ANLY-590 Midterm

Artificial Neural Networks and Deep Learning

Section 2

October 2018

1 Classification Network

Consider an ANN with two input features, two hidden nodes (with biases) and a single output node (with a bias). Use ReLU activations in the hidden layer. This will be a binary classification network, so the output layer activation function will be a sigmoid. For this network, we'll use a binary cross-entropy Loss function.

1.1

Draw a diagram of the network described above. Give labels to each node and weight in the network.

1.2

Write an expression for the output of this network, as function of X_1 and X_2 and weights.

1.3

Consider the weight parameter that connects X_1 to the first hidden node h_1, let's call this parameter w_{11}. Derive an expression for the partial derivative of the loss with respect to this parameter,$\frac{\partial L}{\partial w_{11}}$.

1.4

Suppose you have the following set of weight matrices:

$$W = \begin{bmatrix}1 & 2\\ 3 & 2\end{bmatrix} \qquad \vec{b} = [2, 3]^T \tag{1}$$

$$\vec{V} = \begin{bmatrix}1\\ 2\end{bmatrix} \qquad c = [2] \tag{2}$$

$$\tag{3}$$

For this input:

$$\vec{X}^T = \begin{bmatrix}2 & -1\end{bmatrix}$$

what is the output probability $\hat{y}$ predicted for this input?

1.5

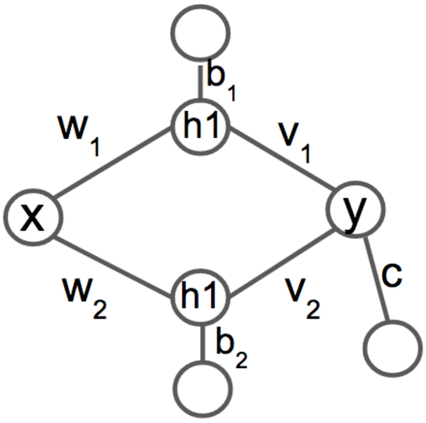
Next, we're going to perform a single step of backprop and update the weight w_{11}. (For simplicity, assume that all the weights besides w_{11} are fixed and that we only need to do gradient descent for w_{11}). For the input X above and for a ground-truth $y = 1$, compute the gradient on w_{11}. For a learning rate of 0.01, what would be the updated value of w_{11}?

1.6

Now suppose we've added L_2-norm regularization to all of the weights. Write an expression for the new Loss function and an expression for $\frac{\partial L}{\partial w_{11}}$.

2 Activation Functions and Capacity

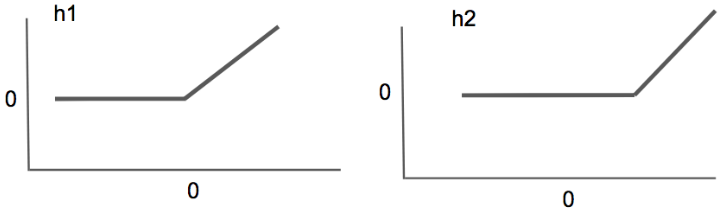
Consider the single-input network shown below for non-linear regression (continuous x and y).



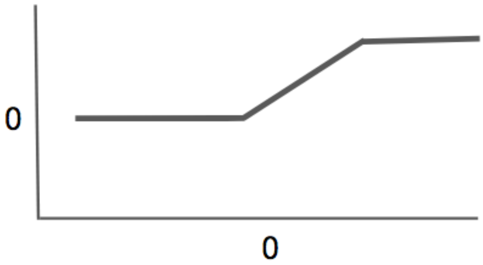
For any given activation function, the kinds of relationships that can be represented at the hidden nodes is fairly flexible, due to the parameters b_k and w_k. Concretely, the w_k parameter tends to impact the "scale" of the relationship, whereas the b_k bias parameter might result in systematic translations.

The outputs of the hidden nodes, h_1 and h_2 are similarly weighted and combined (with a bias) for form the final output of the network, y. The final output of the network y will then flexibly depend on h, as weighted by v_k and c. Thus, this network can potentially represent many kinds of relationships between x and y.

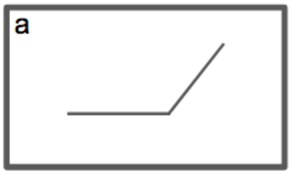
As a concrete example, suppose we have ReLu activations and the outputs of h_1 and h_2 look like the following.

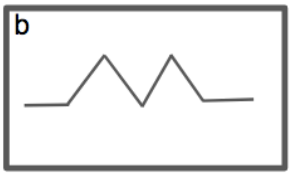


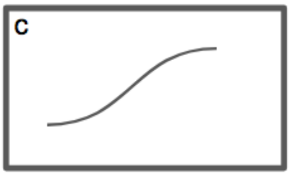
We know that $y = v_1 h_1 + v_2 h_2 + c$, so one example might be if $v_1 = 1$ and $v_2 = -1$, then the final output of the network might look like

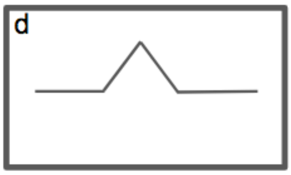


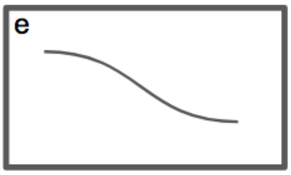
For this question, determine which kinds of output relationships are possible to achieve with the network above and the given activation functions. Consider the bank of possible output functions below.

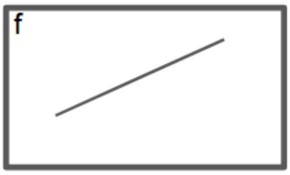


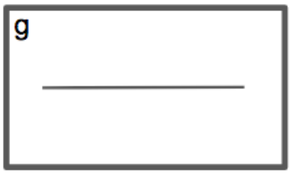


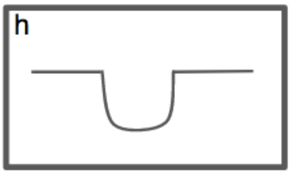


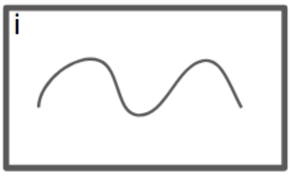












2.1

If the hidden nodes have a ReLu activation, which of the relationships (a through i) can be represented by the network (there may be multiple true answers). Make sure to draw a "proof" for each answer.

2.2

If the hidden nodes have a linear activation, which of the relationships (a through i) can be represented by the network (there may be multiple true answers). Make sure to draw a "proof" for each answer.

3 Open answer questions

Answer the questions below in the open space. Be sure to include your reasoning.

1. What does it mean when dropout is included in a network architecture? Explain at a high-level how and why dropout works.

2. Why is it important to consider the *bias-variance trade-off* when training neural networks? What are some possible approaches to mitigate over- and under-fitting?

3. Why are GPUs used to train neural networks rather than CPUs?

4. For a given data set, why is fitting a neural network non-deterministic?

4 Multi-task Learning

Prompt: Multi-task learning is where we try and predict multiple outputs for a single input. Suppose we work at Zillow. For each of the houses we are tracking, we want to predict not only whether the house will sell, but also what the final sale price will be. We can build models to solve these tasks.

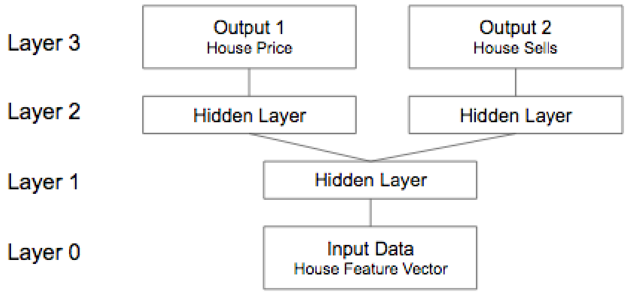
Data: For this question suppose you have the following data for each house:

- Input data: A feature vector of housing attributes such as number of rooms, square footage, zip code,... Call this data set X
- Output Data 1: House price. Call this $Y^{(1)}$.
- Output Data 1: House sells. Call this $Y^{(2)}$.

Re-writing the data looks like $\{(X, Y^{(1)}, Y^{(2)})\}$.

Questions

1. Given the architecture below, suggest the loss functions $L^{(1)}$ and $L^{(2)}$ to be used for outputs $Y^{(1)}$ and $Y^{(2)}$



2. Suppose we were to fit a network on the regression problem $\{(X, Y^{(1)})\}$ and then separately on the classification problem $\{(X, Y^{(2)})\}$. How might the performance of the pair of models compare to a network fit jointly on $\{(X, Y^{(1)}, Y^{(2)})\}$?

3. Why might it be advantageous to train a multi-task network to solve both problems simultaneously.