CECS 551
Assignment 3
Total: 50 Points

General Instruction

- Use `Python 3`, any other programming language is not acceptable.
- You can import modules in the Python Standard Library (please check the full list *here*). If you want to use any other library, please consult with the instructor.
- Submit uncompressed file(s) in the Dropbox folder via BeachBoard (Not email).

---

1. Implement multi-layer neural network **WITHOUT** using external deep learning libraries such as `Keras, Caffe, Theano, TensorFlow, PyTorch...`
   (a) Consider a neural network as shown in Figure 1 that approximates XOR function.
      - The width of the layer 1 is 2, and the width of the layer 2 is 1.
      - The activation functions of the layer 1 are the hyperbolic tangent.
      - The activation function of the layer 2 is the sigmoid.
      - The loss function is the binary cross entropy.

   (b) (25 points) Derive $\frac{\partial L}{\partial \vec{W}^{(1)}}$, $\frac{\partial L}{\partial \vec{w}^{(2)}}$, $\frac{\partial L}{\partial \vec{b}^{(1)}}$, and $\frac{\partial L}{\partial b^{(2)}}$. Please include the answers in the Jupyter notebook. Notice that you can use $\LaTeX$ equation in the Jupyter notebook.

   (c) (25 points) Implement the model **without** using any deep learning libraries. However, you can use import `numpy` in case you need.

```
X = np.array([[0,0],[0,1],[1,0],[1,1]])
y = np.array([0,1,1,0])
```

   You need to optimize the parameters $\vec{W}^{(1)}, \vec{w}^{(2)}, \vec{b}^{(1)},$ and $b^{(2)}$ using simple gradient descent method. For example, $b^{(2)} \leftarrow b^{(2)} - \eta \frac{\partial L}{\partial b^{(2)}}$ where η is a small positive number. Predict $\hat{y}$ using the trained network and show the result.

   (d) Submit `Assignment_3_scratch.ipynb`.

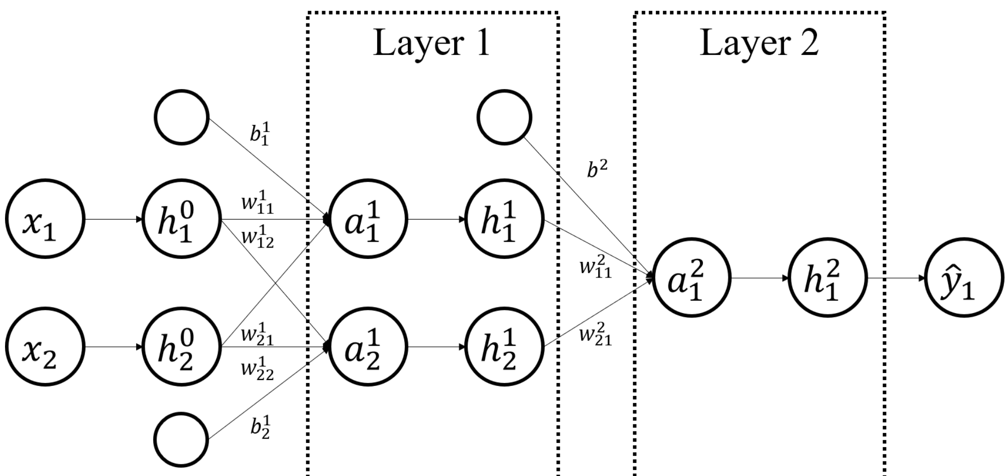

Figure 1: network design