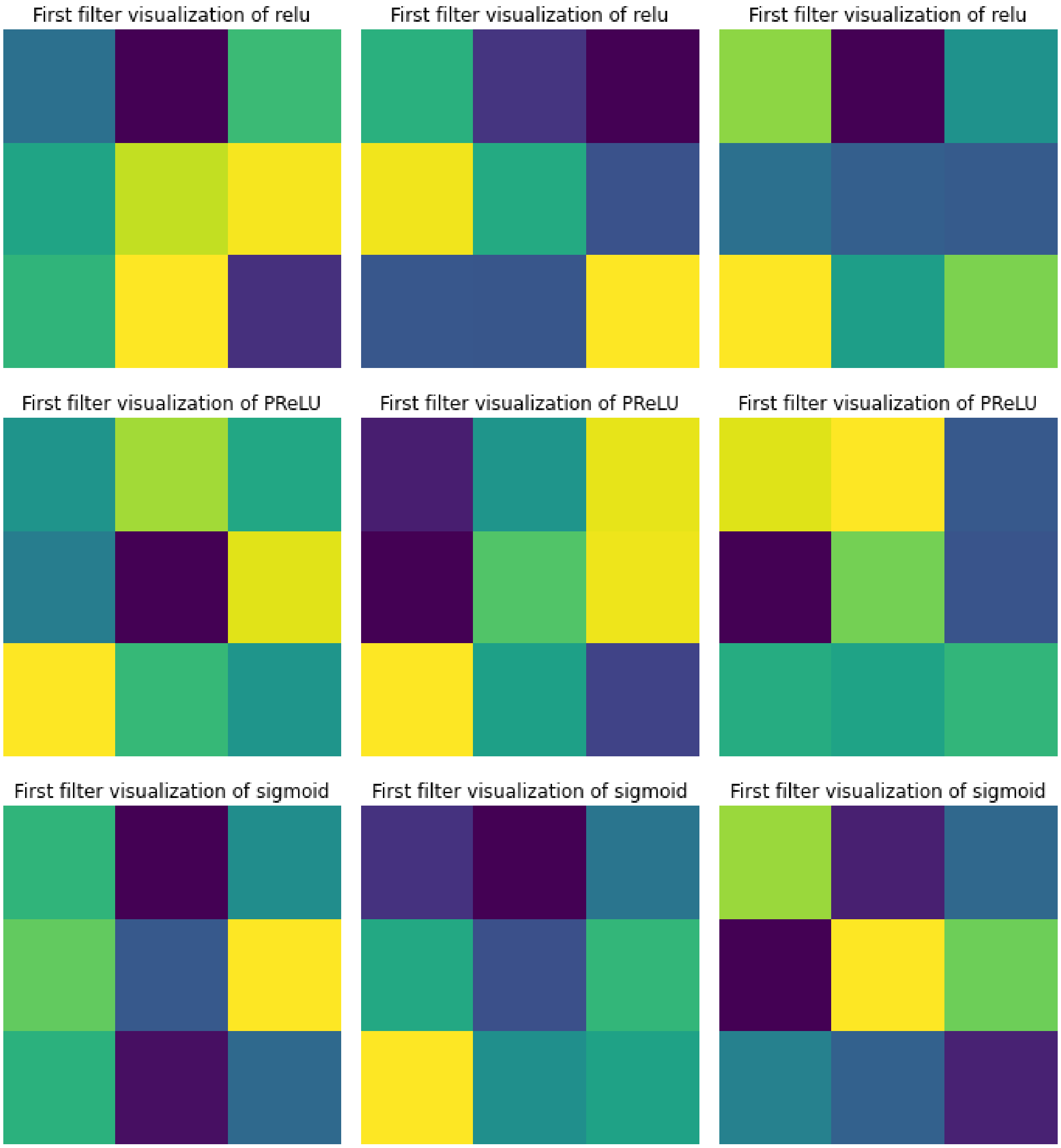
7. APPENDIX

Appendix 1

Filters learnt from CNN model without dropouts



Filters learnt from CNN model with dropouts



Appendix 2
The output of CNN_without_dropout.py

Execution of CNN training without drop out and with relu

```
Model: "CNN_without_dropout_and_with_relu"
_________________________________________________________________
Layer (type)              Output Shape              Param #
=================================================================
first_filters (Conv2D)     (None, 26, 26, 3)        30

batch_normalization_16 (Bat  (None, 26, 26, 3)       12
chNormalization)

MaxPool1 (MaxPooling2D)     (None, 13, 13, 3)        0

second_filters (Conv2D)     (None, 11, 11, 3)        84
```

```
batch_normalization_17 (Bat  (None, 11, 11, 3)        12
chNormalization)

MaxPool2 (MaxPooling2D)     (None, 5, 5, 3)           0

Flatten_of_Convs_Output (Fl  (None, 75)               0
atten)

Hidden_Layer_1 (Dense)      (None, 1024)              77824

Hidden_Layer_2 (Dense)      (None, 512)               524800

Output_Layer (Dense)        (None, 24)                12312

=================================================================
Total params: 615,074
Trainable params: 615,062
Non-trainable params: 12
_________________________________________________________________
Epoch 1/50
687/687 [==============================] - 13s 18ms/step - loss: 0.3636 - sparse_categorical_accuracy: 0.8964 - val_loss: 0.0176 - val_sparse_categorical_accuracy: 0.9980
Epoch 2/50
687/687 [==============================] - 13s 18ms/step - loss: 0.0359 - sparse_categorical_accuracy: 0.9908 - val_loss: 0.0593 - val_sparse_categorical_accuracy: 0.9834
Epoch 3/50
687/687 [==============================] - 12s 18ms/step - loss: 0.0086 - sparse_categorical_accuracy: 0.9982 - val_loss: 7.9113e-04 - val_sparse_categorical_accuracy: 1.0000
Epoch 4/50
687/687 [==============================] - 12s 18ms/step - loss: 4.3893e-04 - sparse_categorical_accuracy: 1.0000 - val_loss: 4.7754e-04 - val_sparse_categorical_accuracy: 1.0000
Epoch 5/50
687/687 [==============================] - 13s 18ms/step - loss: 2.1145e-04 - sparse_categorical_accuracy: 1.0000 - val_loss: 2.7356e-04 - val_sparse_categorical_accuracy: 1.0000
Epoch 6/50
687/687 [==============================] - 12s 18ms/step - loss: 0.0696 - sparse_categorical_accuracy: 0.9824 - val_loss: 0.1188 - val_sparse_categorical_accuracy: 0.9627
Epoch 7/50
687/687 [==============================] - 13s 18ms/step - loss: 0.0086 - sparse_categorical_accuracy: 0.9980 - val_loss: 6.5363e-04 - val_sparse_categorical_accuracy: 1.0000
Epoch 8/50
687/687 [==============================] - 12s 18ms/step - loss: 2.4231e-04 - sparse_categorical_accuracy: 1.0000 - val_loss: 3.0419e-04 - val_sparse_categorical_accuracy: 1.0000
End of training model with activation function =  relu
-------------------------------------------------------------------------------------

              precision    recall  f1-score   support

           0       0.90      1.00      0.95       331
           1       1.00      0.91      0.95       432
           2       0.96      0.93      0.95       310
           3       0.94      0.76      0.84       245
           4       0.88      0.93      0.90       498
           5       0.92      1.00      0.96       247
           6       0.80      0.94      0.86       348
           7       0.95      0.90      0.92       436
           8       0.85      0.78      0.81       288
           9       0.60      0.62      0.61       331
          10       0.79      1.00      0.88       209
          11       0.94      0.86      0.90       394
          12       0.75      0.75      0.75       291
          13       1.00      0.65      0.79       246
          14       0.89      0.92      0.90       347
          15       0.82      0.95      0.88       164
          16       0.39      0.53      0.45       144
          17       0.74      0.57      0.65       246
          18       0.80      0.56      0.66       248
          19       0.65      0.89      0.76       266
          20       0.88      0.79      0.84       346
```

```
         21       0.75      0.89      0.81       206
         22       0.81      0.88      0.84       267
         23       0.77      0.70      0.73       332

    accuracy                          0.83      7172
   macro avg       0.82      0.82      0.82      7172
weighted avg       0.84      0.83      0.83      7172
```

Execution of CNN training without drop out and with PReLU

```
Model: "CNN_without_dropout_and_with_PReLU"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 first_filters (Conv2D)      (None, 26, 26, 3)         30

 batch_normalization_18 (Bat  (None, 26, 26, 3)        12
 chNormalization)

 MaxPool1 (MaxPooling2D)     (None, 13, 13, 3)         0

 second_filters (Conv2D)     (None, 11, 11, 3)         84

 batch_normalization_19 (Bat  (None, 11, 11, 3)        12
 chNormalization)

 MaxPool2 (MaxPooling2D)     (None, 5, 5, 3)           0

 Flatten_of_Convs_Output (Fl  (None, 75)               0
 atten)

 Hidden_Layer_1 (Dense)      (None, 1024)              78848

 Hidden_Layer_2 (Dense)      (None, 512)               525312

 Output_Layer (Dense)        (None, 24)                12312

=================================================================
Total params: 616,610
Trainable params: 616,598
Non-trainable params: 12
_________________________________________________________________
Epoch 1/50
687/687 [==============================] - 14s 19ms/step - loss: 0.4745 - sparse_categorical_accuracy: 0.8595 - val_loss: 0.0319 - val_sparse_categorical_accuracy: 0.9925
Epoch 2/50
687/687 [==============================] - 13s 19ms/step - loss: 0.0210 - sparse_categorical_accuracy: 0.9954 - val_loss: 0.0021 - val_sparse_categorical_accuracy: 1.0000
Epoch 3/50
687/687 [==============================] - 14s 20ms/step - loss: 0.0010 - sparse_categorical_accuracy: 1.0000 - val_loss: 5.9735e-04 - val_sparse_categorical_accuracy: 1.0000
Epoch 4/50
687/687 [==============================] - 13s 19ms/step - loss: 3.5977e-04 - sparse_categorical_accuracy: 1.0000 - val_loss: 3.0338e-04 - val_sparse_categorical_accuracy: 1.0000
Epoch 5/50
687/687 [==============================] - 13s 20ms/step - loss: 1.9836e-04 - sparse_categorical_accuracy: 1.0000 - val_loss: 1.7889e-04 - val_sparse_categorical_accuracy: 1.0000
Epoch 6/50
687/687 [==============================] - 18s 26ms/step - loss: 1.2004e-04 - sparse_categorical_accuracy: 1.0000 - val_loss: 1.2363e-04 - val_sparse_categorical_accuracy: 1.0000
Epoch 7/50
687/687 [==============================] - 14s 20ms/step - loss: 7.1894e-05 - sparse_categorical_accuracy: 1.0000 - val_loss: 7.7699e-05 - val_sparse_categorical_accuracy: 1.0000
Epoch 8/50
687/687 [==============================] - 13s 19ms/step - loss: 4.7827e-05 - sparse_categorical_accuracy: 1.0000 - val_loss: 5.6781e-05 - val_sparse_categorical_accuracy: 1.0000
Epoch 9/50
687/687 [==============================] - 13s 19ms/step - loss: 3.0106e-05 - sparse_categorical_accuracy: 1.0000 - val_loss: 3.3737e-05 - val_sparse_categorical_accuracy: 1.0000
Epoch 10/50
```

```
687/687 [==============================] - 13s 19ms/step - loss: 1.9782e-05 - sparse_categorical_accuracy: 1.0000 - val_loss: 3.0423e-05 - val_sparse_categorical_accuracy: 1.0000
Epoch 11/50
687/687 [==============================] - 13s 19ms/step - loss: 1.2842e-05 - sparse_categorical_accuracy: 1.0000 - val_loss: 1.5482e-05 - val_sparse_categorical_accuracy: 1.0000
Epoch 12/50
687/687 [==============================] - 14s 20ms/step - loss: 8.3873e-06 - sparse_categorical_accuracy: 1.0000 - val_loss: 1.2185e-05 - val_sparse_categorical_accuracy: 1.0000
Epoch 13/50
687/687 [==============================] - 13s 19ms/step - loss: 5.8130e-06 - sparse_categorical_accuracy: 1.0000 - val_loss: 9.6766e-06 - val_sparse_categorical_accuracy: 1.0000
Epoch 14/50
687/687 [==============================] - 13s 19ms/step - loss: 4.0490e-06 - sparse_categorical_accuracy: 1.0000 - val_loss: 6.4517e-06 - val_sparse_categorical_accuracy: 1.0000
Epoch 15/50
687/687 [==============================] - 13s 19ms/step - loss: 0.1156 - sparse_categorical_accuracy: 0.9768 - val_loss: 0.0045 - val_sparse_categorical_accuracy: 0.9993
Epoch 16/50
687/687 [==============================] - 13s 19ms/step - loss: 8.0995e-04 - sparse_categorical_accuracy: 1.0000 - val_loss: 4.4387e-04 - val_sparse_categorical_accuracy: 1.0000
Epoch 17/50
687/687 [==============================] - 13s 19ms/step - loss: 2.6221e-04 - sparse_categorical_accuracy: 1.0000 - val_loss: 2.8788e-04 - val_sparse_categorical_accuracy: 1.0000
End of training model with activation function =  PReLU
----------------------------------------------------------------------------------------
```

	precision	recall	f1-score	support
0	0.90	0.99	0.94	331
1	1.00	0.96	0.98	432
2	1.00	0.99	0.99	310
3	1.00	0.84	0.92	245
4	0.91	0.88	0.89	498
5	0.96	1.00	0.98	247
6	0.83	0.98	0.90	348
7	0.95	0.88	0.91	436
8	0.98	0.75	0.85	288
9	0.68	0.74	0.71	331
10	1.00	1.00	1.00	209
11	0.68	0.68	0.68	394
12	0.51	0.68	0.58	291
13	0.99	0.58	0.73	246
14	0.99	0.99	0.99	347
15	0.87	0.86	0.87	164
16	0.56	0.85	0.67	144
17	0.59	0.55	0.57	246
18	0.78	0.68	0.73	248
19	0.58	0.57	0.57	266
20	0.82	0.53	0.64	346
21	0.50	0.80	0.61	206
22	0.73	0.80	0.76	267
23	0.89	0.88	0.88	332
accuracy			0.82	7172
macro avg	0.82	0.81	0.81	7172
weighted avg	0.83	0.82	0.82	7172

Execution of CNN training without drop out and with sigmoid

```
Model: "CNN_without_dropout_and_with_sigmoid"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 first_filters (Conv2D)      (None, 26, 26, 3)         30

 batch_normalization_20 (Bat  (None, 26, 26, 3)        12
 chNormalization)

 MaxPool1 (MaxPooling2D)     (None, 13, 13, 3)         0
```

```
second_filters (Conv2D)     (None, 11, 11, 3)        84

batch_normalization_21 (Bat  (None, 11, 11, 3)       12
chNormalization)

MaxPool2 (MaxPooling2D)     (None, 5, 5, 3)          0

Flatten_of_Convs_Output (Fl  (None, 75)              0
atten)

Hidden_Layer_1 (Dense)      (None, 1024)             77824

Hidden_Layer_2 (Dense)      (None, 512)              524800

Output_Layer (Dense)        (None, 24)               12312

=================================================================
Total params: 615,074
Trainable params: 615,062
Non-trainable params: 12
_________________________________________________________________
Epoch 1/50
687/687 [==============================] - 13s 18ms/step - loss: 1.2276 - sparse_categorical_accuracy: 0.6192 - val_loss: 0.5882 - val_sparse_categorical_accuracy: 0.8082
Epoch 2/50
687/687 [==============================] - 12s 18ms/step - loss: 0.2293 - sparse_categorical_accuracy: 0.9397 - val_loss: 0.0971 - val_sparse_categorical_accuracy: 0.9867
Epoch 3/50
687/687 [==============================] - 13s 18ms/step - loss: 0.0516 - sparse_categorical_accuracy: 0.9945 - val_loss: 0.0252 - val_sparse_categorical_accuracy: 0.9985
Epoch 4/50
687/687 [==============================] - 12s 18ms/step - loss: 0.0161 - sparse_categorical_accuracy: 0.9995 - val_loss: 0.0094 - val_sparse_categorical_accuracy: 1.0000
Epoch 5/50
687/687 [==============================] - 12s 18ms/step - loss: 0.0082 - sparse_categorical_accuracy: 0.9993 - val_loss: 0.0042 - val_sparse_categorical_accuracy: 1.0000
Epoch 6/50
687/687 [==============================] - 12s 18ms/step - loss: 0.0048 - sparse_categorical_accuracy: 0.9995 - val_loss: 0.0133 - val_sparse_categorical_accuracy: 0.9982
Epoch 7/50
687/687 [==============================] - 12s 18ms/step - loss: 0.0180 - sparse_categorical_accuracy: 0.9952 - val_loss: 0.0015 - val_sparse_categorical_accuracy: 1.0000
Epoch 8/50
687/687 [==============================] - 13s 19ms/step - loss: 9.6865e-04 - sparse_categorical_accuracy: 1.0000 - val_loss: 8.7072e-04 - val_sparse_categorical_accuracy: 1.0000
Epoch 9/50
687/687 [==============================] - 12s 18ms/step - loss: 6.3303e-04 - sparse_categorical_accuracy: 1.0000 - val_loss: 5.9486e-04 - val_sparse_categorical_accuracy: 1.0000
Epoch 10/50
687/687 [==============================] - 12s 18ms/step - loss: 4.3341e-04 - sparse_categorical_accuracy: 1.0000 - val_loss: 3.9770e-04 - val_sparse_categorical_accuracy: 1.0000
Epoch 11/50
687/687 [==============================] - 12s 18ms/step - loss: 3.0382e-04 - sparse_categorical_accuracy: 1.0000 - val_loss: 3.1078e-04 - val_sparse_categorical_accuracy: 1.0000
Epoch 12/50
687/687 [==============================] - 15s 22ms/step - loss: 2.1534e-04 - sparse_categorical_accuracy: 1.0000 - val_loss: 2.1454e-04 - val_sparse_categorical_accuracy: 1.0000
Epoch 13/50
687/687 [==============================] - 16s 23ms/step - loss: 1.5932e-04 - sparse_categorical_accuracy: 1.0000 - val_loss: 2.4535e-04 - val_sparse_categorical_accuracy: 1.0000
Epoch 14/50
687/687 [==============================] - 16s 23ms/step - loss: 1.1367e-04 - sparse_categorical_accuracy: 1.0000 - val_loss: 1.1084e-04 - val_sparse_categorical_accuracy: 1.0000
Epoch 15/50
687/687 [==============================] - 14s 21ms/step - loss: 7.7709e-05 - sparse_categorical_accuracy: 1.0000 - val_loss: 8.0018e-05 - val_sparse_categorical_accuracy: 1.0000
Epoch 16/50
687/687 [==============================] - 16s 23ms/step - loss: 5.4823e-05 - sparse_categorical_accuracy: 1.0000 - val_loss: 6.5993e-05 - val_sparse_categorical_accuracy: 1.0000
Epoch 17/50
```

687/687 [==============================] - 15s 21ms/step - loss: 3.9309e-05 - sparse_categorical_accuracy: 1.0000 - val_loss: 5.3887e-05 - val_sparse_categorical_accuracy: 1.0000
Epoch 18/50
687/687 [==============================] - 14s 21ms/step - loss: 2.6321e-05 - sparse_categorical_accuracy: 1.0000 - val_loss: 3.3582e-05 - val_sparse_categorical_accuracy: 1.0000
Epoch 19/50
687/687 [==============================] - 14s 21ms/step - loss: 1.7944e-05 - sparse_categorical_accuracy: 1.0000 - val_loss: 2.6176e-05 - val_sparse_categorical_accuracy: 1.0000
Epoch 20/50
687/687 [==============================] - 14s 21ms/step - loss: 1.2160e-05 - sparse_categorical_accuracy: 1.0000 - val_loss: 2.0572e-05 - val_sparse_categorical_accuracy: 1.0000
Epoch 21/50
687/687 [==============================] - 14s 21ms/step - loss: 8.9035e-06 - sparse_categorical_accuracy: 1.0000 - val_loss: 1.6589e-05 - val_sparse_categorical_accuracy: 1.0000
Epoch 22/50
687/687 [==============================] - 15s 22ms/step - loss: 0.0191 - sparse_categorical_accuracy: 0.9946 - val_loss: 0.0010 - val_sparse_categorical_accuracy: 1.0000
Epoch 23/50
687/687 [==============================] - 14s 20ms/step - loss: 2.8026e-04 - sparse_categorical_accuracy: 1.0000 - val_loss: 2.4219e-04 - val_sparse_categorical_accuracy: 1.0000
Epoch 24/50
687/687 [==============================] - 14s 20ms/step - loss: 1.2298e-04 - sparse_categorical_accuracy: 1.0000 - val_loss: 1.4930e-04 - val_sparse_categorical_accuracy: 1.0000
End of training model with activation function = sigmoid
---

	precision	recall	f1-score	support
0	0.83	1.00	0.90	331
1	1.00	0.95	0.98	432
2	0.89	1.00	0.94	310
3	0.89	0.93	0.91	245
4	0.86	0.96	0.91	498
5	0.82	1.00	0.90	247
6	0.92	0.87	0.90	348
7	0.92	0.80	0.86	436
8	0.96	0.82	0.88	288
9	0.80	0.76	0.78	331
10	0.87	1.00	0.93	209
11	0.73	0.86	0.79	394
12	0.78	0.54	0.64	291
13	0.96	0.65	0.77	246
14	0.87	0.90	0.89	347
15	0.73	0.87	0.79	164
16	0.55	0.85	0.67	144
17	0.71	0.63	0.67	246
18	0.75	0.65	0.70	248
19	0.70	0.69	0.69	266
20	0.94	0.86	0.89	346
21	0.79	0.81	0.80	206
22	0.91	0.80	0.85	267
23	0.90	0.88	0.89	332
accuracy			0.84	7172
macro avg	0.84	0.84	0.83	7172
weighted avg	0.85	0.84	0.84	7172

Train time(secs) with dropout relu 101.50942802429199
Train time(secs) with dropout PReLU 230.4533770084381
Train time(secs) with dropout sigmoid 329.69622707366943

Appendix 3
The output of CNN_with_dropout.py

Execution of CNN training with drop out and with relu

Model: "CNN_with_dropout_and_with_relu"

____

```
Layer (type)                Output Shape              Param #
=================================================================
first_filters (Conv2D)      (None, 26, 26, 3)         30

batch_normalization_4 (Batc  (None, 26, 26, 3)        12
hNormalization)

MaxPool1 (MaxPooling2D)     (None, 13, 13, 3)         0

second_filters (Conv2D)     (None, 11, 11, 3)         84

batch_normalization_5 (Batc  (None, 11, 11, 3)        12
hNormalization)

MaxPool2 (MaxPooling2D)     (None, 5, 5, 3)           0

Flatten_of_Convs_Output (Fl  (None, 75)               0
atten)

Hidden_Layer_1 (Dense)      (None, 1024)              77824

Dropout_1 (Dropout)         (None, 1024)              0

Hidden_Layer_2 (Dense)      (None, 512)               524800

Dropout_2 (Dropout)         (None, 512)               0

Output_Layer (Dense)        (None, 24)                12312

=================================================================
Total params: 615,074
Trainable params: 615,062
Non-trainable params: 12
_________________________________________________________________
Epoch 1/50
687/687 [==============================] - 15s 21ms/step - loss: 0.4361 - sparse_categorical_accuracy: 0.8690 - val_loss: 0.0251 - val_sparse_categorical_accuracy: 0.9971
Epoch 2/50
687/687 [==============================] - 14s 20ms/step - loss: 0.0296 - sparse_categorical_accuracy: 0.9929 - val_loss: 0.0350 - val_sparse_categorical_accuracy: 0.9874
Epoch 3/50
687/687 [==============================] - 12s 18ms/step - loss: 0.0271 - sparse_categorical_accuracy: 0.9922 - val_loss: 0.0014 - val_sparse_categorical_accuracy: 0.9998
Epoch 4/50
687/687 [==============================] - 12s 18ms/step - loss: 0.0321 - sparse_categorical_accuracy: 0.9907 - val_loss: 0.0464 - val_sparse_categorical_accuracy: 0.9860
Epoch 5/50
687/687 [==============================] - 13s 18ms/step - loss: 0.0106 - sparse_categorical_accuracy: 0.9976 - val_loss: 3.4828e-04 - val_sparse_categorical_accuracy: 1.0000
Epoch 6/50
687/687 [==============================] - 12s 18ms/step - loss: 0.0092 - sparse_categorical_accuracy: 0.9971 - val_loss: 0.0637 - val_sparse_categorical_accuracy: 0.9851
Epoch 7/50
687/687 [==============================] - 12s 18ms/step - loss: 0.0322 - sparse_categorical_accuracy: 0.9901 - val_loss: 0.0084 - val_sparse_categorical_accuracy: 0.9967
Epoch 8/50
687/687 [==============================] - 12s 18ms/step - loss: 0.0128 - sparse_categorical_accuracy: 0.9973 - val_loss: 0.0022 - val_sparse_categorical_accuracy: 0.9991
Epoch 9/50
687/687 [==============================] - 12s 18ms/step - loss: 0.0123 - sparse_categorical_accuracy: 0.9960 - val_loss: 0.0412 - val_sparse_categorical_accuracy: 0.9856
```

End of training model with activation function = relu

Execution of CNN training with drop out and with PReLU

Model: "CNN_with_dropout_and_with_PReLU"

```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 first_filters (Conv2D)      (None, 26, 26, 3)         30

 batch_normalization_6 (Batc  (None, 26, 26, 3)        12
 hNormalization)

 MaxPool1 (MaxPooling2D)     (None, 13, 13, 3)         0

 second_filters (Conv2D)     (None, 11, 11, 3)         84

 batch_normalization_7 (Batc  (None, 11, 11, 3)        12
 hNormalization)

 MaxPool2 (MaxPooling2D)     (None, 5, 5, 3)           0

 Flatten_of_Convs_Output (Fl  (None, 75)               0
 atten)

 Hidden_Layer_1 (Dense)      (None, 1024)              78848

 Dropout_1 (Dropout)         (None, 1024)              0

 Hidden_Layer_2 (Dense)      (None, 512)               525312

 Dropout_2 (Dropout)         (None, 512)               0

 Output_Layer (Dense)        (None, 24)                12312

=================================================================
Total params: 616,610
Trainable params: 616,598
Non-trainable params: 12
_________________________________________________________________
Epoch 1/50
687/687 [==============================] - 15s 20ms/step - loss: 0.4430 - sparse_categorical_accuracy: 0.8647 - val_loss: 0.0661 - val_sparse_categorical_accuracy: 0.9834
Epoch 2/50
687/687 [==============================] - 14s 20ms/step - loss: 0.0373 - sparse_categorical_accuracy: 0.9898 - val_loss: 0.0411 - val_sparse_categorical_accuracy: 0.9876
Epoch 3/50
687/687 [==============================] - 14s 21ms/step - loss: 0.0291 - sparse_categorical_accuracy: 0.9911 - val_loss: 0.0072 - val_sparse_categorical_accuracy: 0.9980
Epoch 4/50
687/687 [==============================] - 14s 21ms/step - loss: 0.0291 - sparse_categorical_accuracy: 0.9907 - val_loss: 0.0287 - val_sparse_categorical_accuracy: 0.9920
Epoch 5/50
687/687 [==============================] - 15s 21ms/step - loss: 0.0156 - sparse_categorical_accuracy: 0.9958 - val_loss: 0.0099 - val_sparse_categorical_accuracy: 0.9975
Epoch 6/50
687/687 [==============================] - 14s 20ms/step - loss: 0.0317 - sparse_categorical_accuracy: 0.9909 - val_loss: 0.0480 - val_sparse_categorical_accuracy: 0.9902
Epoch 7/50
```

```
687/687 [==============================] - 14s 20ms/step - loss: 0.0153 - sparse_categorical_accuracy: 0.9955 - val_loss: 0.0040 - val_sparse_categorical_accuracy: 0.9985
Epoch 8/50
687/687 [==============================] - 14s 20ms/step - loss: 0.0179 - sparse_categorical_accuracy: 0.9950 - val_loss: 0.0019 - val_sparse_categorical_accuracy: 0.9993
Epoch 9/50
687/687 [==============================] - 14s 20ms/step - loss: 0.0184 - sparse_categorical_accuracy: 0.9949 - val_loss: 0.0147 - val_sparse_categorical_accuracy: 0.9940
Epoch 10/50
687/687 [==============================] - 14s 20ms/step - loss: 0.0224 - sparse_categorical_accuracy: 0.9941 - val_loss: 0.0045 - val_sparse_categorical_accuracy: 0.9982
End of training model with activation function =  PReLU
-----------------------------------------------------------------------------------------

              precision    recall  f1-score   support

           0       0.96      0.95      0.95       331
           1       1.00      0.91      0.95       432
           2       0.89      0.94      0.91       310
           3       0.81      1.00      0.89       245
           4       0.96      1.00      0.98       498
           5       0.85      1.00      0.92       247
           6       0.86      0.77      0.81       348
           7       0.85      0.91      0.88       436
           8       0.88      0.92      0.90       288
           9       0.80      0.91      0.85       331
          10       0.90      0.95      0.92       209
          11       0.93      0.66      0.77       394
          12       0.73      0.73      0.73       291
          13       1.00      0.82      0.90       246
          14       0.98      0.76      0.86       347
          15       0.79      1.00      0.88       164
          16       0.59      0.56      0.57       144
          17       0.71      0.88      0.79       246
          18       0.88      0.62      0.73       248
          19       0.61      0.68      0.64       266
          20       0.99      0.80      0.88       346
          21       0.87      1.00      0.93       206
          22       0.80      0.85      0.83       267
          23       0.83      0.94      0.88       332

    accuracy                           0.86      7172
   macro avg       0.85      0.86      0.85      7172
weighted avg       0.87      0.86      0.86      7172

Execution of CNN training with drop out and with sigmoid

Model: "CNN_with_dropout_and_with_sigmoid"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 first_filters (Conv2D)      (None, 26, 26, 3)         30

 batch_normalization_8 (Batc  (None, 26, 26, 3)        12
 hNormalization)

 MaxPool1 (MaxPooling2D)     (None, 13, 13, 3)         0

 second_filters (Conv2D)     (None, 11, 11, 3)         84

 batch_normalization_9 (Batc  (None, 11, 11, 3)        12
```

```
hNormalization)

MaxPool2 (MaxPooling2D)     (None, 5, 5, 3)          0

Flatten_of_Convs_Output (Fl  (None, 75)              0
atten)

Hidden_Layer_1 (Dense)      (None, 1024)             77824

Dropout_1 (Dropout)         (None, 1024)             0

Hidden_Layer_2 (Dense)      (None, 512)              524800

Dropout_2 (Dropout)         (None, 512)              0

Output_Layer (Dense)        (None, 24)               12312

=================================================================
Total params: 615,074
Trainable params: 615,062
Non-trainable params: 12
_________________________________________________________________
Epoch 1/50
687/687 [==============================] - 14s 19ms/step - loss: 1.2669 - sparse_categorical_accuracy: 0.6137 - val_loss: 0.4645 - val_sparse_categorical_accuracy: 0.8660
Epoch 2/50
687/687 [==============================] - 13s 19ms/step - loss: 0.2872 - sparse_categorical_accuracy: 0.9132 - val_loss: 0.1684 - val_sparse_categorical_accuracy: 0.9583
Epoch 3/50
687/687 [==============================] - 13s 19ms/step - loss: 0.0909 - sparse_categorical_accuracy: 0.9797 - val_loss: 0.0373 - val_sparse_categorical_accuracy: 0.9954
Epoch 4/50
687/687 [==============================] - 13s 19ms/step - loss: 0.0342 - sparse_categorical_accuracy: 0.9951 - val_loss: 0.0137 - val_sparse_categorical_accuracy: 0.9987
Epoch 5/50
687/687 [==============================] - 13s 19ms/step - loss: 0.0176 - sparse_categorical_accuracy: 0.9977 - val_loss: 0.0278 - val_sparse_categorical_accuracy: 0.9933
Epoch 6/50
687/687 [==============================] - 13s 19ms/step - loss: 0.0113 - sparse_categorical_accuracy: 0.9983 - val_loss: 0.0311 - val_sparse_categorical_accuracy: 0.9940
Epoch 7/50
687/687 [==============================] - 13s 19ms/step - loss: 0.0081 - sparse_categorical_accuracy: 0.9985 - val_loss: 0.0047 - val_sparse_categorical_accuracy: 0.9996
Epoch 8/50
687/687 [==============================] - 16s 23ms/step - loss: 0.0077 - sparse_categorical_accuracy: 0.9986 - val_loss: 0.0025 - val_sparse_categorical_accuracy: 0.9993
Epoch 9/50
687/687 [==============================] - 17s 25ms/step - loss: 0.0075 - sparse_categorical_accuracy: 0.9980 - val_loss: 0.0046 - val_sparse_categorical_accuracy: 0.9989
Epoch 10/50
687/687 [==============================] - 15s 22ms/step - loss: 0.0060 - sparse_categorical_accuracy: 0.9985 - val_loss: 0.0013 - val_sparse_categorical_accuracy: 0.9996
Epoch 11/50
687/687 [==============================] - 13s 20ms/step - loss: 0.0029 - sparse_categorical_accuracy: 0.9993 - val_loss: 0.0033 - val_sparse_categorical_accuracy: 0.9989
Epoch 12/50
687/687 [==============================] - 14s 20ms/step - loss: 0.0068 - sparse_categorical_accuracy: 0.9978 - val_loss: 0.0504 - val_sparse_categorical_accuracy: 0.9823
Epoch 13/50
687/687 [==============================] - 16s 23ms/step - loss: 0.0026 - sparse_categorical_accuracy: 0.9994 - val_loss: 0.0051 - val_sparse_categorical_accuracy: 0.9980
```

Epoch 14/50
687/687 [==============================] - 16s 24ms/step - loss: 0.0068 - sparse_categorical_accuracy: 0.9980 - val_loss: 4.7899e-04 - val_sparse_categorical_accuracy: 1.0000
Epoch 15/50
687/687 [==============================] - 15s 22ms/step - loss: 4.8140e-04 - sparse_categorical_accuracy: 1.0000 - val_loss: 2.9430e-04 - val_sparse_categorical_accuracy: 0.9998
Epoch 16/50
687/687 [==============================] - 16s 24ms/step - loss: 1.9631e-04 - sparse_categorical_accuracy: 1.0000 - val_loss: 1.5435e-04 - val_sparse_categorical_accuracy: 1.0000
Epoch 17/50
687/687 [==============================] - 16s 24ms/step - loss: 1.4179e-04 - sparse_categorical_accuracy: 1.0000 - val_loss: 1.4343e-04 - val_sparse_categorical_accuracy: 1.0000
Epoch 18/50
687/687 [==============================] - 17s 24ms/step - loss: 0.0033 - sparse_categorical_accuracy: 0.9995 - val_loss: 0.2592 - val_sparse_categorical_accuracy: 0.9330
Epoch 19/50
687/687 [==============================] - 14s 21ms/step - loss: 0.0132 - sparse_categorical_accuracy: 0.9955 - val_loss: 0.0073 - val_sparse_categorical_accuracy: 0.9976
Epoch 20/50
687/687 [==============================] - 14s 21ms/step - loss: 8.4481e-04 - sparse_categorical_accuracy: 0.9999 - val_loss: 6.6530e-05 - val_sparse_categorical_accuracy: 1.0000
End of training model with activation function = sigmoid
---

	precision	recall	f1-score	support
0	0.92	1.00	0.96	331
1	1.00	0.95	0.98	432
2	1.00	1.00	1.00	310
3	0.78	0.89	0.83	245
4	0.94	0.92	0.93	498
5	0.97	1.00	0.99	247
6	0.87	0.88	0.88	348
7	0.84	0.94	0.89	436
8	0.85	0.98	0.91	288
9	1.00	0.68	0.81	331
10	1.00	1.00	1.00	209
11	0.81	0.82	0.81	394
12	0.69	0.57	0.62	291
13	1.00	0.88	0.94	246
14	0.97	0.94	0.96	347
15	0.93	0.93	0.93	164
16	0.40	0.45	0.42	144
17	0.64	0.74	0.69	246
18	0.82	0.77	0.79	248
19	0.75	0.73	0.74	266
20	0.88	0.95	0.92	346
21	0.90	0.99	0.94	206
22	0.97	0.93	0.95	267
23	0.94	0.88	0.91	332
accuracy			0.88	7172
macro avg	0.87	0.87	0.87	7172
weighted avg	0.88	0.88	0.88	7172

Train time(secs) with dropout relu 117.01991486549377
Train time(secs) with dropout PReLU 141.09013104438782
Train time(secs) with dropout sigmoid 324.0808439254761

Appendix 4

The output of ANN_without_dropout.py

Execution of ANN training without drop out and with relu

```
Epoch 1/50
687/687 [==============================] - 10s 14ms/step - loss: 1.7816 - sparse_categorical_accuracy: 0.4354 - val_loss: 1.0198 - val_sparse_categorical_accuracy: 0.6729
Epoch 2/50
687/687 [==============================] - 10s 14ms/step - loss: 0.7632 - sparse_categorical_accuracy: 0.7494 - val_loss: 0.5289 - val_sparse_categorical_accuracy: 0.8233
Epoch 3/50
687/687 [==============================] - 10s 14ms/step - loss: 0.3960 - sparse_categorical_accuracy: 0.8691 - val_loss: 0.2513 - val_sparse_categorical_accuracy: 0.9273
Epoch 4/50
687/687 [==============================] - 10s 14ms/step - loss: 0.2253 - sparse_categorical_accuracy: 0.9287 - val_loss: 0.1516 - val_sparse_categorical_accuracy: 0.9497
Epoch 5/50
687/687 [==============================] - 10s 15ms/step - loss: 0.1711 - sparse_categorical_accuracy: 0.9454 - val_loss: 0.1818 - val_sparse_categorical_accuracy: 0.9357
Epoch 6/50
687/687 [==============================] - 10s 14ms/step - loss: 0.1227 - sparse_categorical_accuracy: 0.9619 - val_loss: 0.0605 - val_sparse_categorical_accuracy: 0.9863
Epoch 7/50
687/687 [==============================] - 10s 14ms/step - loss: 0.1149 - sparse_categorical_accuracy: 0.9668 - val_loss: 0.4643 - val_sparse_categorical_accuracy: 0.8363
Epoch 8/50
687/687 [==============================] - 10s 15ms/step - loss: 0.0697 - sparse_categorical_accuracy: 0.9798 - val_loss: 0.4308 - val_sparse_categorical_accuracy: 0.8472
Epoch 9/50
687/687 [==============================] - 10s 14ms/step - loss: 0.0917 - sparse_categorical_accuracy: 0.9723 - val_loss: 0.1055 - val_sparse_categorical_accuracy: 0.9690
Epoch 10/50
687/687 [==============================] - 10s 14ms/step - loss: 0.0386 - sparse_categorical_accuracy: 0.9889 - val_loss: 0.0061 - val_sparse_categorical_accuracy: 0.9998
Epoch 11/50
687/687 [==============================] - 10s 15ms/step - loss: 0.0653 - sparse_categorical_accuracy: 0.9790 - val_loss: 0.1181 - val_sparse_categorical_accuracy: 0.9574
Epoch 12/50
687/687 [==============================] - 10s 15ms/step - loss: 0.0433 - sparse_categorical_accuracy: 0.9865 - val_loss: 0.0013 - val_sparse_categorical_accuracy: 1.0000
Epoch 13/50
687/687 [==============================] - 10s 14ms/step - loss: 0.0011 - sparse_categorical_accuracy: 1.0000 - val_loss: 8.4324e-04 - val_sparse_categorical_accuracy: 1.0000
Epoch 14/50
687/687 [==============================] - 10s 15ms/step - loss: 7.0653e-04 - sparse_categorical_accuracy: 1.0000 - val_loss: 6.5420e-04 - val_sparse_categorical_accuracy: 1.0000
Epoch 15/50
687/687 [==============================] - 9s 13ms/step - loss: 0.1700 - sparse_categorical_accuracy: 0.9526 - val_loss: 0.0554 - val_sparse_categorical_accuracy: 0.9822
Epoch 16/50
687/687 [==============================] - 10s 14ms/step - loss: 0.0101 - sparse_categorical_accuracy: 0.9992 - val_loss: 0.0190 - val_sparse_categorical_accuracy: 0.9965
Epoch 17/50
687/687 [==============================] - 9s 14ms/step - loss: 0.0592 - sparse_categorical_accuracy: 0.9819 - val_loss: 0.0029 - val_sparse_categorical_accuracy: 1.0000
Model: "ANN_without_dropout_and_with_relu"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 Hidden_Layer_1 (Dense)      (None, 1024)              803840

 Hidden_Layer_2 (Dense)      (None, 512)               524800
```

Output_Layer (Dense)   (None, 24)   12312

===
Total params: 1,340,952
Trainable params: 1,340,952
Non-trainable params: 0

End of training model with activation function = relu
--

	precision	recall	f1-score	support
0	0.81	1.00	0.90	331
1	0.99	0.93	0.96	432
2	1.00	0.93	0.96	310
3	0.85	0.97	0.91	245
4	0.91	1.00	0.95	498
5	0.78	1.00	0.88	247
6	0.91	0.79	0.84	348
7	0.95	0.95	0.95	436
8	0.82	0.93	0.87	288
9	0.91	0.60	0.73	331
10	0.78	1.00	0.88	209
11	0.80	0.69	0.74	394
12	0.86	0.64	0.74	291
13	1.00	0.82	0.90	246
14	0.95	1.00	0.97	347
15	0.76	0.83	0.79	164
16	0.53	0.80	0.64	144
17	0.66	0.65	0.65	246
18	0.66	0.67	0.66	248
19	0.60	0.55	0.57	266
20	0.83	0.76	0.80	346
21	0.62	0.74	0.68	206
22	0.77	0.83	0.80	267
23	0.85	0.68	0.76	332
accuracy			0.83	7172
macro avg	0.82	0.82	0.81	7172
weighted avg	0.84	0.83	0.83	7172

Execution of ANN training without drop out and with PReLU

Epoch 1/50
687/687 [==============================] - 11s 15ms/step - loss: 1.7216 - sparse_categorical_accuracy: 0.4542 - val_loss: 0.9833 - val_sparse_categorical_accuracy: 0.6487
Epoch 2/50
687/687 [==============================] - 10s 15ms/step - loss: 0.6689 - sparse_categorical_accuracy: 0.7746 - val_loss: 0.3372 - val_sparse_categorical_accuracy: 0.9031
Epoch 3/50
687/687 [==============================] - 11s 16ms/step - loss: 0.3150 - sparse_categorical_accuracy: 0.8974 - val_loss: 0.1228 - val_sparse_categorical_accuracy: 0.9701
Epoch 4/50
687/687 [==============================] - 10s 14ms/step - loss: 0.1685 - sparse_categorical_accuracy: 0.9460 - val_loss: 0.1522 - val_sparse_categorical_accuracy: 0.9523
Epoch 5/50
687/687 [==============================] - 10s 14ms/step - loss: 0.1406 - sparse_categorical_accuracy: 0.9532 - val_loss: 0.0910 - val_sparse_categorical_accuracy: 0.9705
Epoch 6/50

```
687/687 [==============================] - 10s 14ms/step - loss: 0.0973 - sparse_categorical_accuracy: 0.9692 - val_loss: 0.1423 - val_sparse_categorical_accuracy: 0.9488
Epoch 7/50
687/687 [==============================] - 10s 14ms/step - loss: 0.0276 - sparse_categorical_accuracy: 0.9932 - val_loss: 0.0101 - val_sparse_categorical_accuracy: 0.9982
Epoch 8/50
687/687 [==============================] - 10s 14ms/step - loss: 0.1640 - sparse_categorical_accuracy: 0.9458 - val_loss: 0.1998 - val_sparse_categorical_accuracy: 0.9337
Epoch 9/50
687/687 [==============================] - 10s 15ms/step - loss: 0.0073 - sparse_categorical_accuracy: 0.9984 - val_loss: 0.0010 - val_sparse_categorical_accuracy: 1.0000
Epoch 10/50
687/687 [==============================] - 10s 14ms/step - loss: 8.3593e-04 - sparse_categorical_accuracy: 1.0000 - val_loss: 6.3991e-04 - val_sparse_categorical_accuracy: 1.0000
Epoch 11/50
687/687 [==============================] - 11s 15ms/step - loss: 5.5168e-04 - sparse_categorical_accuracy: 1.0000 - val_loss: 5.0015e-04 - val_sparse_categorical_accuracy: 1.0000
Epoch 12/50
687/687 [==============================] - 10s 15ms/step - loss: 4.4000e-04 - sparse_categorical_accuracy: 1.0000 - val_loss: 3.2029e-04 - val_sparse_categorical_accuracy: 1.0000
Epoch 13/50
687/687 [==============================] - 10s 15ms/step - loss: 0.2332 - sparse_categorical_accuracy: 0.9490 - val_loss: 0.1692 - val_sparse_categorical_accuracy: 0.9414
Epoch 14/50
687/687 [==============================] - 10s 15ms/step - loss: 0.0283 - sparse_categorical_accuracy: 0.9912 - val_loss: 0.0012 - val_sparse_categorical_accuracy: 1.0000
Epoch 15/50
687/687 [==============================] - 10s 15ms/step - loss: 7.8926e-04 - sparse_categorical_accuracy: 1.0000 - val_loss: 6.1704e-04 - val_sparse_categorical_accuracy: 1.0000
Model: "ANN_without_dropout_and_with_PReLU"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 Hidden_Layer_1 (Dense)      (None, 1024)              804864

 Hidden_Layer_2 (Dense)      (None, 512)               525312

 Output_Layer (Dense)        (None, 24)                12312

=================================================================
Total params: 1,342,488
Trainable params: 1,342,488
Non-trainable params: 0
_________________________________________________________________
End of training model with activation function =  PReLU
------------------------------------------------------------------------------------

              precision    recall  f1-score   support

           0       0.81      1.00      0.89       331
           1       1.00      1.00      1.00       432
           2       0.91      0.98      0.94       310
           3       0.86      0.92      0.89       245
           4       0.90      1.00      0.94       498
           5       0.84      0.99      0.91       247
           6       0.88      0.87      0.88       348
           7       0.96      0.91      0.93       436
           8       0.91      0.78      0.84       288
           9       0.96      0.65      0.77       331
          10       0.77      1.00      0.87       209
          11       0.80      0.78      0.79       394
```

12	0.87	0.57	0.69	291
13	0.99	0.83	0.90	246
14	0.94	0.98	0.96	347
15	0.73	0.90	0.80	164
16	0.67	0.72	0.69	144
17	0.76	0.55	0.64	246
18	0.77	0.67	0.72	248
19	0.73	0.66	0.69	266
20	0.67	0.72	0.70	346
21	0.58	0.80	0.67	206
22	0.68	0.81	0.74	267
23	0.83	0.72	0.77	332
accuracy			0.84	7172
macro avg	0.83	0.82	0.82	7172
weighted avg	0.84	0.84	0.83	7172

Execution of ANN training without drop out and with sigmoid

Epoch 1/50
687/687 [==============================] - 11s 15ms/step - loss: 1.9655 - sparse_categorical_accuracy: 0.3829 - val_loss: 1.3131 - val_sparse_categorical_accuracy: 0.5531
Epoch 2/50
687/687 [==============================] - 10s 14ms/step - loss: 0.9378 - sparse_categorical_accuracy: 0.6937 - val_loss: 0.7942 - val_sparse_categorical_accuracy: 0.7408
Epoch 3/50
687/687 [==============================] - 9s 13ms/step - loss: 0.5731 - sparse_categorical_accuracy: 0.8185 - val_loss: 0.4035 - val_sparse_categorical_accuracy: 0.8634
Epoch 4/50
687/687 [==============================] - 9s 14ms/step - loss: 0.3252 - sparse_categorical_accuracy: 0.9057 - val_loss: 0.3216 - val_sparse_categorical_accuracy: 0.8864
Epoch 5/50
687/687 [==============================] - 10s 14ms/step - loss: 0.2001 - sparse_categorical_accuracy: 0.9451 - val_loss: 0.1132 - val_sparse_categorical_accuracy: 0.9818
Epoch 6/50
687/687 [==============================] - 9s 13ms/step - loss: 0.1145 - sparse_categorical_accuracy: 0.9727 - val_loss: 0.0489 - val_sparse_categorical_accuracy: 0.9964
Epoch 7/50
687/687 [==============================] - 9s 13ms/step - loss: 0.0879 - sparse_categorical_accuracy: 0.9786 - val_loss: 0.1043 - val_sparse_categorical_accuracy: 0.9658
Epoch 8/50
687/687 [==============================] - 9s 13ms/step - loss: 0.0670 - sparse_categorical_accuracy: 0.9834 - val_loss: 0.0601 - val_sparse_categorical_accuracy: 0.9858
Epoch 9/50
687/687 [==============================] - 9s 13ms/step - loss: 0.0678 - sparse_categorical_accuracy: 0.9822 - val_loss: 0.0194 - val_sparse_categorical_accuracy: 0.9993
Epoch 10/50
687/687 [==============================] - 9s 14ms/step - loss: 0.0086 - sparse_categorical_accuracy: 0.9998 - val_loss: 0.0057 - val_sparse_categorical_accuracy: 1.0000
Epoch 11/50
687/687 [==============================] - 9s 13ms/step - loss: 0.0981 - sparse_categorical_accuracy: 0.9727 - val_loss: 0.0074 - val_sparse_categorical_accuracy: 1.0000
Epoch 12/50
687/687 [==============================] - 10s 14ms/step - loss: 0.0088 - sparse_categorical_accuracy: 0.9992 - val_loss: 0.0059 - val_sparse_categorical_accuracy: 1.0000
Epoch 13/50
687/687 [==============================] - 10s 14ms/step - loss: 0.0822 - sparse_categorical_accuracy: 0.9732 - val_loss: 0.0184 - val_sparse_categorical_accuracy: 0.9987
Model: "ANN_without_dropout_and_with_sigmoid"

Layer (type)	Output Shape	Param #

```
=================================================================
 Hidden_Layer_1 (Dense)      (None, 1024)            803840

 Hidden_Layer_2 (Dense)      (None, 512)             524800

 Output_Layer (Dense)        (None, 24)              12312

=================================================================
Total params: 1,340,952
Trainable params: 1,340,952
Non-trainable params: 0
_________________________________________________________________
End of training model with activation function =  sigmoid
-------------------------------------------------------------------------------------

              precision    recall  f1-score   support

           0       0.88      0.97      0.93       331
           1       0.99      0.97      0.98       432
           2       0.71      0.99      0.83       310
           3       0.96      0.87      0.91       245
           4       0.92      0.95      0.94       498
           5       0.90      0.91      0.91       247
           6       0.81      0.93      0.87       348
           7       1.00      0.91      0.95       436
           8       0.83      0.78      0.80       288
           9       0.79      0.70      0.74       331
          10       0.71      0.91      0.80       209
          11       0.82      0.79      0.80       394
          12       0.78      0.60      0.68       291
          13       1.00      0.66      0.80       246
          14       0.94      0.94      0.94       347
          15       0.70      0.88      0.78       164
          16       0.51      0.56      0.53       144
          17       0.52      0.56      0.54       246
          18       0.74      0.69      0.71       248
          19       0.69      0.50      0.58       266
          20       0.68      0.64      0.66       346
          21       0.58      0.81      0.68       206
          22       0.74      0.89      0.81       267
          23       0.91      0.67      0.77       332

    accuracy                           0.81      7172
   macro avg       0.80      0.80      0.79      7172
weighted avg       0.82      0.81      0.81      7172

Train time(secs) without dropout relu          167.58163785934448
Train time(secs) without dropout PReLU         153.41403079032898
Train time(secs) without dropout sigmoid       124.03638672828674
```

Appendix 5
The output of ANN_with_dropout.py

```
Execution of ANN training with drop out and with relu

Epoch 1/50
687/687 [==============================] - 11s 15ms/step - loss: 2.1051 - sparse_categorical_accuracy: 0.3243 - val_loss: 1.3289 - val_sparse_categorical_accuracy: 0.5653
Epoch 2/50
687/687 [==============================] - 10s 14ms/step - loss: 1.2648 - sparse_categorical_accuracy: 0.5642 - val_loss: 0.7573 - val_sparse_categorical_accuracy: 0.7594
```

```
Epoch 3/50
687/687 [==============================] - 10s 14ms/step - loss: 0.9272 - sparse_categorical_accuracy: 0.6804 - val_loss: 0.6695 - val_sparse_categorical_accuracy: 0.7733
Epoch 4/50
687/687 [==============================] - 9s 14ms/step - loss: 0.7194 - sparse_categorical_accuracy: 0.7496 - val_loss: 0.3675 - val_sparse_categorical_accuracy: 0.8927
Epoch 5/50
687/687 [==============================] - 9s 14ms/step - loss: 0.6020 - sparse_categorical_accuracy: 0.7865 - val_loss: 0.2726 - val_sparse_categorical_accuracy: 0.9211
Epoch 6/50
687/687 [==============================] - 10s 14ms/step - loss: 0.4856 - sparse_categorical_accuracy: 0.8315 - val_loss: 0.2183 - val_sparse_categorical_accuracy: 0.9370
Epoch 7/50
687/687 [==============================] - 10s 15ms/step - loss: 0.4201 - sparse_categorical_accuracy: 0.8530 - val_loss: 0.1823 - val_sparse_categorical_accuracy: 0.9457
Epoch 8/50
687/687 [==============================] - 10s 14ms/step - loss: 0.4120 - sparse_categorical_accuracy: 0.8565 - val_loss: 0.1312 - val_sparse_categorical_accuracy: 0.9630
Epoch 9/50
687/687 [==============================] - 10s 14ms/step - loss: 0.3542 - sparse_categorical_accuracy: 0.8765 - val_loss: 0.2156 - val_sparse_categorical_accuracy: 0.9253
Epoch 10/50
687/687 [==============================] - 9s 14ms/step - loss: 0.3423 - sparse_categorical_accuracy: 0.8788 - val_loss: 0.1009 - val_sparse_categorical_accuracy: 0.9796
Epoch 11/50
687/687 [==============================] - 9s 14ms/step - loss: 0.2966 - sparse_categorical_accuracy: 0.8975 - val_loss: 0.0971 - val_sparse_categorical_accuracy: 0.9732
Epoch 12/50
687/687 [==============================] - 10s 14ms/step - loss: 0.3010 - sparse_categorical_accuracy: 0.8971 - val_loss: 0.1271 - val_sparse_categorical_accuracy: 0.9676
Epoch 13/50
687/687 [==============================] - 10s 14ms/step - loss: 0.2558 - sparse_categorical_accuracy: 0.9113 - val_loss: 0.0449 - val_sparse_categorical_accuracy: 0.9913
Epoch 14/50
687/687 [==============================] - 10s 14ms/step - loss: 0.2501 - sparse_categorical_accuracy: 0.9139 - val_loss: 0.0645 - val_sparse_categorical_accuracy: 0.9876
Epoch 15/50
687/687 [==============================] - 10s 14ms/step - loss: 0.2669 - sparse_categorical_accuracy: 0.9062 - val_loss: 0.0527 - val_sparse_categorical_accuracy: 0.9871
Epoch 16/50
687/687 [==============================] - 10s 14ms/step - loss: 0.2357 - sparse_categorical_accuracy: 0.9204 - val_loss: 0.0670 - val_sparse_categorical_accuracy: 0.9836
Epoch 17/50
687/687 [==============================] - 10s 14ms/step - loss: 0.2364 - sparse_categorical_accuracy: 0.9190 - val_loss: 0.1040 - val_sparse_categorical_accuracy: 0.9683
Epoch 18/50
687/687 [==============================] - 10s 14ms/step - loss: 0.2741 - sparse_categorical_accuracy: 0.9073 - val_loss: 0.0879 - val_sparse_categorical_accuracy: 0.9761
Epoch 19/50
687/687 [==============================] - 9s 14ms/step - loss: 0.2373 - sparse_categorical_accuracy: 0.9185 - val_loss: 0.0675 - val_sparse_categorical_accuracy: 0.9787
Model: "ANN_with_dropout_and_with_relu"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 Hidden_Layer_1 (Dense)      (None, 1024)              803840    

 Dropout_Layer_1 (Dropout)   (None, 1024)              0         

 Hidden_Layer_2 (Dense)      (None, 512)               524800    
```

Dropout_Layer_2 (Dropout)   (None, 512)   0

Output_Layer (Dense)   (None, 24)   12312

```
=================================================================
Total params: 1,340,952
Trainable params: 1,340,952
Non-trainable params: 0
_________________________________________________________________
End of training model with activation function =  relu
-------------------------------------------------------------------------------------

              precision    recall  f1-score   support

           0       0.86      1.00      0.92       331
           1       0.96      0.98      0.97       432
           2       0.92      1.00      0.96       310
           3       0.84      0.94      0.88       245
           4       0.89      1.00      0.94       498
           5       0.92      0.95      0.93       247
           6       0.86      0.81      0.83       348
           7       0.96      0.95      0.95       436
           8       0.82      0.78      0.80       288
           9       0.86      0.61      0.71       331
          10       0.66      0.75      0.70       209
          11       0.69      0.80      0.74       394
          12       0.73      0.44      0.55       291
          13       0.95      0.92      0.94       246
          14       0.91      1.00      0.95       347
          15       0.64      1.00      0.78       164
          16       0.40      0.42      0.41       144
          17       0.68      0.66      0.67       246
          18       0.65      0.74      0.69       248
          19       0.63      0.55      0.59       266
          20       0.76      0.47      0.58       346
          21       0.53      0.86      0.65       206
          22       0.80      0.75      0.77       267
          23       0.84      0.52      0.64       332

    accuracy                           0.80      7172
   macro avg       0.78      0.79      0.77      7172
weighted avg       0.81      0.80      0.80      7172
```

Execution of ANN training with drop out and with PReLU

```
Epoch 1/50
687/687 [==============================] - 11s 14ms/step - loss: 2.0118 - sparse_categorical_accuracy: 0.3559 - val_loss: 1.1725 - val_sparse_categorical_accuracy: 0.6321
Epoch 2/50
687/687 [==============================] - 11s 15ms/step - loss: 1.0110 - sparse_categorical_accuracy: 0.6553 - val_loss: 0.8512 - val_sparse_categorical_accuracy: 0.6933
Epoch 3/50
687/687 [==============================] - 10s 15ms/step - loss: 0.6105 - sparse_categorical_accuracy: 0.7872 - val_loss: 0.2940 - val_sparse_categorical_accuracy: 0.9140
Epoch 4/50
687/687 [==============================] - 11s 15ms/step - loss: 0.4325 - sparse_categorical_accuracy: 0.8466 - val_loss: 0.2514 - val_sparse_categorical_accuracy: 0.9151
Epoch 5/50
687/687 [==============================] - 11s 16ms/step - loss: 0.3111 - sparse_categorical_accuracy: 0.8907 - val_loss: 0.1746 - val_sparse_categorical_accuracy: 0.9381
Epoch 6/50
```

687/687 [==============================] - 11s 16ms/step - loss: 0.2460 - sparse_categorical_accuracy: 0.9112 - val_loss: 0.1210 - val_sparse_categorical_accuracy: 0.9645
Epoch 7/50
687/687 [==============================] - 10s 15ms/step - loss: 0.2151 - sparse_categorical_accuracy: 0.9261 - val_loss: 0.0440 - val_sparse_categorical_accuracy: 0.9902
Epoch 8/50
687/687 [==============================] - 10s 15ms/step - loss: 0.1899 - sparse_categorical_accuracy: 0.9349 - val_loss: 0.0483 - val_sparse_categorical_accuracy: 0.9922
Epoch 9/50
687/687 [==============================] - 10s 15ms/step - loss: 0.1785 - sparse_categorical_accuracy: 0.9382 - val_loss: 0.0375 - val_sparse_categorical_accuracy: 0.9911
Epoch 10/50
687/687 [==============================] - 10s 15ms/step - loss: 0.1766 - sparse_categorical_accuracy: 0.9386 - val_loss: 0.0831 - val_sparse_categorical_accuracy: 0.9683
Epoch 11/50
687/687 [==============================] - 11s 16ms/step - loss: 0.1718 - sparse_categorical_accuracy: 0.9412 - val_loss: 0.1068 - val_sparse_categorical_accuracy: 0.9669
Epoch 12/50
687/687 [==============================] - 11s 16ms/step - loss: 0.1553 - sparse_categorical_accuracy: 0.9482 - val_loss: 0.0181 - val_sparse_categorical_accuracy: 0.9954
Epoch 13/50
687/687 [==============================] - 10s 15ms/step - loss: 0.1524 - sparse_categorical_accuracy: 0.9512 - val_loss: 0.0685 - val_sparse_categorical_accuracy: 0.9745
Epoch 14/50
687/687 [==============================] - 10s 15ms/step - loss: 0.1224 - sparse_categorical_accuracy: 0.9589 - val_loss: 0.0194 - val_sparse_categorical_accuracy: 0.9949
Epoch 15/50
687/687 [==============================] - 10s 15ms/step - loss: 0.1390 - sparse_categorical_accuracy: 0.9572 - val_loss: 0.0339 - val_sparse_categorical_accuracy: 0.9883
Epoch 16/50
687/687 [==============================] - 10s 15ms/step - loss: 0.1536 - sparse_categorical_accuracy: 0.9512 - val_loss: 0.0143 - val_sparse_categorical_accuracy: 0.9962
Epoch 17/50
687/687 [==============================] - 11s 15ms/step - loss: 0.1255 - sparse_categorical_accuracy: 0.9602 - val_loss: 0.0584 - val_sparse_categorical_accuracy: 0.9801
Model: "ANN_with_dropout_and_with_PReLU"

```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 Hidden_Layer_1 (Dense)      (None, 1024)              804864

 Dropout_Layer_1 (Dropout)   (None, 1024)              0

 Hidden_Layer_2 (Dense)      (None, 512)               525312

 Dropout_Layer_2 (Dropout)   (None, 512)               0

 Output_Layer (Dense)        (None, 24)                12312

=================================================================
Total params: 1,342,488
Trainable params: 1,342,488
Non-trainable params: 0
_________________________________________________________________
End of training model with activation function =  PReLU
----------------------------------------------------------------------------------------

              precision    recall  f1-score   support

           0       0.89      1.00      0.94       331
           1       0.94      1.00      0.97       432
```

2	0.98	1.00	0.99	310
3	0.99	0.78	0.87	245
4	0.81	1.00	0.89	498
5	0.79	1.00	0.88	247
6	0.93	0.84	0.88	348
7	1.00	0.90	0.95	436
8	0.69	0.85	0.76	288
9	0.97	0.39	0.56	331
10	0.73	1.00	0.84	209
11	0.88	0.75	0.81	394
12	0.86	0.44	0.58	291
13	0.85	0.84	0.85	246
14	1.00	0.98	0.99	347
15	0.79	0.87	0.83	164
16	0.71	0.71	0.71	144
17	0.66	0.75	0.70	246
18	0.59	0.66	0.62	248
19	0.82	0.74	0.78	266
20	0.69	0.72	0.70	346
21	0.45	0.70	0.55	206
22	0.77	0.79	0.78	267
23	0.72	0.61	0.66	332
accuracy			0.81	7172
macro avg	0.81	0.80	0.80	7172
weighted avg	0.83	0.81	0.81	7172

Execution of ANN training with drop out and with sigmoid

```
Epoch 1/50
687/687 [==============================] - 10s 14ms/step - loss: 2.1291 - sparse_categorical_accuracy: 0.3269 - val_loss: 1.2657 - val_sparse_categorical_accuracy: 0.6041
Epoch 2/50
687/687 [==============================] - 9s 14ms/step - loss: 1.1334 - sparse_categorical_accuracy: 0.6222 - val_loss: 0.8160 - val_sparse_categorical_accuracy: 0.7279
Epoch 3/50
687/687 [==============================] - 9s 14ms/step - loss: 0.7407 - sparse_categorical_accuracy: 0.7545 - val_loss: 0.6199 - val_sparse_categorical_accuracy: 0.7929
Epoch 4/50
687/687 [==============================] - 10s 14ms/step - loss: 0.5189 - sparse_categorical_accuracy: 0.8278 - val_loss: 0.3326 - val_sparse_categorical_accuracy: 0.8973
Epoch 5/50
687/687 [==============================] - 9s 14ms/step - loss: 0.3581 - sparse_categorical_accuracy: 0.8839 - val_loss: 0.2660 - val_sparse_categorical_accuracy: 0.9179
Epoch 6/50
687/687 [==============================] - 10s 15ms/step - loss: 0.2625 - sparse_categorical_accuracy: 0.9170 - val_loss: 0.2672 - val_sparse_categorical_accuracy: 0.9177
Epoch 7/50
687/687 [==============================] - 10s 14ms/step - loss: 0.2166 - sparse_categorical_accuracy: 0.9326 - val_loss: 0.0894 - val_sparse_categorical_accuracy: 0.9791
Epoch 8/50
687/687 [==============================] - 10s 14ms/step - loss: 0.1423 - sparse_categorical_accuracy: 0.9565 - val_loss: 0.0992 - val_sparse_categorical_accuracy: 0.9705
Epoch 9/50
687/687 [==============================] - 10s 15ms/step - loss: 0.1328 - sparse_categorical_accuracy: 0.9581 - val_loss: 0.1715 - val_sparse_categorical_accuracy: 0.9465
Epoch 10/50
687/687 [==============================] - 10s 14ms/step - loss: 0.1189 - sparse_categorical_accuracy: 0.9650 - val_loss: 0.0344 - val_sparse_categorical_accuracy: 0.9949
Epoch 11/50
```

```
687/687 [==============================] - 10s 14ms/step - loss: 0.0904 - sparse_categorical_accuracy: 0.9738 - val_loss: 0.0575 - val_sparse_categorical_accuracy: 0.9838
Epoch 12/50
687/687 [==============================] - 10s 15ms/step - loss: 0.0872 - sparse_categorical_accuracy: 0.9738 - val_loss: 0.0421 - val_sparse_categorical_accuracy: 0.9885
Epoch 13/50
687/687 [==============================] - 10s 14ms/step - loss: 0.0719 - sparse_categorical_accuracy: 0.9785 - val_loss: 0.0818 - val_sparse_categorical_accuracy: 0.9669
Epoch 14/50
687/687 [==============================] - 10s 14ms/step - loss: 0.0813 - sparse_categorical_accuracy: 0.9758 - val_loss: 0.1021 - val_sparse_categorical_accuracy: 0.9619
Epoch 15/50
687/687 [==============================] - 10s 14ms/step - loss: 0.0929 - sparse_categorical_accuracy: 0.9701 - val_loss: 0.0548 - val_sparse_categorical_accuracy: 0.9845
Epoch 16/50
687/687 [==============================] - 10s 15ms/step - loss: 0.0570 - sparse_categorical_accuracy: 0.9829 - val_loss: 0.0852 - val_sparse_categorical_accuracy: 0.9736
Epoch 17/50
687/687 [==============================] - 10s 14ms/step - loss: 0.0710 - sparse_categorical_accuracy: 0.9773 - val_loss: 0.0202 - val_sparse_categorical_accuracy: 0.9964
Epoch 18/50
687/687 [==============================] - 10s 15ms/step - loss: 0.0598 - sparse_categorical_accuracy: 0.9819 - val_loss: 0.0135 - val_sparse_categorical_accuracy: 0.9976
Epoch 19/50
687/687 [==============================] - 10s 14ms/step - loss: 0.0853 - sparse_categorical_accuracy: 0.9734 - val_loss: 0.0227 - val_sparse_categorical_accuracy: 0.9958
Model: "ANN_with_dropout_and_with_sigmoid"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 Hidden_Layer_1 (Dense)      (None, 1024)              803840

 Dropout_Layer_1 (Dropout)   (None, 1024)              0

 Hidden_Layer_2 (Dense)      (None, 512)               524800

 Dropout_Layer_2 (Dropout)   (None, 512)               0

 Output_Layer (Dense)        (None, 24)                12312

=================================================================
Total params: 1,340,952
Trainable params: 1,340,952
Non-trainable params: 0
_________________________________________________________________
End of training model with activation function =  sigmoid
-----------------------------------------------------------------------------------------

              precision    recall  f1-score   support

           0       0.81      1.00      0.90       331
           1       0.96      0.99      0.97       432
           2       0.96      0.93      0.95       310
           3       0.81      1.00      0.89       245
           4       0.96      0.95      0.95       498
           5       0.80      0.99      0.88       247
           6       0.93      0.77      0.84       348
           7       0.95      0.91      0.93       436
           8       0.83      0.91      0.87       288
           9       0.94      0.69      0.79       331
          10       0.98      0.96      0.97       209
```

11	*0.88*	*0.80*	*0.84*	*394*
12	*0.88*	*0.66*	*0.75*	*291*
13	*0.99*	*0.75*	*0.85*	*246*
14	*0.96*	*0.97*	*0.96*	*347*
15	*0.55*	*0.95*	*0.69*	*164*
16	*0.71*	*0.28*	*0.40*	*144*
17	*0.55*	*0.64*	*0.59*	*246*
18	*0.69*	*0.76*	*0.72*	*248*
19	*0.59*	*0.63*	*0.61*	*266*
20	*0.67*	*0.65*	*0.66*	*346*
21	*0.53*	*0.69*	*0.60*	*206*
22	*0.73*	*0.61*	*0.66*	*267*
23	*0.82*	*0.81*	*0.82*	*332*
accuracy			*0.82*	*7172*
macro avg	*0.81*	*0.80*	*0.80*	*7172*
weighted avg	*0.83*	*0.82*	*0.82*	*7172*

Train time(secs) with dropout relu *185.1030809879303*
Train time(secs) with dropout PReLU *179.4084279537201*
Train time(secs) with dropout sigmoid *188.11008310317993*

Appendix 6

F1-scores for CNN models

	CNN without Dropouts			CNN with Dropouts		
	ReLu	PReLu	sigmoid	ReLu	PReLu	sigmoid
a	0.95	0.94	0.90	0.95	0.95	0.96
b	0.95	0.98	0.98	0.95	0.95	0.98
c	0.95	0.99	0.94	1.00	0.91	1.00
d	0.84	0.92	0.91	0.82	0.89	0.83
e	0.90	0.89	0.91	0.90	0.98	0.93
f	0.96	0.98	0.90	0.85	0.92	0.99
g	0.86	0.90	0.90	0.85	0.81	0.88
h	0.92	0.91	0.86	0.85	0.88	0.89
i	0.81	0.85	0.88	0.77	0.90	0.91
k	0.61	0.71	0.78	0.84	0.85	0.81
l	0.88	1.00	0.93	0.73	0.92	1.00
m	0.90	0.68	0.79	0.71	0.77	0.81
n	0.75	0.58	0.64	0.68	0.73	0.62
o	0.79	0.73	0.77	0.78	0.90	0.94
p	0.90	0.99	0.89	0.95	0.86	0.96
q	0.88	0.87	0.79	0.86	0.88	0.93
r	0.45	0.67	0.67	0.28	0.57	0.42
s	0.65	0.67	0.67	0.62	0.79	0.69
t	0.66	0.73	0.70	0.72	0.73	0.79
u	0.76	0.57	0.69	0.77	0.64	0.74
v	0.84	0.64	0.89	0.85	0.88	0.92
w	0.81	0.61	0.80	0.72	0.93	0.94
x	0.84	0.76	0.85	0.70	0.83	0.95
y	0.73	0.88	0.89	0.84	0.88	0.91

F1-scores for ANN models

	ANN without Dropouts			ANN with Dropouts		
	ReLu	PReLu	sigmoid	ReLu	PReLu	sigmoid
a	0.90	0.89	0.93	0.92	0.94	0.90
b	0.96	1.00	0.98	0.97	0.97	0.97
c	0.96	0.94	0.83	0.96	0.99	0.95
d	0.91	0.89	0.91	0.88	0.87	0.89
e	0.95	0.94	0.94	0.94	0.89	0.95
f	0.88	0.91	0.91	0.93	0.88	0.88
g	0.84	0.88	0.87	0.83	0.88	0.84
h	0.95	0.93	0.95	0.95	0.88	0.93
i	0.87	0.84	0.80	0.80	0.76	0.87
k	0.73	0.77	0.74	0.71	0.56	0.79
l	0.88	0.87	0.80	0.70	0.84	0.97
m	0.74	0.79	0.80	0.74	0.81	0.84
n	0.74	0.69	0.68	0.55	0.58	0.75
o	0.90	0.90	0.80	0.94	0.85	0.85
p	0.97	0.96	0.94	0.95	0.99	0.96
q	0.79	0.80	0.78	0.78	0.83	0.69
r	0.64	0.69	0.53	0.41	0.71	0.40
s	0.65	0.64	0.54	0.67	0.70	0.59
t	0.66	0.72	0.71	0.69	0.62	0.72
u	0.57	0.69	0.58	0.59	0.78	0.61
v	0.80	0.70	0.66	0.58	0.70	0.66
w	0.68	0.67	0.68	0.65	0.55	0.60
x	0.80	0.74	0.81	0.77	0.78	0.66
y	0.76	0.77	0.77	0.64	0.66	0.82

8. References

[1] “NHIS - National Health Interview Survey,” *Centers for Disease Control and Prevention*, 02-Dec-2021. [Online]. Available: https://www.cdc.gov/nchs/nhis/index.htm?CDC_AA_refVal=https%3A%2F%2Fwww.cdc.gov%2Fnchs%2Fnhis.htm. [Accessed: 21-Dec-2021].

[2]S. Saha, “A Comprehensive Guide to Convolutional Neural Networks - the ELI5 way” *towards data science,* 15-Dec-2018. [Online]. Available: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53 [Accessed: 28-Nov-2021].

[3] By: IBM Cloud Education, “What are neural networks?,” *IBM*. [Online]. Available: https://www.ibm.com/cloud/learn/neural-networks. [Accessed: 28-Nov-2021].

[4] “A beginner's Guide to Neural Networks and deep learning,” *Pathmind*. [Online]. Available: https://wiki.pathmind.com/neural-network. [Accessed: 28-Nov-2021].

[5][1]“K-Nearest Neighbor(KNN) algorithm for Machine Learning - Javatpoint,” *www.javatpoint.com*. [Online]. Available: https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning. [Accessed: 28-Nov-2021].