Exercise 1

Glorot初始化和He初始化所要解决的问题是什么？

答案：

Glorot初始化和He初始化被设计为至少在训练开始时使输出标准偏差尽可能接近输入标准偏差。这减少了消失/爆炸梯度的问题。

Exercise 2

是否可以将所有的权重初始化为相同的值，只要该值是使用 He 初始化随机选择的？

答案：

不可以，所有权重应单独取样；它们不应该都具有相同的初始值。随机采样权重的一个重要目标是打破对称性：如果所有权重都具有相同的初始值，即使该值不为零，那么对称性不会被打破（即，给定层中的所有神经元都是等价的），反向传播将无法打破对称性。具体地说，这意味着任何给定层中所有神经元都将始终具有相同的权重。这就像每层只有一个神经元，而且速度要慢得多。这样的配置几乎不可能收敛到一个好的解决方案。

Exercise 3

可以将偏差项初始化为0吗？

答案：

将偏置项初始化为零是非常好的。有些人喜欢像权重一样初始化它们，这也没问题；这没有多大区别。

Exerice 4

在哪种情况下，您希望使用我们在本章中讨论的每个激活函数？

答案：

ReLU通常是隐藏层的良好默认值，因为它速度快，效果好。在某些情况下，其精确输出零的能力也很有用（例如，参见第17章）。此外，它有时可以从优化的实现以及硬件加速中受益。与ReLU相比，ReLU的leaky ReLU变体可以提高模型的质量，而不会过多地阻碍其速度。对于大型神经网络和更复杂的问题，GLU、Swish和Mish可以为您提供质量稍高的模型，但它们有计算成本。双曲正切（tanh）在输出层中非常有用，如果您需要输出固定范围内的数字（默认值在-1和1之间），但现在它在隐藏层中不常用，除非在递归网络中。当您需要估计概率（例如，用于二进制分类）时，S形激活函数在输出层也很有用，但它很少用于隐藏层（例如，对于变分自动编码器的编码层，有例外；参见第17章）。当您需要确保输出始终为正时，softplus激活功能在输出层中非常有用。softmax激活函数在输出层中用于估计互斥类的概率，但在隐藏层中很少使用（如果有的话）。

Exercise 5

当你使用SGD优化器时，如果你将动量超参数设置得太接近1（例如，0.99999），会发生什么？

答案：

如果在使用SGD优化器时将动量超参数设置得太接近1（例如，0.99999），那么算法可能会加快速度，希望大致向全局最小值移动，但其动量将使其刚好超过最小值。然后它会减速并返回，再次加速，再次超调，等等。在收敛之前，它可能会以这种方式振荡多次，因此总的来说，与较小动量值相比，收敛所需的时间要长得多。

Exercise 6

列出可以生成稀疏模型的三种方法。

答案：

生成稀疏模型（即，大多数权重等于零）的一种方法是正常训练模型，然后将极小的权重归零。要获得更多稀疏性，可以应用 ℓ_1 训练期间的正则化，这将优化器推向稀疏。第三种选择是使用TensorFlow模型优化工具包。

Exercise 7

辍学生会减慢训练的速度吗？它是否减缓推理（例如，对新实例做出预测）？那MC dropout呢？

答案：

是的，dropout确实会减慢训练速度，一般来说大约是2倍。然而，它对推理速度没有影响，因为它只在训练期间打开。MCdropout与训练期间的dropout完全相同，但在推理过程中它仍然活跃，因此每次推理都会稍微放慢。更重要的是，当使用MC Dropout时，您通常希望运行10次或更多次推断以获得更好的预测。这意味着做出预测的速度会减慢10倍或更多。

Exercise 8

在CIFAR10图像数据集上练习训练深度神经网络：

1. 构建一个包含20个隐藏层的100个神经元的DNN（这太多了，但这是这个练习的重点）。使用He初始化和Swish激活函数。
2. 利用Nadam优化和早期停止，在CIFAR10数据集上对网络进行训练。您可以使用tf.keras.datasets.cifar10.load_ data()来加载它。数据集由60,000个32 ×32像素彩色图像（50000用于训练，10000用于测试）和10个类组成，所以你需要一个包含10个神经元的softmax输出层。请记住，每次更改模型的架构或超参数时，都要搜索正确的学习速率。
3. 现在尝试添加批处理归一化并比较学习曲线：它的收敛速度比以前快了吗？它能产生一个更好的模型吗？它会如何影响训练速度呢？
4. 尝试用SELU替换批标准化，并进行必要的调整，确保网络自规范化（即标准化输入功能，使用LeCun正常初始化，确保DNN只包含一系列密集层，等）。
5. 尝试用alpha dropout来正则化模型。然后，在不重新训练你的模型的情况下，看看你是否可以使用MC dropout来获得更好的准确性。
6. 使用1cycle调度重新训练你的模型，看看它是否提高了训练速度和模型精度。

In [8]:
```
import tensorflow as tf
```

```
from matplotlib import pyplot as plt
import numpy as np
from pathlib import Path
from time import strftime
import tensorboard
```

8.1

In [3]:
```
tf.random.set_seed(42)

model = tf.keras.Sequential()
model.add(tf.keras.layers.Flatten(input_shape=[32, 32, 3]))
for _ in range(20):
    model.add(tf.keras.layers.Dense(100,activation="swish",
                                    kernel_initializer="he_normal"))
```

8.2

In [4]:
```
# Let's add the output layer to the model:
model.add(tf.keras.layers.Dense(10, activation="softmax"))
```

让我们使用学习率为5e-5的Nadam优化器。我尝试了1e-5、3e-5、1e-4、3e-4、1e-3、3e-3和1e-2的学习率，并比较了各自10个时期的学习曲线（使用下面的TensorBoard回调）。3e-5和1e-4的学习率相当不错，所以我尝试了5e-5，结果稍微好一点。

In [5]:
```
optimizer = tf.keras.optimizers.Nadam(learning_rate=5e-5)

model.compile(loss="sparse_categorical_crossentropy",
              optimizer=optimizer,
              metrics=["accuracy"])
```

In [6]:
```
# load the CIFAR10 dataset

cifar10 = tf.keras.datasets.cifar10.load_data()
(X_train_full, y_train_full), (X_test, y_test) = cifar10

X_train = X_train_full[5000:]
y_train = y_train_full[5000:]
X_valid = X_train_full[:5000]
y_valid = y_train_full[:5000]
```

```
Downloading data from https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz
170498071/170498071 [==============================] - 65s 0us/step
```

In [11]:
```
# create the callbacks we need and train the model

early_stopping_cb = tf.keras.callbacks.EarlyStopping(patience=20,
                                                     restore_best_weights=True)

model_checkpoint_cb = tf.keras.callbacks.ModelCheckpoint("my_cifar10_model",
                                                         save_best_only=True)

run_index = 1 # increment every time you train the model

run_logdir = Path() / "my_cifar10_logs" / f"run_{run_index:03d}"

tensorboard_cb = tf.keras.callbacks.TensorBoard(run_logdir)

callbacks = [early_stopping_cb, model_checkpoint_cb, tensorboard_cb]
```

In [10]:
```
%load_ext tensorboard
%tensorboard --logdir=./my_cifar10_logs
```

In [12]:
```
model.fit(X_train, y_train, epochs=100,
          validation_data=(X_valid, y_valid),
          callbacks=callbacks)
```

```
Epoch 1/100
1400/1407 [============================>.] - ETA: 0s - loss: 4.5069 - accuracy: 0.1447IN
FO:tensorflow:Assets written to: my_cifar10_model\assets
1407/1407 [==============================] - 15s 9ms/step - loss: 4.4966 - accuracy: 0.1
450 - val_loss: 2.1814 - val_accuracy: 0.2132
Epoch 2/100
1407/1407 [==============================] - 9s 7ms/step - loss: 2.0899 - accuracy: 0.22
96 - val_loss: 2.3673 - val_accuracy: 0.1808
```

```
Epoch 3/100
1407/1407 [==============================] - ETA: 0s - loss: 1.9683 - accuracy: 0.2745IN
FO:tensorflow:Assets written to: my_cifar10_model\assets
1407/1407 [==============================] - 12s 8ms/step - loss: 1.9683 - accuracy: 0.2
745 - val_loss: 2.0644 - val_accuracy: 0.2576
Epoch 4/100
1405/1407 [============================>.] - ETA: 0s - loss: 1.8869 - accuracy: 0.3079IN
FO:tensorflow:Assets written to: my_cifar10_model\assets
1407/1407 [==============================] - 12s 9ms/step - loss: 1.8869 - accuracy: 0.3
079 - val_loss: 1.8561 - val_accuracy: 0.3216
Epoch 5/100
1407/1407 [==============================] - ETA: 0s - loss: 1.8235 - accuracy: 0.3296IN
FO:tensorflow:Assets written to: my_cifar10_model\assets
1407/1407 [==============================] - 12s 9ms/step - loss: 1.8235 - accuracy: 0.3
296 - val_loss: 1.8161 - val_accuracy: 0.3418
Epoch 6/100
1403/1407 [============================>.] - ETA: 0s - loss: 1.7760 - accuracy: 0.3529IN
FO:tensorflow:Assets written to: my_cifar10_model\assets
1407/1407 [==============================] - 12s 9ms/step - loss: 1.7758 - accuracy: 0.3
531 - val_loss: 1.7659 - val_accuracy: 0.3640
Epoch 7/100
1407/1407 [==============================] - ETA: 0s - loss: 1.7308 - accuracy: 0.3729IN
FO:tensorflow:Assets written to: my_cifar10_model\assets
1407/1407 [==============================] - 13s 9ms/step - loss: 1.7308 - accuracy: 0.3
729 - val_loss: 1.7393 - val_accuracy: 0.3594
Epoch 8/100
1407/1407 [==============================] - ETA: 0s - loss: 1.6952 - accuracy: 0.3913IN
FO:tensorflow:Assets written to: my_cifar10_model\assets
1407/1407 [==============================] - 13s 9ms/step - loss: 1.6952 - accuracy: 0.3
913 - val_loss: 1.6996 - val_accuracy: 0.3930
Epoch 9/100
1403/1407 [============================>.] - ETA: 0s - loss: 1.6597 - accuracy: 0.4042IN
FO:tensorflow:Assets written to: my_cifar10_model\assets
1407/1407 [==============================] - 13s 9ms/step - loss: 1.6596 - accuracy: 0.4
044 - val_loss: 1.6618 - val_accuracy: 0.4052
Epoch 10/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.6314 - accuracy: 0.4
133 - val_loss: 1.7114 - val_accuracy: 0.3758
Epoch 11/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.6043 - accuracy: 0.4
258 - val_loss: 1.6912 - val_accuracy: 0.3870
Epoch 12/100
1405/1407 [============================>.] - ETA: 0s - loss: 1.5798 - accuracy: 0.4338IN
FO:tensorflow:Assets written to: my_cifar10_model\assets
1407/1407 [==============================] - 13s 9ms/step - loss: 1.5796 - accuracy: 0.4
339 - val_loss: 1.6257 - val_accuracy: 0.4102
Epoch 13/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.5605 - accuracy: 0.4
405 - val_loss: 1.6300 - val_accuracy: 0.4196
Epoch 14/100
1406/1407 [============================>.] - ETA: 0s - loss: 1.5453 - accuracy: 0.4459IN
FO:tensorflow:Assets written to: my_cifar10_model\assets
1407/1407 [==============================] - 13s 9ms/step - loss: 1.5453 - accuracy: 0.4
459 - val_loss: 1.5948 - val_accuracy: 0.4202
Epoch 15/100
1405/1407 [============================>.] - ETA: 0s - loss: 1.5249 - accuracy: 0.4538IN
FO:tensorflow:Assets written to: my_cifar10_model\assets
1407/1407 [==============================] - 13s 9ms/step - loss: 1.5250 - accuracy: 0.4
538 - val_loss: 1.5681 - val_accuracy: 0.4408
Epoch 16/100
1403/1407 [============================>.] - ETA: 0s - loss: 1.5065 - accuracy: 0.4611IN
FO:tensorflow:Assets written to: my_cifar10_model\assets
1407/1407 [==============================] - 13s 9ms/step - loss: 1.5066 - accuracy: 0.4
610 - val_loss: 1.5648 - val_accuracy: 0.4422
Epoch 17/100
1400/1407 [============================>.] - ETA: 0s - loss: 1.4910 - accuracy: 0.4658IN
```

```
FO:tensorflow:Assets written to: my_cifar10_model\assets
1407/1407 [==============================] - 13s 9ms/step - loss: 1.4912 - accuracy: 0.4
658 - val_loss: 1.5417 - val_accuracy: 0.4380
Epoch 18/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.4752 - accuracy: 0.4
716 - val_loss: 1.5858 - val_accuracy: 0.4364
Epoch 19/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.4624 - accuracy: 0.4
779 - val_loss: 1.5496 - val_accuracy: 0.4492
Epoch 20/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.4478 - accuracy: 0.4
800 - val_loss: 1.5572 - val_accuracy: 0.4442
Epoch 21/100
1403/1407 [============================>.] - ETA: 0s - loss: 1.4369 - accuracy: 0.4857IN
FO:tensorflow:Assets written to: my_cifar10_model\assets
1407/1407 [==============================] - 13s 9ms/step - loss: 1.4371 - accuracy: 0.4
856 - val_loss: 1.5394 - val_accuracy: 0.4462
Epoch 22/100
1406/1407 [============================>.] - ETA: 0s - loss: 1.4254 - accuracy: 0.4920IN
FO:tensorflow:Assets written to: my_cifar10_model\assets
1407/1407 [==============================] - 13s 9ms/step - loss: 1.4255 - accuracy: 0.4
920 - val_loss: 1.5204 - val_accuracy: 0.4600
Epoch 23/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.4128 - accuracy: 0.4
957 - val_loss: 1.5315 - val_accuracy: 0.4524
Epoch 24/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.4017 - accuracy: 0.4
957 - val_loss: 1.5572 - val_accuracy: 0.4390
Epoch 25/100
1405/1407 [============================>.] - ETA: 0s - loss: 1.3893 - accuracy: 0.5007IN
FO:tensorflow:Assets written to: my_cifar10_model\assets
1407/1407 [==============================] - 13s 9ms/step - loss: 1.3893 - accuracy: 0.5
008 - val_loss: 1.5085 - val_accuracy: 0.4638
Epoch 26/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.3769 - accuracy: 0.5
054 - val_loss: 1.5472 - val_accuracy: 0.4524
Epoch 27/100
1405/1407 [============================>.] - ETA: 0s - loss: 1.3669 - accuracy: 0.5106IN
FO:tensorflow:Assets written to: my_cifar10_model\assets
1407/1407 [==============================] - 13s 9ms/step - loss: 1.3669 - accuracy: 0.5
106 - val_loss: 1.4853 - val_accuracy: 0.4700
Epoch 28/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.3561 - accuracy: 0.5
166 - val_loss: 1.5107 - val_accuracy: 0.4550
Epoch 29/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.3519 - accuracy: 0.5
138 - val_loss: 1.4922 - val_accuracy: 0.4742
Epoch 30/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.3365 - accuracy: 0.5
176 - val_loss: 1.5701 - val_accuracy: 0.4538
Epoch 31/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.3291 - accuracy: 0.5
234 - val_loss: 1.4900 - val_accuracy: 0.4728
Epoch 32/100
1407/1407 [==============================] - ETA: 0s - loss: 1.3199 - accuracy: 0.5257IN
FO:tensorflow:Assets written to: my_cifar10_model\assets
1407/1407 [==============================] - 13s 9ms/step - loss: 1.3199 - accuracy: 0.5
257 - val_loss: 1.4817 - val_accuracy: 0.4840
Epoch 33/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.3104 - accuracy: 0.5
279 - val_loss: 1.5388 - val_accuracy: 0.4668
Epoch 34/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.2994 - accuracy: 0.5
321 - val_loss: 1.5463 - val_accuracy: 0.4618
Epoch 35/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.2970 - accuracy: 0.5
```

```
344 - val_loss: 1.5056 - val_accuracy: 0.4732
Epoch 36/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.2881 - accuracy: 0.5
381 - val_loss: 1.4993 - val_accuracy: 0.4704
Epoch 37/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.2792 - accuracy: 0.5
406 - val_loss: 1.4998 - val_accuracy: 0.4884
Epoch 38/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.2684 - accuracy: 0.5
457 - val_loss: 1.4970 - val_accuracy: 0.4684
Epoch 39/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.2610 - accuracy: 0.5
477 - val_loss: 1.4943 - val_accuracy: 0.4760
Epoch 40/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.2542 - accuracy: 0.5
485 - val_loss: 1.5132 - val_accuracy: 0.4640
Epoch 41/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.2475 - accuracy: 0.5
504 - val_loss: 1.4972 - val_accuracy: 0.4802
Epoch 42/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.2371 - accuracy: 0.5
550 - val_loss: 1.5273 - val_accuracy: 0.4726
Epoch 43/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.2322 - accuracy: 0.5
561 - val_loss: 1.5460 - val_accuracy: 0.4680
Epoch 44/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.2262 - accuracy: 0.5
595 - val_loss: 1.5675 - val_accuracy: 0.4596
Epoch 45/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.2183 - accuracy: 0.5
634 - val_loss: 1.5325 - val_accuracy: 0.4716
Epoch 46/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.2102 - accuracy: 0.5
633 - val_loss: 1.5001 - val_accuracy: 0.4804
Epoch 47/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.1987 - accuracy: 0.5
701 - val_loss: 1.5032 - val_accuracy: 0.4786
Epoch 48/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.1951 - accuracy: 0.5
696 - val_loss: 1.5194 - val_accuracy: 0.4690
Epoch 49/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.1872 - accuracy: 0.5
745 - val_loss: 1.5661 - val_accuracy: 0.4628
Epoch 50/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.1814 - accuracy: 0.5
768 - val_loss: 1.5048 - val_accuracy: 0.4784
Epoch 51/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.1718 - accuracy: 0.5
787 - val_loss: 1.5201 - val_accuracy: 0.4788
Epoch 52/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.1668 - accuracy: 0.5
803 - val_loss: 1.5369 - val_accuracy: 0.4602
```

```
Out[12]: <keras.callbacks.History at 0x1783a861400>
```

```
In [13]: model.evaluate(X_valid, y_valid)
```

```
157/157 [==============================] - 0s 2ms/step - loss: 1.4817 - accuracy: 0.4840
Out[13]: [1.4817038774490356, 0.48399999737739563]
```

验证损失最小的模型在验证集上的准确度约为46.8%。它花费了29个时期达到最低的验证丢失，在我的笔记本电脑（没有GPU）上每个时期大约10秒。让我们看看是否可以使用批处理规范化来改进模型。

8.3

下面的代码与上面的代码非常相似，只做了一些更改：

1. 我在每个密集层之后（激活功能之前）添加了BN层，除了输出层。
2. 我将学习率改为5e-4。我用1e-5、3e-5、5e-5、1e-4、3e-4、5e-4、1e-3和3e-3进行了实验，我选择了20个时期后验证性能最好的一个。
3. 我将运行目录重命名为run*bn**，将模型文件名重命名为my_cifar10_bn_mode。

In [14]:
```
tf.random.set_seed(42)

model = tf.keras.Sequential()
model.add(tf.keras.layers.Flatten(input_shape=[32, 32, 3]))
for _ in range(20):
    model.add(tf.keras.layers.Dense(100, kernel_initializer="he_normal"))
    model.add(tf.keras.layers.BatchNormalization())
    model.add(tf.keras.layers.Activation("swish"))

model.add(tf.keras.layers.Dense(10, activation="softmax"))

optimizer = tf.keras.optimizers.Nadam(learning_rate=5e-4)
model.compile(loss="sparse_categorical_crossentropy",
              optimizer=optimizer,
              metrics=["accuracy"])

early_stopping_cb = tf.keras.callbacks.EarlyStopping(patience=20,
                                                     restore_best_weights=True)
model_checkpoint_cb = tf.keras.callbacks.ModelCheckpoint("my_cifar10_bn_model",
                                                         save_best_only=True)
run_index = 1 # increment every time you train the model
run_logdir = Path() / "my_cifar10_logs" / f"run_bn_{run_index:03d}"
tensorboard_cb = tf.keras.callbacks.TensorBoard(run_logdir)
callbacks = [early_stopping_cb, model_checkpoint_cb, tensorboard_cb]

model.fit(X_train, y_train, epochs=100,
          validation_data=(X_valid, y_valid),
          callbacks=callbacks)

model.evaluate(X_valid, y_valid)
```

```
Epoch 1/100
1405/1407 [============================>.] - ETA: 0s - loss: 2.0331 - accuracy: 0.2619IN
FO:tensorflow:Assets written to: my_cifar10_bn_model\assets
1407/1407 [==============================] - 29s 16ms/step - loss: 2.0329 - accuracy: 0.
2620 - val_loss: 1.8883 - val_accuracy: 0.3350
Epoch 2/100
1401/1407 [============================>.] - ETA: 0s - loss: 1.7749 - accuracy: 0.3628IN
FO:tensorflow:Assets written to: my_cifar10_bn_model\assets
1407/1407 [==============================] - 22s 15ms/step - loss: 1.7747 - accuracy: 0.
3628 - val_loss: 1.7966 - val_accuracy: 0.3562
Epoch 3/100
1405/1407 [============================>.] - ETA: 0s - loss: 1.6731 - accuracy: 0.4052IN
FO:tensorflow:Assets written to: my_cifar10_bn_model\assets
1407/1407 [==============================] - 22s 15ms/step - loss: 1.6731 - accuracy: 0.
4054 - val_loss: 1.7831 - val_accuracy: 0.3686
Epoch 4/100
1404/1407 [============================>.] - ETA: 0s - loss: 1.6049 - accuracy: 0.4265IN
FO:tensorflow:Assets written to: my_cifar10_bn_model\assets
1407/1407 [==============================] - 21s 15ms/step - loss: 1.6049 - accuracy: 0.
4265 - val_loss: 1.6593 - val_accuracy: 0.4116
Epoch 5/100
1407/1407 [==============================] - ETA: 0s - loss: 1.5463 - accuracy: 0.4493IN
FO:tensorflow:Assets written to: my_cifar10_bn_model\assets
1407/1407 [==============================] - 21s 15ms/step - loss: 1.5463 - accuracy: 0.
4493 - val_loss: 1.6295 - val_accuracy: 0.4292
```

```
Epoch 6/100
1407/1407 [==============================] - 15s 10ms/step - loss: 1.5001 - accuracy: 0.
4699 - val_loss: 1.8032 - val_accuracy: 0.3732
Epoch 7/100
1407/1407 [==============================] - 14s 10ms/step - loss: 1.4546 - accuracy: 0.
4844 - val_loss: 1.6868 - val_accuracy: 0.4172
Epoch 8/100
1406/1407 [============================>.] - ETA: 0s - loss: 1.4156 - accuracy: 0.4980IN
FO:tensorflow:Assets written to: my_cifar10_bn_model\assets
1407/1407 [==============================] - 21s 15ms/step - loss: 1.4156 - accuracy: 0.
4980 - val_loss: 1.5918 - val_accuracy: 0.4430
Epoch 9/100
1405/1407 [============================>.] - ETA: 0s - loss: 1.3848 - accuracy: 0.5106IN
FO:tensorflow:Assets written to: my_cifar10_bn_model\assets
1407/1407 [==============================] - 21s 15ms/step - loss: 1.3847 - accuracy: 0.
5106 - val_loss: 1.5285 - val_accuracy: 0.4546
Epoch 10/100
1405/1407 [============================>.] - ETA: 0s - loss: 1.3548 - accuracy: 0.5233IN
FO:tensorflow:Assets written to: my_cifar10_bn_model\assets
1407/1407 [==============================] - 21s 15ms/step - loss: 1.3550 - accuracy: 0.
5232 - val_loss: 1.4800 - val_accuracy: 0.4656
Epoch 11/100
1403/1407 [============================>.] - ETA: 0s - loss: 1.3282 - accuracy: 0.5298IN
FO:tensorflow:Assets written to: my_cifar10_bn_model\assets
1407/1407 [==============================] - 21s 15ms/step - loss: 1.3281 - accuracy: 0.
5298 - val_loss: 1.4449 - val_accuracy: 0.4844
Epoch 12/100
1407/1407 [==============================] - 14s 10ms/step - loss: 1.2986 - accuracy: 0.
5392 - val_loss: 1.5180 - val_accuracy: 0.4614
Epoch 13/100
1407/1407 [==============================] - 14s 10ms/step - loss: 1.2724 - accuracy: 0.
5495 - val_loss: 1.5796 - val_accuracy: 0.4458
Epoch 14/100
1407/1407 [==============================] - 14s 10ms/step - loss: 1.2475 - accuracy: 0.
5575 - val_loss: 1.4996 - val_accuracy: 0.4778
Epoch 15/100
1407/1407 [==============================] - 14s 10ms/step - loss: 1.2279 - accuracy: 0.
5675 - val_loss: 1.8289 - val_accuracy: 0.4158
Epoch 16/100
1407/1407 [==============================] - 14s 10ms/step - loss: 1.2079 - accuracy: 0.
5737 - val_loss: 1.4688 - val_accuracy: 0.4888
Epoch 17/100
1407/1407 [==============================] - 14s 10ms/step - loss: 1.1909 - accuracy: 0.
5786 - val_loss: 1.4732 - val_accuracy: 0.4964
Epoch 18/100
1407/1407 [==============================] - 14s 10ms/step - loss: 1.1666 - accuracy: 0.
5860 - val_loss: 1.5374 - val_accuracy: 0.4696
Epoch 19/100
1407/1407 [==============================] - 14s 10ms/step - loss: 1.1496 - accuracy: 0.
5941 - val_loss: 1.5607 - val_accuracy: 0.4742
Epoch 20/100
1407/1407 [==============================] - 14s 10ms/step - loss: 1.1330 - accuracy: 0.
6022 - val_loss: 1.5536 - val_accuracy: 0.4666
Epoch 21/100
1407/1407 [==============================] - 14s 10ms/step - loss: 1.1151 - accuracy: 0.
6060 - val_loss: 1.6986 - val_accuracy: 0.4246
Epoch 22/100
1407/1407 [==============================] - 14s 10ms/step - loss: 1.1017 - accuracy: 0.
6110 - val_loss: 1.6077 - val_accuracy: 0.4750
Epoch 23/100
1407/1407 [==============================] - 14s 10ms/step - loss: 1.0892 - accuracy: 0.
6173 - val_loss: 1.5339 - val_accuracy: 0.4778
Epoch 24/100
1407/1407 [==============================] - 14s 10ms/step - loss: 1.0686 - accuracy: 0.
6235 - val_loss: 1.5180 - val_accuracy: 0.4866
Epoch 25/100
```

```
1407/1407 [==============================] - 14s 10ms/step - loss: 1.0574 - accuracy: 0.
6266 - val_loss: 1.6930 - val_accuracy: 0.4360
Epoch 26/100
1407/1407 [==============================] - 14s 10ms/step - loss: 1.0429 - accuracy: 0.
6326 - val_loss: 1.4493 - val_accuracy: 0.5016
Epoch 27/100
1407/1407 [==============================] - 14s 10ms/step - loss: 1.0307 - accuracy: 0.
6365 - val_loss: 1.4688 - val_accuracy: 0.5090
Epoch 28/100
1407/1407 [==============================] - 14s 10ms/step - loss: 1.0179 - accuracy: 0.
6395 - val_loss: 1.7259 - val_accuracy: 0.4512
Epoch 29/100
1407/1407 [==============================] - 14s 10ms/step - loss: 1.0076 - accuracy: 0.
6476 - val_loss: 1.4642 - val_accuracy: 0.4978
Epoch 30/100
1407/1407 [==============================] - 14s 10ms/step - loss: 0.9960 - accuracy: 0.
6459 - val_loss: 1.5535 - val_accuracy: 0.4802
Epoch 31/100
1407/1407 [==============================] - 14s 10ms/step - loss: 0.9803 - accuracy: 0.
6531 - val_loss: 1.5549 - val_accuracy: 0.4934
157/157 [==============================] - 0s 3ms/step - loss: 1.4449 - accuracy: 0.4844
```

Out[14]:

```
[1.444880723953247, 0.4844000041484833]
```

模型收敛速度是否比以前更快？

快得多！前一个模型花费了29个时期达到最低的验证损失，而新模型仅在12个时期内实现了相同的损失，并继续取得进展，直到第17个时期。BN层稳定了训练，允许我们使用更大的学习率，因此收敛速度更快。

BN是否生产出更好的模型？

对最终的模型也要好得多，验证准确率为50.7%，而不是46.7%。它仍然不是一个很好的模型，但至少比以前好得多（卷积神经网络会做得更好，但这是一个不同的主题，见第14章）。

BN如何影响训练速度？

尽管模型收敛得更快，但由于BN层需要额外的计算，每个历元大约需要20秒而不是10秒。但总体而言，达到最佳模型的训练时间（壁时间）缩短了约10%。

8.4

In [15]:

```python
tf.random.set_seed(42)

model = tf.keras.Sequential()
model.add(tf.keras.layers.Flatten(input_shape=[32, 32, 3]))
for _ in range(20):
    model.add(tf.keras.layers.Dense(100,
                                    kernel_initializer="lecun_normal",
                                    activation="selu"))

model.add(tf.keras.layers.Dense(10, activation="softmax"))

optimizer = tf.keras.optimizers.Nadam(learning_rate=7e-4)
model.compile(loss="sparse_categorical_crossentropy",
              optimizer=optimizer,
              metrics=["accuracy"])

early_stopping_cb = tf.keras.callbacks.EarlyStopping(
    patience=20, restore_best_weights=True)
model_checkpoint_cb = tf.keras.callbacks.ModelCheckpoint(
    "my_cifar10_selu_model", save_best_only=True)
run_index = 1 # increment every time you train the model
```

```
run_logdir = Path() / "my_cifar10_logs" / f"run_selu_{run_index:03d}"
tensorboard_cb = tf.keras.callbacks.TensorBoard(run_logdir)
callbacks = [early_stopping_cb, model_checkpoint_cb, tensorboard_cb]

X_means = X_train.mean(axis=0)
X_stds = X_train.std(axis=0)
X_train_scaled = (X_train - X_means) / X_stds
X_valid_scaled = (X_valid - X_means) / X_stds
X_test_scaled = (X_test - X_means) / X_stds

model.fit(X_train_scaled, y_train, epochs=100,
          validation_data=(X_valid_scaled, y_valid),
          callbacks=callbacks)

model.evaluate(X_valid_scaled, y_valid)
```

```
Epoch 1/100
1401/1407 [============================>.] - ETA: 0s - loss: 1.9215 - accuracy: 0.3103IN
FO:tensorflow:Assets written to: my_cifar10_selu_model\assets
1407/1407 [==============================] - 15s 10ms/step - loss: 1.9210 - accuracy: 0.
3105 - val_loss: 1.8195 - val_accuracy: 0.3500
Epoch 2/100
1403/1407 [============================>.] - ETA: 0s - loss: 1.7063 - accuracy: 0.3918IN
FO:tensorflow:Assets written to: my_cifar10_selu_model\assets
1407/1407 [==============================] - 12s 9ms/step - loss: 1.7062 - accuracy: 0.3
918 - val_loss: 1.6988 - val_accuracy: 0.3978
Epoch 3/100
1407/1407 [==============================] - ETA: 0s - loss: 1.6101 - accuracy: 0.4315IN
FO:tensorflow:Assets written to: my_cifar10_selu_model\assets
1407/1407 [==============================] - 12s 8ms/step - loss: 1.6101 - accuracy: 0.4
315 - val_loss: 1.6472 - val_accuracy: 0.4074
Epoch 4/100
1407/1407 [==============================] - 11s 8ms/step - loss: 1.5429 - accuracy: 0.4
559 - val_loss: 1.6516 - val_accuracy: 0.4290
Epoch 5/100
1402/1407 [============================>.] - ETA: 0s - loss: 1.4831 - accuracy: 0.4807IN
FO:tensorflow:Assets written to: my_cifar10_selu_model\assets
1407/1407 [==============================] - 12s 9ms/step - loss: 1.4831 - accuracy: 0.4
807 - val_loss: 1.5880 - val_accuracy: 0.4468
Epoch 6/100
1405/1407 [============================>.] - ETA: 0s - loss: 1.4361 - accuracy: 0.4952IN
FO:tensorflow:Assets written to: my_cifar10_selu_model\assets
1407/1407 [==============================] - 13s 9ms/step - loss: 1.4363 - accuracy: 0.4
951 - val_loss: 1.5195 - val_accuracy: 0.4646
Epoch 7/100
1407/1407 [==============================] - 11s 8ms/step - loss: 1.3936 - accuracy: 0.5
130 - val_loss: 1.5357 - val_accuracy: 0.4658
Epoch 8/100
1401/1407 [============================>.] - ETA: 0s - loss: 1.3521 - accuracy: 0.5260IN
FO:tensorflow:Assets written to: my_cifar10_selu_model\assets
1407/1407 [==============================] - 13s 9ms/step - loss: 1.3517 - accuracy: 0.5
262 - val_loss: 1.5055 - val_accuracy: 0.4762
Epoch 9/100
1406/1407 [============================>.] - ETA: 0s - loss: 1.3200 - accuracy: 0.5404IN
FO:tensorflow:Assets written to: my_cifar10_selu_model\assets
1407/1407 [==============================] - 12s 8ms/step - loss: 1.3199 - accuracy: 0.5
404 - val_loss: 1.4854 - val_accuracy: 0.4724
Epoch 10/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.2882 - accuracy: 0.5
530 - val_loss: 1.5002 - val_accuracy: 0.4842
Epoch 11/100
1407/1407 [==============================] - 11s 8ms/step - loss: 1.2565 - accuracy: 0.5
678 - val_loss: 1.5196 - val_accuracy: 0.4880
Epoch 12/100
1407/1407 [==============================] - 11s 8ms/step - loss: 1.2346 - accuracy: 0.5
756 - val_loss: 1.4986 - val_accuracy: 0.4944
```

```
Epoch 13/100
1406/1407 [============================>.] - ETA: 0s - loss: 1.2103 - accuracy: 0.5812IN
FO:tensorflow:Assets written to: my_cifar10_selu_model\assets
1407/1407 [==============================] - 12s 9ms/step - loss: 1.2102 - accuracy: 0.5
812 - val_loss: 1.4701 - val_accuracy: 0.5060
Epoch 14/100
1407/1407 [==============================] - 11s 7ms/step - loss: 1.1848 - accuracy: 0.5
927 - val_loss: 1.4937 - val_accuracy: 0.5036
Epoch 15/100
1407/1407 [==============================] - 11s 8ms/step - loss: 1.1622 - accuracy: 0.6
003 - val_loss: 1.5035 - val_accuracy: 0.4956
Epoch 16/100
1407/1407 [==============================] - 11s 8ms/step - loss: 1.1375 - accuracy: 0.6
086 - val_loss: 1.5110 - val_accuracy: 0.4992
Epoch 17/100
1407/1407 [==============================] - 11s 7ms/step - loss: 1.1109 - accuracy: 0.6
171 - val_loss: 1.5429 - val_accuracy: 0.4976
Epoch 18/100
1407/1407 [==============================] - 11s 8ms/step - loss: 1.5025 - accuracy: 0.5
964 - val_loss: 1.6042 - val_accuracy: 0.4456
Epoch 19/100
1407/1407 [==============================] - 11s 8ms/step - loss: 1.2667 - accuracy: 0.5
574 - val_loss: 1.5147 - val_accuracy: 0.4944
Epoch 20/100
1407/1407 [==============================] - 11s 8ms/step - loss: 1.1743 - accuracy: 0.5
900 - val_loss: 1.5047 - val_accuracy: 0.4930
Epoch 21/100
1407/1407 [==============================] - 11s 8ms/step - loss: 1.1290 - accuracy: 0.6
054 - val_loss: 1.5106 - val_accuracy: 0.5070
Epoch 22/100
1407/1407 [==============================] - 11s 8ms/step - loss: 1.0922 - accuracy: 0.6
213 - val_loss: 1.5213 - val_accuracy: 0.5044
Epoch 23/100
1407/1407 [==============================] - 10s 7ms/step - loss: 1.0662 - accuracy: 0.6
331 - val_loss: 1.5026 - val_accuracy: 0.4992
Epoch 24/100
1407/1407 [==============================] - 11s 8ms/step - loss: 1.0444 - accuracy: 0.6
396 - val_loss: 1.5463 - val_accuracy: 0.5016
Epoch 25/100
1407/1407 [==============================] - 11s 7ms/step - loss: 1.0266 - accuracy: 0.6
490 - val_loss: 1.5375 - val_accuracy: 0.5086
Epoch 26/100
1407/1407 [==============================] - 11s 8ms/step - loss: 1.0093 - accuracy: 0.6
535 - val_loss: 1.5551 - val_accuracy: 0.5036
Epoch 27/100
1407/1407 [==============================] - 11s 8ms/step - loss: 0.9944 - accuracy: 0.6
602 - val_loss: 1.5460 - val_accuracy: 0.5086
Epoch 28/100
1407/1407 [==============================] - 11s 8ms/step - loss: 0.9868 - accuracy: 0.6
645 - val_loss: 1.5868 - val_accuracy: 0.4986
Epoch 29/100
1407/1407 [==============================] - 11s 8ms/step - loss: 0.9837 - accuracy: 0.6
641 - val_loss: 1.6124 - val_accuracy: 0.5006
Epoch 30/100
1407/1407 [==============================] - 11s 8ms/step - loss: 0.9734 - accuracy: 0.6
678 - val_loss: 1.5607 - val_accuracy: 0.5086
Epoch 31/100
1407/1407 [==============================] - 11s 8ms/step - loss: 0.9676 - accuracy: 0.6
705 - val_loss: 1.5528 - val_accuracy: 0.5136
Epoch 32/100
1407/1407 [==============================] - 11s 8ms/step - loss: 0.9650 - accuracy: 0.6
725 - val_loss: 1.5849 - val_accuracy: 0.5106
Epoch 33/100
1407/1407 [==============================] - 11s 8ms/step - loss: 0.9457 - accuracy: 0.6
795 - val_loss: 1.6060 - val_accuracy: 0.5078
157/157 [==============================] - 0s 2ms/step - loss: 1.4701 - accuracy: 0.5060
```

```
Out[15]: [1.4701128005981445, 0.5059999823570251]
```

8.5

```
In [16]: tf.random.set_seed(42)

model = tf.keras.Sequential()
model.add(tf.keras.layers.Flatten(input_shape=[32, 32, 3]))
for _ in range(20):
    model.add(tf.keras.layers.Dense(100,
                                    kernel_initializer="lecun_normal",
                                    activation="selu"))

model.add(tf.keras.layers.Dense(10, activation="softmax"))

optimizer = tf.keras.optimizers.Nadam(learning_rate=7e-4)
model.compile(loss="sparse_categorical_crossentropy",
              optimizer=optimizer,
              metrics=["accuracy"])

early_stopping_cb = tf.keras.callbacks.EarlyStopping(
    patience=20, restore_best_weights=True)
model_checkpoint_cb = tf.keras.callbacks.ModelCheckpoint(
    "my_cifar10_selu_model", save_best_only=True)
run_index = 1 # increment every time you train the model
run_logdir = Path() / "my_cifar10_logs" / f"run_selu_{run_index:03d}"
tensorboard_cb = tf.keras.callbacks.TensorBoard(run_logdir)
callbacks = [early_stopping_cb, model_checkpoint_cb, tensorboard_cb]

X_means = X_train.mean(axis=0)
X_stds = X_train.std(axis=0)
X_train_scaled = (X_train - X_means) / X_stds
X_valid_scaled = (X_valid - X_means) / X_stds
X_test_scaled = (X_test - X_means) / X_stds

model.fit(X_train_scaled, y_train, epochs=100,
          validation_data=(X_valid_scaled, y_valid),
          callbacks=callbacks)

model.evaluate(X_valid_scaled, y_valid)
```

```
Epoch 1/100
1401/1407 [============================>.] - ETA: 0s - loss: 1.9416 - accuracy: 0.3058IN
FO:tensorflow:Assets written to: my_cifar10_selu_model\assets
1407/1407 [==============================] - 15s 9ms/step - loss: 1.9409 - accuracy: 0.3
061 - val_loss: 1.8503 - val_accuracy: 0.3380
Epoch 2/100
1400/1407 [============================>.] - ETA: 0s - loss: 1.7156 - accuracy: 0.3937IN
FO:tensorflow:Assets written to: my_cifar10_selu_model\assets
1407/1407 [==============================] - 12s 9ms/step - loss: 1.7156 - accuracy: 0.3
936 - val_loss: 1.7217 - val_accuracy: 0.3888
Epoch 3/100
1404/1407 [============================>.] - ETA: 0s - loss: 1.6165 - accuracy: 0.4314IN
FO:tensorflow:Assets written to: my_cifar10_selu_model\assets
1407/1407 [==============================] - 13s 9ms/step - loss: 1.6163 - accuracy: 0.4
316 - val_loss: 1.6346 - val_accuracy: 0.4148
Epoch 4/100
1402/1407 [============================>.] - ETA: 0s - loss: 1.5486 - accuracy: 0.4590IN
FO:tensorflow:Assets written to: my_cifar10_selu_model\assets
1407/1407 [==============================] - 16s 12ms/step - loss: 1.5489 - accuracy: 0.
4590 - val_loss: 1.6141 - val_accuracy: 0.4374
Epoch 5/100
1406/1407 [============================>.] - ETA: 0s - loss: 1.4910 - accuracy: 0.4809IN
FO:tensorflow:Assets written to: my_cifar10_selu_model\assets
```

```
1407/1407 [==============================] - 16s 12ms/step - loss: 1.4910 - accuracy: 0.
4809 - val_loss: 1.5595 - val_accuracy: 0.4500
Epoch 6/100
1407/1407 [==============================] - ETA: 0s - loss: 1.4496 - accuracy: 0.4951IN
FO:tensorflow:Assets written to: my_cifar10_selu_model\assets
1407/1407 [==============================] - 16s 11ms/step - loss: 1.4496 - accuracy: 0.
4951 - val_loss: 1.5365 - val_accuracy: 0.4686
Epoch 7/100
1406/1407 [============================>.] - ETA: 0s - loss: 1.4057 - accuracy: 0.5059IN
FO:tensorflow:Assets written to: my_cifar10_selu_model\assets
1407/1407 [==============================] - 17s 12ms/step - loss: 1.4057 - accuracy: 0.
5059 - val_loss: 1.5159 - val_accuracy: 0.4676
Epoch 8/100
1406/1407 [============================>.] - ETA: 0s - loss: 1.3604 - accuracy: 0.5263IN
FO:tensorflow:Assets written to: my_cifar10_selu_model\assets
1407/1407 [==============================] - 16s 11ms/step - loss: 1.3605 - accuracy: 0.
5263 - val_loss: 1.4854 - val_accuracy: 0.4850
Epoch 9/100
1407/1407 [==============================] - 15s 10ms/step - loss: 1.3327 - accuracy: 0.
5373 - val_loss: 1.5233 - val_accuracy: 0.4758
Epoch 10/100
1407/1407 [==============================] - 15s 10ms/step - loss: 1.2979 - accuracy: 0.
5494 - val_loss: 1.5711 - val_accuracy: 0.4760
Epoch 11/100
1407/1407 [==============================] - 15s 10ms/step - loss: 1.2718 - accuracy: 0.
5574 - val_loss: 1.5028 - val_accuracy: 0.4970
Epoch 12/100
1407/1407 [==============================] - 15s 10ms/step - loss: 1.2359 - accuracy: 0.
5724 - val_loss: 1.4909 - val_accuracy: 0.4894
Epoch 13/100
1407/1407 [==============================] - ETA: 0s - loss: 1.2131 - accuracy: 0.5832IN
FO:tensorflow:Assets written to: my_cifar10_selu_model\assets
1407/1407 [==============================] - 16s 11ms/step - loss: 1.2131 - accuracy: 0.
5832 - val_loss: 1.4842 - val_accuracy: 0.4994
Epoch 14/100
1403/1407 [============================>.] - ETA: 0s - loss: 1.1889 - accuracy: 0.5891IN
FO:tensorflow:Assets written to: my_cifar10_selu_model\assets
1407/1407 [==============================] - 16s 12ms/step - loss: 1.1887 - accuracy: 0.
5893 - val_loss: 1.4600 - val_accuracy: 0.5078
Epoch 15/100
1407/1407 [==============================] - 14s 10ms/step - loss: 1.1692 - accuracy: 0.
5980 - val_loss: 1.5186 - val_accuracy: 0.5004
Epoch 16/100
1407/1407 [==============================] - 15s 10ms/step - loss: 1.2189 - accuracy: 0.
5920 - val_loss: 1.6896 - val_accuracy: 0.4166
Epoch 17/100
1407/1407 [==============================] - 15s 10ms/step - loss: 1.3732 - accuracy: 0.
5164 - val_loss: 1.5307 - val_accuracy: 0.4706
Epoch 18/100
1407/1407 [==============================] - 14s 10ms/step - loss: 1.2418 - accuracy: 0.
5678 - val_loss: 1.5000 - val_accuracy: 0.4924
Epoch 19/100
1407/1407 [==============================] - 14s 10ms/step - loss: 1.1787 - accuracy: 0.
5908 - val_loss: 1.4942 - val_accuracy: 0.4946
Epoch 20/100
1407/1407 [==============================] - 15s 11ms/step - loss: 1.1385 - accuracy: 0.
6074 - val_loss: 1.5281 - val_accuracy: 0.4940
Epoch 21/100
1407/1407 [==============================] - 14s 10ms/step - loss: 1.1041 - accuracy: 0.
6188 - val_loss: 1.4871 - val_accuracy: 0.5134
Epoch 22/100
1407/1407 [==============================] - 14s 10ms/step - loss: 1.0830 - accuracy: 0.
6305 - val_loss: 1.4866 - val_accuracy: 0.5208
Epoch 23/100
1407/1407 [==============================] - 14s 10ms/step - loss: 1.0632 - accuracy: 0.
6369 - val_loss: 1.5162 - val_accuracy: 0.5112
```

```
Epoch 24/100
1407/1407 [==============================] - 14s 10ms/step - loss: 1.0519 - accuracy: 0.
6410 - val_loss: 1.4900 - val_accuracy: 0.5158
Epoch 25/100
1407/1407 [==============================] - 14s 10ms/step - loss: 1.0477 - accuracy: 0.
6426 - val_loss: 1.5088 - val_accuracy: 0.5186
Epoch 26/100
1407/1407 [==============================] - 14s 10ms/step - loss: 1.0316 - accuracy: 0.
6490 - val_loss: 1.5076 - val_accuracy: 0.5124
Epoch 27/100
1407/1407 [==============================] - 14s 10ms/step - loss: 1.0107 - accuracy: 0.
6567 - val_loss: 1.5252 - val_accuracy: 0.5120
Epoch 28/100
1407/1407 [==============================] - 14s 10ms/step - loss: 0.9956 - accuracy: 0.
6642 - val_loss: 1.5267 - val_accuracy: 0.5112
Epoch 29/100
1407/1407 [==============================] - 14s 10ms/step - loss: 0.9836 - accuracy: 0.
6676 - val_loss: 1.5865 - val_accuracy: 0.5180
Epoch 30/100
1407/1407 [==============================] - 14s 10ms/step - loss: 0.9561 - accuracy: 0.
6785 - val_loss: 1.5166 - val_accuracy: 0.5198
Epoch 31/100
1407/1407 [==============================] - 14s 10ms/step - loss: 0.9452 - accuracy: 0.
6801 - val_loss: 1.5474 - val_accuracy: 0.5160
Epoch 32/100
1407/1407 [==============================] - 14s 10ms/step - loss: 0.9454 - accuracy: 0.
6820 - val_loss: 1.6213 - val_accuracy: 0.5114
Epoch 33/100
1407/1407 [==============================] - 14s 10ms/step - loss: 0.9155 - accuracy: 0.
6943 - val_loss: 1.5579 - val_accuracy: 0.5164
Epoch 34/100
1407/1407 [==============================] - 14s 10ms/step - loss: 0.9023 - accuracy: 0.
6993 - val_loss: 1.6489 - val_accuracy: 0.5146
157/157 [==============================] - 1s 3ms/step - loss: 1.4600 - accuracy: 0.5078
```

Out[16]:

```
[1.459976315498352, 0.5077999830245972]
```

In [17]:

```python
class MCAlphaDropout(tf.keras.layers.AlphaDropout):
    def call(self, inputs):
        return super().call(inputs, training=True)

mc_model = tf.keras.Sequential([
    (
        MCAlphaDropout(layer.rate)
        if isinstance(layer, tf.keras.layers.AlphaDropout)
        else layer
    )
    for layer in model.layers
])

# The first will run the model many times (10 by default) and it will return the mean pr
def mc_dropout_predict_probas(mc_model, X, n_samples=10):
    Y_probas = [mc_model.predict(X) for sample in range(n_samples)]
    return np.mean(Y_probas, axis=0)

# The second will use these mean probabilities to predict the most likely class for each
def mc_dropout_predict_classes(mc_model, X, n_samples=10):
    Y_probas = mc_dropout_predict_probas(mc_model, X, n_samples)
    return Y_probas.argmax(axis=1)

tf.random.set_seed(42)

y_pred = mc_dropout_predict_classes(mc_model, X_valid_scaled)
accuracy = (y_pred == y_valid[:, 0]).mean()
accuracy
```

```
157/157 [==============================] - 1s 3ms/step
157/157 [==============================] - 1s 3ms/step
157/157 [==============================] - 1s 3ms/step
157/157 [==============================] - 1s 3ms/step
157/157 [==============================] - 1s 3ms/step
157/157 [==============================] - 1s 3ms/step
157/157 [==============================] - 1s 3ms/step
157/157 [==============================] - 1s 3ms/step
157/157 [==============================] - 1s 3ms/step
157/157 [==============================] - 1s 4ms/step
```

Out[17]:

```
0.5078
```

8.6

In [18]:

```
tf.random.set_seed(42)

model = tf.keras.Sequential()
model.add(tf.keras.layers.Flatten(input_shape=[32, 32, 3]))
for _ in range(20):
    model.add(tf.keras.layers.Dense(100,
                                    kernel_initializer="lecun_normal",
                                    activation="selu"))

model.add(tf.keras.layers.AlphaDropout(rate=0.1))
model.add(tf.keras.layers.Dense(10, activation="softmax"))

optimizer = tf.keras.optimizers.SGD()
model.compile(loss="sparse_categorical_crossentropy",
              optimizer=optimizer,
              metrics=["accuracy"])

batch_size = 128
rates, losses = find_learning_rate(model, X_train_scaled, y_train, epochs=1,
                                   batch_size=batch_size)
plot_lr_vs_loss(rates, losses)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[18], line 19
     14 model.compile(loss="sparse_categorical_crossentropy",
     15               optimizer=optimizer,
     16               metrics=["accuracy"])
     18 batch_size = 128
---> 19 rates, losses = find_learning_rate(model, X_train_scaled, y_train, epochs=1,
     20                                    batch_size=batch_size)
     21 plot_lr_vs_loss(rates, losses)

NameError: name 'find_learning_rate' is not defined
```

In [ ]:

```
tf.random.set_seed(42)

model = tf.keras.Sequential()
model.add(tf.keras.layers.Flatten(input_shape=[32, 32, 3]))
for _ in range(20):
    model.add(tf.keras.layers.Dense(100,
                                 kernel_initializer="lecun_normal",
                                 activation="selu"))

model.add(tf.keras.layers.AlphaDropout(rate=0.1))
model.add(tf.keras.layers.Dense(10, activation="softmax"))

optimizer = tf.keras.optimizers.SGD(learning_rate=2e-2)
model.compile(loss="sparse_categorical_crossentropy",
              optimizer=optimizer,
              metrics=["accuracy"])
```

In [ ]:
```
n_epochs = 15
n_iterations = math.ceil(len(X_train_scaled) / batch_size) * n_epochs
onecycle = OneCycleScheduler(n_iterations, max_lr=0.05)
history = model.fit(X_train_scaled, y_train, epochs=n_epochs, batch_size=batch_size,
                    validation_data=(X_valid_scaled, y_valid),
                    callbacks=[onecycle])
```