# CSE 474/574
# INTRODUCTION TO MACHINE LEARNING
# PROJECT ASSIGNMENT 2 REPORT

Group Number: 86

Members:

Arshabh Semwal(Person No. 50419031)
Utkarsh Bansal(Person No. 50415035)

## 1. Introduction

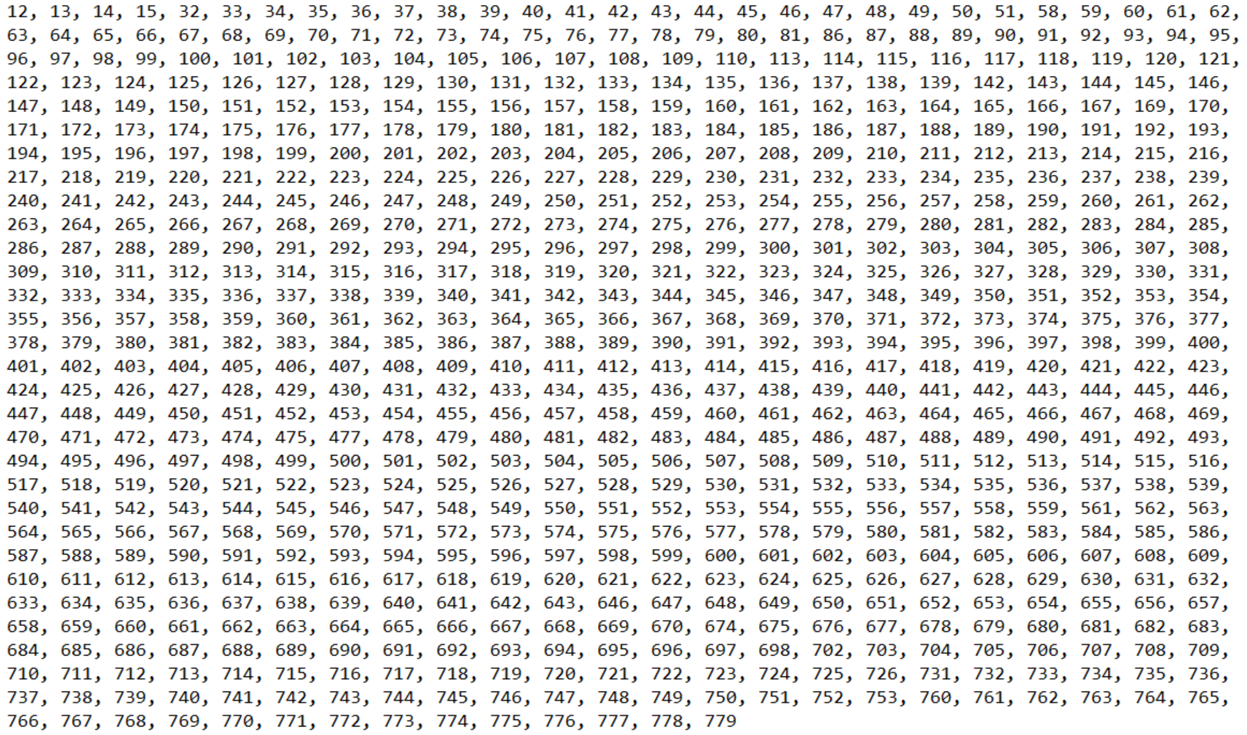
As part of the project assignment 2, we are given the MNIST dataset for image classification. Our task is to train a multilayer perceptron to classify images in the dataset.

We are also given a celeb face dataset in which we are required to detect whether the person is wearing glasses or not. This task needs to be done using the multilayer perceptron which was implemented for the MNIST dataset. These results also need to be compared with the deep neural network and CNN provided.

## 2. Feature Selection

Feature selection is a powerful method of reducing the complexity of the neural network. We have removed the features which have 0 value for all the training examples. This is because these features will not provide any information that will help us in classifying the images.

The following features were used for training the neural network:



```
12, 13, 14, 15, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 58, 59, 60, 61, 62,
63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95,
96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 113, 114, 115, 116, 117, 118, 119, 120, 121,
122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137, 138, 139, 142, 143, 144, 145, 146,
147, 148, 149, 150, 151, 152, 153, 154, 155, 156, 157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 169, 170,
171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193,
194, 195, 196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207, 208, 209, 210, 211, 212, 213, 214, 215, 216,
217, 218, 219, 220, 221, 222, 223, 224, 225, 226, 227, 228, 229, 230, 231, 232, 233, 234, 235, 236, 237, 238, 239,
240, 241, 242, 243, 244, 245, 246, 247, 248, 249, 250, 251, 252, 253, 254, 255, 256, 257, 258, 259, 260, 261, 262,
263, 264, 265, 266, 267, 268, 269, 270, 271, 272, 273, 274, 275, 276, 277, 278, 279, 280, 281, 282, 283, 284, 285,
286, 287, 288, 289, 290, 291, 292, 293, 294, 295, 296, 297, 298, 299, 300, 301, 302, 303, 304, 305, 306, 307, 308,
309, 310, 311, 312, 313, 314, 315, 316, 317, 318, 319, 320, 321, 322, 323, 324, 325, 326, 327, 328, 329, 330, 331,
332, 333, 334, 335, 336, 337, 338, 339, 340, 341, 342, 343, 344, 345, 346, 347, 348, 349, 350, 351, 352, 353, 354,
355, 356, 357, 358, 359, 360, 361, 362, 363, 364, 365, 366, 367, 368, 369, 370, 371, 372, 373, 374, 375, 376, 377,
378, 379, 380, 381, 382, 383, 384, 385, 386, 387, 388, 389, 390, 391, 392, 393, 394, 395, 396, 397, 398, 399, 400,
401, 402, 403, 404, 405, 406, 407, 408, 409, 410, 411, 412, 413, 414, 415, 416, 417, 418, 419, 420, 421, 422, 423,
424, 425, 426, 427, 428, 429, 430, 431, 432, 433, 434, 435, 436, 437, 438, 439, 440, 441, 442, 443, 444, 445, 446,
447, 448, 449, 450, 451, 452, 453, 454, 455, 456, 457, 458, 459, 460, 461, 462, 463, 464, 465, 466, 467, 468, 469,
470, 471, 472, 473, 474, 475, 477, 478, 479, 480, 481, 482, 483, 484, 485, 486, 487, 488, 489, 490, 491, 492, 493,
494, 495, 496, 497, 498, 499, 500, 501, 502, 503, 504, 505, 506, 507, 508, 509, 510, 511, 512, 513, 514, 515, 516,
517, 518, 519, 520, 521, 522, 523, 524, 525, 526, 527, 528, 529, 530, 531, 532, 533, 534, 535, 536, 537, 538, 539,
540, 541, 542, 543, 544, 545, 546, 547, 548, 549, 550, 551, 552, 553, 554, 555, 556, 557, 558, 559, 561, 562, 563,
564, 565, 566, 567, 568, 569, 570, 571, 572, 573, 574, 575, 576, 577, 578, 579, 580, 581, 582, 583, 584, 585, 586,
587, 588, 589, 590, 591, 592, 593, 594, 595, 596, 597, 598, 599, 600, 601, 602, 603, 604, 605, 606, 607, 608, 609,
610, 611, 612, 613, 614, 615, 616, 617, 618, 619, 620, 621, 622, 623, 624, 625, 626, 627, 628, 629, 630, 631, 632,
633, 634, 635, 636, 637, 638, 639, 640, 641, 642, 643, 646, 647, 648, 649, 650, 651, 652, 653, 654, 655, 656, 657,
658, 659, 660, 661, 662, 663, 664, 665, 666, 667, 668, 669, 670, 674, 675, 676, 677, 678, 679, 680, 681, 682, 683,
684, 685, 686, 687, 688, 689, 690, 691, 692, 693, 694, 695, 696, 697, 698, 702, 703, 704, 705, 706, 707, 708, 709,
710, 711, 712, 713, 714, 715, 716, 717, 718, 719, 720, 721, 722, 723, 724, 725, 726, 731, 732, 733, 734, 735, 736,
737, 738, 739, 740, 741, 742, 743, 744, 745, 746, 747, 748, 749, 750, 751, 752, 753, 760, 761, 762, 763, 764, 765,
766, 767, 768, 769, 770, 771, 772, 773, 774, 775, 776, 777, 778, 779
```

Fig.1: Selected Features

Total number of selected features: 717

## 3. Neural Network with 1 hidden layer

### 3.1 Analyzing results for all values of regularization parameter and number of hidden units

The following table gives the different values of accuracies achieved by training the model using different values of ƛ and number of hidden nodes.

| S.No. | Hidden Units | Lambda (λ) | Training Accuracy | Validation Accuracy | Testing Accuracy | Training Time (seconds) |
|---|---|---|---|---|---|---|
| 1. | 4 | 0 | 58.936 | 58.32 | 58.52 | 38.994042 |
| 2. | | 10 | 60.212 | 59.78 | 59.52 | 39.523467 |
| 3. | | 20 | 58.450 | 58.13 | 57.80 | 37.987799 |
| 4. | | 30 | 59.780 | 59.19 | 58.97 | 39.064510 |
| 5. | | 40 | 59.486 | 59.22 | 58.65 | 38.667804 |
| 6. | | 50 | 58.542 | 57.95 | 57.71 | 37.745211 |
| 7. | | 60 | 57.364 | 56.77 | 56.83 | 37.277275 |
| 8. | 8 | 0 | 87.572 | 86.82 | 87.61 | 44.909581 |
| 9. | | 10 | 88.124 | 87.73 | 88.07 | 47.239825 |
| 10. | | 20 | 86.950 | 86.38 | 87.28 | 42.208883 |
| 11. | | 30 | 89.874 | 89.30 | 89.52 | 45.051012 |
| 12. | | 40 | 88.486 | 87.97 | 88.60 | 45.329324 |
| 13. | | 50 | 87.654 | 87.41 | 87.82 | 41.417437 |
| 14. | | 60 | 89.064 | 88.53 | 89.11 | 56.215964 |
| 15. | 12 | 0 | 91.522 | 90.48 | 91.03 | 54.815639 |
| 16. | | 10 | 91.586 | 90.67 | 91.36 | 66.665178 |
| 17. | | 20 | 91.618 | 90.60 | 91.30 | 53.649781 |
| 18. | | 30 | 91.814 | 90.90 | 91.45 | 56.192001 |
| 19. | | 40 | 91.564 | 91.02 | 91.33 | 53.761525 |
| 20. | | 50 | 91.626 | 90.75 | 91.32 | 53.129061 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 21. | | 60 | 91.234 | 90.49 | 91.18 | 57.460227 |
| 22. | 16 | 0 | 93.048 | 92.33 | 92.56 | 77.209275 |
| 23. | | 10 | 93.556 | 92.70 | 93.08 | 67.846944 |
| 24. | | 20 | 93.274 | 92.62 | 92.90 | 61.504588 |
| 25. | | 30 | 93.166 | 92.42 | 92.90 | 64.187501 |
| 26. | | 40 | 93.088 | 92.48 | 92.89 | 73.483336 |
| 27. | | 50 | 92.840 | 92.24 | 92.64 | 61.088850 |
| 28. | | 60 | 92.658 | 92.03 | 92.61 | 60.144770 |
| 29. | 20 | 0 | 93.262 | 92.73 | 93.13 | 59.724584 |
| 30. | | 10 | 93.584 | 93.07 | 93.47 | 66.998732 |
| 31. | | 20 | 93.404 | 92.73 | 93.24 | 61.141419 |
| 32. | | 30 | 93.244 | 92.48 | 93.21 | 67.956336 |
| 33. | | 40 | 92.932 | 92.38 | 93.11 | 61.078771 |
| 34. | | 50 | 92.590 | 92.03 | 92.71 | 65.193018 |
| 35. | | 60 | 92.328 | 91.89 | 92.34 | 67.572325 |

The maximum accuracy on testing set is 93.47% on a neural network with 20 hidden layer and the value of λ as 10.

### 3.2 Regularization parameter vs accuracy for different number of hidden units

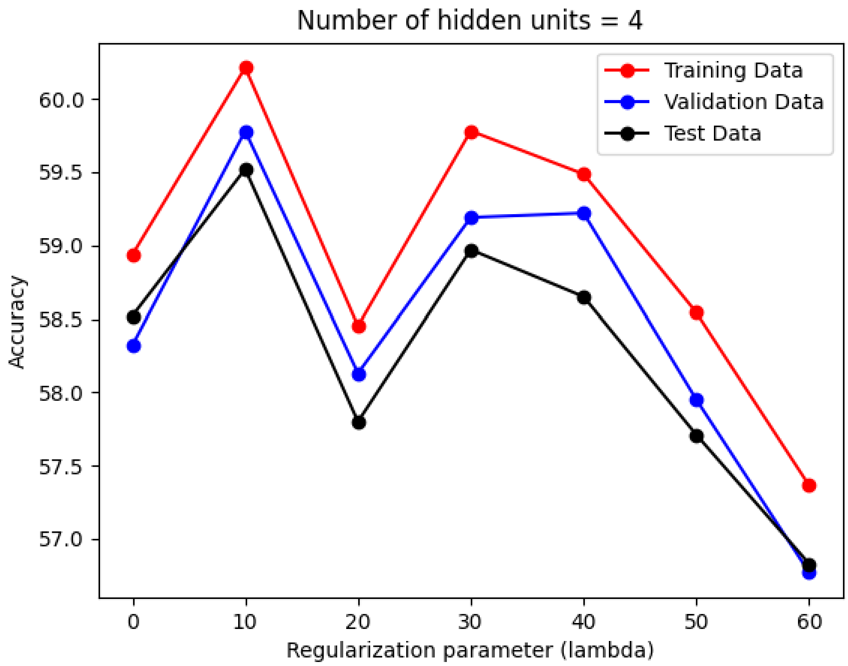


Fig. 2: 4 hidden units

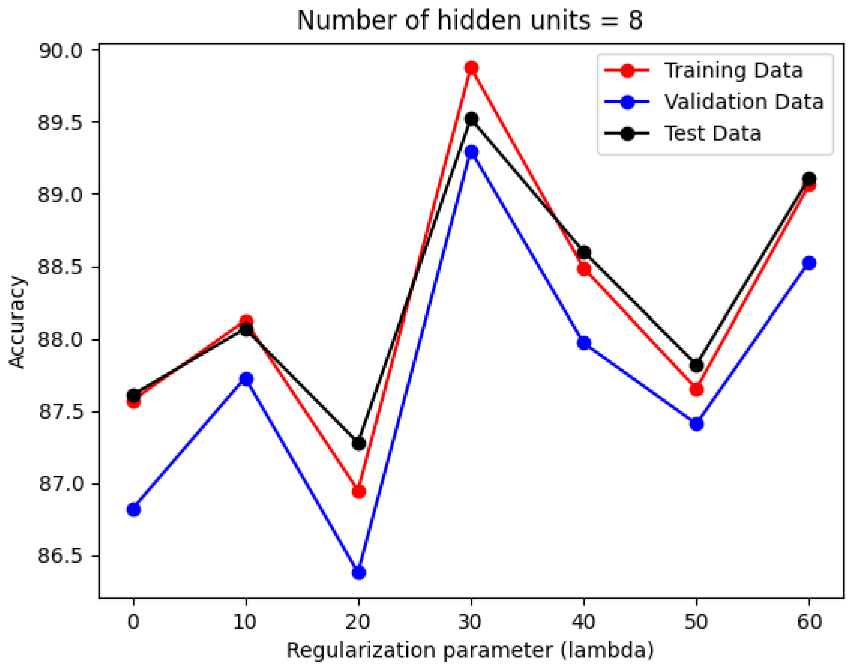


Fig. 3: 8 hidden units

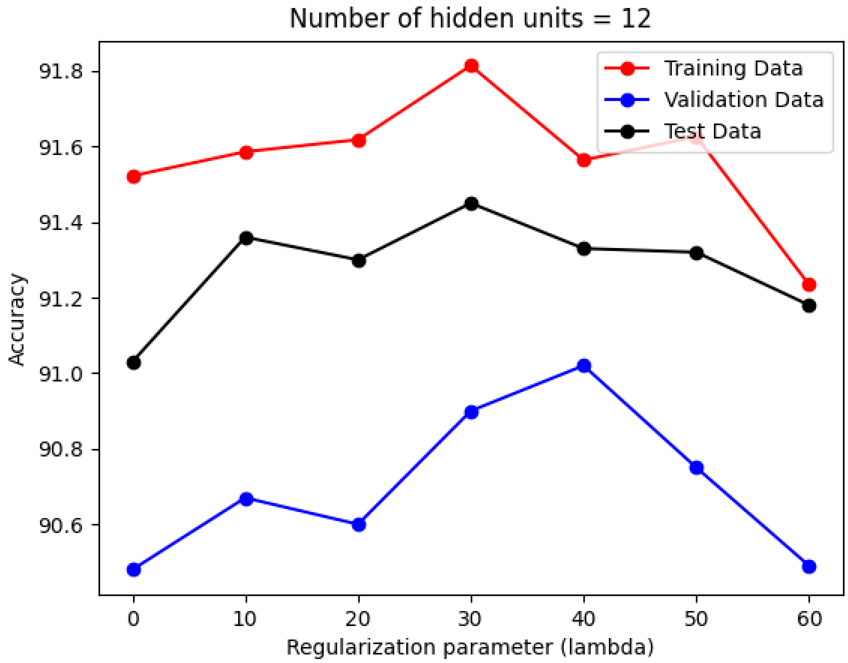


Fig. 4: 12 hidden units

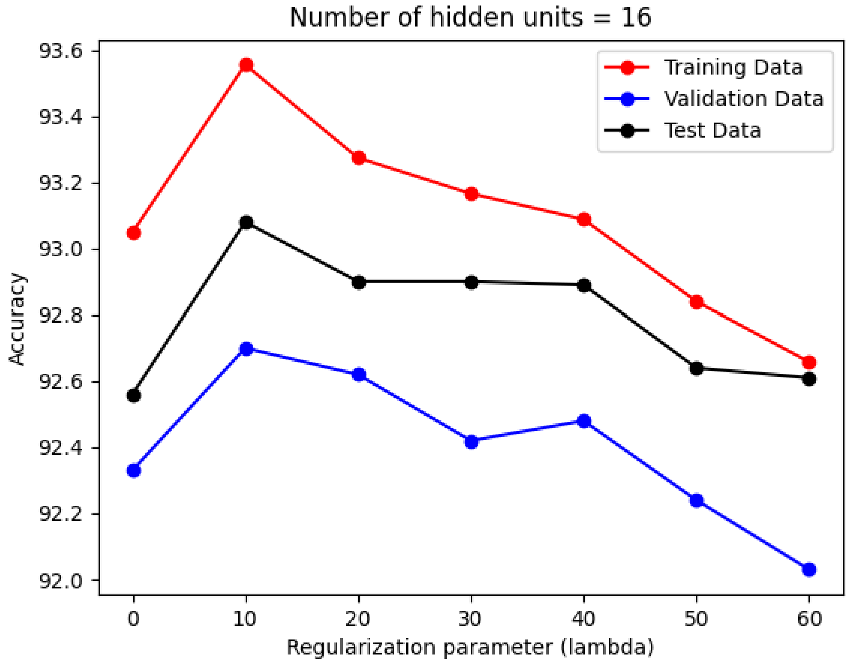


Fig. 5: 16 hidden units

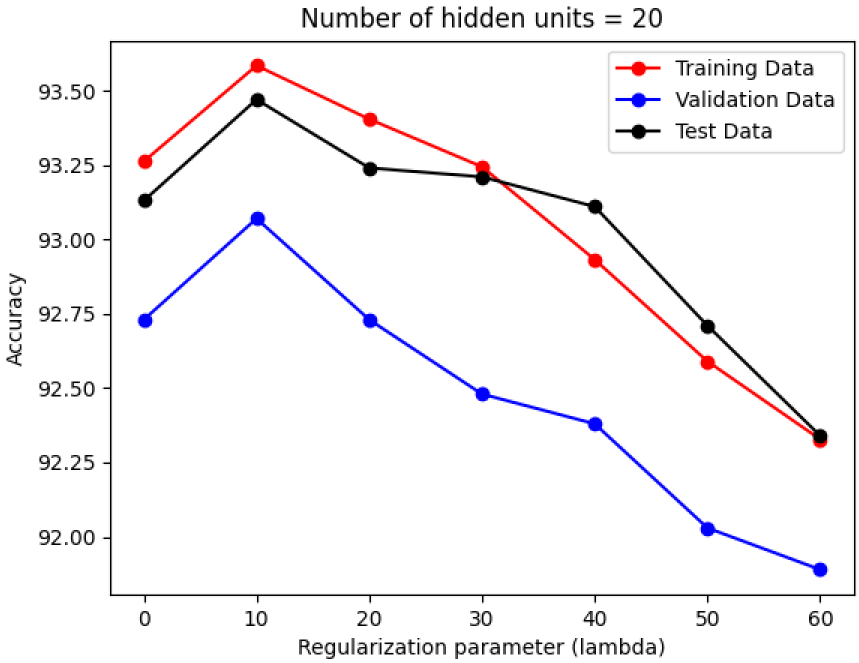


Fig. 6: 20 hidden units

From the above given graphs we can infer the following points:

1. The best accuracy on all datasets is achieved when the number of hidden units is 20. This is because more hidden layer features are generated for the model on which it makes the final decision.
2. When hidden units are 20, the best accuracy on validation data is found on the regularization parameter ƛ = 10. The validation set accuracy achieved is **93.07%**. Also, the test set accuracy achieved is **93.47%.** After 10, the weights get too constrained because of ƛ and the model is not able to give the best results.

### 3.3 Training time vs Number of hidden units

The graph below shows the relation between training time and number of hidden units used:

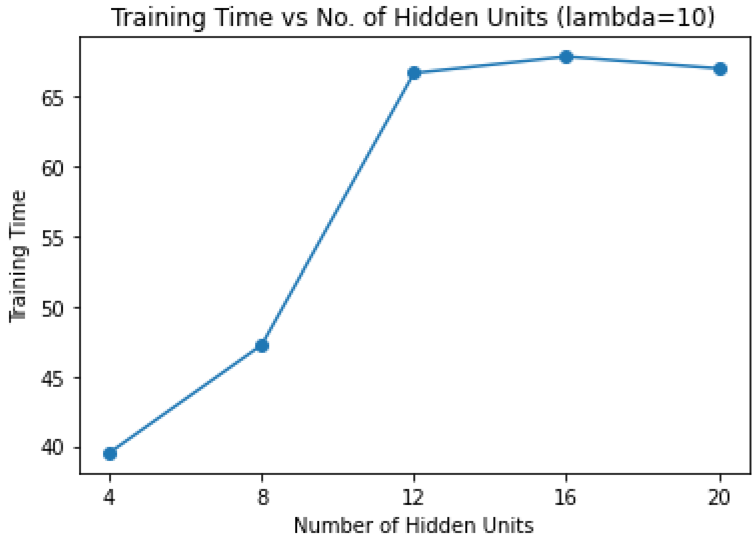


Fig. 7: Training Time vs Number of Hidden Units (At lambda = 10)

**Observations:**

1. As the number of hidden units increases, the training time increases.
2. This is because the model becomes more complex, and more parameters have to be dealt with in every step.

### 3.4 Validation Accuracy vs Number of hidden units

The graph below shows the relation between validation accuracy achieved and the number of hidden units used:

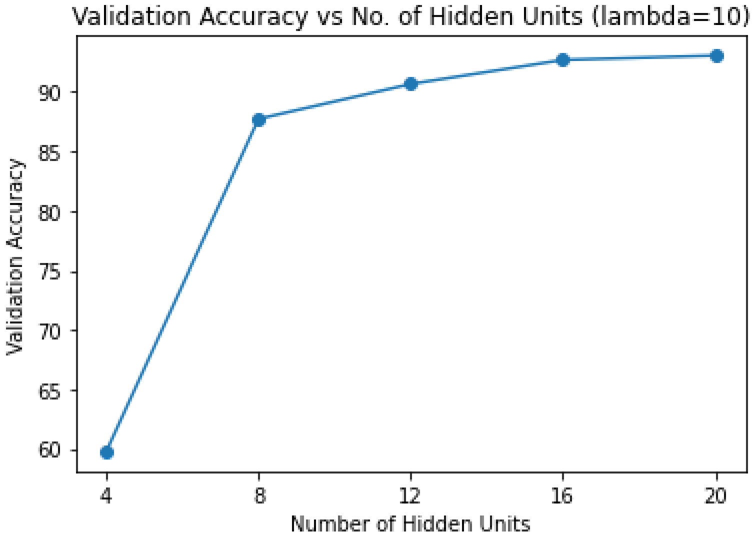


Fig. 8: Validation accuracy vs Number of hidden units

**Observations:**

1. As we increase the number of hidden units, the accuracy of the model increases.
2. This is because more parameters are tuned in every step and the model ends up getting a much better understanding of the data.
3. One thing to note is that more hidden units do not always mean better accuracy. This is because after a certain point, due to overfitting, the model will start performing extremely well on the training data but will give worse results on the validation and the test data.

## 4. Comparison of Neural Network with a Deep Neural Network (Using Tensorflow) in terms of Accuracy and Training time:

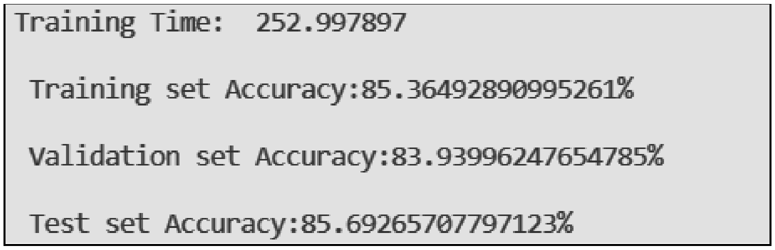
We have compared the performance of single neural net and deep neural network with 2 layers, 3 layers, 5 layers and 7 layers respectively and the results are presented below in tabular form:

| No. of Layers | Script | Accuracy | Training time |
|---|---|---|---|
| 1 | facennScript.py | 0.8569265 | 4m 13s |
| 2 | deepnnScript.py | 0.8145344 | 4m 14s |
| 3 | deepnnScript.py | 0.8016654 | 4m 26s |
| 5 | deepnnScript.py | 0.7649508 | 5m 10s |
| 7 | deepnnScript.py | 0.75246024 | 6m 28s |

There are two clear inferences when we check accuracy and training time with number of layers:

1. As the number of layers increases the training time increases. Which is natural and expected as increasing hidden layers and extra parameters to be computed.
2. Accuracy is decreasing as hidden layers increase which at first seems counterintuitive, since a general expectation is an increase of accuracy with the increase in the number of hidden layers. But this result actually brings out a very important flaw in increasing the number of hidden layers limitlessly - a large number of hidden layers causes overfitting of the model in test data, as the number of parameters increases, the correlation between the parameters also increases which impacts the accuracy of the model in test and validation set. There is a need to optimise the amount of hidden layers that must be used for a model, to improve accuracy and avoid overfitting.

Accuracy of 1 hidden layer Neural Network



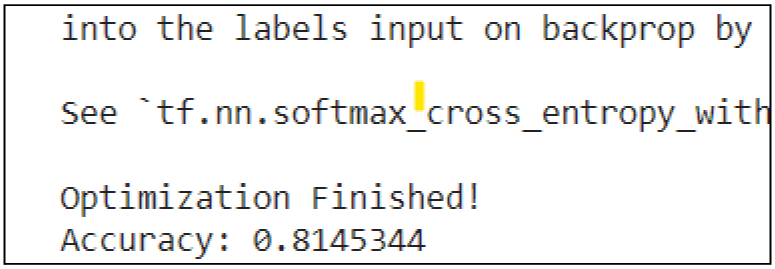
```
Training Time:  252.997897

 Training set Accuracy:85.36492890995261%

 Validation set Accuracy:83.93996247654785%

 Test set Accuracy:85.69265707797123%
```

Accuracy of 2 hidden layer Deep NN

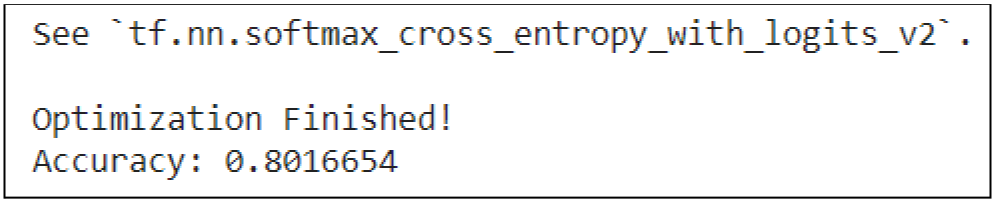


```
into the labels input on backprop by

See `tf.nn.softmax_cross_entropy_with

Optimization Finished!
Accuracy: 0.8145344
```

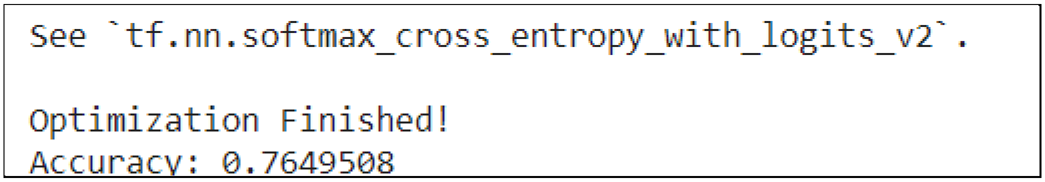
Accuracy of 3 hidden layer Deep NN



```
See `tf.nn.softmax_cross_entropy_with_logits_v2`.

Optimization Finished!
Accuracy: 0.8016654
```

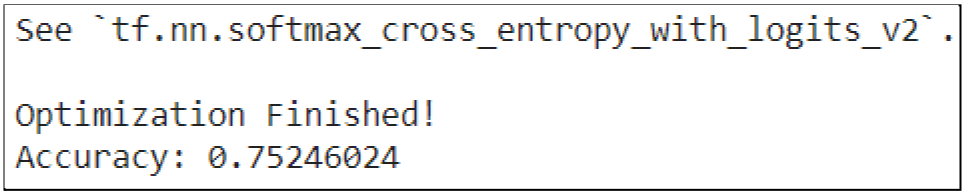
Accuracy of 5 hidden layer Deep NN



```
See `tf.nn.softmax_cross_entropy_with_logits_v2`.

Optimization Finished!
Accuracy: 0.7649508
```

Accuracy of 7 hidden layer Deep NN



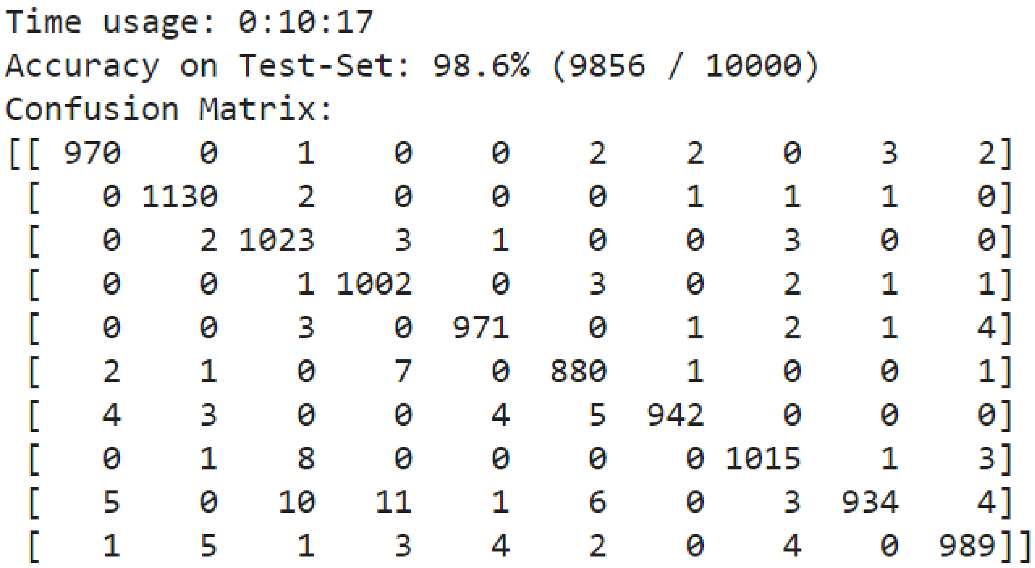
```
See `tf.nn.softmax_cross_entropy_with_logits_v2`.

Optimization Finished!
Accuracy: 0.75246024
```

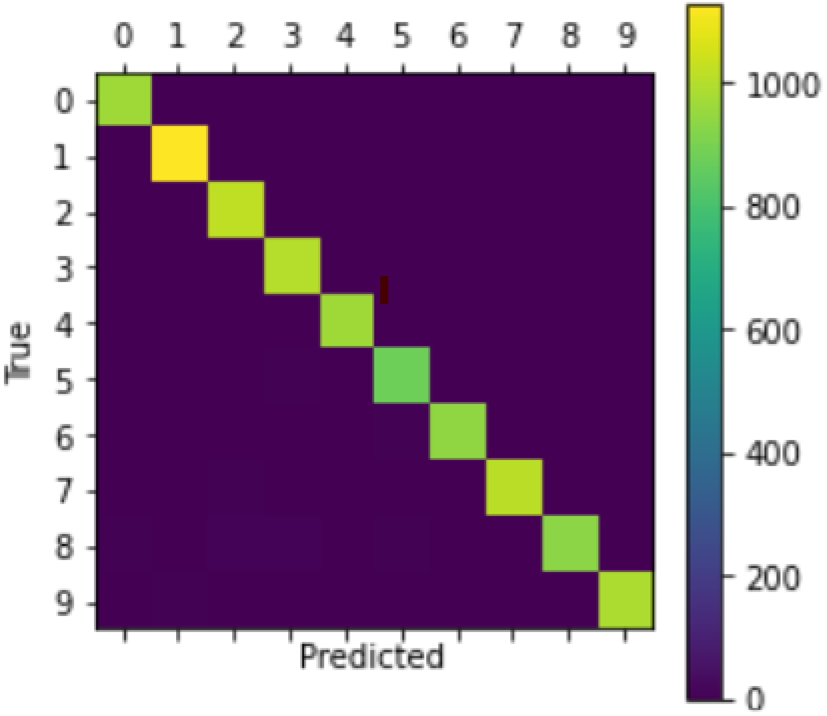
### 5. Results from Convolutional Neural Network in terms of Accuracy and Training time

We ran the convolution neural network on the given dataset and following are the inferences we get by interpreting the output:

- In comparison with the single layer Neural network and multi layer Deep neural network the CNN performs much better. In true essence it outperformed the other two with substantial margin and got an accuracy of 98.6% in testing data.
- Filter matrix is the major reason for this astounding accuracy. The various filters reduce the dimensionality of the images and extract various different types of relevant data(features) which enables CNN to be this effective in handling images.
- This might be the result of the ability of CNN to tune its hyperparameters based on the images independent of the prior knowledge in selecting the features.
- With each iteration the accuracy of CNN improves. And after 10,000 iteration the accuracy is 98.6% as we can see in the image below



```
Time usage: 0:10:17
Accuracy on Test-Set: 98.6% (9856 / 10000)
Confusion Matrix:
[[ 970    0    1    0    0    2    2    0    3    2]
 [    0 1130    2    0    0    0    1    1    1    0]
 [    0    2 1023    3    1    0    0    3    0    0]
 [    0    0    1 1002    0    3    0    2    1    1]
 [    0    0    3    0  971    0    1    2    1    4]
 [    2    1    0    7    0  880    1    0    0    1]
 [    4    3    0    0    4    5  942    0    0    0]
 [    0    1    8    0    0    0    0 1015    1    3]
 [    5    0   10   11    1    6    0    3  934    4]
 [    1    5    1    3    4    2    0    4    0  989]]
```



The time taken to train the CNN is 10 min and 17 seconds.

## 6. Conclusion

The following conclusions can be made after observing the results:

1. Feature selection is an important preprocessing step. With the help of this we can remove the unnecessary parameters from the data. This will reduce the training time and the complexity of the model.
2. If the number of hidden units are too less, then the neural network will not give a good result for data that contains a large amount of features.
3. Too many hidden units will also negatively affect the results as the neural network will do well only on the training data (due to overfitting) and not on the validation or the test data. This implies we should always find an optimum value for the number of hidden units to achieve the best results.
4. The regularization parameter (lambda) also needs to be tuned so that we make sure we are not overfitting on the training data. Large values of lambda can still ruin the accuracy as it will push the fit equation towards a line and not a curve. This will lead to bad results on both training and test data.
5. When we increase the number of hidden layers in the neural network, it becomes a deep neural network. A deep neural network does not guarantee better results as increasing the number of layers could also lead to overfitting on the data.

6. Convolutional Neural Networks have been very successful for image classification. This is because kernel convolutions create new features which provide the model with a lot more information.