HW2：Cifar-10 Classification with MLP and CNN

计02 刘明道 2020011156

基础问题

`self.training` 如何工作？为什么在训练和测试时需要不同?

- 在 PyTorch 的实现中，`.train()` 递归更新自己和子模块的 `.training` 为 `True`，`.eval()` 则会递归更新为 `False`。
- 而 `self.training` 用于标识模型当前的状态，即是在训练还是在推理。
- 训练和测试时的模块需要有不同的行为，因此需要进行区分。
- 在本次实验中，该属性主要影响了 BatchNorm 和 Dropout 的行为

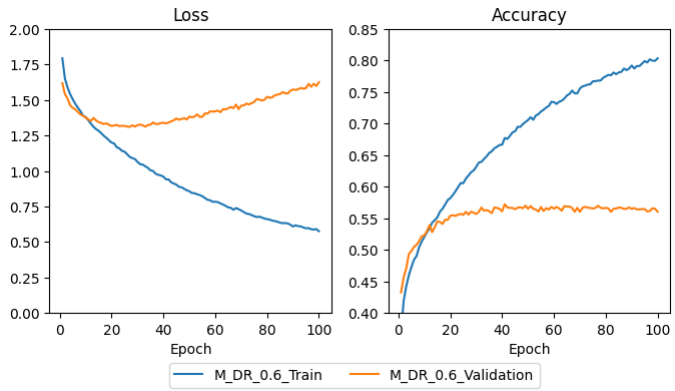
Module	`.train()`	`.eval()`
BatchNorm	计算该batch的均值和方差 μ_i 和 σ_i^2 ，然后使用指数移动平均更新 `running_mean` 和 `running_var`	直接使用 `running_mean` 和 `running_var` 对输入进行标准化
Dropout	对输入以概率 p 随机置零，然后将输入乘以 $\frac{1}{1-p}$	恒同映射

MLP 实验

在验证集上表现最好的一组实验的超参为

参数	**值**
Learning Rate	0.001
Hidden Size	1024
Dropout Rate	0.6
Weigth Decay	0.001
Batch Size	100
Max Epoch	100

训练过程中的 Loss 与 Accuracy 如下

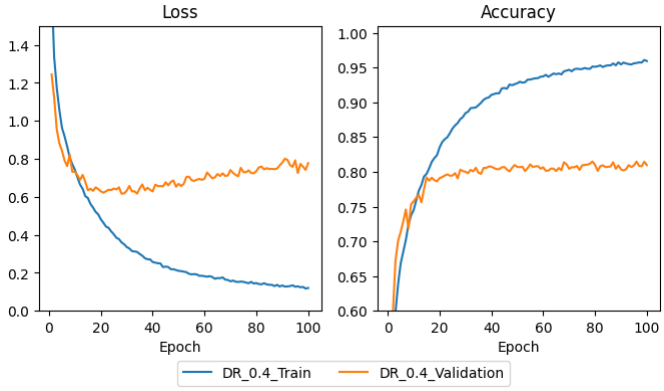


CNN 实验

在 Dropout Rate 非零的实验中，验证集上表现最好的一组实验的超参为

- Conv Layers：channel 为 128 和 512，kernel size 为 5 和 7，padding 为 2 和 3
- Max Pooling Layers：kernel size 为 5 和 5，padding 为 2 和 2，stride 为 3 和 4
- Dropout Layers：dropout rate 为 0.4 和 0.4
- 其他超参为
  - Learning Rate 0.001
  - Optimizer: AdmaW，Weight Decay 1×10^{-5}
  - Batch Size 100
  - Max Epoch 100

训练过程中的 Loss 与 Accuracy 如下



为什么训练集和验证集上的 loss 值不同？这对调参有何帮助？

- 在训练最初期，Training Loss 可能会比 Validation Loss 略高一些，这可能是因为
  - 此时模型在一个 Epoch 中提升较大，Training Loss 是整个训练过程中 loss 的平均，而 Validation Loss 则是每个 Epoch 训练结束后的测量
  - 测量 Validation Loss 时， 模型 `.training=False` 。 `Dropout` 退化为恒同映射，此时利用了比训练过程更多的神经元，因此模型性能也会更强
- 在训练的后期，Training Loss 比 Validation Loss 显著低，而 Validation Loss 甚至有所增加。这可能是因为
  - 训练集和测试集的样本特征分布并不完全一致。

    - 在训练初期，模型学到的特征中有许多较为普遍的特征，这些特征同时存在于训练集与测试集中，所以 Training Loss 和 Validation Loss 都在明显下降
    - 在训练后期，模型拟合了更多仅存在于训练集中的特征，这些特征可以帮助模型更好地完成训练集中的任务，因此 Training Loss 进一步下降。但是由于这些特征是训练集特有的，模型无法更好地完成验证集中的任务，甚至会因过拟合训练集而效果更差，因此 Validation Loss 难以下降甚至回升。
- 对调参的帮助
    - 如果在训练过程中 Validation Loss 开始上升而 Training Loss 仍在下降，说明模型可能出现了较为严重的过拟合，且过拟合已经开始对性能产生明显影响。此时可以考虑增加 Dropout Rate，增大 Weight Decay 等方法减轻过拟合，或者修改 Max_Epoch，缩小训练周期。
    - 如果 Training Loss 与 Validation Loss 十分接近，甚至比后者高（如 Dropout Rate 很大时），那么模型可能在训练过程中是欠拟合的，没能有效捕捉特征。此时可以降低 Dropout Rate 来让模型学习更复杂的特征。

准确率

我们选取在 Validation Set 上表现最好的 checkpoint 评估其在 Test Set 上的准准确率。

实验	Training Accuracy / %	Validation Accuracy / %	Testing Accuracy / %
MLP	80.33	57.23	56.09
CNN	96.10	81.49	80.66

MLP 与 CNN 对比

从测试集准确率来看，CNN 的效果明显优于 MLP。可能的原因包括

- CNN 在进行卷积运算时，会从有空间相关性的点进行特征提取，还会使用 Max Pooling 将不同位置的信息进行聚合，因此可以更好地利用空间相对位置关系的信息；而 MLP 将图像拉直为 1 维向量，模型失去了像素之间空间关系的先验信息。
- 在本次作业中，MLP 有 3.16M 参数，CNN 有 3.27 M 参数。两者数量接近。CNN 利用卷积核来提取信息，同一输出通道不用区域之间的参数是共享的。CNN 的卷积参数被利用到了较大的范围中，而不只是某个特定的区域，因此利用效率更高，泛化能力更强，提取特征更有效；而 MLP 为了连接所有的像素点，参数与固定的位置的输入相关，参数利用率低，也更容易在训练集上过拟合。

Dropout & BatchNorm 的效果

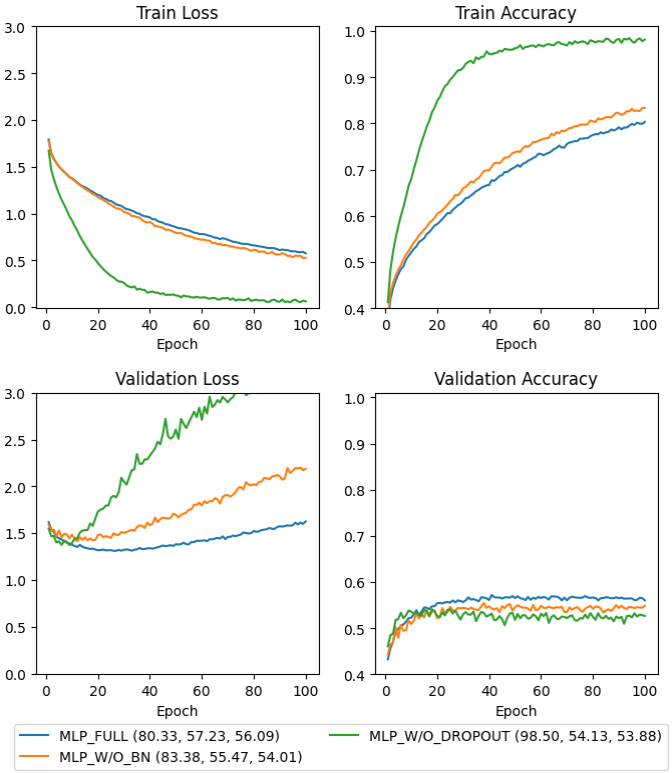
下面在 MLP 和 CNN 中列出分别去掉 Dropout 和 BatchNorm 后的训练过程。

图中 FULL 表示之前提到的实验配置；W/O_BN 表示在先前实验配置的基础上去除 BatchNorm 层；W/O_DROP 表示在先前实验的基础上去掉 Dropout 层。

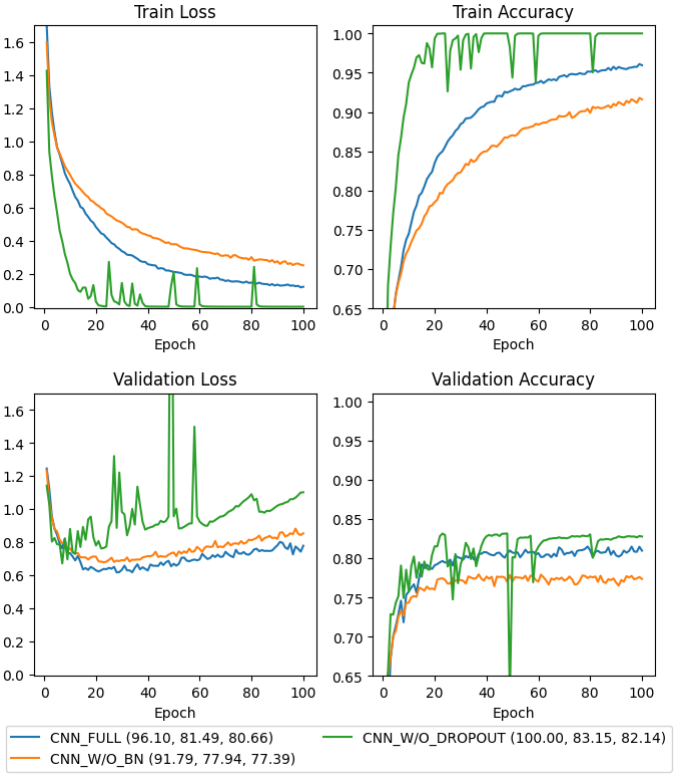
图例中括号中的数字表示：

$(\text{train set 最高准确率，val set 最高准确率，在 val set 准确率最高时的 test set 准确率})$

MLP


Train Loss
Train Accuracy
Validation Loss
Validation Accuracy
Epoch
MLP_FULL (80.33, 57.23, 56.09)
MLP_W/O_BN (83.38, 55.47, 54.01)
MLP_W/O_DROPOUT (98.50, 54.13, 53.88)

CNN


Train Loss
Train Accuracy
Validation Loss
Validation Accuracy
Epoch
CNN_FULL (96.10, 81.49, 80.66)
CNN_W/O_BN (91.79, 77.94, 77.39)
CNN_W/O_DROPOUT (100.00, 83.15, 82.14)

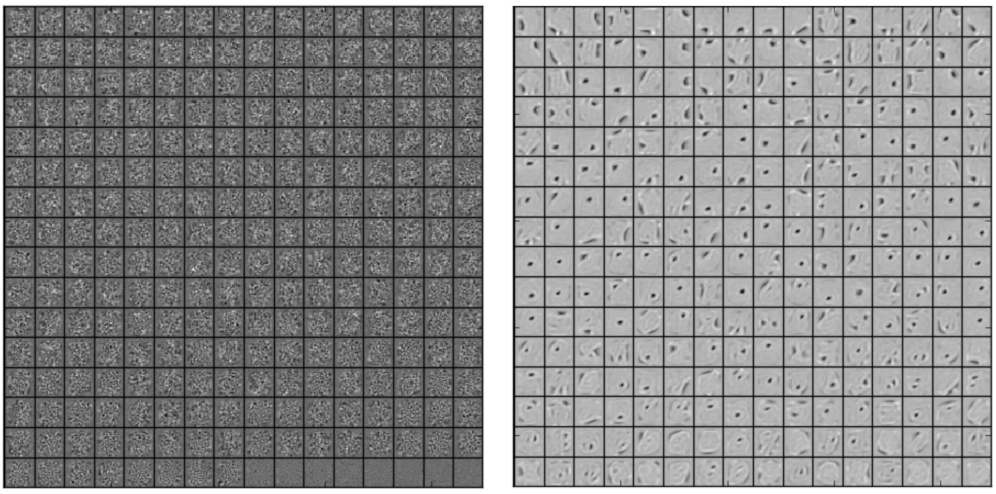
BatchNorm 的效果

- 相比没有 BatchNorm 来说，使用 BatchNorm 使 MLP 和 CNN 的测试集准确率分别提升了 2.08% 和 3.27% 。
- 在训练的过程中，模型的参数在每次 Step 后都会更新。如果没有 BatchNorm，每次反向传播后，后续网络输入的统计特性将会发生一定的变化，而**后一层将需要不断的调整参数来适应这一变化。**对于深度较大的网络而言，这种统计特征的改变还会被放大，导致深度网络中各层的输入的统计特征发生显著漂移。
- 而使用了 BatchNorm 后，我们直接要求网络的均值方差为定值，尽量减少输入的分布漂移，**使后一层无需不断调整适应前一层网络统计特性的变化，从而使每层的训练更加专注于学习新的特征。**因此，使用 BatchNorm 的网络泛化性能更好。对于 CNN 这种层数较多的网络，使用 BatchNorm 还能显著加快收敛速度。

Dropout 的效果

- 从准确率来说，相比没有使用 Dropout， Dropout 层使得 MLP 的测试集准确率提升了 2.21%，但使 CNN 的测试集准确率降低了 1.48%。在 CNN 和 MLP 上，Dropout 减小了训练过程中模型在测试集和验证集上的准确率差异。
- 在关闭 Dropout 时，模型的 Training Loss 迅速收敛到接近零，而 Validation Loss 在训练后期显著回升甚至发散；同时关闭 Dropout 也会使模型振荡更严重。
- 在没有 Dropout 层时，模型迅速背会了训练集中的特征；这样的过拟合很可能是因为不同的层次之间有较强的依赖性，后续网络习得的参数对前一层网络的全部参数依赖严重，**这导致网络可以通过形成复杂的前后依赖关系来过拟合训练集的数据。**
- 而 Dropout 层在训练过程中，相当于取了全部网络参数的一个子集进行训练。对于每层来说，它的训练目标则是对**前一层网络不同子集的输出，都有良好的效果**，因此可以学到更具泛化性的特征。从 Dropout 有关论文 的定性结果中明显看出，使用了 Dropout 的网络更能有效提取特征，而不只是生硬拟合

7.1 Effect on Features



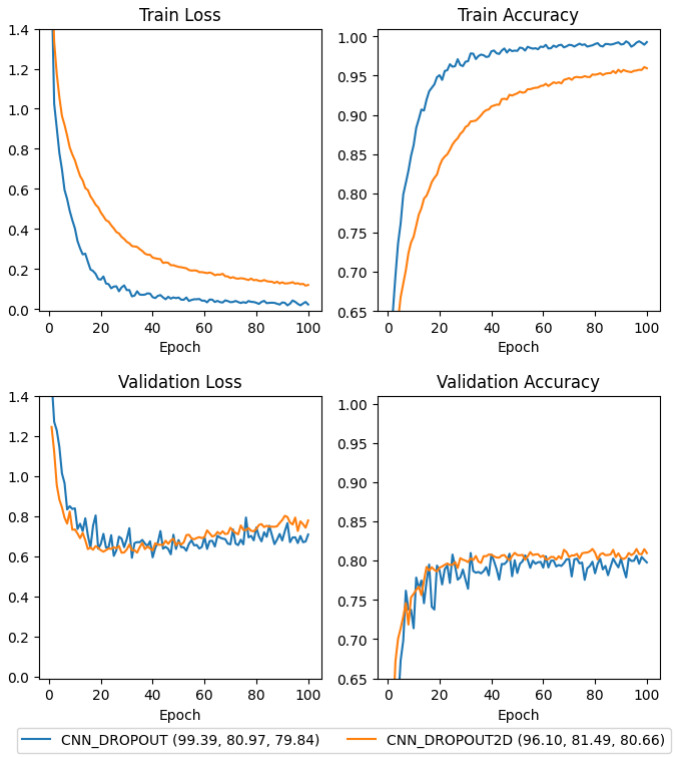
(a) Without dropout　　(b) Dropout with $p = 0.5$.

Figure 7: Features learned on MNIST with one hidden layer autoencoders having 256 rectified linear units.

- 对于 MLP 模型来说，如前文所述，其参数效率不是很高，第一层的参数总是与特定区域相联系，利用率并不很高，此时 Dropout 可以在减轻过拟合的同时提升测试准确率；然而对于 CNN 来说，其参数利用率已经很高，**此时使用 Dropout 虽然会缓解过拟合，但也会抑制模型在训练过程中对特征的提取，使得不同的层次之间的配合能力减弱**，造成最终准确率稍有下降。

其他问题

Dropout2D

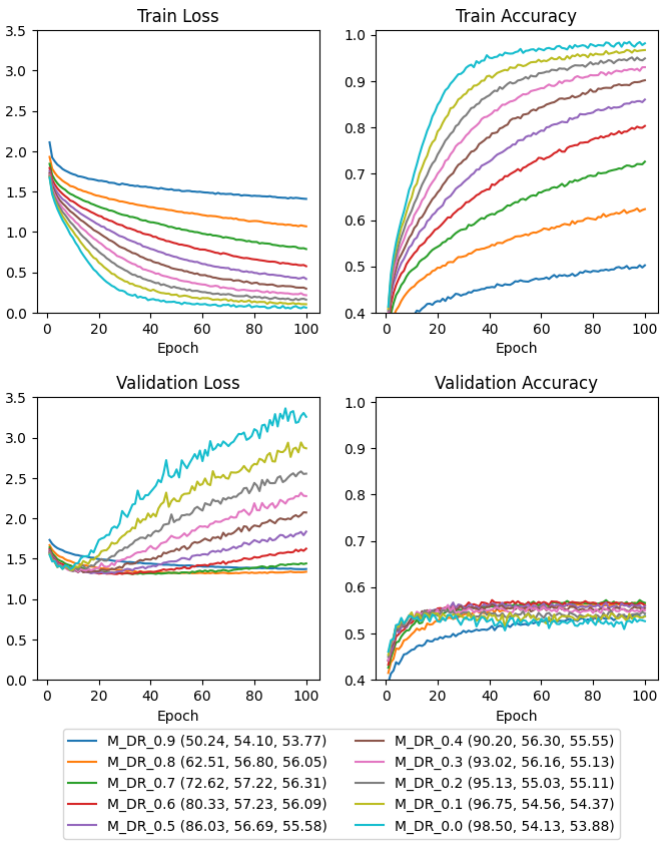


- 相比 Dropout，Dropout2D 提升了最终的准确率，更有效地缓解了过拟合现象，使模型在训练过程中更加稳定。
- 对于 CNN 的输出来说，通道中的元素与它在空间上临近的元素有很强的相关性，此时如果使用 Dropout，**那么同一通道上空间相关性较高的点也会有较大概率被保留一部分**，此时后一层的网络将仍然可以利用这些元素来构造较为复杂的依赖关系，这就减弱了 Dropout 的正则化效果。
- 而如果使用了 Dropout2D，那么与前一层输出的整个通道将会被同时置零，此时后一层的网络就必须依赖**其他通道**提取的特征进行学习，这有效降低了网络对前一层特定通道参数的依赖，使得 Dropout 的正则化效果得以充分发挥。

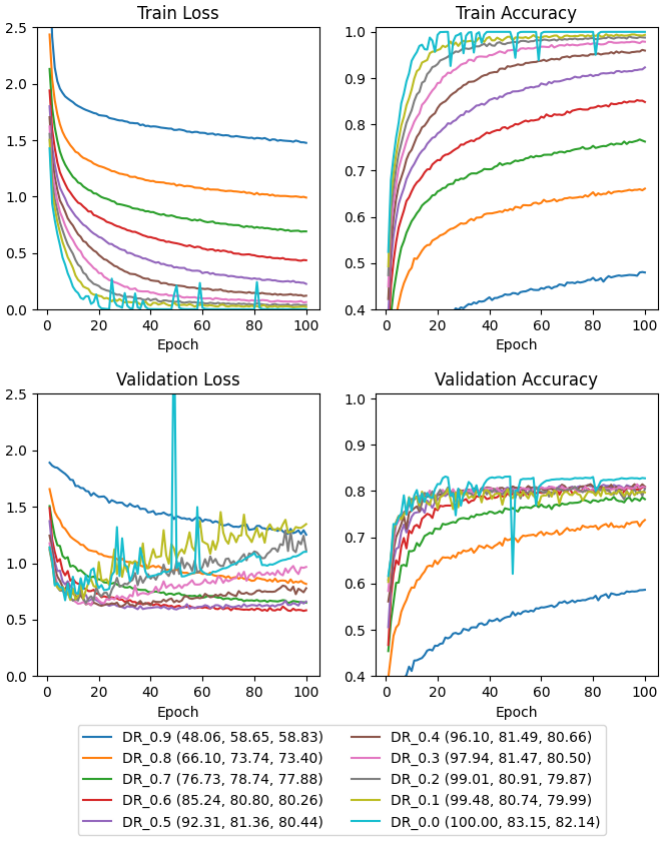
超参数

Dropout Rate

MLP



CNN

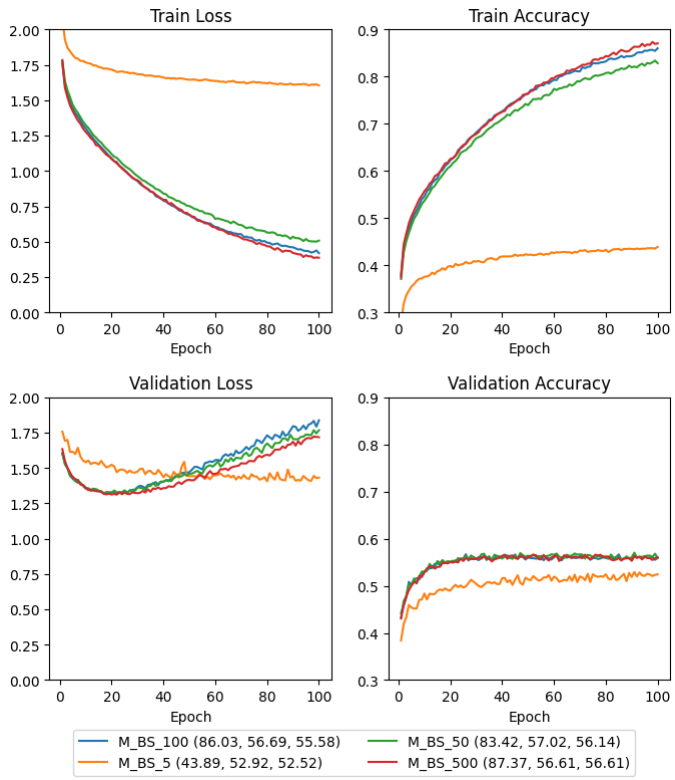


这一节我们探索 Dropout Rate 对 Dropout 层作用效果的影响。我们使用与最优实验相同的超参数，仅调节 Dropout Rate $p \in \{0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$，记录训练过程中 在训练集和验证集上的 Accuracy 与 Loss 曲线，并给出在各数据集上的准确率准确率（见图例）。

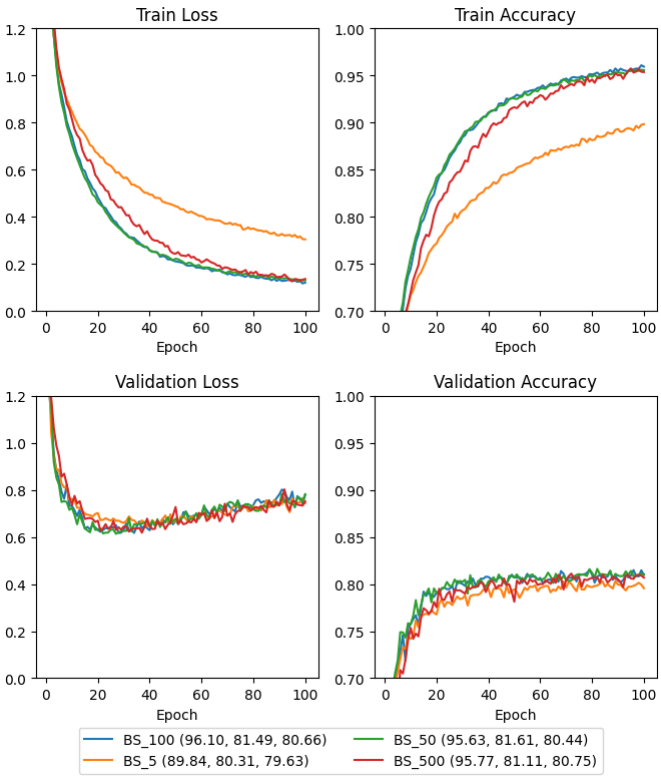
- 从准确率角度来看，随着 Dropout Rate 提升，过拟合越来越缓解，直到在训练集与验证集上表现相当（如 CNN 的 $p = 0.7, 0.8, 0.9$）
- Dropout Rate 过高时，模型欠拟合，表现为 Validation 和 Train Accuracy 在 100 个 Epoch 后准确率仍然较低，训练集与验证集表现接近，甚至训练集比测试集准确率更低。**此时，模型每一层得到的有效输入太少，层与层之间缺乏必要的连接，参数更新过于缓慢。**过高的 Dropout 显著限制了模型在训练过程中得到的能力。
- Dropout Rate 过低将导致模型出现极为严重的过拟合，在表现为 Training Loss 极为接近 0，而 Validation Loss 发散。
- **对不同的模型结构，最适的 Dropout Rate 有所不同。**对 MLP 而言，层与层之间的全连接很容易形成复杂的依赖，进而形成过拟合，因此最适的 Dropout Rate 就相对较大，如本实验中的 $p = 0.6$。而对于 CNN 来说，其参数本身被用于不同的区域，利用率较高，如果 Dropout Rate 过大，将可能限制模型在训练过程中学到的特征提取能力。对于本实验来说， CNN 取 $p \in [0.1, 0.6]$ 时性能接近，而 $p = 0$ 时性能则有明显提升。一个很小的 p 可能对本实验的 CNN 架构较为合适。

Batch Size

MLP



CNN



这一节我们探索 BatchSize 对 BatchNorm 层作用效果的影响。同样，我们使用与最优实验相同的超参数，仅调节 Batch Size $B \in \{5, 50, 100, 500\}$ 。

- Batch Size 过小时，模型的在训练过程中振荡更明显（MLP），且最终准确率更低（MLP，CNN）；Batch Size $B \in \{50, 100, 500\}$ 性能差异则不明显。
- 可能的原因包括
  - Batch Size 过小时，输入随机性较高，**训练过程中 BatchNorm 层对均值和方差的估计的不确定性增大**，有时可能会与数据整体情况产生明显偏差，造成 BatchNorm 处理后输入振荡更加剧烈。这种训练过程中的不稳定性甚至导致 MLP 模型在训练集上的最高准确率低于在验证集上的最高准确率。
  - 另一方面，输入随机性较高也会导致每次的梯度更新的随机性提高，这也可能使模型的训练过程更不稳定。