


# Deep Learning CS583 Fall 2021

## Midterm Exam

Instructor: Jia Xu

Oct 21st, 2021

**Honor Pledge:** I pledge on my honor that I have not given or received any unauthorized assistance on this assignment/examination. Communicating with others or Internet use except for Zoom is strictly prohibited. Signature:

Student name: Archana Kalburgi 

Student ID: 10469491

Student email address: akalburg@stevens.edu

- Read these instructions carefully
- Fill-in your personal info, as indicated above and sign on the honor pledge.
- You have 2 hours and an half hour extension, in total two and half hours.
- There are ten questions. Each question worths the same (1 point). There is also one optional question (1 point).
- Both computer-typed and hand-writing in the very clear form are accepted.
- This is an open-book test.
- You should work on the exam only by yourself.
- Submit your PDF/Doc/Pages **by 12:00 PM Oct 21st** in person or on Canvas under Midterm exam Section B.

good luck!

## 1 Question

- Applying back-propagation to train a neural network is guaranteed to find the global optimal.

A. TRUE. ☒ B. FALSE. *B. False*

- Regardless of the choice of the activation function, it makes the network function as a linear mapping from inputs to outputs to set all weights close to zero in a neural network.

A. TRUE. ☒ B. FALSE. *B. False*

## 2 Question

- A multi-layer feedforward network with linear activation functions is more expressive than a single-layer feedforward network with linear activation functions.

A. TRUE.

☒ B. FALSE.

*B: False*

C. I do not know. (0.1 Point)

- Suppose that you are training a neural network for classification, but you notice that the training loss is much lower than the validation (test set) loss. Which of the following can be used to address the issue (select all that apply)?

A. Use a network with fewer layers.

☒ B. Decrease dropout probability (Dropout turns off neurons with a probability)

C. Increase the regularization weight

☒ D. Increase the size of each hidden layer

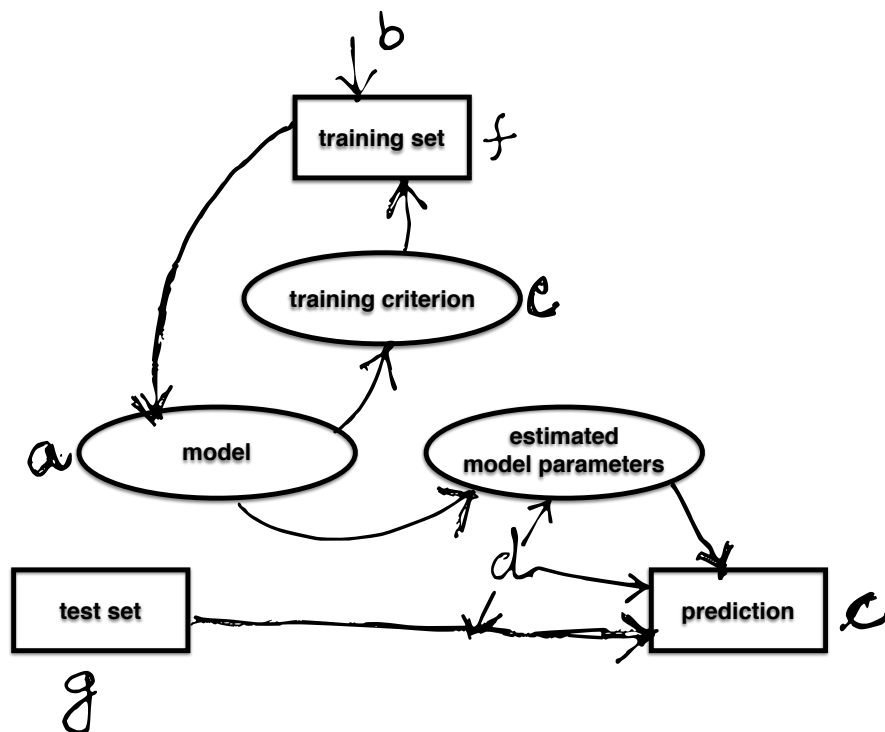
*B, D*

### 3 Question

A classification or a regression system such as those in machine translation, or in other tasks such as question answering, image recognition, speech recognition, follows a workflow. This workflow is composed of several components, as depicted in the graph below.

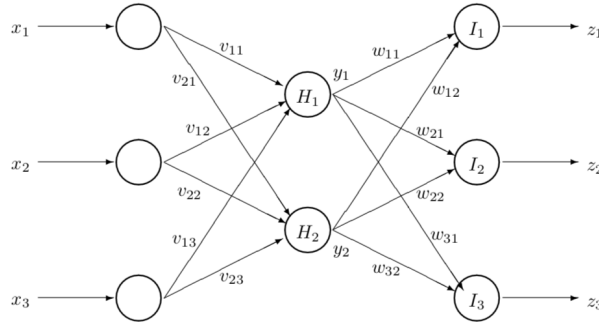
Your tasks in this question:

1. Connect the boxes with directed edges and indicate the relationship between the components connected with your edges.
2. Write the following terms as instances of their belonging component box, considering no noise in the training set but errors in the prediction output (this can be many-to-many alignment if you think necessary):
  - (a) neural networks
  - (b) smoothing using linear interpolation
  - (c)  $P(e) = 0.5$
  - (d) maximum likelihood
  - (e) mean squared error
  - (f) "the cat sat on the matt"
  - (g) "matt the on"



## 4 Question

- A training pattern, consisting of an input vector  $x = [x_1, x_2, x_3]^T$  and desired outputs  $t = [t_1, t_2, t_3]^T$ , is presented to the following neural network. What is the usual sequence of events for training the network using the back-propagation algorithm?



- ✓ A. (1) calculate  $z_k = f(I_k)$ , (2) update  $W_{kj}$ , (3) calculate  $y_j = f(H_j)$ , (4) update  $v_{ji}$ .
- B. (1) calculate  $y_j = f(H_j)$ , (2) update  $v_{ji}$ , (3) calculate  $z_k = f(I_k)$ , (4) update  $w_{kj}$ .
- C. (1) calculate  $y_j = f(H_j)$ , (2) calculate  $z_k = f(I_k)$ , (3) update  $v_{ji}$ , (4) update  $w_{kj}$ .
- D. (1) calculate  $y_j = f(H_j)$ , (2) calculate  $z_k = f(I_k)$ , (3) update  $w_{kj}$ , (4) update  $v_{ji}$ .

A

## 5 Question

- $\hat{c} = \operatorname{argmax}_c \Pr(c|x)$ , where  $c$  is the class, and  $x$  is the observation. How can we formally derive  $\hat{c} = \operatorname{argmax}_c \Pr(x|c) \cdot \Pr(c)$ ?  
(you should at least say which theorem/decision rule you are using)

We can derive  $\hat{c}$  using Bayes rule-  
(Naive Bayes Classifier)

$$\begin{aligned}\hat{c} &= \operatorname{argmax} \Pr(c|x) \\ &= \operatorname{argmax} \frac{\Pr(x|c) \cdot \Pr(c)}{\Pr(c)}\end{aligned}$$

$$= \operatorname{argmax} \Pr(x|c) \Pr(c)$$

$$\begin{aligned}\Pr(x|c) &= \Pr(x_1|c) \cdot \Pr(x_2|x_1, c) \Pr(x_3|x_1, x_2, c) \\ &\quad \dots \dots \Pr(x_n|x_1, x_2 \dots x_{n-1}, c)\end{aligned}$$

• assuming  $x$  has features from  $x_1 \dots x_n$

• This is a conditional independent prob, assume the features are independent of class  $c$

$$\therefore \Pr(x|c) = \prod_{i=1}^N \Pr(x_i|x_1, x_2 \dots x_{n-1}, c)$$

## 6 Question

- We have seen that averaging the outputs from multiple models typically gives better results than using just one model. Why is it helpful? Briefly explain.

1. Collection of models reduces the variance, making the model robust to outliers. That way our model will not be very sensitive to noisy data.

Individual model form weak learners.  
Combining the output of all the weak learners, we can get better results.

Disadvantage associated with this, is, all the model's output are given equal weight.  
We can overcome this problem using weighted output giving weight to some model that perform well.

## 7 Question

In Bayesian learning, we consider not just one, but many different weight vectors. Each of those is assigned a probability by which it is weighted in producing the final output.

- Write down Bayes' rule as it applies to supervised neural network learning. Clearly define the symbols that you are using.

$$P(w|x) = \frac{P(x|w) \cdot P(w)}{P(x)}$$

$w$ : weights  
 $x$ : observation

- Clearly indicate which part of the formula is the "prior distribution", which is the "likelihood term", and which is the "posterior distribution".

prior distribution:  $P(w)$ . prior distrib<sup>n</sup> of weight  
likelihood term:  $P(x|w)$   
Posterior distribution:  $P(w|x)$

- In this context, how is Maximum A Posteriori (MAP) learning different from Maximum Likelihood (ML) learning?

• MAP is a special case of MLE.

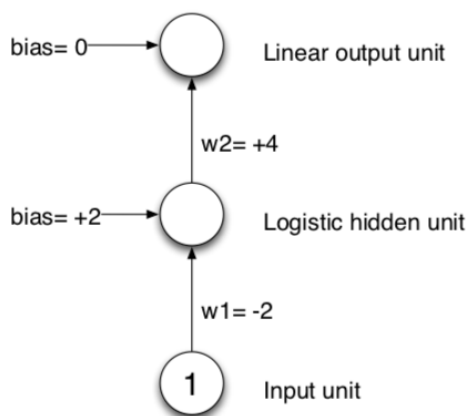
MAP gives a value that maximises the posterior  
& MLE gives the value that maximises likelihood

• We cannot use MAP estimation here because we do not know the weight distribution ahead of time. In this case we can use MLE to produce weight vector



## 8 Question

Here you see a very small neural network: it has one input unit, one hidden unit (logistic), and one output unit (linear). Let's consider one training case. For that training case, the input value is 1 (as shown in the diagram), and the target output value is 1. We're using the standard squared error loss function:  $E = (t - y)^2/2$ . The numbers in this question have been constructed in such a way that you don't need a calculator.



- What is the output of the hidden unit and the output unit, for this training case?

Hidden unit o/p:  $w_1x + b_1 = (-2)(1) + 2 = 0$       output unit  $= w_2x + b_2$   
 $= 4(0) + 0$   
 $= 0$

- What is the loss, for this training case?

$$E = (1 - 0)^2/2 = 0.5$$

- What is the derivative of the loss w.r.t.  $w_2$ , for this training case?

derivative:  $\frac{\partial E}{\partial w_2} = y - t, y - t = 0 - 1 = -1$

- What is the derivative of the loss w.r.t.  $w_1$ , for this training case?

given  $b_1 = 2$   
 $b_2 = 0$   
 $w_1 = -2$   
 $w_2 = 4$   
 i/p:  $x = 1$

$$\frac{\partial E}{\partial w_2} = \frac{\partial E}{\partial y} \cdot \frac{\partial y}{\partial w_2} = (y - t)x = y - t$$

$$\frac{\partial E}{\partial w_1} = \frac{\partial E}{\partial y} \cdot \frac{\partial y}{\partial w_1} = (y - t)x = y - t$$

$$= 0 - 1$$

$$= -1$$

$$y = w_1x + b_1 + w_2x + b_2$$

$$E = (t - y)^2/2$$

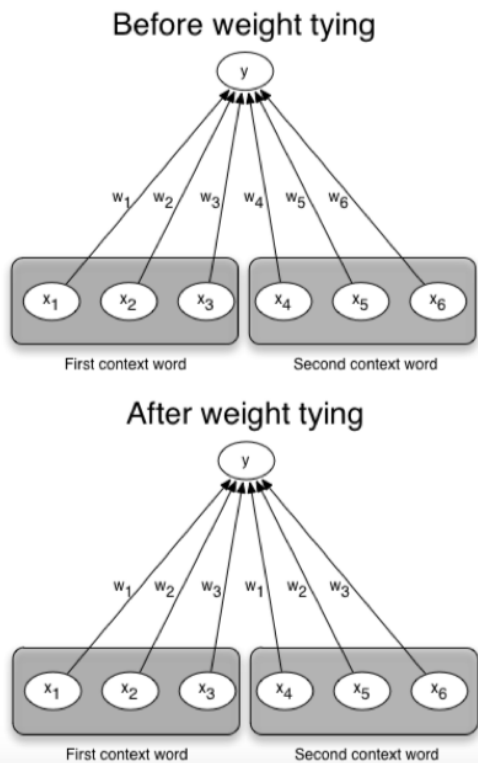
$$\frac{\partial E}{\partial y} = \frac{1}{2} \times 2 \times (t - y) \cdot (-1) = y - t$$

## 9 Question

Suppose that we have a vocabulary of 3 words, "a", "b", and "c", and we want to predict the next word in a sentence given the previous two words. For this network, we don't want to use feature vectors for words: we simply use the local encoding, i.e. a 3-component vector with one entry being 1 and the other two entries being 0.

In the language models that we have seen so far, each of the context words has its own dedicated section of the network, so we would encode this problem with two 3-dimensional inputs. That makes for a total of 6 dimensions. For example, if the two preceding words (the "context" words) are "c" and "b", then the input would be  $(0, 0, 1, 0, 1, 0)$ . Clearly, the more context words we want to include, the more input units our network must have. More inputs means more parameters, and thus increases the risk of overfitting. Here is a proposal to reduce the number of parameters in the model:

Consider a single neuron that is connected to this input, and call the weights that connect the input to this neuron  $w_1, w_2, w_3, w_4, w_5$ , and  $w_6$ .  $w_1$  connects the neuron to the first input unit,  $w_2$  connects it to the second input unit, etc. Notice how for every neuron, we need as many weights as there are input dimensions (6 in our case), which will be the number of words times the length of the context. A way to reduce the number of parameters is to tie certain weights together, so that they share a parameter. One possibility is to tie the weights coming from input units that correspond to the same word but at different context positions. In our example that would mean that  $w_1 = w_4$ ,  $w_2 = w_5$ , and  $w_3 = w_6$  (see the "after" diagram). **Explain the main weakness that that change creates.**



One of the main problem it creates is, It will extract irrelevant information from such a encoding. Because some weights are same they become irrelevant. This can cause overfitting.

## 10 Question

- For a fully-connected deep network with one hidden layer, increasing the number of hidden units should have what effect on bias and variance? Explain briefly.

There will be decrease in bias and increase in the variance. high variance = low training error, higher test error  
higher bias = higher error in training & testing

- What is the risk with tuning hyperparameters using a test dataset?

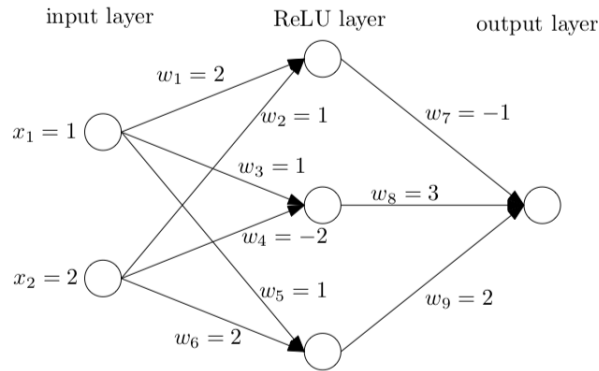
In this case we are giving our model to see the test data. This will cause our model to develop a bias towards the test set.

This will not help because we won't know how good our model is performing on the unseen data.

Activation fun : Sigmoid

## 11 Question (Optional, 1 point)

Assume the artificial neural network below, with mean square error loss and gold output 0. Compute the values of all weights  $w_i$  after performing a SGD update with learning rate 0.1.



---

---

---

---