**CSE 574: Introduction to Machine Learning**

**Programming Assignment 2**

**Group 79**

**Chihuangji Wang (50130295)**

**Wei Zhang (50240861)**

**Sonal Singh (50290784)**

**Overview**

The assignment aims to implement Multilayer Perceptron Neural Network Model and evaluate its performance in classifying handwritten digits and face recognition. The data for handwritten digits is obtained from the MNIST dataset and for face recognition, CelebFaces Attributes dataset is used. The assignment also compares the performance of Single Layer Neural Network, Deep Neural Network and Convolutional Neural Network using the TensorFlow Library.

**Evaluation**

1. ***Determine hyper-parameters and feature selection:***

We try λ from 0 to 60, with an increment of 5, and the number of hidden layers with 4,8,12,16, 20, 24, 28, and 32. The Figure 1 below shows the accuracy for training, validation, and test data sets with different options of hyper-parameters, organized by different hidden layer unit. The figure 2 below also shows the relationship between accuracy and different options of hyper-parameters, organized by different λ. Basically, we choose the optimal hyper parameters based on test accuracy, but also taking accuracies for training and validation into consideration.



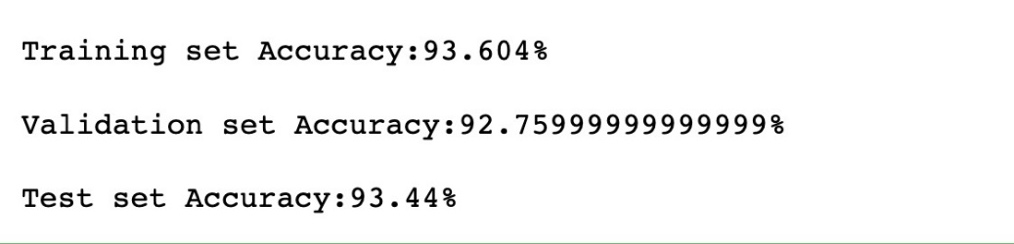
Figure 1: accuracy of NN with different hyper-parameters (x axis: # of λ; y axis: accuracy)



Figure 2: accuracy of NN with different hyper-parameters (x axis: hidden layer unites, y axis: accuracy)

Based on the Figure 2, we can see that all three accuracies are quite similar to each other with different hyper parameters. Also, it can be observed from the Figure 1 that the overall test accuracy is higher than other alternatives when hidden layer units is 28. When λ (lambda) increase from 0 to 5, then 10, under the condition that hidden layer units is 28, the test accuracy also increases and reaches the highest, about 93.15%, when λ (lambda) is 10. Additionally, accuracy for validation and training start to decline after λ (lambda) increase after 10. All of these indicate that which means that the optimal hyper-parameters are hidden layer units is 28 and λ (lambda) is 10. We also considered whether there is an issue of overfitting. Since training accuracy is not much higher than testing accuracy (93.5% and 93%), this also allows us to posit that our model does not overfit at all.

Then we re-trained Neural Network with the optimal hyper-parameters, the result is shown as below. The test accuracy is 93.44%.



For feature selection: We only selected features that can provide variation and help to our network. To be more specific, features with the same values across all training data set are removed. So, we finally got 719 features out of 784, deleting 65 of them. Below is the list of the index of features we chose and we store them into params.pickle.

[Selected features]

**[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, 141, 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, 645, 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]**

1. ***Training time for Neural Network***

Below is the table indicating training time Neural Network used with different units of hidden layers and λ. As we can easily tell, with the increase of units of hidden layers, the training time slightly increases. This is because more hidden units mean more computation toward gradient and weights which eventually will lead to longer training time. However, training time did not change too much with the changes of λ, just about 10 – 40 seconds.

Also, we can observe that when the unit of hidden layers reach to the optimal number, which is 28, the training time decreases compared with those with same Lambda but with different hidden layer unit. It just testifies that the optimal value of hidden layers unit is 28.



1. ***Comparison between single layer Neural Network and deep Neural Network***

**Results from single layer Neural Network on CelebA dataset**

We trained the single layer Neural Network with 256 hidden units and λ (lambda) being 10, on CelebA dataset. The results of accuracy are shown as below:

Training Accuracy: 85.44 %

Validation Accuracy: 84.09%

Test Accuracy: 85.31%

Time used: 501.49 seconds

**Results from deep Neural Network on CelebA dataset**

Then we trained deep neural network (using TensorFlow) and compare the result with single layer neural network in terms of accuracy and training time. The table below shows the accuracy on test data and time needed for training deep Neural Network with different hidden layer units. With the increase in the number of hidden layers, the training time increases but the accuracy deceases. We think this is due to the issue of overfitting.

|  |  |  |
| --- | --- | --- |
| Hidden layers | Accuracy | Training time (seconds) |
| 1 | 82.59% | 81.01 |
| 2 | 80.43% | 85.84 |
| 3 | 75.28% | 107.55 |
| 5 | 74.72% | 108.64 |
| 7 | 75.21% | 123.64 |

**Comparison of NN and deep NN**

If just comparing test accuracy from both single layer NN and deep NN, we can tell that NN performance is better. However, NN takes much longer training time than deep NN because multiple hidden layers need more time for computation of gradients and weights. Theoretically, deep NN should be intrinsically more powerful than simple NN (in our case, single layer NN). But we found that the highest test accuracy comes from single NN, which may be the small data size and potential overfitting problem when only single layer is considered in NN. (Both single layer NN and deep NN programs were run on Google Colab.)

1. ***Results from CNN***

Lastly, we trained convolutional neural network and get the results of test data accuracy when iterations are 1, 100, 1000 and 10,000. The results are shown in the table below. The results fit our expectations that more iterations need more time but will yield higher accuracy on test data set. Finally, the accuracy rate is 98.6%, which is high but only takes about 16 minutes.

|  |  |  |
| --- | --- | --- |
| Iteration | Accuracy | Time (seconds) |
| 1 | **7.5%** | **0** |
| 100 | **69.0%** | **11** |
| 1,000 | **92.9%** | **71** |
| 10,000 | **98.6%** | **953** |

The figure 3 below shows the accuracy on test data from the Convolutional Neural Network. Once the iteration reaches 6,000, the accuracy stays stable (between 95% to 100%).

Figure 3: accuracy of CNN with different iterations

**Confusion Matrices:**

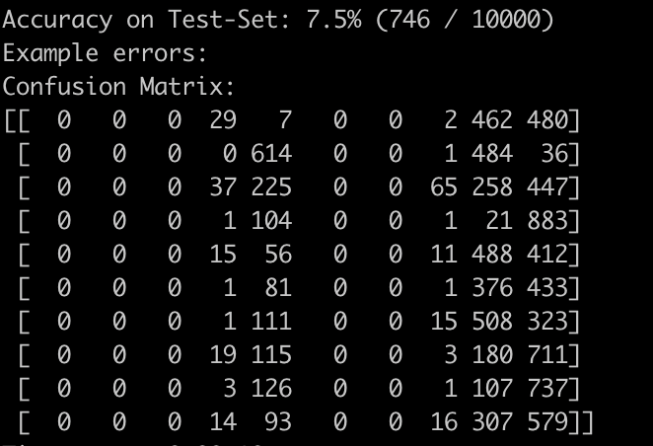
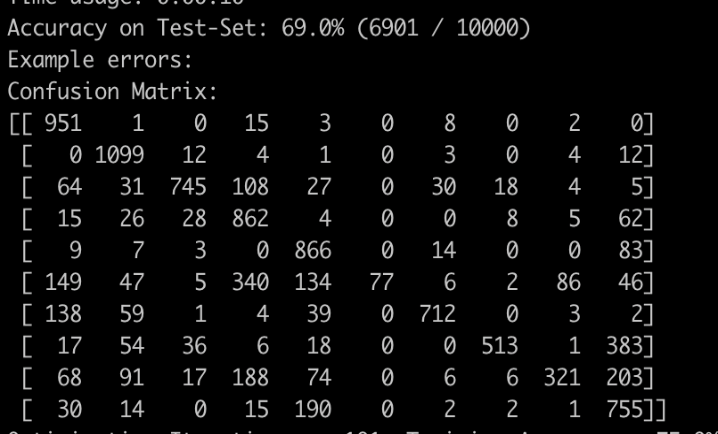
****

Figure 4: Confusion matrix for iteration: 1 Figure 5: Confusion matrix for iteration: 100

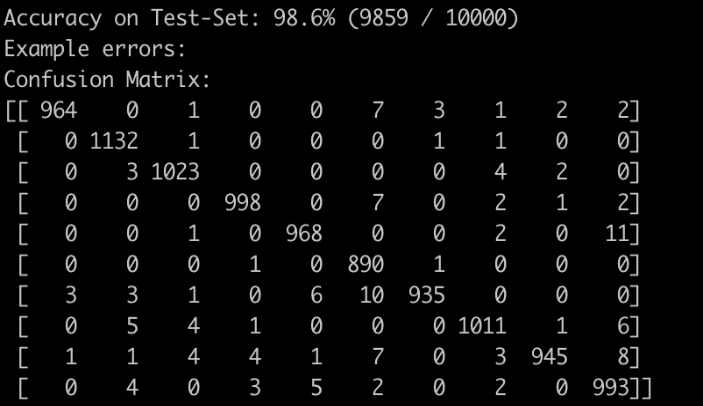
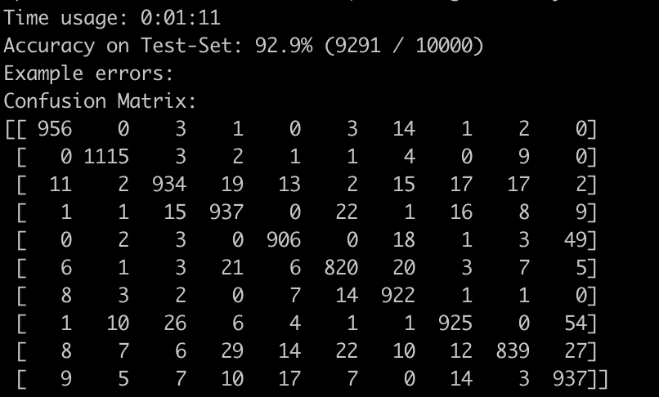


Figure 6: Confusion matrix for iteration: 1000 Figure 7: Confusion matrix for iteration: 10000

From the confusion matrices, we can notice that as the iterations increase, the accuracy increases. For iteration 1, the accuracy with which the model identifies the digits is very less. For iteration 10000, the data accuracy has increased considerably.

**Sample of Example Errors:**

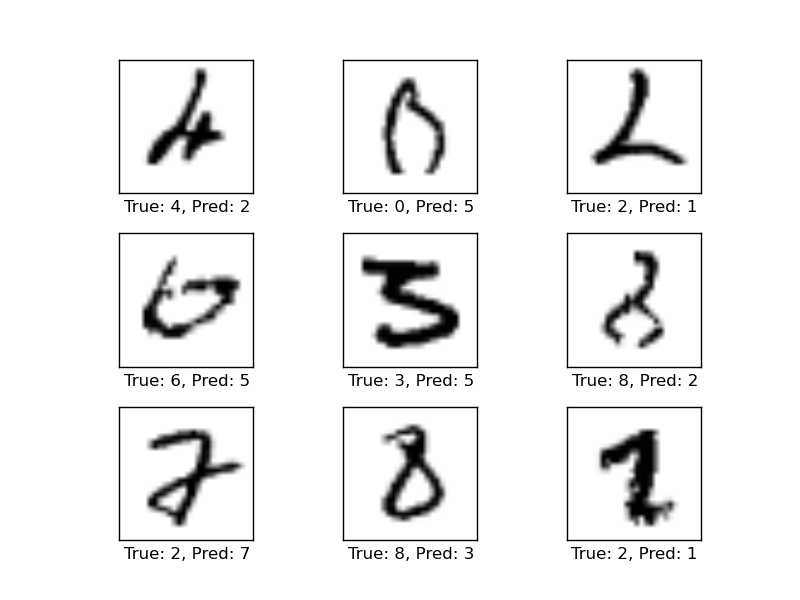
****

Figure: First 9 example errors for iteration: 10000