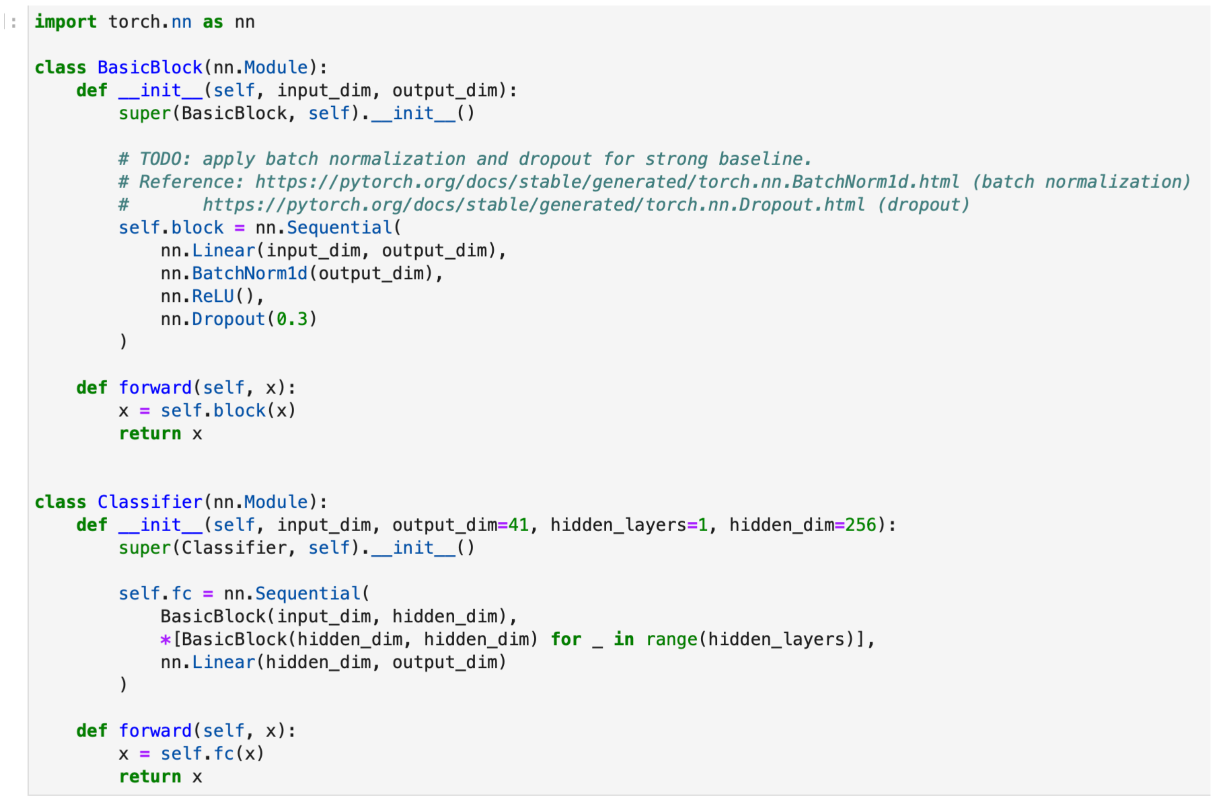
HW2-classfication

Created	@2025年6月9日 18:43
Class	李宏毅机器学习

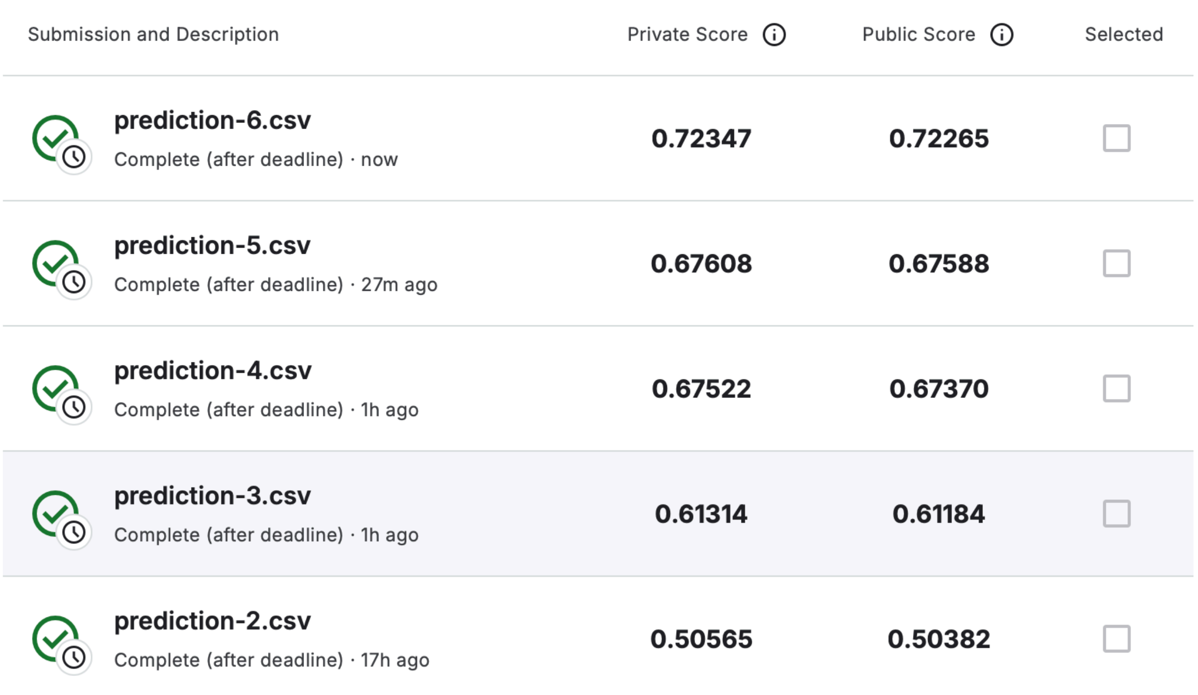
音素分类

基础模型结构：



```python
import torch.nn as nn

class BasicBlock(nn.Module):
    def __init__(self, input_dim, output_dim):
        super(BasicBlock, self).__init__()

        # TODO: apply batch normalization and dropout for strong baseline.
        # Reference: https://pytorch.org/docs/stable/generated/torch.nn.BatchNorm1d.html (batch normalization)
        #       https://pytorch.org/docs/stable/generated/torch.nn.Dropout.html (dropout)
        self.block = nn.Sequential(
            nn.Linear(input_dim, output_dim),
            nn.BatchNorm1d(output_dim),
            nn.ReLU(),
            nn.Dropout(0.3)
        )

    def forward(self, x):
        x = self.block(x)
        return x


class Classifier(nn.Module):
    def __init__(self, input_dim, output_dim=41, hidden_layers=1, hidden_dim=256):
        super(Classifier, self).__init__()

        self.fc = nn.Sequential(
            BasicBlock(input_dim, hidden_dim),
            *[BasicBlock(hidden_dim, hidden_dim) for _ in range(hidden_layers)],
            nn.Linear(hidden_dim, output_dim)
        )

    def forward(self, x):
        x = self.fc(x)
        return x
```

增大concat_nframes，增加batchnorm和dropout层，逐渐增大隐藏层层数和神经元个数。



Submission and Description	Private Score	Public Score	Selected
prediction-6.csv Complete (after deadline) · now	**0.72347**	**0.72265**	☐
prediction-5.csv Complete (after deadline) · 27m ago	**0.67608**	**0.67588**	☐
prediction-4.csv Complete (after deadline) · 1h ago	**0.67522**	**0.67370**	☐
prediction-3.csv Complete (after deadline) · 1h ago	**0.61314**	**0.61184**	☐
prediction-2.csv Complete (after deadline) · 17h ago	**0.50565**	**0.50382**	☐

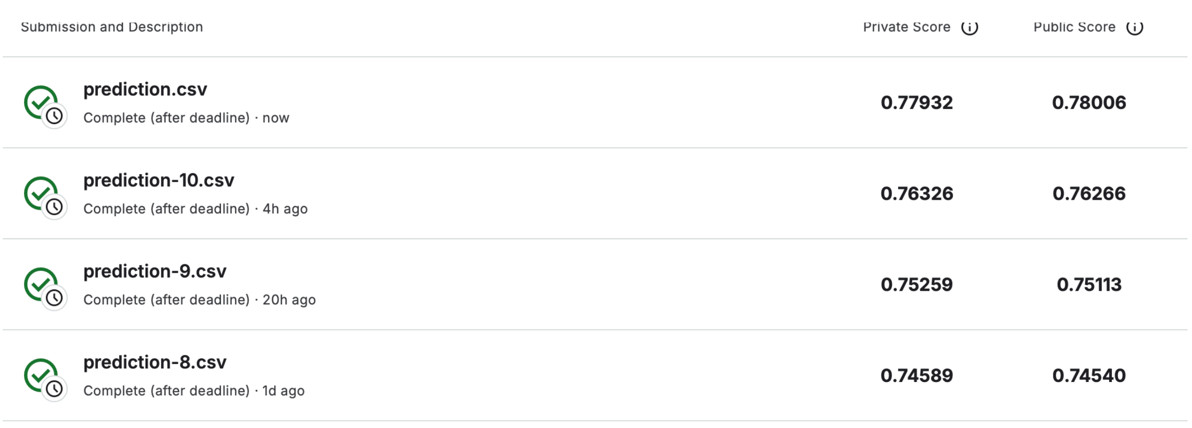
准确率从50%提升72%左右

下面优化模型为RNN

新模型结构

```python
class Classifier(nn.Module):
    def __init__(self, input_dim, output_dim=41, hidden_layers=1, hidden_dim=256):
        super(Classifier, self).__init__()


        # Create BiLSTM
        self.input_size = 39    # 这一项是RNN的"input_dim", RNN需要对"单"个数据进行处理
        self.hidden_size = 512  # 这一项是RNN的"hidden_dim"
        self.num_layers = 6     # 这一项是RNN的"hidden_layers"
        self.rnn = nn.LSTM(input_size=self.input_size,
                           hidden_size=self.hidden_size,
                           num_layers=self.num_layers,
                           batch_first=True,
                           dropout=0.3,
                           bidirectional=True)

        # 后接全连接层
        self.fc = nn.Sequential(
            BasicBlock(2 * self.hidden_size, hidden_dim),
            nn.Linear(hidden_dim, output_dim)
        )

    def forward(self, x):
        # 通过RNN层, 得到输出和最后一个隐藏状态, 注意输出的shape
        # x.shape: (batch_size, seq_len, RNN_input_size)
        x, _ = self.rnn(x)  # => (batch_size, seq_len, RNN_hidden_size)

        # 取最后一个时间步的输出作为分类的输入
        x = x[:, -1]        # => (batch_size, RNN_hidden_size)

        # 通过线性层, 得到最终的分类结果
        x = self.fc(x)      # => (batch_size, labels)

        return x
```

结果:



Submission and Description	Private Score ⓘ	Public Score ⓘ
prediction.csv Complete (after deadline) · now	0.77932	0.78006
prediction-10.csv Complete (after deadline) · 4h ago	0.76326	0.76266
prediction-9.csv Complete (after deadline) · 20h ago	0.75259	0.75113
prediction-8.csv Complete (after deadline) · 1d ago	0.74589	0.74540

使用rnn之后经过两三个epoch acc分数就能到0.75，但是后续val_acc增长过于缓慢，而且训练太慢了，如果继续训练应该可以接近baseline 0.85。