S3-Assignment-Solution

Due Apr 12 at 6:30am **Points** 165 **Questions** 16 **Available** until Apr 12 at 6:30am
Time Limit 20 Minutes

Instructions

Instructions:

1. You have 20 minutes to attempt the S3-Assignment-Solution.
2. Make sure you have played around with the COLAB FILE shared earlier. Here is the link again
3. Once you start the solution, you cannot go back and re-attempt it
4. You will not find answers online, so please make sure you are ready for the quiz
5. For Multiple Answer Questions, ALL the answers must be correct to score any point

This quiz was locked Apr 12 at 6:30am.

Attempt History

	Attempt	Time	Score
LATEST	Attempt 1	4 minutes	140 out of 165

Score for this quiz: **140** out of 165
Submitted Apr 11 at 4:05pm
This attempt took 4 minutes.

Question 1 — 5 / 5 pts

How many dimensions are there in a tensor defined as below?

```
torch.rand(1, 1, 1, 1)
```

Correct!
- [x] 4
- [ ] 1

Question 2 — 0 / 20 pts

Assume that we moved our complete (cats vs dogs) image dataset to numpy arrays. Then we use torch.from_numpy to convert these images to tensor. Then we apply a specific data augmentation strategy called "CutOut" which blocks a portion of the image directly on these tensors. What will happen to the accuracy of a model trained on this strategy?

CutOut strategy is shown below:

Correct Answer
- [ ] Our model will not train and get stuck at 50% accuracy.

You Answered
- [x] CutOut is really a great augmentation strategy. Our model created with this strategy will have higher accuracy than the model trained without this strategy.

Question 3 — 0 / 5 pts

Why do you think we are observing this behavior?

You Answered
- [x] CutOut blocks a portion of an image. This causes the network to look for other cues to predict the object in the image. This reduces overfitting and improves performance.

Correct Answer

The way we have implemented the strategy, we will end up adding black blocks on images while changing the original image. After few operations, whole image will just be black. Then network would see just black images for dogs and cats, and thereby failing in recognizing either, getting stuck at 50% accuracy.

Question 4 — 10 / 10 pts

We saw above that some times numpy and tensors share same storage and changing one changes the other. If we define a rank-2 tensor with ones (dtype of f16), and then convert it into a numpy data type using tensor.numpy() and store it in a variable called "num", and then we perform this operation `num = num * 0.5`, will the original tensor have 1.0s or 0.5s as its element values?

Correct!
- [x] 1.0s
- [ ] 0.5s

Question 5 — 0 / 0 pts

If the operation `num = num*5` is changed to `num[:] = num*5` will the original tensor have 1.0s or 0.5s as its element values?

- [ ] 1.0s

Correct!
- [x] 0.5s

Question 6 — 5 / 5 pts

Is the transpose of concatenated a & b tensor on dimension 1, same as the contatenated tensor of a & b on dimension 0?

- [ ] True

Correct!
- [x] False

Question 7 — 0 / 0 pts

`a` is defined as `torch.arange(start=0, end=10)`. We will create `b` using the two operations as below. Is

1. indices variable created by the modulo operation on arange between 0 and 10. Then a new variable
2. indices variable created by the modulo operation on arange between 1 and 11. Then a new variable

Correct Answer
- [ ] True

You Answered
- [x] False

Question 8 — 10 / 10 pts

Consider a tensor defined as `torch.randn((6, 5))` .

Is the shape of the new tensor created by taking the 0th, 2nd and 4th row of the old tensor, same as the shape of the a newer tensor created by taking the 0th, 2nd and 4th row of the old tensor, after transposing it by operation `torch.transpose(tensor, 0, 1)` ?

- [ ] True

Correct!
- [x] False

Question 9 — 20 / 20 pts

Consider a tensor `a` created with [1, 2, 3] and [1, 2, 3] of size (2, 3) is reshaped with operation `.reshape`

Also consider a tensor `b` created with [[2, 1]] and of size (1, 2), later operated with `view(2, -1)` operatic

If we do a dot product of a and b (using `torch.mm`) and perform the sum of all the elements (using `torch.sum`),
what do we get?

Correct!
18

Correct Answers
18

Question 10 — 20 / 20 pts

Looking at the results above (check code) it can be said that the pixel values in the blue channels would be very small compared to red channel. True/False?

Correct!
- [x] True
- [ ] False

Question 11 — 20 / 20 pts

Why the gradient of a is all 5s above (refer code)?

- [ ] There is a bug in the code and we are not calculating gradient of a.

Correct!
- [x] Because that is what it should be based on how result is defined

Question 12 — 20 / 20 pts

In the code above (refer the notebook code), why do we have 2 in `2.0*(y_pred - y)` ?

Correct!
- [x] We are calculating the gradient of y which is derivate of (y_red - y)^2. When we perform derivative of a squared entity, "2" comes as a multiplier.
- [ ] It serves no purpose there, and we can still run the code without 2.
- [ ] We have added 2 to increase the "punishment value" of our network. Creating a higher penalizing value allows us to train the network better.

Question 13 — 10 / 10 pts

In the code above (refer the notebook code), what does `grad_h[h < 0] = 0` signify?

Correct!
- [x] This operation refers to the derivative of ReLU function
- [ ] When calculating gradients, we do not want negative gradients to flow into the network, that is why we are clipping negative gradients.

Question 14 — 10 / 10 pts

In the code above (refer the notebook code), how many "epochs" have we trained the model for?

Correct!
500

Correct Answers
500

Question 15 — 10 / 10 pts

In the code above (refer the notebook code), If we take the trained model, and run it on fresh inputs, the
trained model will be able to predict fresh output with high accuracy.

- [ ] True

Correct!
- [x] False

Question 16 — 0 / 0 pts

In the code above (refer the notebook code), if we dont use clone in `grad_h = grad_h_relu.clone()` the model will still train without any issues.

Correct!
- [x] True
- [ ] False

Quiz Score: **140** out of 165