**2021 CCF BDCI基于飞桨实现花样滑冰选手骨骼点动作识别-第7名方案 报告文档**

1. **模型及算法**
2. **算法的总体框架**

关节数据

骨骼数据

数据预处理、数据增强

STGCN

AAGCN

MSG3D

模型

融合

**Score**

**Score1**

**Score2**

**Score3**

图1 算法总体框架图

**1.1数据部分**

比赛提供的数据是骨骼点（关节点）数据，我们通过相邻两个关节点相减的方法得到骨骼数据，分别用关节数据和骨骼数据训练模型。

**1.2数据预处理和数据增强**

**数据预处理。**由于最初的每个样本数据中包含许多无用的帧，因此我们先用AutoPadding选出包含有用信息的帧，这样不仅能够降低数据的大小，减少训练时显存的消耗，还能减轻样本噪声对模型训练的影响。

**数据增强。**我们使用了PoseCompact、Resize、RandomResizedCrop、Flip等数据增强的方法，来提高模型的泛化能力。代码参考：[mmaction2/mmaction/datasets/pipelines at master · open-mmlab/mmaction2 · GitHub (fastgit.org)](https://hub.fastgit.org/open-mmlab/mmaction2/tree/master/mmaction/datasets/pipelines)。我们还使用Mixup（在AAGCN以及MSG3D模型）和manifold mixup数据增强方法（在STGCN模型上）。

Mixup：<https://arxiv.org/abs/1710.09412>

Manifold Mixup：<https://arxiv.org/abs/1806.05236>

**1.3模型部分**

我们一共使用了三个模型，分别是STGCN、AAGCN和MSG3D。

**STGCN模型**首先提出了基于图卷积的方法来实现骨骼点数据特征的提取。通过图卷积，模型能够很好地捕获关节点之间的关系，从而提高模型对骨骼点数据的识别。

**AAGCN模型**有两点不同于STGCN，首先其使用了自适应的图卷积，参数化图卷积操作，同时在图卷积部分引入了注意力机制，加入了空间注意力、时间注意力和通道注意力，从而提高了模型的灵活性和泛化能力。

**MSG3D模型**通过Disentangled Multi-Scale Aggregation去除了k级相邻关节点之间的冗余信息，以及通过Cross-Spacetime Skip Connections考虑了不同帧之间相邻关节点的影响（例如，第3帧中的关节2对第1帧中的关节1的影响，假设1,2关节相邻），最终能够提取出多尺度的时间-空间特征，更加全面地进行特征的提取。

若想具体了解这些模型，请阅读原论文，代码参考官方提供的代码。

STGCN：<https://arxiv.org/abs/1801.07455>

AAGCN：<https://arxiv.org/pdf/1912.06971v1.pdf>

MSG3D：<https://arxiv.org/pdf/2003.14111v2.pdf>

1. **代码内容说明**

我们使用的是paddlevideo（<https://github.com/PaddlePaddle/PaddleVideo>）。里面内容较多，这里就其中我们主要需要的几个文件进行说明。

**configs：**这里面是训练和测试模型的相关配置。我们使用的配置文件是recognition/stgcn/stgcn\_fsd.yaml 、recognition/aagcn/aagcn\_fsd.yaml 和 recognition/msg3d/ msg3d\_fsd.yaml。

**log：**这里面是训练过程中的日志，包含训练和验证损失、top1、top5以及lr变化曲线。

**output：**训练过程中模型保存的地方。

**best\_model：**里面存放了在验证集上最优的模型和优化器参数，这也是我们最后提交的结果的模型。

**paddlevideo：**

**loader文件夹**中是数据预处理各种方法、pipeline。我们主要使用的是work/PaddleVideo/paddlevideo/loader/pipelines/DA.py、 mix.py以及skeleton\_pipeline.py。

**modeling文件夹**中是具体模型的backbone和head。我们使用的模型为STGCN、AAGCN和MSG3D。按照名称寻找即可（注：AAGCN的head使用的是STGCN的head）。

**score：**这里面保存的是各个模型在测试集（test-b）上的score，用于后续的模型融合。

**运行文件**

main-1.ipynb,main-2.ipynb,main-3.ipynb分别是用于STGCN、AAGCN和MSG3D的训练，其中有重复的部分。

**submission.csv**就是最终提交的结果。

**三、复现**

我们使用了三个模型STGCN、AAGCN和MSG3D，每个模型在关节数据和骨骼数据分别运行了两次，因此最终共有12个模型（以STGCN为例，模型名为STGCN

-Joint,STGCN-Joint-2,STGCN-Bone以及STGCN-Bone-2），这些都是在验证集上最优的模型，保存在best\_model文件夹里面。

由于我们使用了多个模型融合的方法，因此需要多次训练不同的模型，然后将最终的结果整合。

**3.1复现的相关配置**

Batch size均为32，dropout为0.3（STGCN和MSG3D）、0.0（AAGCN），优化器我们使用SGD，momentum为0.9，L2为0.0001，学习率使用的是CustomWarmupCosineDecay，数据增强的具体参数详见对应的yaml文件，训练80epoch（STGCN和AAGCN）和60epoch（MSG3D）。

**3.2具体复现步骤**

以训练STGCN模型，使用main-1.ipynb为例。

Step1：将训练数据分割成训练集和验证集。（整个复现过程只需运行一次）

Step2：产生骨骼数据（训练集、验证集以及测试集）。（只需运行一次）

Step3：找到stgcn\_fsd.yaml文件进行参数配置，默认配置是我们实现最终结果的配置。这里需要注意使用的是关节数据还是骨骼数据。

Step4：训练模型并在验证集上测试。

Step5：用在验证集上最优的模型对测试集进行预测，保存score。

上述步骤结束之后就可以得到模型STGCN-Joint,修改yaml文件中的模型名，再次运行就可以得到STGCN-Joint-2。STGCN-Bone和STGCN-Bone-2类似，只需要修改数据和模型名即可。

三个模型都训练并测试完结束之后，共有12个score数据，通过model fusion.ipynb就可以把这些score整合起来，得到预测类别并保存到submission.csv中。

**3.3训练日志**

下面是我们训练过程中模型在训练集和验证集上的部分结果，我们使用的是VisualDL可视化工具，所有的日志文件我们存放在log文件夹中。

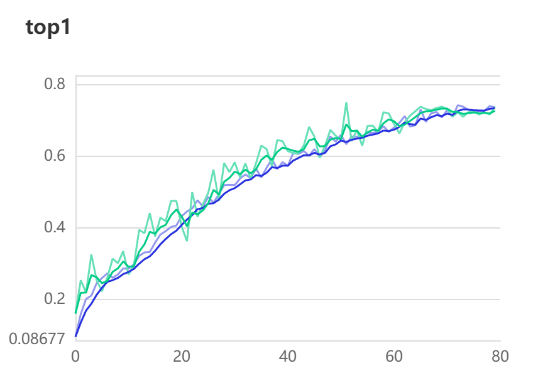
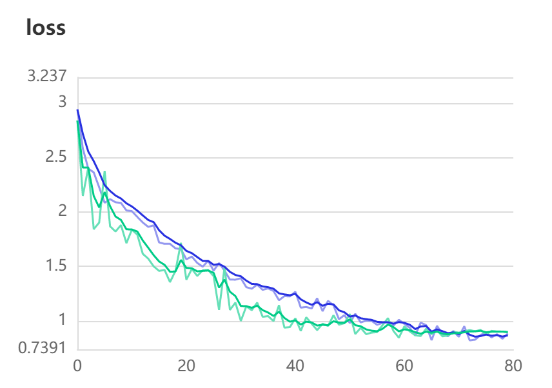


图2 STGCN-Joint模型训练过程中的误差和top1准确率的变化曲线，蓝色为训练集，绿色为验证集

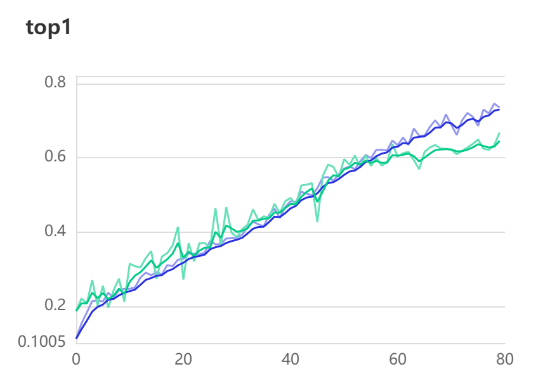
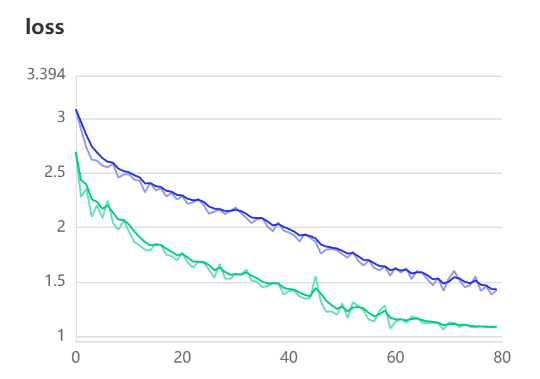


图3 AAGCN-Bone模型训练过程中的误差和top1准确率的变化曲线，蓝色为训练集，绿色为验证集

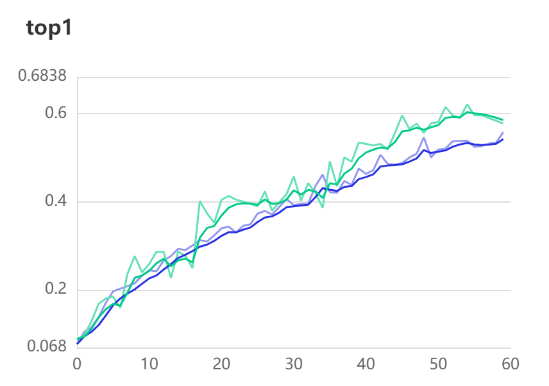
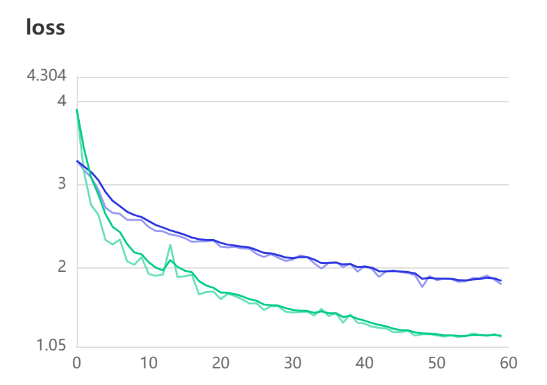


图4 MSG3D-Bone模型训练过程中的误差和top1准确率的变化曲线，蓝色为训练集，绿色为验证集