HW2-report

计11 周韧平 2021010699

`self.training` 工作原理，训练和测试不同的原因

`self.training` 为基类 `nn.Module` 中定义的成员变量，是一个布尔值。它用于区分模型是在训练模式（`self.training == True`）还是在评估模式（`self.training == False`）。在代码中，通过调用 `model.train()` 和 `model.eval()` 去递归的改变子模块的 `self.training` 变量值，从而控制模型在训练和测试两种模式下的不同行为。具体来说，在本次实验中，训练和测试有两点不同

- **Dropout**: 在训练模式下，Dropout 层会以 p 的概率随机"丢弃"一些神经元的输出，使其输出为 0，并对活动的神经元模型输出乘以 $\frac{1}{1-p}$ ，这有助于防止模型过拟合。但在评估模式下，Dropout 层应将所有神经元视为"活动"的，并不丢弃任何输出。
- **Batch Norm**: 在训练时，Batch Norm 使用当前 batch 的统计信息（均值和方差）对数据进行归一化。而在评估时，它使用累计的移动平均统计数据，这些数据是在整个训练过程中收集的。

除此以外，训练和测试还可能在数据集、正则化技术、是否反向传播等行为上有所不同，二者的目的也有所不同，训练模式是为了通过最小化损失函数值来调整模型参数，而测试则是为了检测模型在测试集上的表现。

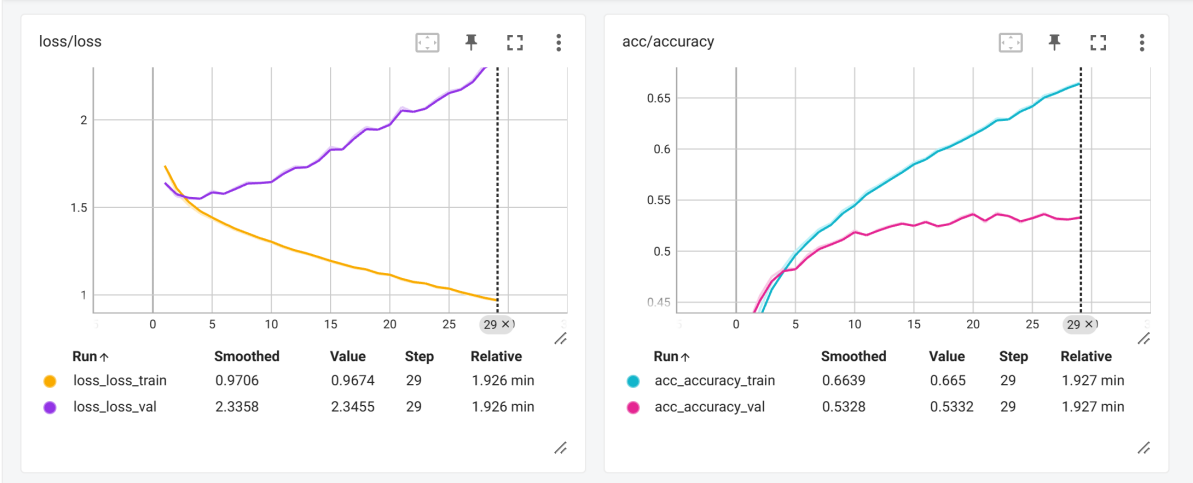
MLP实验

在验证集上表现最好的一组超参是

```
{
    "drop_rate": 0.5,
    "hidden_features": 1024,
    "learning_rate": 5e-4,
}
```

和训练有关的超参为

- Learning rate: 0.001
- Max_epochs: 30
- Batch Size: 100



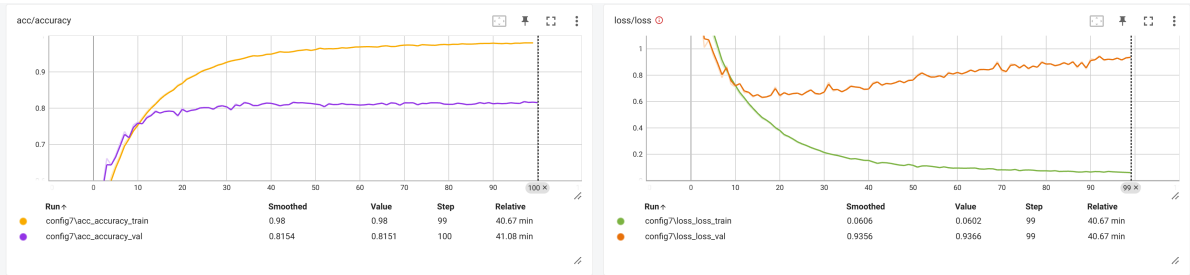
CNN实验

在验证集上表现最好的一组超参是

```
{
    "layer1": {
        "conv":{
            "in_channel":3,
            "out_channel":256,
            "kernel":5,
            "stride":1,
            "padding":2
        },
        "maxpool":{
            "kernel":5,
            "stride":3,
            "padding":2
        }
    },
    "layer2": {
        "conv":{
            "in_channel":256,
            "out_channel":1024,
            "kernel":7,
            "stride":1,
            "padding":3
        },
        "maxpool":{
            "kernel":5,
            "stride":3,
            "padding":2
        }
    },
    "drop_rate":[0.4,0.4]
}
```

和训练有关的超参为

- Learning rate：0.001
- Max_epochs: 100
- Batch Size: 100



分析 training loss 和 validation loss 不同的原因以及如何帮助调整超参

- 在训练初期，train loss往往比 test loss 稍大，这可能是以下两个原因导致

1. 在模型训练初期，一个 epoch 内模型提升较大，train loss 是对整个epoch内的loss取平均，而validation loss 则是在一个 epoch 结束后再去测量的，因此会比 train loss 稍低
2. 在模型训练初期，过拟合情况并不明显，模型对 train 集合和 validation 集合的识别能力还没有很大的区别，而由于 validation 集合测试时使用 `model.eval()` 开启了评估模式，在 `Dropout` 这一层使用了更多的神经元，因此能够比 train 集合表现更好。

- 随着训练 epoch 数增加，train loss 依然稳定下降，但 validation loss 不降反升，这可能是由于训练集和验证机的样本特征并不完全一致，在训练一开始，模型学到的更多是普遍的、通用的特征，而随着训练的深入，由于没有采用数据增强技术等手段，模型反复看到训练集中同样的图片，逐渐学到更多训练集特有的特征，而这些特征是验证集所没有的，也就出现了过拟合的现象。
- 对调参的帮助：
  - 如果 train loss 和validation loss 接近甚至 validation集合上模型表现更好，则说明模型出现了欠拟合现象，此时应当适当减小 `dropout_rate`，或通过调整为参数量更大的网络结构来提升模型的学习能力，并适当延长训练批次
  - 如果出现了 train loss 下降但 validation loss 开始上升的情况，说明模型出现了严重的过拟合现象，此时应当采取相应的手段来避免过拟合，如增大 `dropout_rate`，尝试引入正则化技术等手段
  - 通过观察 validation 集上最低点出现的位置，可以适当调整 max_epoch 的大小，避免后续不必要的训练，或者引入早停等技术手段实现动态训练。

汇报测试集上的表现，并分析MLP和CNN网络表现不同的原因

准确率

下面给出MLP和CNN两种网络在 Train 集，Validation 集和 Test 集上的表现，其中 Validation 和 Test 集取的是在 Validation 集上表现最好时保存的模型 checkpoint

网络结构	Train accuracy	Validation accuracy	Test accuracy
MLP	0.6650	0.5361	0.5293
CNN	0.9800	0.8178	0.8067

两种网络对比分析

从准确率来看，CNN 模型远强于 MLP 模型，可能原因包括以下两点

- 从模型参数量来看，CNN模型参数量为13M，而MLP模型只有3M，更大的参数量提升了模型的复杂度，如果能够避免过拟合问题的话，模型可以更加准确的分类图像，并具有更强的泛化性
- 从其它实验来看，即使将CNN模型的参数量调低至和MLP同一水平，CNN模型依然明显优于后者。这是因为在图像分类任务上，CNN模型参数利用更加高效。CNN模型的卷积层具有局部感知野（Local Receptive Fields）和参数共享（Weight Sharing）特性，即一个小的固定的卷积核被用于图像的不同位置，这种做法一方面降低了参数量，降低了过拟合的风险，另一方面也使得模型对图像的空间特征更加敏感，在处理cifar10这样的图像分类任务上相比MLP更有优势

最后，尽管准确率上CNN远强于MLP，但实际训练中我们也会发现MLP在推理和训练速度上都要更快，每个epoch训练的速度是后者的4-5倍，且在训练时MLP占用显存较小。

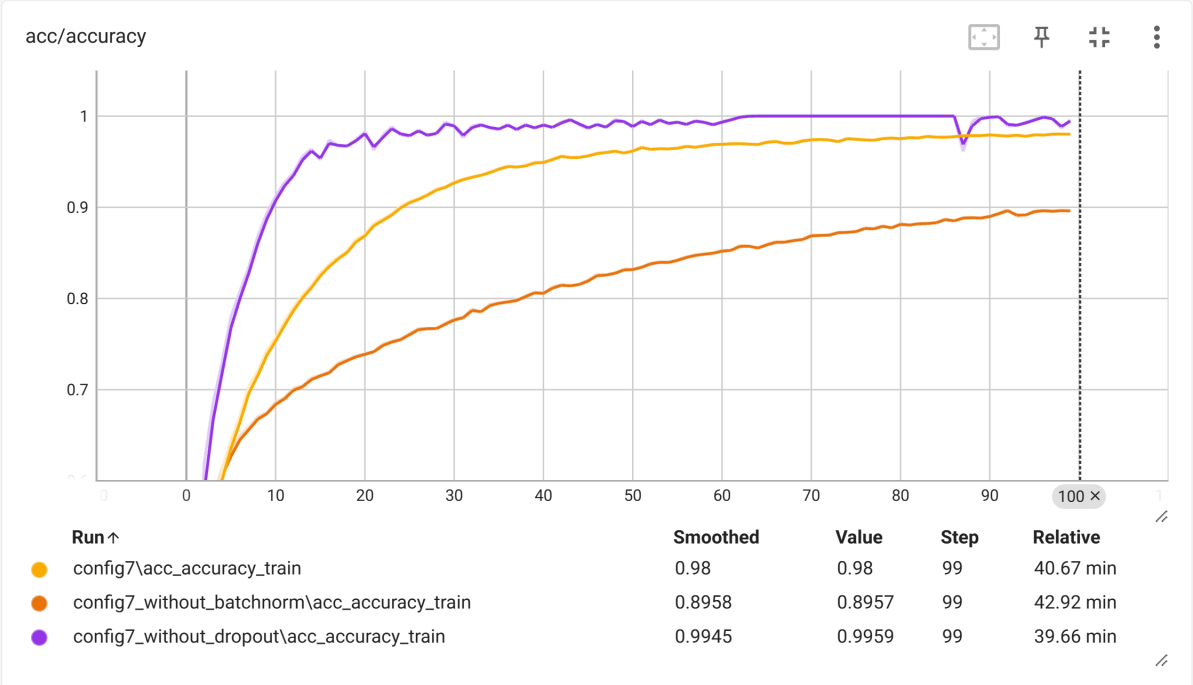
Batch Norm & Dropout 层消融实验

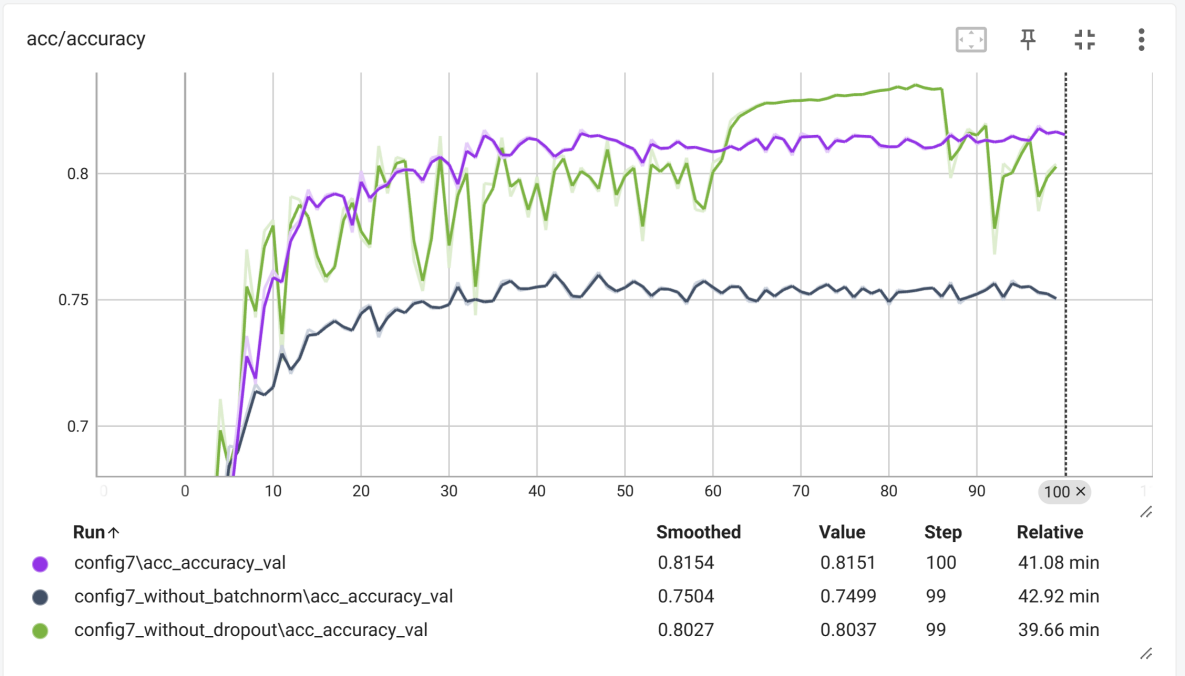
本节中，我将CNN和MLP表现最好的模型分别去掉 Batch Norm 层和 Dropout 层，其它超参数不变进行训练，得到测试集上结果如下

网络	完整结构	无 Dropout 层	无 Batch Norm 层
CNN	0.807	0.829 (+0.022)	0.752 (-0.055)
MLP	0.529	0.527 (-0.002)	0.555 (+0.026)

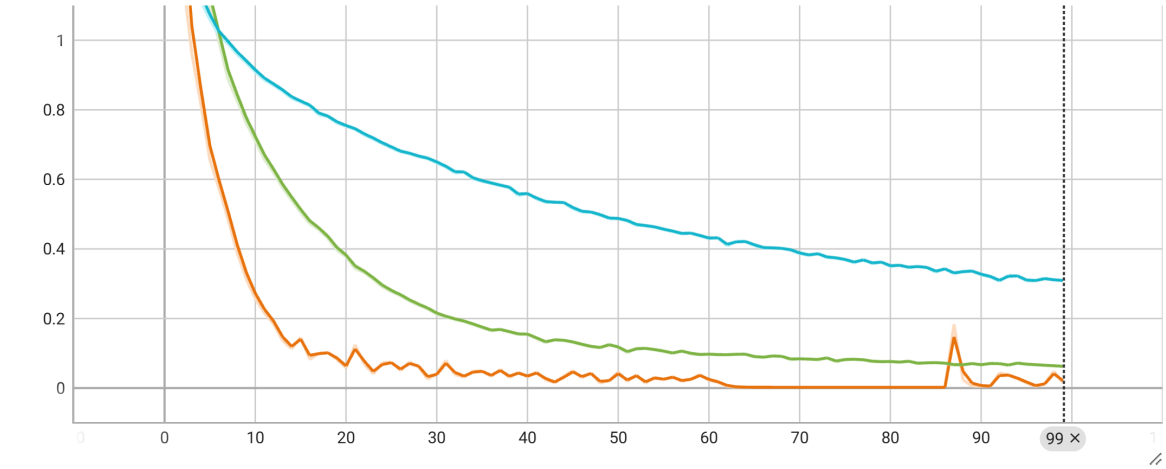
训练过程中 train 和 validation 集合上的 loss 和 accuracy 曲线如下

CNN



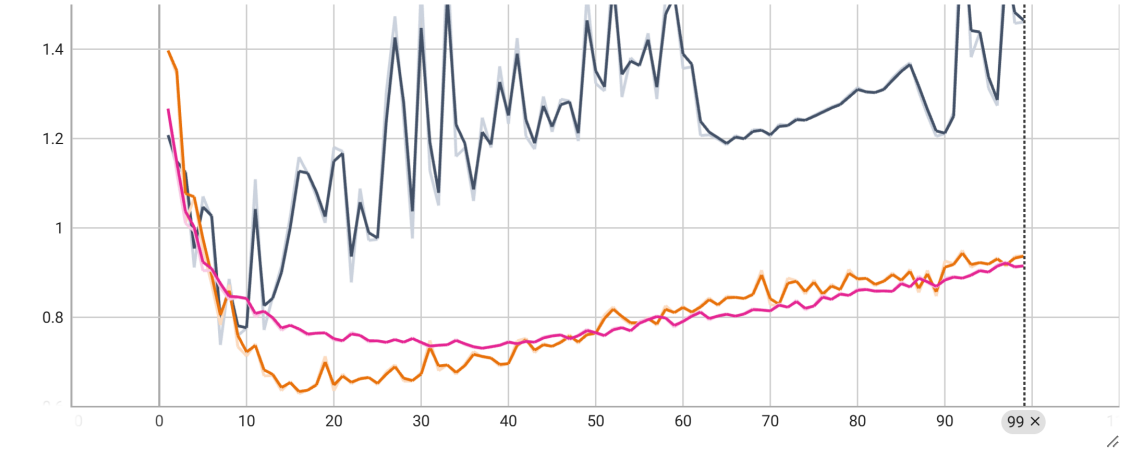


loss/loss



Run	Smoothed	Value	Step	Relative
config7\loss_loss_train	0.0606	0.0602	99	40.67 min
config7_without_batchnorm\loss_loss_train	0.3078	0.3073	99	42.92 min
config7_without_dropout\loss_loss_train	0.0191	0.014	99	39.66 min

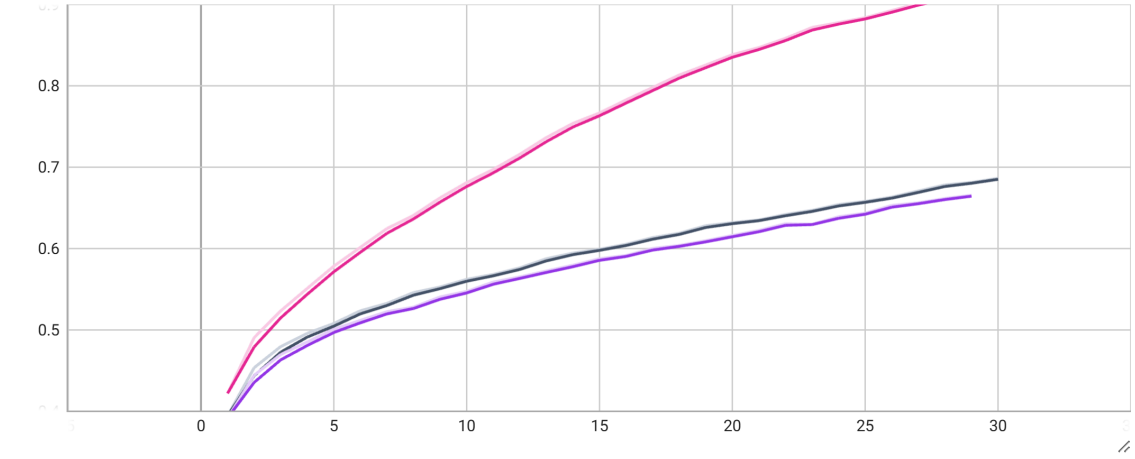
loss/loss



Run	Smoothed	Value	Step	Relative
config7\loss_loss_val	0.9356	0.9366	99	40.67 min
config7_without_batchnorm\loss_loss_val	0.9138	0.9142	99	42.92 min
config7_without_dropout\loss_loss_val	1.4649	1.4605	99	39.66 min

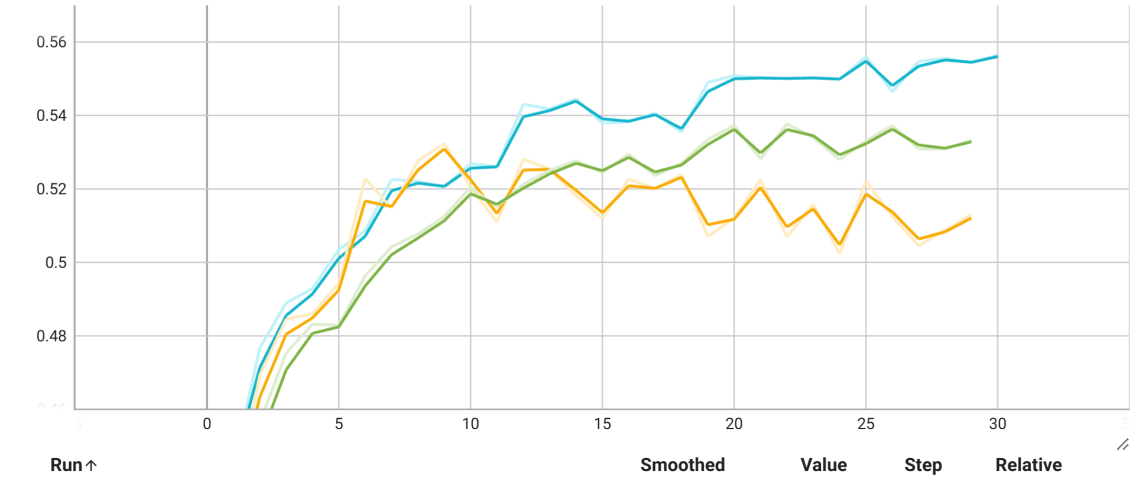
MLP

acc/accuracy

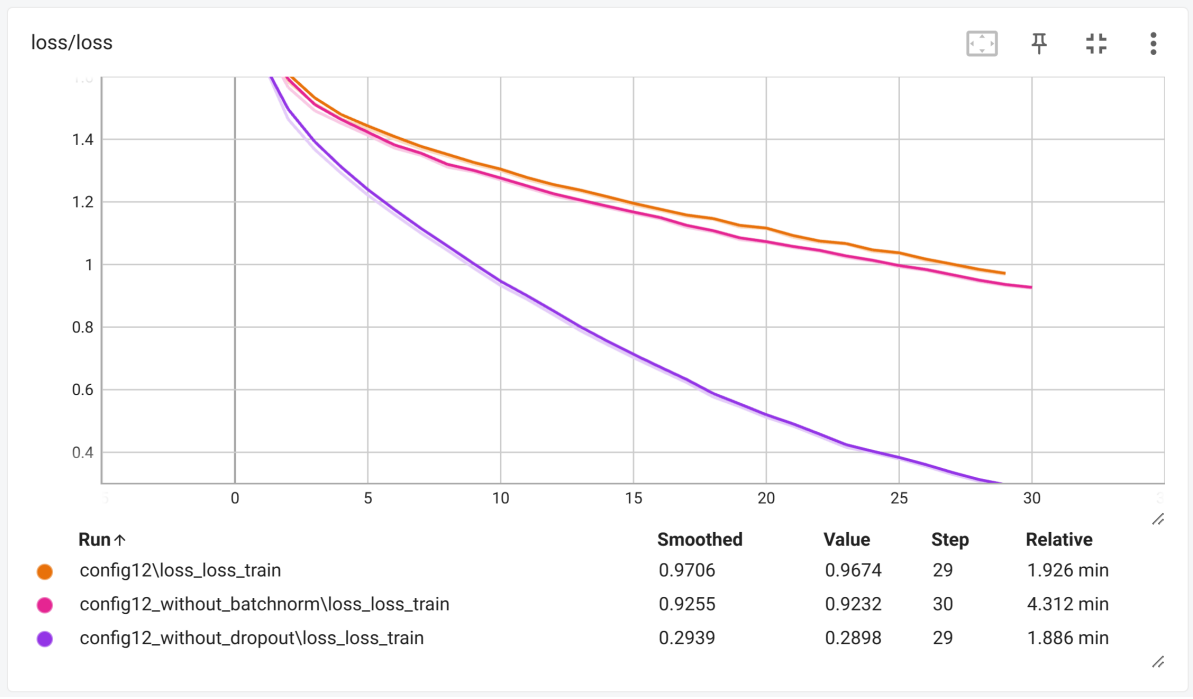


Run	Smoothed	Value	Step	Relative
config12\acc_accuracy_train	0.6639	0.665	29	1.927 min
config12_without_batchnorm\acc_accuracy_train	0.6847	0.6859	30	4.312 min
config12_without_dropout\acc_accuracy_train	0.913	0.9146	29	1.887 min

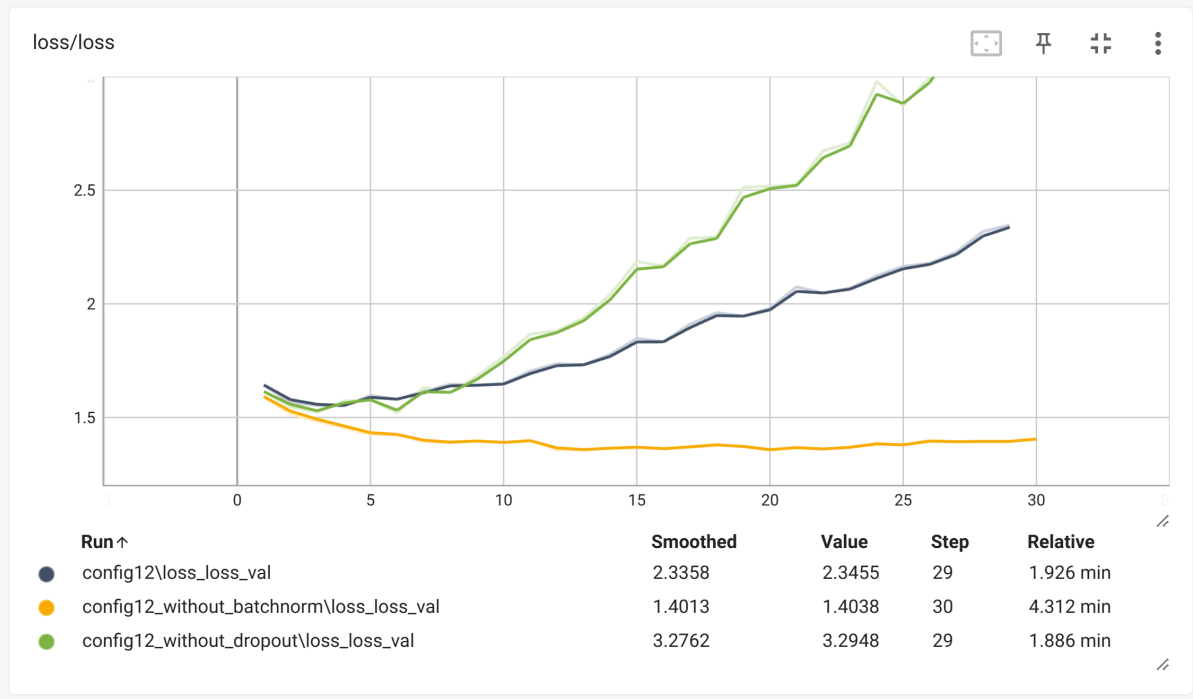
acc/accuracy



Run	Smoothed	Value	Step	Relative
config12\acc_accuracy_val	0.5328	0.5332	29	1.927 min
config12_without_batchnorm\acc_accuracy_val	0.556	0.5564	30	4.312 min
config12_without_dropout\acc_accuracy_val	0.512	0.5129	29	1.887 min



Run	Smoothed	Value	Step	Relative
config12\loss_loss_train	0.9706	0.9674	29	1.926 min
config12_without_batchnorm\loss_loss_train	0.9255	0.9232	30	4.312 min
config12_without_dropout\loss_loss_train	0.2939	0.2898	29	1.886 min



Run	Smoothed	Value	Step	Relative
config12\loss_loss_val	2.3358	2.3455	29	1.926 min
config12_without_batchnorm\loss_loss_val	1.4013	1.4038	30	4.312 min
config12_without_dropout\loss_loss_val	3.2762	3.2948	29	1.886 min

实验分析

Dropout

从准确率来看，Dropout层使MLP在测试集的准确率提升了0.2%，但使CNN在测试集的准确率下降了2.2%。分析可能的原因为：

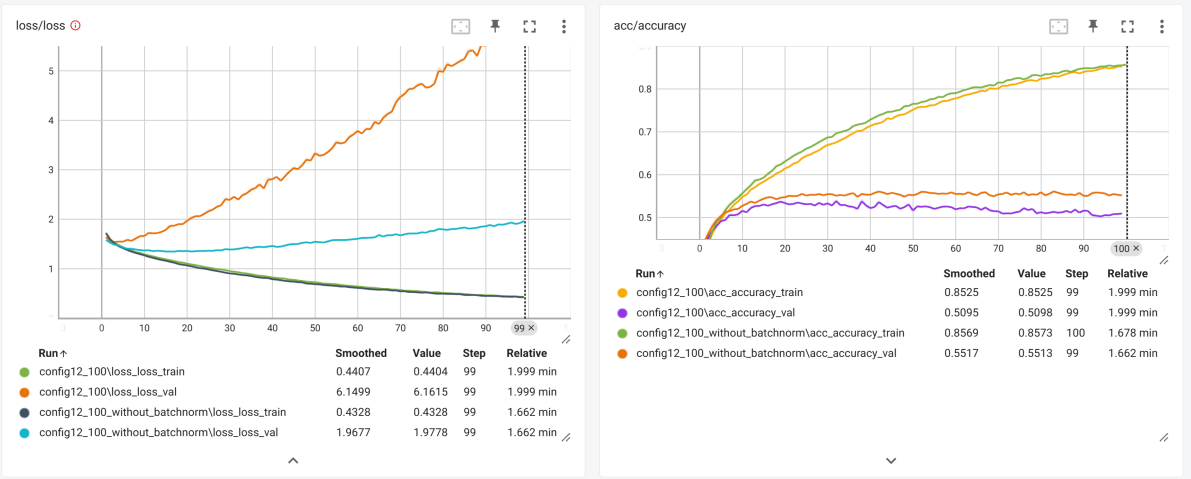
- Dropout在训练过程中随机地断开一些神经元的连接，从而使得网络在训练时限制了某些特定神经元之间的强依赖性，并强迫网络学习到更具有泛化能力的特征。通过观察 train loss 可以发现，不管是MLP还是CNN网络，引入dropout 层后 train loss 下降明显变慢，这是因为 Dropout 会在训练期间关闭一部分神经元，使得每次这可能会导致网络在学习过程中损失一些信息，因此下降更慢。
- 对于 MLP 模型，因为具有大量的全连接层的结构，容易出现神经元之间的强依赖关系，并导致过度拟合的问题，因此增加Dropout层可以让模型权重的更新更加平稳，**对于每个神经元来说，他可能会随机被关闭而见不到一些样本，这使得每个神经元的更新并不依赖于某些特定的样本，因而有效降低模型的过度拟合，提高模型的泛化能力**，从而提升测试集的准确率。

- 而对于 CNN 模型，在特征提取方面已经使用了卷积操作和池化操作来提取图像中的局部特征，这些局部特征的关联性较强，并不会像全连接层那样出现神经元之间的强依赖关系。因此，在CNN模型上使用Dropout层，可能削弱了特征之间的联系，导致模型在测试集上的准确率有所下降。
- 另一个有趣的现象是CNN模型在去掉dropout层后不管是loss还是accuracy曲线波动幅度都明显变大，训练明显变得更不稳定。对此的解释是**去掉Dropout层后，由于每个神经元都会参与到每次训练中，网络的更新可能会更依赖于一些特别的神经元，这些神经元在某些样本产生较大的梯度时，会导致loss曲线和accuracy曲线发生明显的波动**。值得注意的是，正是 60-70 batch之间的一次抖动，使得validation集合上无dropout层取得了更高的准确率，因此这种训练中的不稳定性也可能带来正向的作用。

Batch Norm

从准确率来看，Batch Norm 层使CNN在测试集的准确率提升了5.5%，但使MLP在测试集的准确率下降了2.6%。分析可能的原因为：

- 对于CNN模型，由于每个卷积层都有一个Batch Norm层，**这些 Batch Norm 层可以减少数据统计分布的变化，从而使得每个卷积层的训练更加稳定**，提高了模型的泛化性能。因此，使用BN层可以显著提高CNN模型在测试集上的准确率。
- 但对于MLP模型而言， Batch Norm 层并没有提升模型的能力，进一步的研究（下图）表明，随着步数的增加，使用 Batch norm 层反而使得模型过拟合的问题更加严重。可能的一个解释是：Batch Norm 允许模型更深，更广，具有更多的参数。这意味着模型可以更轻松地记住训练数据的细节，而不是仅仅泛化到数据中的一般模式。如果超参数设置不当，模型更容易产生过拟合。



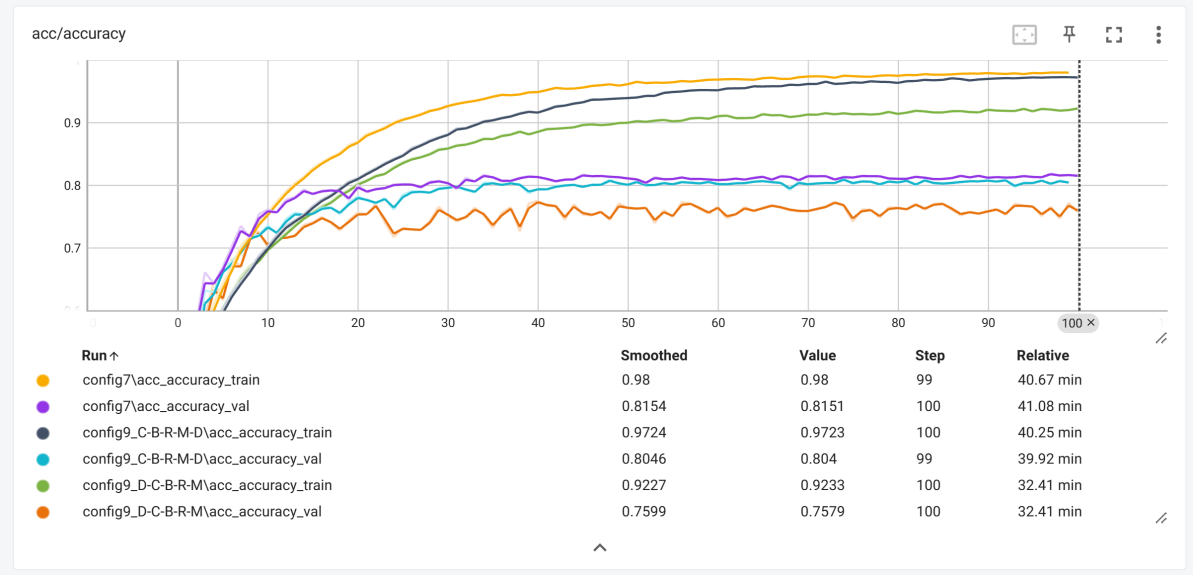
其它研究

CNN中网络模块顺序对训练的影响

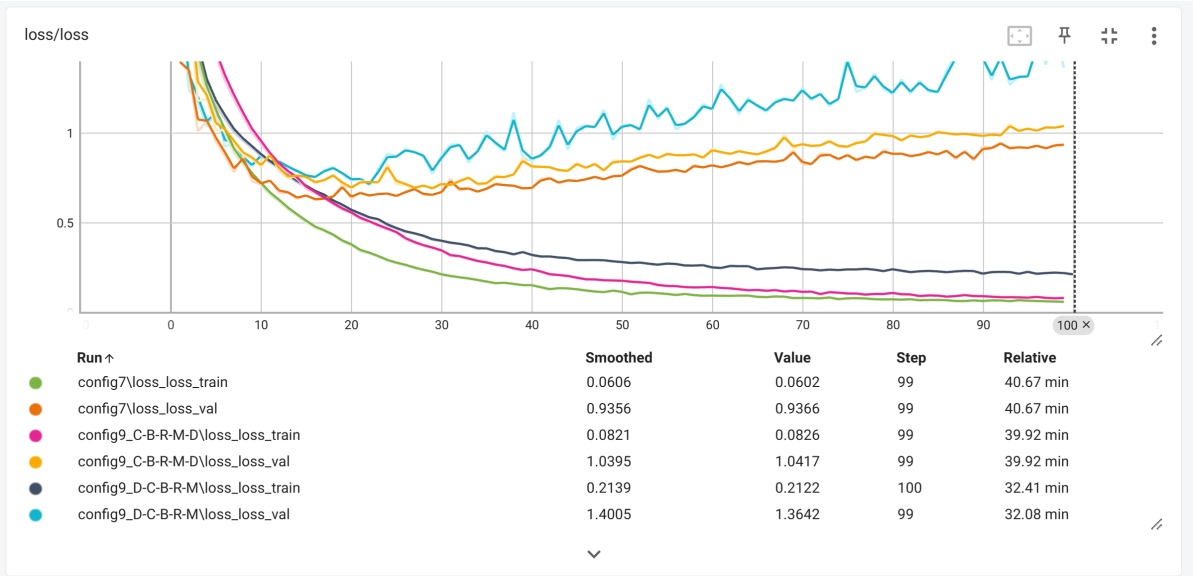
考虑到 CNN 和 MaxPool 层顺序调换会改变原有顺序下的通道结构，进而改变模型参数，可能导致实验引入其它影响因素。因此本节实验仅改变了 Dropout层和 Batch Norm层的位置，设计两组实验探究对模型训练产生的影响。对于本节中的实验，Default 结构为前文CNN所用的 C-B-R-D-M 结构（Conv-Batchnorm-Relu-Dropout-Maxpool），所有超参数也与前文保持一致。

Dropout

基于 Dropout，我设计了 C-B-R-D-M（Default），C-B-R-M-D，D-C-B-R-M 三种网络结构。训练在 train 集合和 validation 集合上的 loss 与 accuracy 如下



Run	Smoothed	Value	Step	Relative
config7\acc_accuracy_train	0.98	0.98	99	40.67 min
config7\acc_accuracy_val	0.8154	0.8151	100	41.08 min
config9_C-B-R-M-D\acc_accuracy_train	0.9724	0.9723	100	40.25 min
config9_C-B-R-M-D\acc_accuracy_val	0.8046	0.804	99	39.92 min
config9_D-C-B-R-M\acc_accuracy_train	0.9227	0.9233	100	32.41 min
config9_D-C-B-R-M\acc_accuracy_val	0.7599	0.7579	100	32.41 min



Run	Smoothed	Value	Step	Relative
config7\loss_loss_train	0.0606	0.0602	99	40.67 min
config7\loss_loss_val	0.9356	0.9366	99	40.67 min
config9_C-B-R-M-D\loss_loss_train	0.0821	0.0826	99	39.92 min
config9_C-B-R-M-D\loss_loss_val	1.0395	1.0417	99	39.92 min
config9_D-C-B-R-M\loss_loss_train	0.2139	0.2122	100	32.41 min
config9_D-C-B-R-M\loss_loss_val	1.4005	1.3642	99	32.08 min

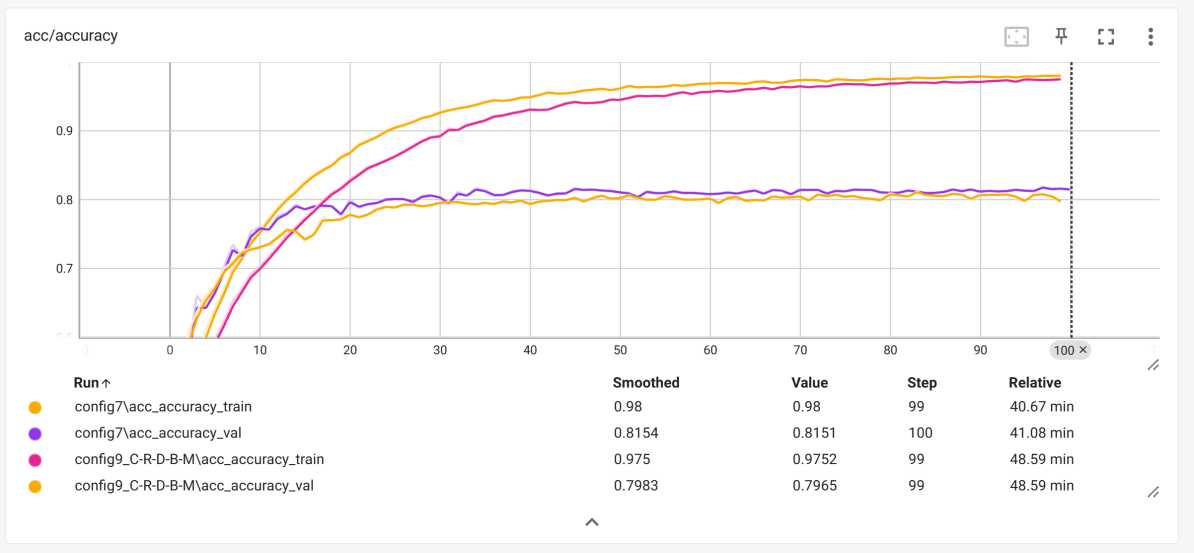
三者最终在 Test 集合上的表现为

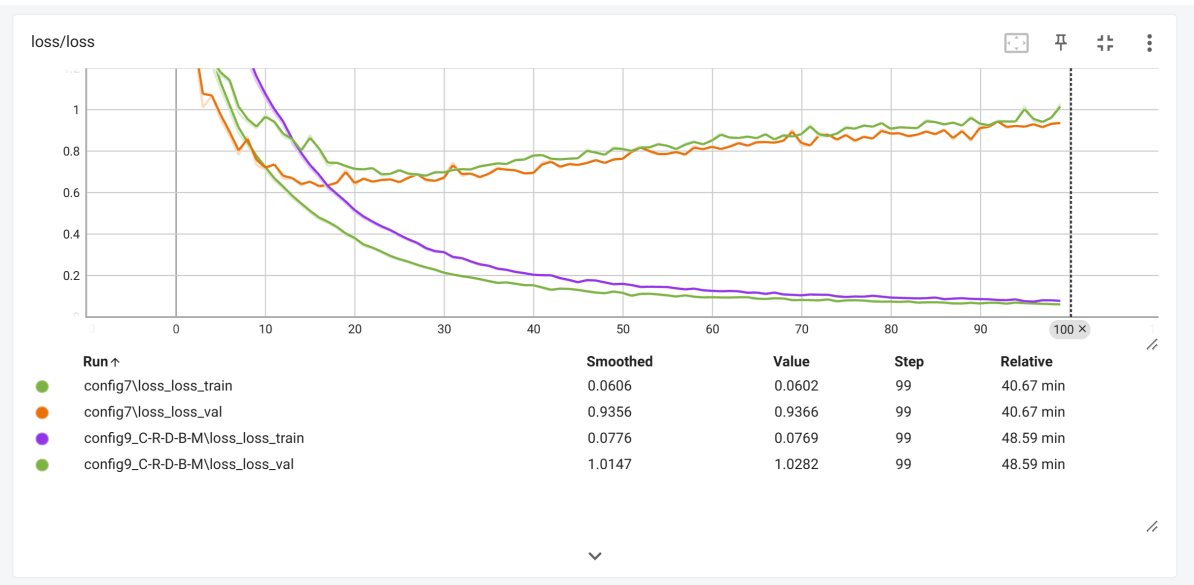
	C-B-R-D-M（Default）	C-B-R-M-D	D-C-B-R-M
Accuracy(%)	80.7%	79.9%	77.0%
Time-100 epochs(min)	41	40	32

- 实验结果表明，Dropout层在 Maxpool 层前后不会对训练产生太大的影响，二者的 loss 和 accuracy 都十分接近，训练时长也相对接近。这是因为 Maxpool 本质上是没有可训参数的，maxpooling 通道与通道之间也不存在信息交换，对基于通道对齐的 Dropout2D 策略来说，二者应该是等价的，因此表现出相似的训练效果
- 和默认结构相比，Dropout 层前置到卷积层之前会使训练效果变差，在测试集上准确率降低了3%左右。一种可能的解释为：dropout本质上是对“神经元”的遗忘作用，倘若将 Dropout置于卷积层之前，会导致其不能有效掩盖神经元层的信息，倘若只看第一层的话，此时Dropout掩盖的是原始图片的信息，这本质上就变为了一种数据增强的策略，而非 Dropout 所要做得 “使神经元遗忘” 的目的
- 一个有趣的观察是，D-C-B-R-M 的训练用时明显更短，这是因为 Dropout 前置后作用的通道数变少，在用伯努利分布采样时用时更短

Batch Norm

基于 Batch Norm，我设计了 C-B-R-D-M（Default），C-R-D-B-M 两种网络结构。训练在 train 集合和 validation 集合上的 loss 与 accuracy 如下





二者最终在 Test 集合上的表现为

	C-B-R-D-M（Default）	C-R-D-B-M
Accuracy(%)	80.7%	79.9%
Time-100 epochs(min)	41	49

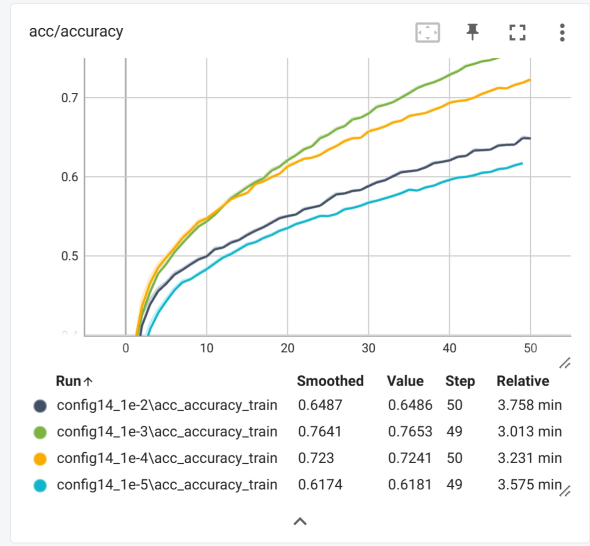
- 实验结果表明，C-B-R-D-M 和 C-R-D-B-M 两种网络结构，训练效果上后者稍弱于前者（test集合上低0.8%）但总体而言差距不大，推测可能的原因为，Relu 层和 Dropout 层并没有明显改变数据集的分布，因此在 BatchNorm 中计算出的均值和方差也没有产生太大的差异，因此对整体训练效果并没有产生显著的影响。

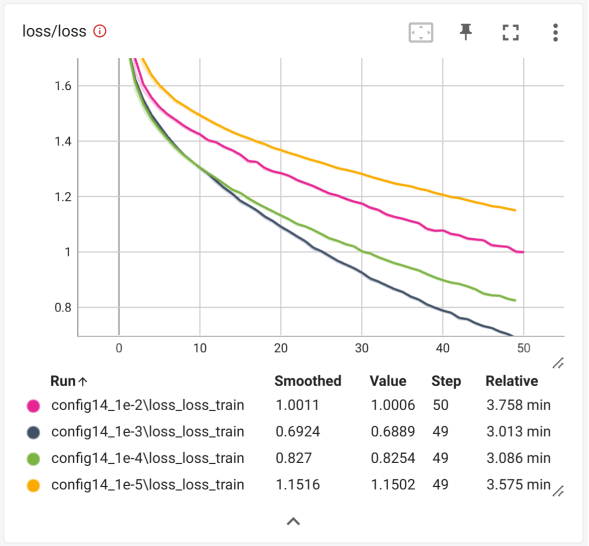
Learning rate && Dropout rate 对训练的影响

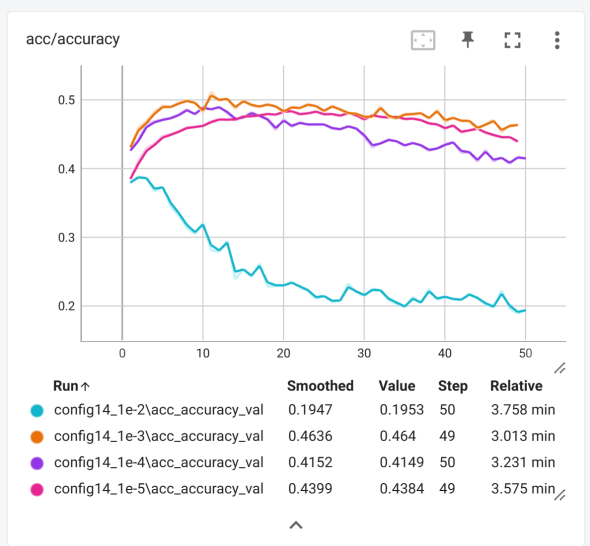
本节中，我以 MLP 模型为例，探究 Learning rate 和 Dropout rate 对模型训练的影响，除探究的参数外，其余超参数和网络结构设定均与前文一致

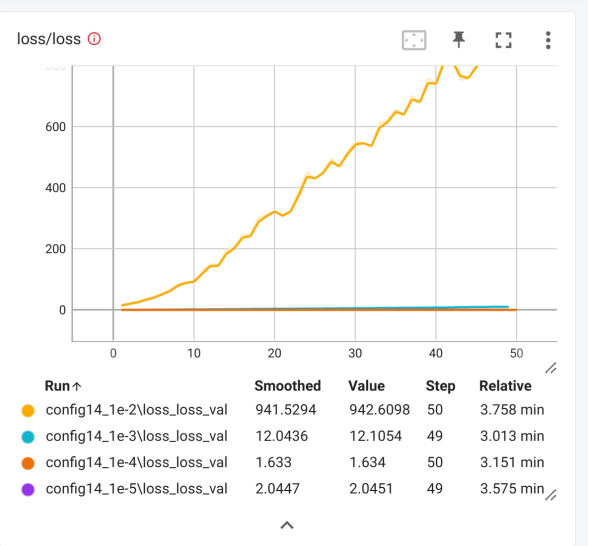
Learning rate

本节中，我取 learning rate=1e-5,1e-4,1e-3,1e-2，探究 Learning rate 对模型训练的影响，实验结果如下





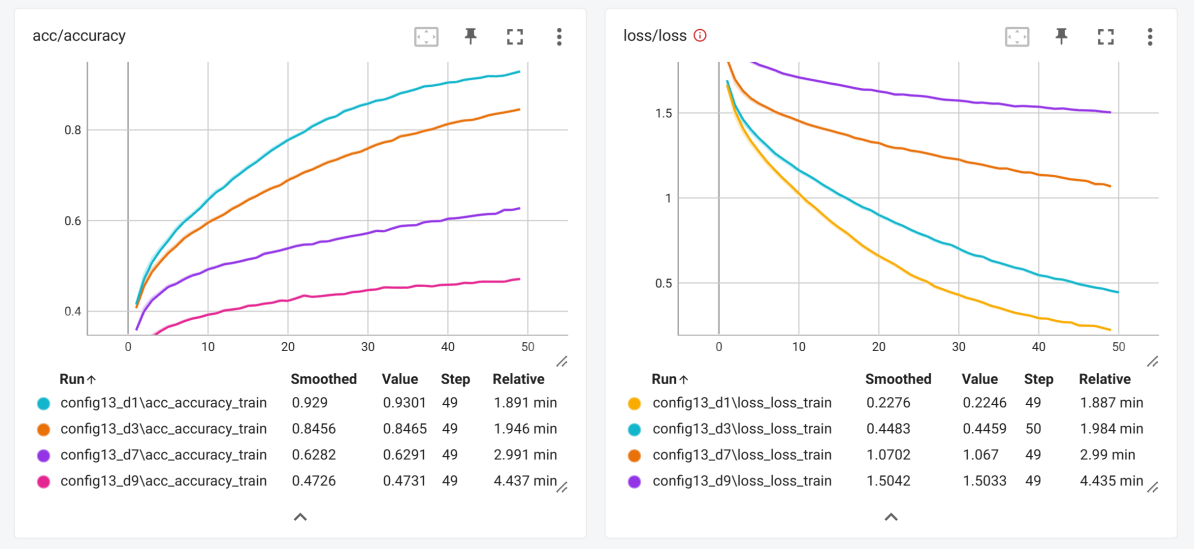


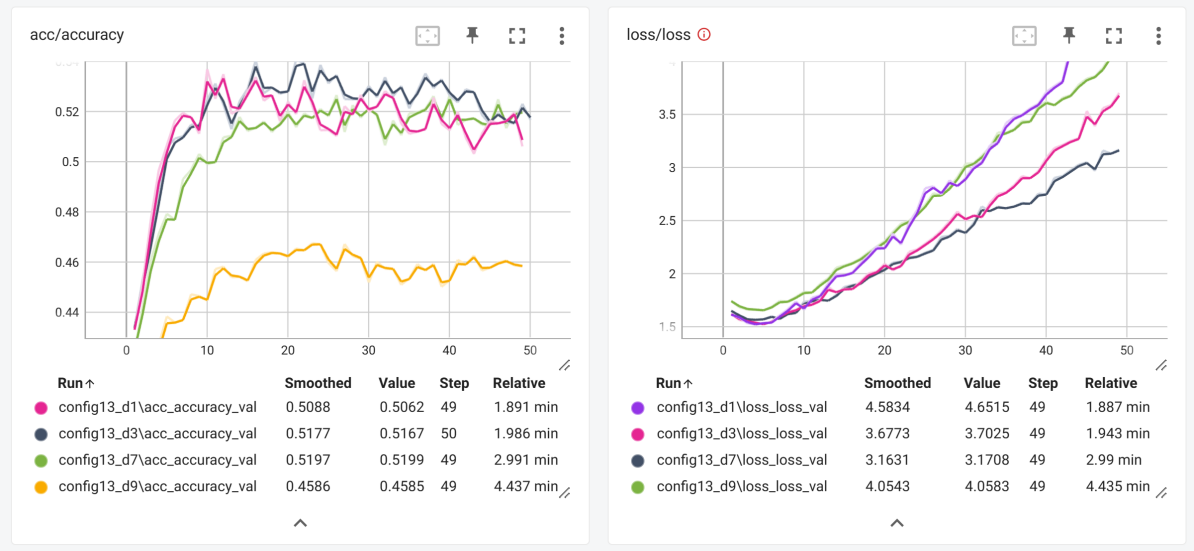


- 在1e-3到1e-5区间，随着学习率的增大，模型在训练集上的loss下降/accuracy上升速度加快，幅度也变大，但在validation上，三者准确率最大值没有明显差异，且模型学习率越大，其越早出现过拟合现象。这是因为模型学习率增大，其单次梯度更新幅度也越大，因此在训练集上曲线斜率也较大，但仅仅改变学习率并不能避免过拟合问题的出现，学习率越大的模型，越快的学得训练集中的特征，也就越早出现过拟合问题
- 在 $learning\_rate = 1e-2$ 时，模型虽然在训练集上仍表现正常，但在测试集上其 accuracy 和 loss 均显著低于/高于其它几组实验，这是因为 learning rate 过大后，模型之前的防止过拟合策略完全失效，且模型会以极快的速度学会训练集的特征（注意观察在2epochs是validation准确率达到最大），早早出现过拟合的现象。

Dropout rate

本节中，我取 Dropout rate=0.1,0.3,0.7,0.9，探究 Dropout rate 对模型训练的影响，实验结果如下





- 在train集合上，随着 Dropout rate 的增加，模型收敛时的 accuracy 变小，loss 变大，说明在训练集合上的拟合程度逐渐变低，这是因为 Dropout rate 的增加会促使更多的神经元“忘掉”前面的信息，因此对训练集的学习程度也会随之下降
- 造 0.1-0.7 区间， Dropout rate 增加会使模型在validation 集上收敛速度变慢，且过拟合现象出现的位置推后，甚至在 Dropout rate = 0.7 时，50 epochs 内模型都没有出现明显的过拟合现象，这是因为随着 Dropout rate的增加，模型在训练时对 train 数据集遗忘的信息增多，神经元之间的强依赖关系减弱，更不容易出现过拟合现象
- 另一个值得注意的现象是随着 Dropout rate 增加，模型在 validation 集上的波动幅度变小，这是因为随着网络遗忘更多的神经元，对每个神经元来说，其梯度受特定样本更新影响的概率降低，而更可能收到一类样本平均的影响，因此从整个网络来看，网络更不容易出现随着某几个样本权重发生剧烈变化的情况，能够在 validation 集合上表现得更加平稳
- 随着 Dropout rate 达到0.9，网络在训练集和验证集上的表现均很差，这是因为 Dropout rate 过高会影响模型的学习能力，使其无法有效学到训练集中的知识，最终导致模型性能下降。