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: Girish

Student name: Girish Amar Budhrani

Student ID: 10477624

Student email address: gbudhran@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!

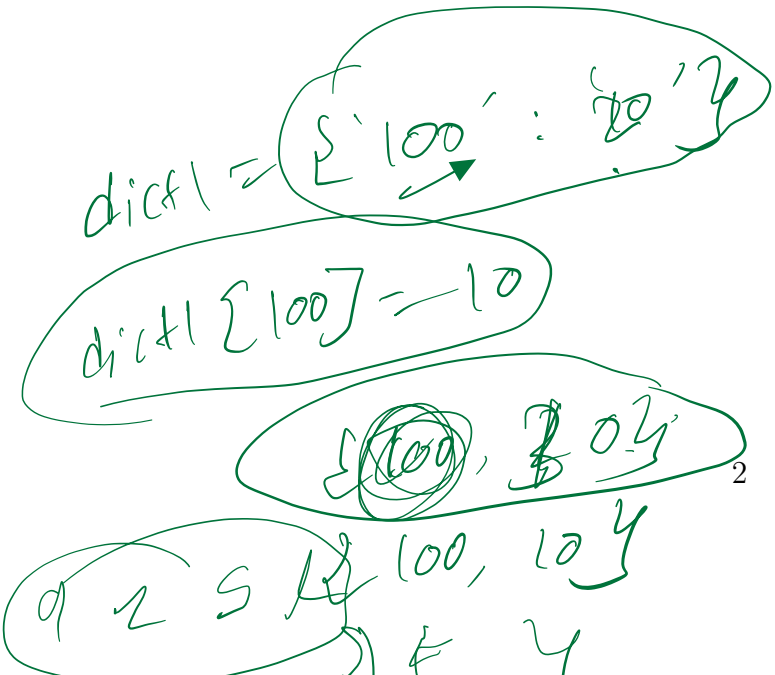
1 Question

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

  A. TRUE. B. FALSE.

- 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.



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.

  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)?

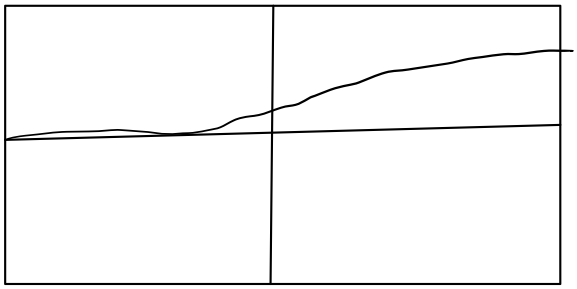
  A. Use a network with fewer layers

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

  C. Increase the regularization weight

  D. Increase the size of each hidden layer

(1) sigmoid activation function.



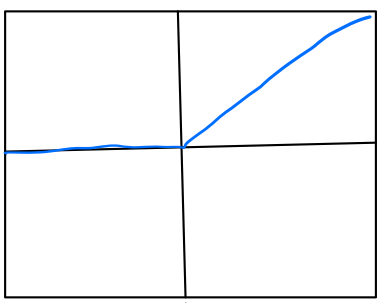
formula for sigmod function is

$$y = \frac{1}{1 + e^{-z}}$$

(2) Rectified linear Unit (ReLU)

formula for ReLU is

$$y = \max(0, z)$$

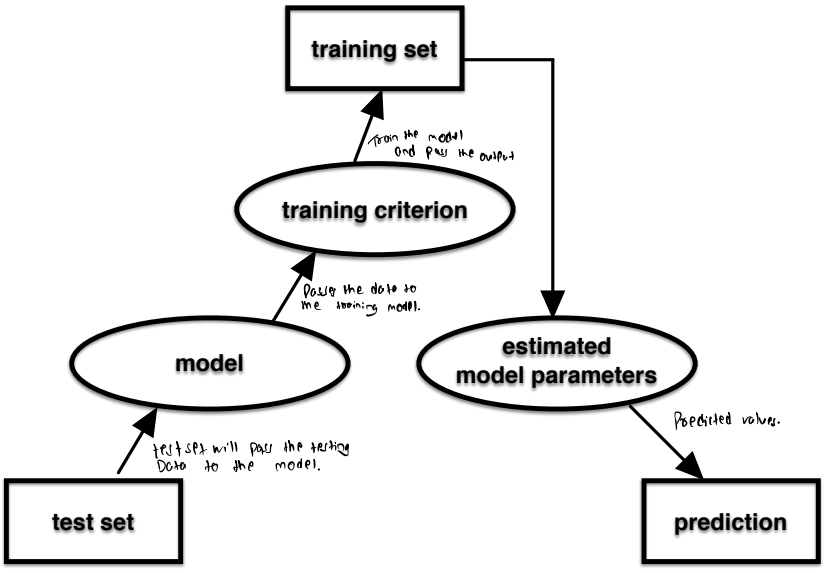


3 Question

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.

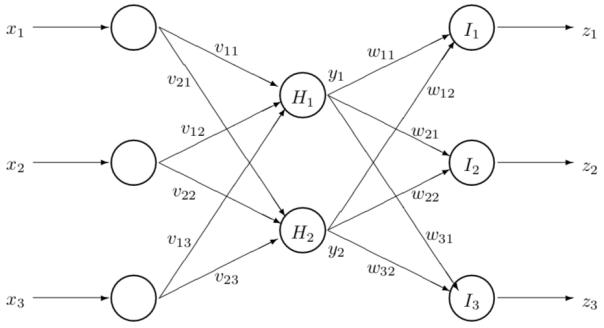
Your tasks in this question:

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 $x = [x1, x2, x3]^T$ and desired outputs $t = [t1, t2, t3]^T$, 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_{jv}.

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) calcualte $y_j = f(H_j)$, (2) calculate $z_k = f(I_k)$, (3) update w_{kj}, (4) update v_{ji}.

5 Question

- $\hat{c} = argmax_c Pr(c|x)$, where c is the class, and x is the observation. How can we formally derive $\hat{c} = argmax_c Pr(x|c) \cdot Pr(c)$?

  (you should at least say which theorem/decision rule you are using)

$\hat{C} = \text{argmax}_c P_\gamma(C|x)$ (Discriminative model.) → (1)

By Baye's Rule

$$P_\gamma(C|x) = \frac{P_\gamma(C, x)}{P_\gamma(x)}$$

$$= \frac{P_\gamma(C, x)}{\sum_x P_\gamma(C, x)}$$

$$= \frac{P_\gamma(C) \cdot P_\gamma(x|C)}{\sum_x P(C) P(x|C)}$$

$\therefore P_\gamma(C|x) = P_\gamma(C) \cdot P_\gamma(x|C)$ (as denominator is constant we can ignore it.)

As $P_\gamma(C|x) = P_\gamma(C) \cdot P_\gamma(x|C)$ we can replace it in equation (1)

$\therefore \hat{C} = \text{argmax}_C P_\gamma(x|C) \cdot P_\gamma(C)$

$\hat{C} = \text{argmax}_c P_\gamma(c|x)$ is an discriminative decision rule and $\hat{C} = \text{argmax}_c P_\gamma(x|C) \cdot P_\gamma(C)$ is an generative decision rule.
Both have the same usage as generative and discriminative decision rule are used for training the model and for predicting the output.
But the main difference is Discriminative compares the two or more features of the model and predicts the output and on the other hand generative model learns each and every model in a deep and then predect the output.

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.

Averaging the model is always the **best option** as it returns the average of n number of model and due which it returns the best probability. When we run the single model it is not sure that the model will return the same percent of probability again and again, as it is dependent on the data and in this situation if we run the model with n number of times then we can take the best models from it, For example if we run an model 20 times and take the top 10 models from it (who's stability is high and the performance of that single model is also high) in short we have to select the top lo **high performance diverse mode**l and calculate the average of all the lo models, in this way our model's accuracy will be far more better then the single model

Take an example-

①. Runs a model single time and the accuracy of that model is 85 %.

② runs a model n number of times and selected the top 5 from it: -

Model 1 accuracy: 85%, model 2 accuracy: 95%, model 3 accuracy is 90%, model 4 accuracy is 89% and the model 5 accuracy is 79%.

Now it we take the average of the 5 models then our average model accuracy will be 87.6%.

Hence taking the average of the model is always a better option.

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.

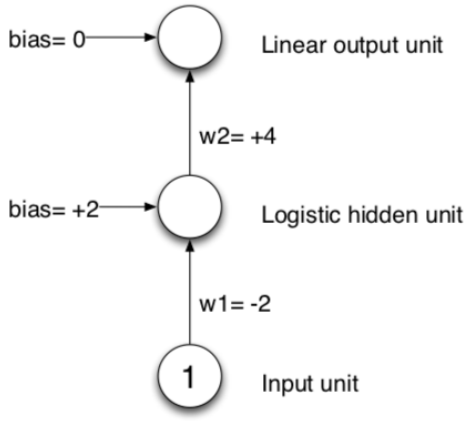
- Write down Bayes' rule as it applies to supervised neural network learning. Clearly define the symbols that you are using.

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

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

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 = $1 \times (-2) + 2 = 0$

  output unit = $0 \times 4 + 0 = 0$

- What is the loss, for this training case?

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

  ∴ the loss is 0.5//.

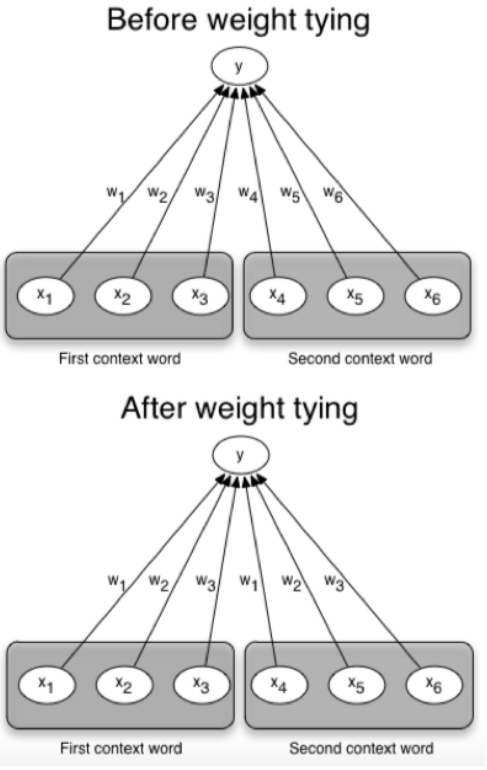
- What is the derivative of the loss w.r.t. w2, for this training case?

  $\frac{dE}{dw2} = \frac{d\,0.5}{dw2} = 0.5//.$

- What is the derivative of the loss w.r.t. w1, for this training case?

  $\frac{dE}{dw1} = \frac{d\,0.5}{dw1} = 0.5//.$

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 w1, w2, w3, w4, w5, and w6. w1 connects the neuron to the first input unit, w2 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 w1=w4, w2=w5, and w3=w6 (see the "after" diagram). **Explain the main weakness that that change creates.**

In this we successfully removed the risk of overfiting the model but as the weight for w1 = w4, w2 = w5 and w3 = w6, the same weight is passed to the second context and the first context word, This change create a main weakness that it is possible that our system will return the same values as the weigh is same.

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.

=> for a fully connected deep network when we introduce a new hidden unit then we have to introduce a new bias for that new hidden layer and variance.

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

=> The risk in tuning the hyperparameter in a test data set is sometime we get high performance diverse model but sometime we get less performance diverse model which is bad for our model.

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.