Assignment 3

Neural Networks and Deep Learning

CSCI 5922

Fall 2017

Student: **Bo Cao**

bo.cao-1@colorado.edu
boca7588@colorado.edu
Github: https://github.com/BryanBo-Cao/neuralnets-deeplearning

Part 1

Use tensorflow to build a feedforward neural net to predict occupancy. This net should do exactly the same thing that your code in Part 2 of Assignment 2 does.

(1a) Run a simulation using tensorflow that is identical to Assignment 2, part 2f, in which you vary the number of hidden units and make a plot. Superimpose the plot you made from Assignment 2, part 2f.

Me:
In assignment 2, part 2f, I showed both Train Accuracy and Test Accuracy. In this assignment only Test Accuracy is shown.

Assign2 part 2f's test accuracies are shown in blue while assign3 part 1a's test accuracies are displayed in red:

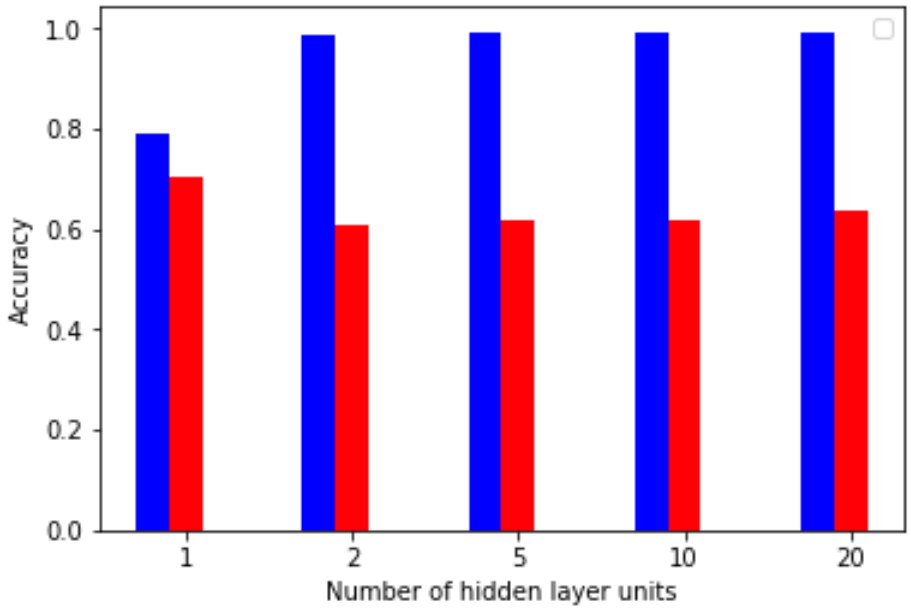

Figure 1.1

Code: neuralnets-deeplearning/assignment3-BoCao/src/assign3_part1a_*.ipynb

(1b) DIscuss the results: are they the same as with your own code? If one works better than the other, explain why you think that is.

Me:
The results are not identical to assignment 2 part 2f. My result from assignment 2 worked better. One reason behind this could be that TensorFlow is too high level that it hides many details behind it such as how the gradients descent was conducted based on the loss/cost function that I had barely any control of it. In assignment 2 I checked every detail of it to make sure every process worked.

(1c) Add a second hidden layer, and train a few architectures with 2 hidden layers. Report what architectures you tried (expressed as 5-h1-h2-1, i.e., 5 input, h1 hidden in first layer, h2 hidden in second layer, and one output unit), and which ones, if any, outperform your single-hidden-layer network.

It will not be a big deal to add the second hidden layer once you have all the rest of your code in place.

Me:
I appended a second layer with 5 hidden units to all of the experiment in (1a). All the neural network architectures are 5-h1-h5-1, 5-h2-h5-1, 5-h5-h5-1, 5-h10-h5-1 and 5-h20-h5-1 respectively. All settings are learning rate = 0.05, epochs = 1000 and batch size = 100.

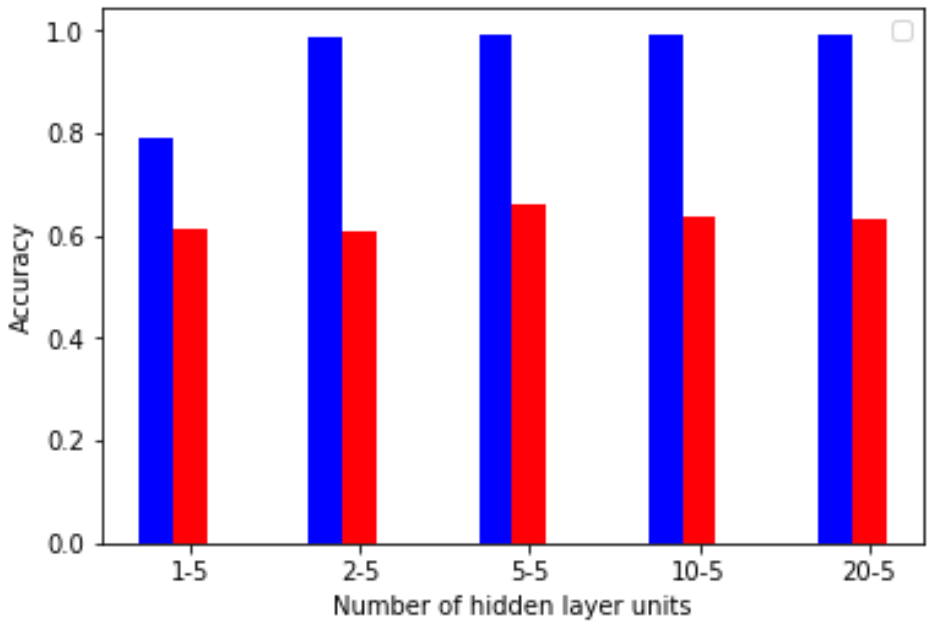

Figure 1.2

Except the first one with 1 unit and 5 units in the hidden layers respective, other architecture results are slightly better than the corresponding ones in (1a).

Code: neuralnets-deeplearning/assignment3-BoCao/src/assign3_part1c_*.ipynb

Part 2

Replicate the Hinton Family Trees architecture in tensorflow. My notes describe the architecture and number of neurons in each layer. You can also refer to the original paper.

The neural network architecture is identical to the figure 2.1. Softmax is used in the output layer. Argmax is used to pick up the highest probability after softmax to predict the classification.

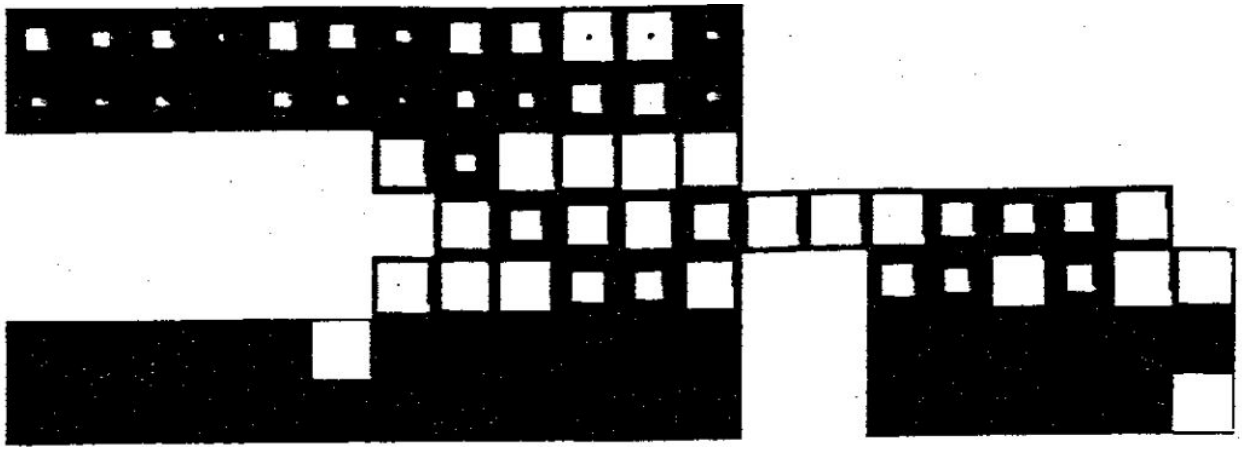

Figure 2.1

(2a) Randomly split the data set into 89 examples for training and 15 for testing. Train and evaluate 20 such random splits of the data and report the mean and standard deviation of the test set accuracy. (Report accuracy not squared error. A response should be counted as correct if the most active unit is the target person2.)

Me:
mean_test_accuracy: **0.126666671783**
std_test_accuracy: **0.078598843526**

test_accuracies: [0.26666668, 0.13333334, 0, 0.06666667, 0.06666667, 0.06666667, 0.13333334, 0.13333334, 0.06666667, 0, 0, 0.2, 0.13333334, 0.2, 0.2, 0.13333334, 0.13333334, 0.13333334, 0.26666668, 0.2]

Code: neuralnets-deeplearning/assignment3-BoCao/src/assign3_part2a.ipynb

(2b) Train a network on all 104 examples and examine the weights from the one-hot person1 input representation to the distributed person1 representation. Figure out a sensible way to graphically display the weights in the 6 hidden units.

Me:

As shown in Figure 2.2, names are shown in x axis while the 6 hidden units' weights are shown in the y axis. The lighter the grid is, the higher value the grid has and vice versa.

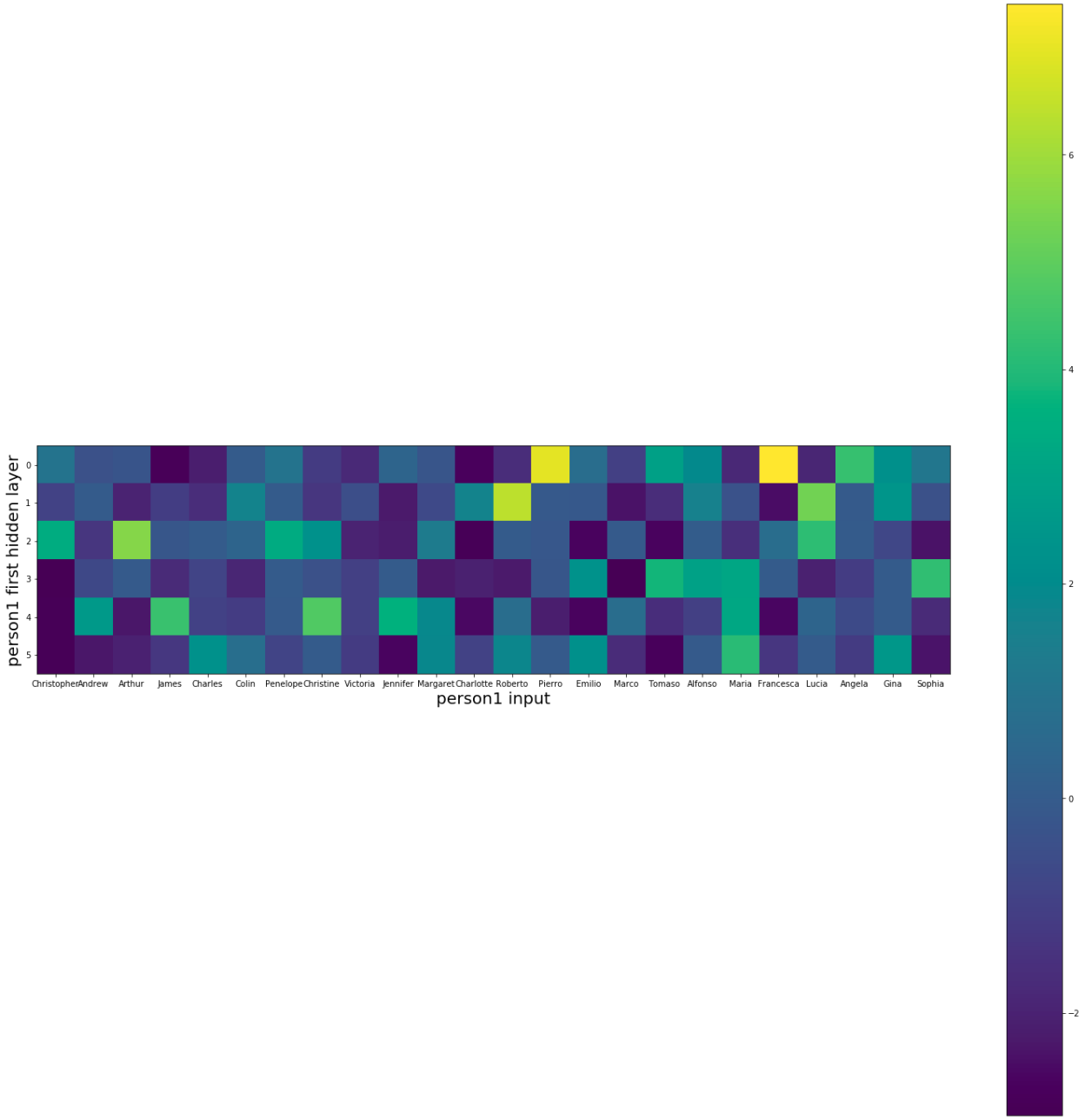

Figure 2.2

Weights from person1 input (n=24) to one hidden layer (n=6):

[[1.00565255e+00 -9.14377928e-01 3.42166686e+00 -2.93981409e+00
-2.86820269e+00 -2.85656118e+00]
[-3.83583784e-01 -2.43334216e-06 -1.37076330e+00 -7.18076706e-01
2.60043764e+00 -2.27148986e+00]

```
[ -2.47297525e-01  -1.95480537e+00   5.61012936e+00  -8.45330954e-03
  -2.30483866e+00  -2.02179766e+00]
[ -2.80550551e+00  -1.05661154e+00  -1.79436207e-01  -1.73387790e+00
   4.38742113e+00  -1.26363659e+00]
[ -2.14506841e+00  -1.65710449e+00  -8.63776077e-05  -8.24423075e-01
  -9.64827776e-01   2.26810312e+00]
[ -2.07595807e-03   1.66284049e+00   3.74939799e-01  -1.88601279e+00
  -1.10952759e+00   7.78896868e-01]
[  9.92429972e-01  -2.37308932e-03   3.36848760e+00  -5.56222047e-04
  -3.72214289e-03  -8.32081318e-01]
[ -1.14485836e+00  -1.36702180e+00   2.26484847e+00  -3.75575781e-01
   4.77532768e+00  -2.83255562e-04]
[ -1.76632476e+00  -4.81454372e-01  -1.98543334e+00  -1.00859785e+00
  -1.01043892e+00  -1.17278767e+00]
[  3.98837745e-01  -2.18824410e+00  -2.16573071e+00  -8.48582946e-04
   3.60707402e+00  -2.61807775e+00]
[ -2.23963737e-01  -6.51018620e-01   1.34809721e+00  -2.25291967e+00
   1.86495185e+00   1.83388758e+00]
[ -2.70092225e+00   1.65699494e+00  -2.86438537e+00  -2.03759694e+00
  -2.53299952e+00  -9.10900354e-01]
[ -1.65698433e+00   6.40054369e+00  -4.76936111e-05  -2.22757673e+00
   7.44099319e-01   1.81180882e+00]
[  6.98388004e+00  -6.43200874e-02  -1.43039703e-01  -2.02835083e-01
  -2.13245130e+00  -3.56695289e-03]
[  7.46097088e-01  -9.43214893e-02  -2.65713215e+00   2.33183026e+00
  -2.64619374e+00   2.22460461e+00]
[ -1.00801277e+00  -2.36243534e+00  -3.17194462e-02  -2.95673919e+00
   7.49965310e-01  -1.67505383e+00]
[  2.93477869e+00  -1.64701438e+00  -2.65925217e+00   3.83113766e+00
  -1.59132600e+00  -2.73138356e+00]
[  1.94979835e+00   1.59789276e+00   5.57591096e-02   2.96054578e+00
  -9.89535570e-01  -7.52156135e-03]
[ -1.78472185e+00  -3.35673809e-01  -1.55013299e+00   3.18553996e+00
   3.20061684e+00   4.10025454e+00]
[  7.40292263e+00  -2.47056127e+00   7.55102456e-01  -5.09094098e-05
  -2.57872868e+00  -1.33230186e+00]
[ -1.86952901e+00   5.31885338e+00   4.19532490e+00  -2.03420877e+00
   3.92848969e-01  -3.64285021e-04]
[  4.36633587e+00  -7.93157052e-03  -1.84008060e-03  -1.09639764e+00
  -6.25632048e-01  -1.17309380e+00]
[  2.25783920e+00   2.43171406e+00  -7.59607315e-01  -1.57441173e-06
  -8.62372573e-04   2.50774479e+00]
[  1.11533725e+00  -3.87471914e-01  -2.37697148e+00   4.24395704e+00
  -1.72958350e+00  -2.32072878e+00]]
```

Code: neuralnets-deeplearning/assignment3-BoCao/src/assign3_part2b.ipynb

(2c) For at least 2 of the hidden units, interpret what the network has learned in its mapping from inputs. You'll have to refer to my create_dataset.py code to determine the interpretation of the person1 input neurons. (I tried to keep the same ordering as Hinton uses.)

Me:

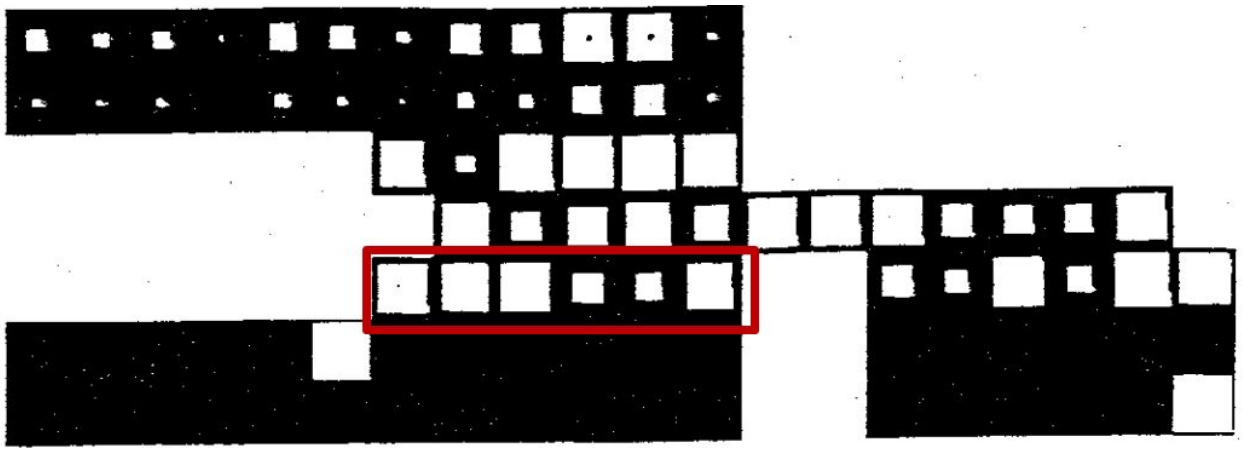

Figure 2.3

In Figure 2.3, this hidden layer learns features of a person among all the 24 persons. It can also learn the difference between Italian and English names.

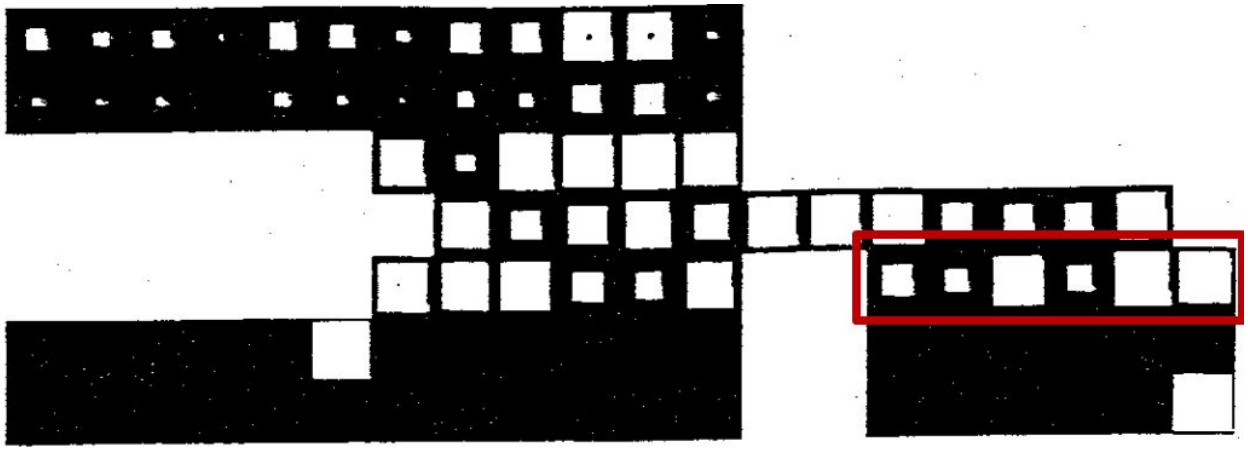

Figure 2.4

In Figure 2.4, this hidden layer learns the features of a relation among all the 12 relations. It can also learn the distinction between generations.

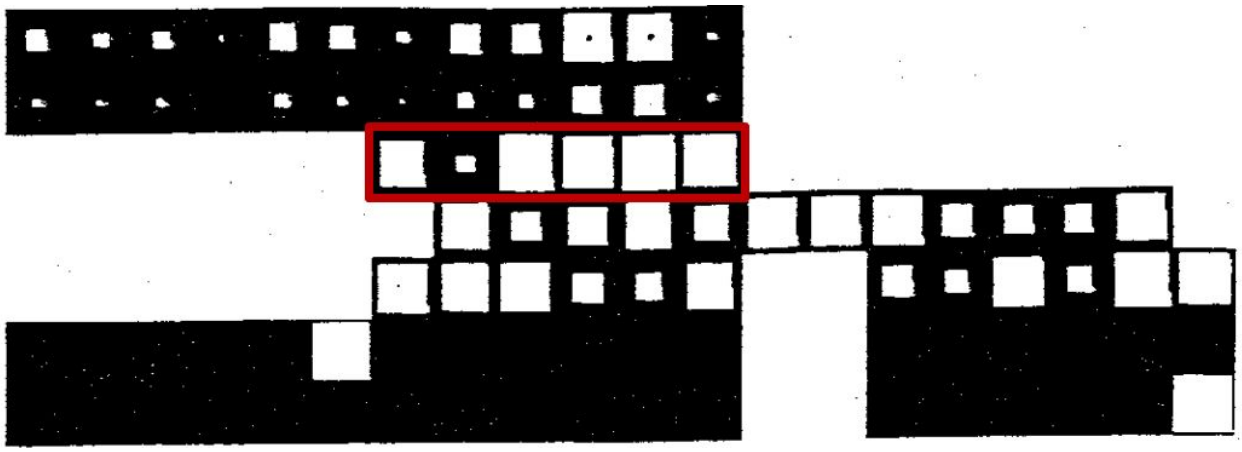

Figure 2.5

In Figure 2.5, this hidden layer learns the knowledge that, given a person's features and a relation's features, how do these features map into a specific person.

Part 3 (Extra Credit)

Compare the architecture Hinton describes for Family trees with a generic feedforward architecture with one hidden layer consisting of 12 neurons.

Me:
The neural network architecture is:
input(n=36) (fully connected to) hidden layer(n=12) (fully connected to) output layer(n=24)

(3a) Conduct an experiment like the one in (2a) using the generic architecture. Report the mean and standard deviation of the test set accuracy.

Me:
mean_test_accuracy: 0.0866666711867
std_test_accuracy: 0.0763034915946
Setting is learning rate = 0.05, epochs = 100000.
test_accuracies: [0.06666667, 0.13333334, 0.13333334, 0, 0.13333334, 0.06666667, 0.06666667, 0.06666667, 0.26666668, 0.06666667, 0, 0.06666667, 0.06666667, 0.26666668, 0, 0.13333334, 0, 0.06666667, 0, 0.13333334]

Code: neuralnets-deeplearning/assignment3-BoCao/src/assign3_part3.ipynb

(3b) Do you see any difference in performance between the structured net (2a) and the generic net (3a)?

Me:
From my code the result of net (2a) with (**mean_test_accuracy: 0.126666671783 std_test_accuracy: 0.078598843526**) is better than net (3a).

(3c) It wasn't difficult to interpret at least some of the hidden units in the structured net. Can you interpret what any of the hidden units are doing for the generic net? Explain.

Me:
The hidden units (num = 12) are trying to learn the features from a person and a relation together, and given these they try to guess, or determine how confidence it should be to identify another person. It might also learn the difference between Italian and English names and distinction between generations simultaneously.

Code:
All code is in
https://github.com/BryanBo-Cao/neuralnets-deeplearning/tree/master/assignment3-BoCao/src
as well.