DL Refresher Quiz

Meta Learning, BITS Goa 2020 II

February 28, 2021

Q. What could happen if we initialised all the weights of a deep network to 0.

(a) The gradients w.r.t. to the parameters will be same

(b) The gradients could possibly vanish

(c) All the neurons will learn the same feature

(d) The network will not learn anything useful

(e) There will not be any problem. It will work as usual.

Ans: a, b, c, d
Reason: Symmetry breaks i.e. all of what is said in a, b, c, d

Q. Assume you have a large dataset. To train a neural net on your data, you have used stochastic gradient descent (SGD) with mini-batches. You have observed the following trend for batch-size versus number of epochs taken to reach a given loss on a validation set (say: 0.05). Assume that the learning rate is same for all the batch-size experiments.

Batch-size	# of epochs
2^0	2^{14}
2^2	2^{12}
2^4	2^8
2^6	2^7
2^8	2^6
2^{10}	2^5
2^{12}	2^5
2^{14}	2^5

Your friend provided the following reasons behind faster convergence for larger batch sizes:

1. Larger batch sizes reduce the variance in the gradient estimation of SGD.

2. Larger batch sizes allow exploring more of the cost surface.

What can you say about the above 2 reasons?

(a) Only 1 is correct

(b) Only 2 is correct

(c) Both 1 and 2 are correct

(d) None of the reasons are correct

Ans: a
Reason: Larger batch sizes don't explore. They exploit instead. Smaller batch sizes tend to be less noisy and have a bit of variance in their gradient estimation.

Q. Assume you have a large dataset. To train a neural net on your data, you have used stochastic gradient descent (SGD) with mini-batches. You have observed the following trend for batch-size versus number of epochs taken to reach a given loss on a validation set (say: 0.05). Assume that the learning rate is same for all the batch-size experiments.

Batch-size	# of epochs
2^0	2^{14}
2^2	2^{12}
2^4	2^8
2^6	2^7
2^8	2^6
2^{10}	2^5
2^{12}	2^5
2^{14}	2^5

Your friend provided the following reasons behind the trend in the last 3 rows of the above table.

1. For the last 3 rows of batch sizes, SGD got trapped in the local minimum.

2. As the batch size grows larger, SGD effectively becomes full batch gradient descent.

What can you say about the above 2 reasons?

(a) Only 1 is correct

(b) Only 2 is correct

(c) Both 1 and 2 are correct

(d) None of the reasons are correct

Ans: b
Reason: Reason 1 is not correct as we don't care whether SGD got trapped in local minimum or not. The targetted validation loss is reached. That's it.

Q. The following table shows four different operations on a neural net model. Each operation influence the bias and the variance of the model. Some of the effects are pre-filled. Fill the rest denoted by I, J, K, L, M.

Operation	Bias	Variance
Regularising the model parameters	I	Decreases
Increasing the size of the hidden layers	J	K
Training the network with dropout	Increases	L
Getting more training data	No change	M

(Choose the option that should replace I, J, K, L, M)

(a) I: Decreases, J: Decreases, K: Increases, L: No effect, M: Decreases

(b) I: Increases, J: Decreases, K: Increases, L: Decreases, M: Decreases

(c) I: Increases, J: Decreases, K: Increases, L: Decreases, M: No effect

(d) I: Increases, J: Increases, K: Decreases, L: Decreases, M: No effect

Ans: b
Reason: straight-forward.

Q. Recall that the softmax function takes in a vector $(z_1, \ldots, z_d)$ and returns a probability distribution as a vector $(y_1, \ldots, y_d)$. This can be written as:

$$s = \sum_i e^{z_i}; \qquad y_j = \frac{e^{z_j}}{s}$$

Which of the following is the correct computational graph for $d = 2$?

(a)
$$\begin{tikzcd}
z_1 \arrow[dr] & & y_1 \\
& s \arrow[ur] \arrow[dr] & \\
z_2 \arrow[ur] & & y_2
\end{tikzcd}$$

(b)
$$\begin{tikzcd}
z_1 \arrow[rr] & & y_1 \\
z_2 \arrow[rr] & & y_2
\end{tikzcd}$$

(c)
$$\begin{tikzcd}
z_1 \arrow[dr] \arrow[rr] & & y_1 \\
& s \arrow[ur] \arrow[dr] & \\
z_2 \arrow[ur] \arrow[rr] & & y_2
\end{tikzcd}$$

(d)
$$\begin{tikzcd}
z_1 \arrow[r] & s \arrow[r] & y_1 \\
z_2 \arrow[r] & s \arrow[r] & y_2
\end{tikzcd}$$

Ans: c

Reason: The dependency of y_i on s due to denominator. Also, y_i is directly dependent on z_i due to numerator.

Q. Consider a neural network with softmax activation function at the output layer. There are k output units. Softmax function converts real-valued outputs to probabilities as follows:

$$\hat{y}_i = \frac{\exp(v_i)}{\sum_{j=1}^{k} \exp(v_j)};\ \forall\, i \in \{1, 2, \ldots, k\}$$

where, v_i is a real-value output. Let L be the cross-entropy loss between true output $\boldsymbol{y}$ and the computed class probability vector $\hat{\boldsymbol{y}}$ and it is calculated as

$$L(\boldsymbol{y}, \hat{\boldsymbol{y}}) = -\sum_{i=1}^{k} y_i \log(\hat{y}_i)$$

where $\boldsymbol{y}$ is a k-length 1-hot encoded vector representation of a class label. Based on these information, which of the following is/are TRUE?

(a) $\frac{\partial \hat{y}_i}{\partial v_j} = \begin{cases} \hat{y}_i(1 - \hat{y}_i) & \text{when } i = j \\ -\hat{y}_i \hat{y}_j & \text{when } i \neq j \end{cases}$

(b) $\frac{\partial \hat{y}_i}{\partial v_j} = \begin{cases} \hat{y}_i(1 - \hat{y}_i) & \text{when } i \neq j \\ -\hat{y}_i \hat{y}_j & \text{when } i = j \end{cases}$

(c) $\frac{\partial L}{\partial v_i} = \hat{y}_i - y_i$

(d) $\frac{\partial L}{\partial v_i} = y_i - \hat{y}_i$

Ans: a, c
Reason: straight-forward calculus

Q. Which of the following is or are TRUE?

(a) Underfitting means that a model is not able to reduce the training error. When training error is much lower than validation error, there is overfitting.

(b) One particular choice for keeping the model simple is weight decay using an L_2 penalty. This leads to weight decay in the update steps of the learning algorithm.

(c) Dropout is also used during testing.

(d) Forward propagation sequentially calculates and stores intermediate variables within the computational graph defined by the neural network. It proceeds from the input to the output layer. Backpropagation sequentially calculates and stores the gradients of intermediate variables and parameters within the neural network in the reversed order.

(e) ReLU activation functions mitigate the vanishing gradient problem. This can accelerate convergence.

(f) True risk is the expectation of the loss over the entire population of data drawn from their true distribution. However, this entire population is usually unavailable. Empirical risk is an average loss over the training data to approximate the true risk. In practice, we perform empirical risk minimization.

Ans: a, b, d, e, f
Reason: We don't use dropout during testing. We want to use the full power of model during testing.

Q. Which of the following statements is or are TRUE?

(a) Translation invariance in images implies that all patches of an image will be treated in the same manner.

(b) Locality means that only a small neighborhood of pixels will be used to compute the corresponding hidden representations.

(c) Multiple filter channels can be used to extend the model parameters of the convolutional layer.

(d) Convolutional layers typically require same number of parameters as fully-connected layers, but they offer faster computation.

(e) Padding and stride can be used to adjust the dimensionality of the data effectively.

Ans: a, b, c, e
Reason: The weight-sharing allows the same paramters to be shared across the image; which is not the case for fully connected layer.

Q. Which of the following is or are TRUE regarding Batch Normalisation (normalising activations for a batch)?

(a) During model training, batch normalization continuously adjusts the intermediate activations of the neural network by utilizing the mean and standard deviation of the minibatch.

(b) We can use batch normalisation for any batch sizes such as 1, 8, 16, 32, 64 and so on.

(c) Batch normalisation stabilises the activations in each layer throughout the neural network.

(d) Computation for batch normalisation during training and testing is same.

Ans: a,c

Reason: b is not correct: minibatch of 1 would make the mean to be the activation itself. Batch norm will result in 0 activation. d is not correct: Batch norm is computed differently during test. Because there is no concept of a batch during testing.

Q. Why is it at all required to choose different learning rates for different parameters of a neural network?

(a) To avoid problem of diminishing learning rate

(b) To avoid overshooting the optimum point

(c) To reduce vertical oscillations while navigating the optimum point

(d) This would aid to reach the optimum point faster

Ans: d
Reason: a and b are not correct at all. c says about vertical oscillation: while this might be correct in 2-dimensional case, does not generalise.

Q. Let the input be a 5×5 matrix. There are 8 filters of size 3×3. Assuming stride of 1, convolution of the matrix with these filters results in an output of dimension [fill in the blank]?

(a) $3 \times 3 \times 1$

(b) $3 \times 3 \times 8$

(c) $5 \times 5 \times 1$

(d) $5 \times 5 \times 8$

Ans: b
Reason: straight forward answer

Q. Let the input a $5 \times 5 \times 3$ tensor. Let a filter be a $3 \times 3 \times 4$ tensor. What is the output dimension for a convolution of the the input with the filter (assume stride $= 1$)?

(a) $3 \times 3 \times 1$

(b) $3 \times 3 \times 4$

(c) $5 \times 5 \times 1$

(d) Convolution not possible.

Ans: d
Reason: Number of channels in input and filter do not match

Q. Suppose we want to detect vertical edges in images and we decide to use 3×3 filters in a CNN. We train our model on a dataset which consists of positive and negative examples of vertical edges. Which one of these could represent a vertical edge detection filter?

(a) $\begin{bmatrix} -10 & 0 & 10 \\ -10 & 0 & 10 \\ -10 & 0 & 10 \end{bmatrix}$

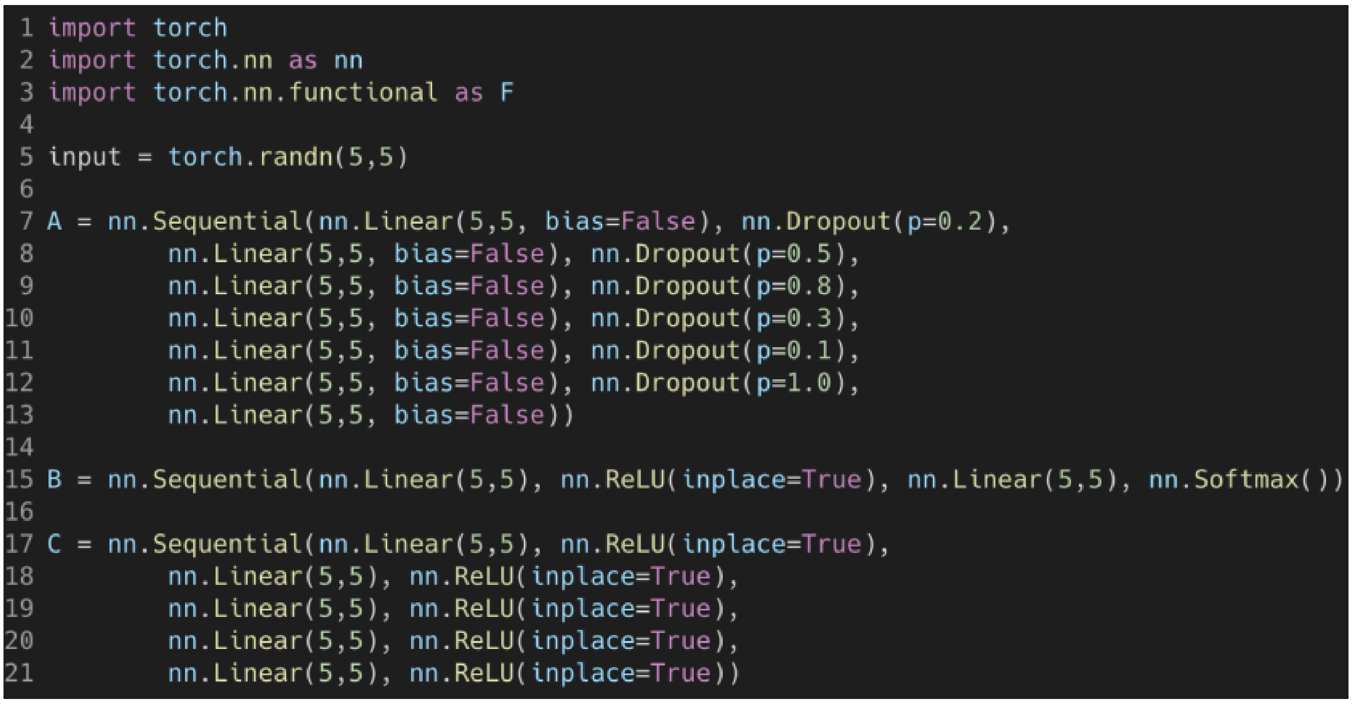
(b) $\begin{bmatrix} -10 & -10 & -10 \\ 0 & 0 & 0 \\ 10 & 10 & 10 \end{bmatrix}$

(c) Not enough information

(d) Both matrix in (a) and matrix (b)

Ans: a
Reason: matrix multiplication will detect this.

Q. Given three network definition as follows:



```python
import torch
import torch.nn as nn
import torch.nn.functional as F

input = torch.randn(5,5)

A = nn.Sequential(nn.Linear(5,5, bias=False), nn.Dropout(p=0.2),
        nn.Linear(5,5, bias=False), nn.Dropout(p=0.5),
        nn.Linear(5,5, bias=False), nn.Dropout(p=0.8),
        nn.Linear(5,5, bias=False), nn.Dropout(p=0.3),
        nn.Linear(5,5, bias=False), nn.Dropout(p=0.1),
        nn.Linear(5,5, bias=False), nn.Dropout(p=1.0),
        nn.Linear(5,5, bias=False))

B = nn.Sequential(nn.Linear(5,5), nn.ReLU(inplace=True), nn.Linear(5,5), nn.Softmax())

C = nn.Sequential(nn.Linear(5,5), nn.ReLU(inplace=True),
        nn.Linear(5,5), nn.ReLU(inplace=True),
        nn.Linear(5,5), nn.ReLU(inplace=True),
        nn.Linear(5,5), nn.ReLU(inplace=True),
        nn.Linear(5,5), nn.ReLU(inplace=True))
```

Weights in network A are initialized using Gaussian distribution and there are no biases. Weights and biases in network B are all initialized to the same value (say 0.0). Weights in Network C are initialized from a uniform distribution of $[-0.00001, 0.00001]$ and biases are set to $= -10000$.

Note: Dropout is a technique that randomly drops connections (with probability p) from the neural network during training.

Suppose `input` is passed to all the networks, in which two networks will the output be the same?

(a) A and B

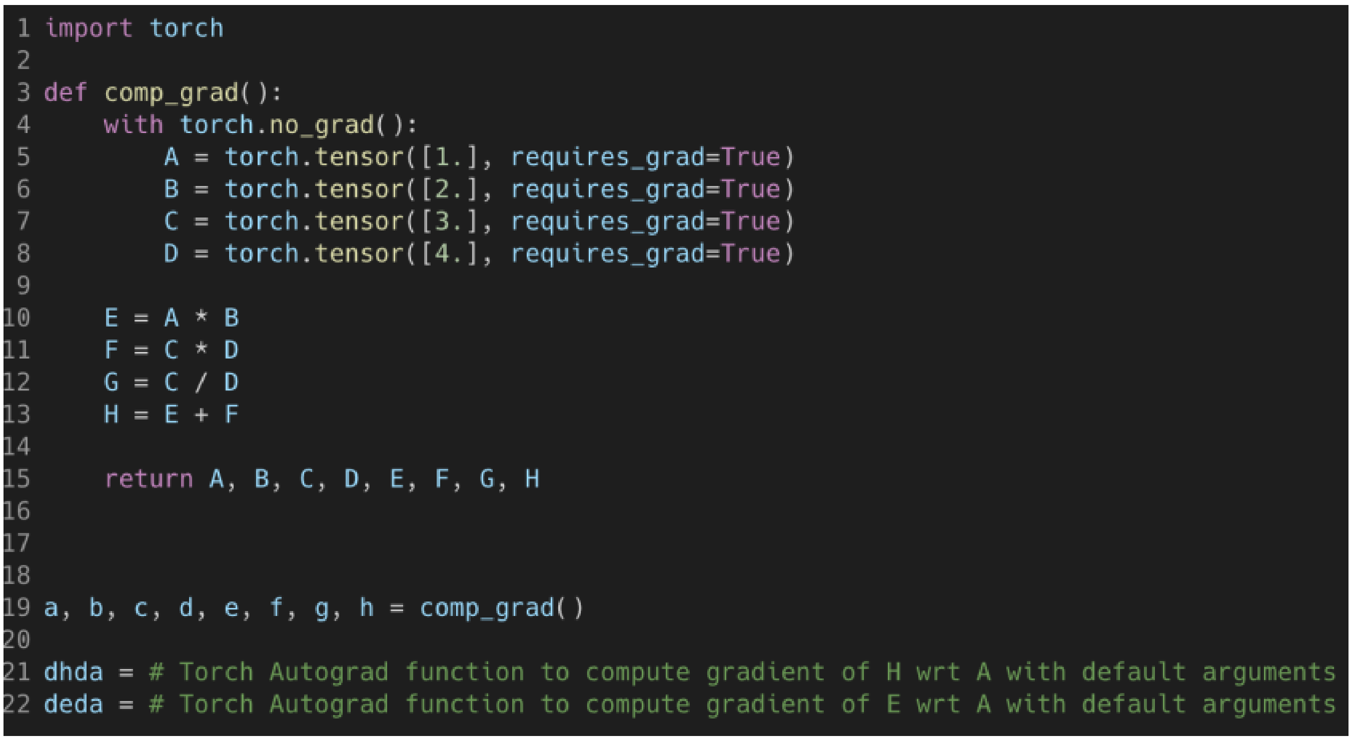
(b) A and C

(c) B and C

(d) None

Ans: Option B
Reason: Due to dropout=1.0 in A everything will be 0.0 and RELU making everything 0 in network C.

Q. Given the following code:



```
1 import torch
2
3 def comp_grad():
4     with torch.no_grad():
5         A = torch.tensor([1.], requires_grad=True)
6         B = torch.tensor([2.], requires_grad=True)
7         C = torch.tensor([3.], requires_grad=True)
8         D = torch.tensor([4.], requires_grad=True)
9
10    E = A * B
11    F = C * D
12    G = C / D
13    H = E + F
14
15    return A, B, C, D, E, F, G, H
16
17
18
19 a, b, c, d, e, f, g, h = comp_grad()
20
21 dhda = # Torch Autograd function to compute gradient of H wrt A with default arguments
22 deda = # Torch Autograd function to compute gradient of E wrt A with default arguments
```

Assuming no syntax errors and for gradient computation you use `torch.autograd.grad`, which of the following would allow the above code to run without any error:

(a) Removing the (last) line that calculates gradients of E w.r.t. A.

(b) Removing the (second last) line that calculates gradient of H w.r.t. A.

(c) Removing the (fourth) line that contains `with torch.no_grad()`.

(d) Nesting all the function code inside a `with torch.no_grad():` condition.

(e) Removing the `E, f, g, h = comp_grad()` line.

(f) Adding another `E, f, g, h = comp_grad()` before the gradient is computed a second time in the last line.

(g) The given code does not need any modifications and should run as is.

Note: Multiple options may be correct. Evaluate each option independently of the others i.e. when the change specified in one option is implemented, the other options are ignored.

Ans: a, b, f
Reason: as the gradients would have to be recalculated for the second computation

Q. Which of the following statements is or are NOT TRUE:

(a) Training the same fully connected model on a (1) normal image and on (2) an image with permuted pixels should lead to different accuracies as the image with permuted pixels does not satisfy the fully connected models prior assumptions.

(b) The output and gradients for a network with only Softmax would be more precise than a network with `logSoftmax`.

(c) The frequency scale used in Mel Frequency Cepstral Coefficients (MFCC) is linear in frequency.

Ans: a, b, c
Reason: for b: `https://discuss.pytorch.org/t/what-is-the-difference-between-log-softmax-and-`
`11801/6`

Q. Suppose you want to do classification, and you define a two layer network. Which of these loss functions do you think could be suitable to train the network?

(a) Cross Entropy

(b) Mean Squared Error (L2 loss)

(c) Binary Cross Entropy

(d) Manhattan Distance (L1 loss)

Ans: a,c
Reason: straight forward

Q. Let C be a function: $C(Y, \hat{Y})$, where $\hat{Y} = \max(0, Z)$. What is $\frac{\partial C}{\partial Z}$?

(a) $\frac{\partial C}{\partial \hat{Y}}$

(b) $\frac{\partial C}{\partial \hat{Y}} \cdot 1,\;$ if $Z > 0; 0$ otherwise

(c) 0

(d) Both (a) and (b)

Ans: (b)
Reason: max differentiation is smoothed.