Understanding Neural Network

October 30, 2018

1 Understanding the Strategy of a Simple Neural Network

1.1 Abstract

In this Jupyter notebook an attempt is made to uncover the strategy of a simple neural network that is trained for classifying rotating squares and triangles (of the same size). To that end, a symbolic representative of the neural network is being studied.

1.2 About the Study

The neural network that is being studied is from `\models\nn_Linear_1024_2_Rect_Linear_2_2_SoftMax_(batc`
For more details about this particular neural network see the corresponding log file. In `understanding_nn.py` we generated and tested a sympy representative for the neural network that is much easier to study.

There are two possible shapes: squares and triangles. In the following, we assume that the shape color is given by $x \in [0,1]$ and the background is given by color $y \in [0,1]$. So if you would give the shape and the colors x and y we can approximately reproduce the picture except for the correct rotation. In `understanding_nn.py` we averaged over the rotation possibilities to obtain a symbolic representation.

In the following script the symbolic representations are loaded for the squares and triangles, resp., with colors x and y. Afterwards, the representations are printed. Small comment: I did not apply the softmax layer for simplicity. Note that due to the monotonicity of the softmax layer, a relatively large value from the last layer means a relatively large probability (and vice versa).

```
In [30]: import os
         import pickle
         from IPython.display import display
         from sympy import init_printing
         init_printing(use_latex=True)

         # load neural network representation for square and triangle, resp.
         root = os.path.dirname(os.getcwd())

         # square (Sq)
         fileNameNNReprSq = root + r'\results\sympyRepresentationOutputLayer3SqNN.pickle'
         with open(fileNameNNReprSq, 'rb') as handle:
             nnReprSq = pickle.load(handle)
```

```
# triangle (Tr)
fileNameNNReprTr = root + r'\results\sympyRepresentationOutputLayer3TrNN.pickle'
with open(fileNameNNReprTr, 'rb') as handle:
    nnReprTr = pickle.load(handle)
```

1.3 Symbolic Representation in case of a Square

```
In [31]: print 'Output for square (positively correlated with probability):'
         display(nnReprSq[0])
         print 'Output for triangle (positively correlated with probability):'
         display(nnReprSq[1])
```

```
Output for square (positively correlated with probability):
```

$$3.0\max\left(0, -9.0x + 9.0y + 0.04\right) + 4.0\max\left(0, 9.0x - 9.0y + 0.04\right) - 1.0$$

```
Output for triangle (positively correlated with probability):
```

$$-3.0\max\left(0, -9.0x + 9.0y + 0.04\right) - 2.0\max\left(0, 9.0x - 9.0y + 0.04\right) + 1.0$$

1.4 Symbolic Representation in case of a Triangle

```
In [32]: print 'Output for square (positively correlated with probability):'
         display(nnReprTr[0])
         print 'Output for triangle (positively correlated with probability):'
         display(nnReprTr[1])
```

```
Output for square (positively correlated with probability):
```

$$3.0\max\left(0, -0.06x - 0.0007y + 0.04\right) + 4.0\max\left(0, -0.04x - 0.3y + 0.04\right) - 1.0$$

```
Output for triangle (positively correlated with probability):
```

$$-3.0\max\left(0, -0.06x - 0.0007y + 0.04\right) - 2.0\max\left(0, -0.04x - 0.3y + 0.04\right) + 1.0$$