

ESE 546

HOMEWORK 2

HARRY GUAN [HARRYG1@SEAS],
COLLABORATORS: HARRY GUAN [HARRYG1@SEAS]

Problem 1.

(a) - *times spent 7 minutes.* In the given ResNet code, the batch normalization (BN) layer is applied before the ReLU activation, following the sequence Conv \rightarrow BN \rightarrow ReLU. This design choice is intentional and generally considered better practice. Applying BN before ReLU ensures that the activations passed into the nonlinearity have a normalized distribution with zero mean and unit variance, which helps stabilize training and improves convergence. If BN were placed after ReLU, many activations would become zero due to ReLU's thresholding, leading to inaccurate mean and variance estimates. As a result, placing BN before ReLU allows the network to maintain more consistent feature scaling and achieve better overall performance.

(b) - *times spent 7 minutes.* The calls `model.train()` and `model.eval()` in PyTorch are used to switch the model between training and evaluation modes. During training, `model.train()` enables layers like dropout and batch normalization to behave in their training state—dropout randomly zeroes activations, and batch norm updates its running statistics. In contrast, `model.eval()` sets these layers to evaluation mode, where dropout is disabled and batch norm uses its learned running averages instead of batch statistics. These calls are important to ensure consistent behavior during validation or testing. In HW1, we didn't include them because our custom library was a simpler implementation that didn't have layers like dropout or batch normalization, so there was no need to change the model's behavior between training and evaluation phases.

(c) - *times spent 7 minutes.*

(d) - *times spent 7 minutes.* Weight decay, or L2 regularization, penalizes large weights by adding a term proportional to the square of their magnitude to the loss function. This helps prevent overfitting by discouraging overly complex models. However, applying weight decay to bias terms is not appropriate because biases simply shift the activation functions and do not control the capacity or smoothness of the model in the same way as weights do. Penalizing them can lead to underfitting or prevent the model from properly centering its activations.

(e) - *times spent 7 minutes.*