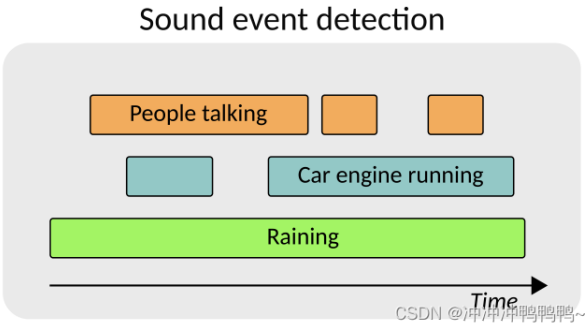
声音事件检测

1. 声音事件检测

1.1. 声音事件检测

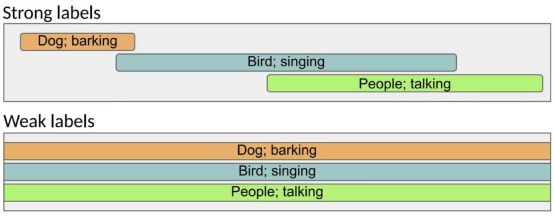
声音事件检测的目的就是识别出一个音频中声音事件的种类，以及检测出声音事件发生和结束的时间。



模型会通过输入已标记的音频进行神经网络训练，从而实现对未知的某一场景下的音频进行事件检测。

传统的声音事件检测数据使用的是强标签，对于音频数据不仅给定了事件类别，还给定了事件发生的起始和终止时间戳，对于这样的输入，学习的任务可以看成分类的任务，我们要对音频中事件的类型，并预测出起始时间戳。

本次任务中我们使用弱标签，弱监督声音事件检测是指在训练过程中只使用了事件级别的标签，而没有使用时间位置信息的一种方法。这样的任务更加有挑战性，因为没有了时间戳，难以直接提取声音事件的隐层信息。



弱监督时间轴预测有以下的难点

1. 缺乏精确的时间位置标签：在弱监督情况下，通常只有事件级别的标签，而没有精确的时间位置信息。这使得模型无法直接学习事件发生的确切时间点，而只能通过事件级别标签进行训练。因此，模型需要具备一定的时间推理能力，能够从事件级别的信息中推断出事件发生的时间轴。
2. 事件的持续时间不确定：由于缺乏精确的时间位置标签，模型无法准确地确定事件的持续时间。事件可能在音频中的不同时间段内发生，并且持续时间可能因事件的不同实例而异。因此，模型需要具备对事件持续时间的灵活建模能力。
3. 多个事件的重叠和交叉：在音频中，多个事件可能同时发生、重叠或交叉。这增加了时间轴预测的复杂性，因为模型需要能够区分和分离不同的事件，并准确地预测它们的时间轴。
4. 数据的不完整性和噪声：在实际应用中，音频数据可能存在不完整性和噪声。例如，音频可能被截断、存在背景噪声或其他干扰。这些因素会对时间轴预测的准确性产生负面影响，因为模型需要能够从不完整和嘈杂的数据中提取有效的事件信息。

1.2. 基线模型

基线模型使用 CRNN 从梅尔频谱图中提取特征以及进行分类，因为是分类任务，因此模型中使用了 softmax 函数，loss 采用了 BCEloss，这两个部分通常是一起使用的，因为在数学形式上它们的内涵是一样的。

通过阅读代码，我认为模型的实现逻辑是这样的。

模型将任务分为两部分，第一部分是分离声音，第二部分是对声音进行事件检测。第一部分是

使用预处理的基线声音分离模型来分离重叠的声音事件，并从背景声音中提取前景声音事件，分离得到声音事件部分和背景音部分。然后通过第二部分声音事件检测模块，提取特征并通过阈值区分背景声音和声音事件，从而得到声音事件的起止时间戳。

对模型的理解我可能还有一些不完善的地方，请多见谅。

模型的评估指标有下面三种，报告后面的结果展示里也会用到这三个指标。

event_based 是通过逐个事件检查预测和真实值的对应程度计算得到的。比如如果一个事件的 label 预测和 ground truth 一样，并且预测的边界在真实事件的参考范围内，则事件被视为 TP，代入下一步的计算中。

segment_based 是基于比系统输出分辨率更粗略的固定时间网格上进行对模型预测和真实标注的比较，从而进行数学运算得到的指标。

tagging_based 是通过直接判断整段音频中是否存在过某一类别的声音而计算得到的指标。

2. CRNN 原理和实现

2.1. CRNN

RCNN 的原论文是应用在图像的识别中的，在本次任务中，我们的输入是 MFCC 特征图，和原论文的核心思想是一样的。

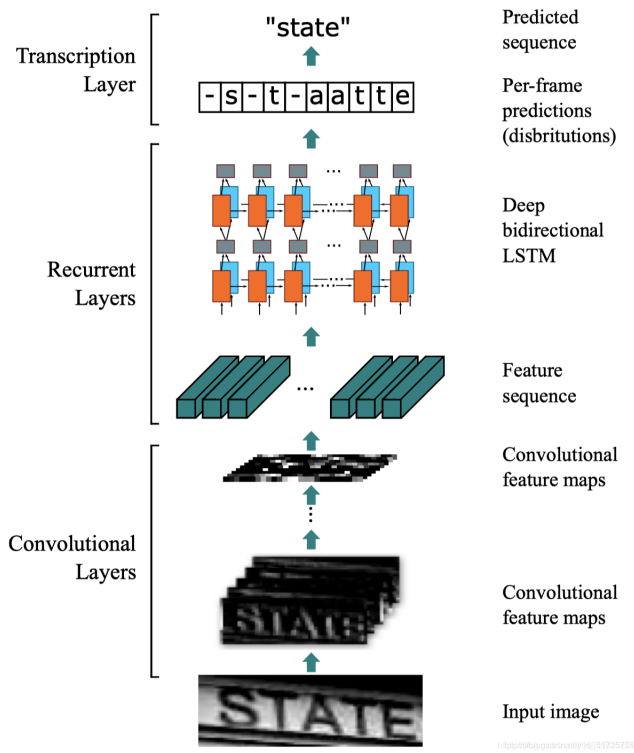
这是原文中用到的模型，图片先经过 CNN 提取特征，形成的特征图切割成序列，输入循环卷积网络 LSTM 中来获取对结果的预测。

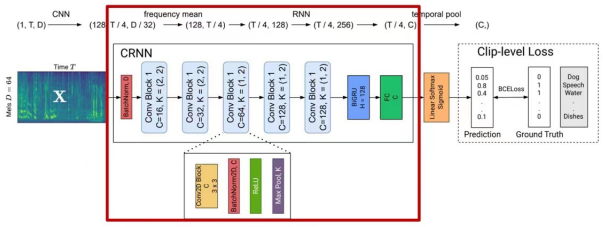
在声音事件检测中，我们也是对 MFCC 特征图进行特征提取，提取的特征图切割成序列输入 RNN，输出结果通过全连接层和激活函数生成 prediction，并与 ground truth 做 BCEloss 来反传优化模型。

2.2. 实现

任务中要实现 CRNN 模型中的 CNN 部分，RNN 部分只需要调用库中的函数即可。

输入的数据为三维张量（三个维度分别代表





batch size、时间步长和频数），首先通过一个 batchnorm1d 层进行归一化（batchnorm1d 的输入是一个三维数据）。接下来对数据进行 unsqueeze 操作升维，数据通过五个卷积模块，每个卷积模块中都含有卷积层、归一化层、激活函数层、池化层，各层的参数图中已给出。要注意的是这里的归一化层要使用 batchnorm2d 因为数据是四维的。接下来对 num_freq 维度求解 mean 来压缩数据维度，再通过 RNN 和 FC 层和 softmax 层从网络中输出，此时的值有负值，再将其通过 sigmoid 函数映射到 0 到 1 之间，得到最终的输出，为预测的该段音频出现每一个声音事件的概率，将其与 ground truth 对比得到 loss。

这一段实现的难点主要是维度的转化，升维用到 unsqueeze，降维我用到了取均值的操作。还有一个难点是要多次调整维度的顺序，以保证对

接口的一致性，我多次使用了 permute 来调整顺序。

3. 实验过程与分析

3.1. baseline

我对实现的 CRNN 进行训练，以下是模型的结果。

表 1: 基线模型

	f_measure	precision	recall
event	0.171394	0.151406	0.225962
segment	0.609222	0.63209	0.602154
tagging	0.626966	0.676021	0.595899

3.2. 优化探究

针对模型的结构，我使用了以下的优化方法

首先，我加深的模型的深度，将卷积层的深度加深，在加深时一定要注意维度的对准和 pooling 的选择，如果不正当使用 pooling 会将特征压缩到 0，这也是我在改正代码时学习到的内容。

加深一层的模型结果如下。

表 2: 加深一层的模型

	f_measure	precision	recall
event	0.170973	0.158216	0.213492
segment	0.584768	0.650827	0.548928
tagging	0.621358	0.656827	0.600358

我又尝试了继续加深，加深了三层，结果如下：

表 3: 加深三层的模型

	f_measure	precision	recall
event	0.16905	0.161381	0.201919
segment	0.60778	0.656633	0.577147
tagging	0.635869	0.648434	0.636004

其实可以看到效果有所提升，但不是很显著，我又想到可以将全连接层加深

表 4: 全连接层加深的模型

	f_measure	precision	recall
event	0.132841	0.126779	0.161972
segment	0.598197	0.638104	0.573879
tagging	0.651989	0.667321	0.647311

可以看到效果没有提升，我还是采用基础的全连接结构。

我又想到可能是深度加深引起的问题，我加入了 dropout 层，加入后的模型效果如下。

表 5: 加入 *dropout* 的模型

	f_measure	precision	recall
event	0.17022	0.157859	0.215535
segment	0.615481	0.630973	0.61446
tagging	0.639908	0.674896	0.618408

也没有太大的提升。

于是我想到更改 RNN 中的模型，源代码中使用的是 GRU，我将其替换为了 RNN 和 LSTM，注意这里一定要使用双向的模型，因为双向一定是优于单向的。

比如一句话：“我的手机坏了，我想要 () 新手机。”

如果是单项的模型，就只能看到括号前的信息，因此难以确定这里填什么，但是如果是双向的，那模型就可以看到括号后的信息，就可以通过前后信息确定要填“新”。当然在这个任务中，双向的模型也能起到类似的作用。

我使用了双向的 RNN 和 LSTM，以下分别是 RNN 和 LSTM 的结果。

表 6: *RNN*

	f_measure	precision	recall
event	0.11073	0.12734	0.10677
segment	0.549005	0.671458	0.478902
tagging	0.605882	0.662504	0.589023

RNN 和 LSTM 的效果不如 GRU，这几个模型我们在语言模型章节已经解释过了，GRU 的模型更加简洁，有更好的长期依赖建模，因此在某些任务上会比 RNN 和 LSTM 更好。

表 7: *LSTM*

	f_measure	**precision**	**recall**
event	0.136197	0.131855	0.173532
segment	0.581286	0.595915	0.589216
tagging	0.617707	0.674792	0.590708

我还尝试了更换池化方式，也没有取得正优化。

我还想到可以更换为 attention 机制，这样的模型理论上会有效果的提升，但由于 Transformer 的接口和 GRU 的接口有一定区别，最终我没能解决数据不匹配的问题，没有成功进行实验。

总结来说 baseline 里的我的一些设置还是很合理的，让基线模型的效果就达到了很好的状态，我的基线模型里还使用了上采样来对其维度，如果不使用这个，模型的效果会打折许多。如果使用加深卷积层的模型，可以将模型的效果更加优化。

4. 增强方法调研

我查阅资料了解有以下的音频数据增强方法

1. 给音频中插入随机噪声（类似图片中的加噪）
2. 转移时间。将音频前移或者后移，对缺失的部分补充静音即可。（类似图片中的移动）
3. 改变音高（liborsa 中有辅助函数）
4. 改变音速（liborsa 中有辅助函数）
5. 时域掩盖，将 t 个连续的时间步长 [t_0,t_0+t] 屏蔽，t 选自从 0 到时间掩码参数 T 的均匀分布，且 t_0 选自 [0,T-t]，T 为音频的总长度。
6. 频域掩盖，将频率域的 [f_0,f_0+f] 部分屏蔽，f 是选自从 0 到频率掩码参数 F 的均匀分布，且 f_0 选自 [0,v-f]，其中 v 为频率通道的数量。
7. 时间扭曲，在音频中选取一个随机点并以距离 w 向左或向右扭曲，该距离从 0 到沿该线的时间扭曲，参数 W 的均匀分布中选择。

可以参考下面的资料

`https://blog.csdn.net/wudibaba21/article/details/110843874`

`https://www.kaggle.com/code/CVxTz/audio-data-augmentation/notebook`