基于无人机的人体行为识别

国赛参赛文档

队伍编号：DIGIX2024TEAM130239

队伍名称：视觉计算与智能认知实验室

代码：https://github.com/superwuu/Behavior-Recognition-Pro

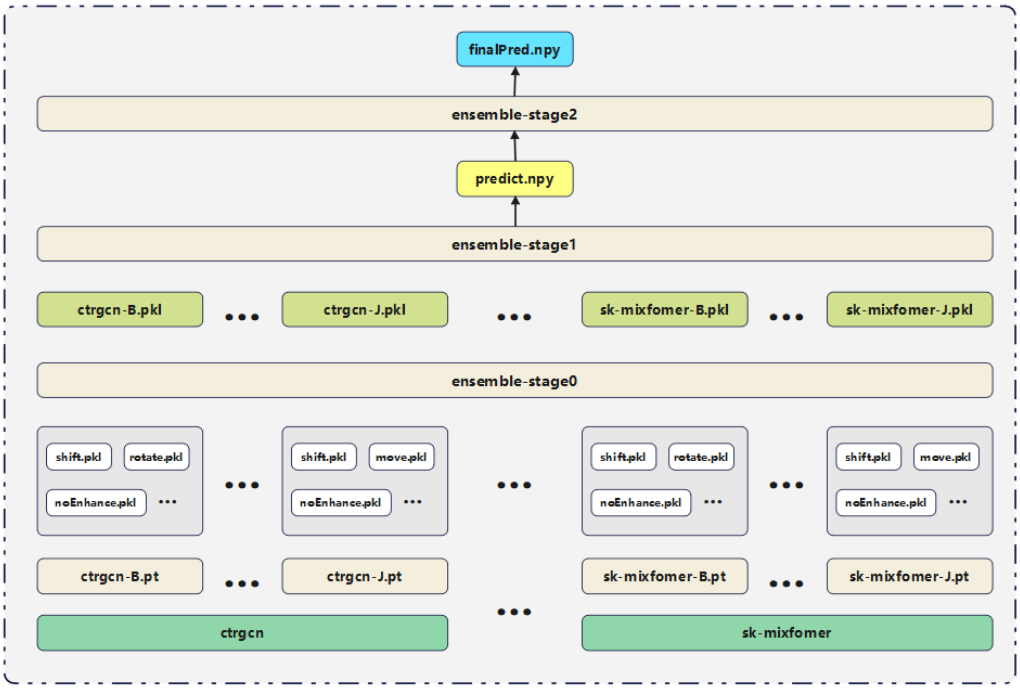
模型权重与推理结果百度网盘链接：https://pan.baidu.com/s/1EVJnTb9U2xShelBH7BboxQ?pwd=inuc

【简介】本次比赛方案以 top 仓库与 TE-GCN 仓库为代码基础构建方案，我们选择使用了 20 个模型、4 种模态及 4 种数据增强方法进行模型的训练与推理，而后使用三阶段多集成方案对不同模型的预测结果进行集成。我们主要的创新工作可总结为：（1）以使用多种不同数据增强的方式模拟使用大量的训练数据；（2）探究模型内外集成组合的方案效果，确定了模型内部集成与多模型集成的集成方案；（3）创新了自适应模型集成权重选择算法，更加合理与高效地进行模型集成；（4）创新清晰集与模糊集分类设想，提高了模型的预测准确率。

一、架构说明

本次比赛我们小组的算法架构如下图所示，在模型训练推理阶段，我们在训练集上对多个模型使用不同模态、不同数据增强方式进行训练后得到 pt 权重文件，而后对 val 数据集与 test 数据集进行推理进行 pkl 文件。在模型集成阶段，我们先将不同模态模型的以

多种数据增强方式得到的推理结果进行小集成，得到每个模型的初步集成结果（Stage0）。而后使用创新的自适应模型权重选择算法对 Stage0 的输出结果进行集成，得到初步输出结果（Stage1）。而后使用创新的清晰集模糊集分类方法对预测结果再次进行优化，得到最终的预测结果（Stage2）。



1.1 模型训练与推理

经过省赛的较量与对国赛数据集进行初步实验，我们发现的一般规律为模型越多、数据量越大，得到的预测准确率也越高。借助 top 仓库与 TE-GCN 仓库，我们在省赛中已使用了 20 个模型，模型数据已足够多，故我们在国赛时更多考虑的优化方案是如何增加训练数据。由于比赛规定不可使用外源数据，我们想到了可以对训练数据进行不同方式的增强，这样可以在一套数据集的基础上“分身”出多套不同的数据，而后进行集成，以此模拟使用大量的数据

进行训练。

为此，我们选择使用四种增强方式，即 random-shift、random-move、random-rotate、random shift & move 等，他们与不进行数据增强一起构成了我们的数据增强选择范围。我们选择的模型与数据增强的组合方式如下图所示。

分类	模型	模态	不增强	random shift	random move	random rotate	random shift & move
Mix_GCN	ctrgcn	joint	☑	☑	☑		☑
		bone	☑	☑	☑	☑	☑
		joint-motion					☑
		bone-motion					☑
	mstgcn	joint	☑	☑	☑		☑
		bone	☑	☑	☑	☑	☑
		joint-motion					☑
		bone-motion					☑
	tdgcn	joint	☑	☑	☑		☑
		bone	☑	☑	☑	☑	☑
		joint-motion					☑
		bone-motion					☑
Mix former	skmixf	joint	☑	☑	☑	☑	☑
		bone	☑	☑	☑	☑	☑
		joint-motion					☑
		bone-motion					☑
	skmixf k2	joint	☑	☑	☑	☑	☑
		bone	☑	☑	☑	☑	☑
tegcn	tegcn	joint	☑			☑	☑
		bone	☑			☑	☑

由上图所示，我们一共选择了 **61** 种组合，对每个组合进行训练后在 val 数据集与 test 数据集上进行推理，得到各自的推理结果。

1.2 模型集成

得到了训练并推理完毕的 61 个 pkl 文件的推理结果后，需要对

多模型的推理结果进行集成。我们在对国赛数据集的实验中发现，若直接对所有文件进行一次集成，并不能得到最佳的效果，为此我们提出了多层次集成方法对结果文件进行处理。在集成的 Stage0 阶段，我们为模型的预测结果进行初步集成，为每个模型输出了一个集成后的结果文件；在 Stage1 阶段，我们对 Stage0 阶段获得每个模型结果进行再次集成，得到预测的输出文件；在 Stage2 阶段，我们对输出文件再次进行筛选，选出模糊集样本，对其使用众数计算重新获取分类结果，最终结合后得到本次比赛的最终预测文件。

（1）Stage0 初步集成阶段

在集成的 Stage0 阶段，我们在 val 数据集上探索了多种组合方式，我们发现在模型内部进行集成比跨模型进行集成得到了更好的效果，因为最终选择了对每个模型集成出一个结果文件。以模型 ctrgcn-B 为例，我们将未增强、使用 random-shift 增强、使用 random-move 增强、使用 random-rorate 增强以及使用 random-shift & move 增强的五个结果文件进行集成，得到 ctrgcn-B 模型的预测文件。在集成方式的选择上，我们探索了平均集成与使用我们提出的自适应算法集成两种方式，发现若在模型集成的两个阶段均使用自适应算法，会导致模型因提前学习到了特例化样本而结果不佳，因此我们在 Stage0 阶段选择了平均集成的方式，即为每个集成的文件分配相同的权重结果。在经过 Stage0 后，得到 20 个模型的集成文件。

（2）Stage1 完整集成阶段

在集成的 Stage1 阶段，为了能够更加客观地评估模型的分类性

能，为不同的模型在集成过程中赋予不同的权重，我们提出了自适应模型集成权重选择算法，将反应模型一般分类性能的类别平均准确率（Accuracy Rate, $\boldsymbol{\mu}$）、反应模型精确分类性能的过滤筛选后类别平均准确率（After-filter Accuracy Rate, $\boldsymbol{AF\mu}$）与反应模型分类置信度的类别分类方差（V ariance，$\boldsymbol{\sigma^2}$）相结合，最终为每个模型输出一个值作为集成过程中该模型的权重。

具体而言，对于每个模型，我们先计算得到其分类分数最高的topK 个类别（k∈{1,3,5},k 为超参，默认为 1）。而后遍历 val 数据中每个样本，若其标签位于 topk 的分类中，则认为该模型在该标签的类别分类正确，该类别计数加一。最终得到一个长度为 155 的列表，记录该模型在每个类别上的对 val 数据集分类成功的概率，对列表计算平均值，即为$\boldsymbol{\mu}$；对列表计算方差，即为$\boldsymbol{\sigma^2}$。设置一个最低阈值（默认为 0.15），将低于该阈值的类别概率剔除，得到一个反应模型精确分类性能的列表，计算平均值，即为$\boldsymbol{AF\mu}$。

得到三个指标后，我们认为一个权重较大、性能好的模型应该同时具备$\boldsymbol{\mu}$与$\boldsymbol{AF\mu}$较大，即识别准确率较高的特点，以及$\boldsymbol{\sigma^2}$较小，即模型分类概率偏差小，性能较为稳定的特点。此外，为了能够学习到最佳的权重，我们为三个指标分别设立了超参数$\boldsymbol{x}$、$\boldsymbol{y}$、$\boldsymbol{z}$，故每个模型的最终权重计算公式为：

$$\boldsymbol{score} = \frac{\boldsymbol{\mu^x \cdot AF\mu^y}}{(\boldsymbol{\sigma^2})^{\boldsymbol{z}}}$$

为了得到$\boldsymbol{x}$、$\boldsymbol{y}$、$\boldsymbol{z}$这三个超参，我们选择网格搜索（Grid Search）与随机数搜索两种方式，构建模型的权重列表，在 val 数据

集上进行集成，记录得到最优结果的参数组合作为我们选择的超参数。在进行 Stage1 阶段后，得到 test 数据集的 predict.npy 预测结果文件。

（3）Stage2 集成后优化阶段

在集成的 Stage2 阶段，为了对预测结果进行进一步的增强，我们提出清晰集与模糊集的分类概念。清晰集即 test 数据集中分类置信度较高的样本，这类样本不需要再次进行分类核验，即认为当前的分类就是最终的预测结果。模糊集即 test 数据中分类置信度较低的样本，由于样本在分类时选择的是概率最大的类别，往往会出现分类错误但正确类别的分类概率与预测概率较为接近的情况，这样样本需要再次进行分类核验，以提高模型的预测准确率。在清晰集与模糊集的区分算法中，我们认为最佳的方式借助深度学习模型，在 val 数据集上进行一次半监督聚类算法，学习模型权重，而后在 test 数据集上进行分类。我们进行了初步实验，但由于国赛时间较为紧张，简单模型分类效果较差，故我们在分类上选择了随机抽样进行，设定模糊集数量为 1500。不可否认的是，我们认为自动化训练模型是一种难度更高但是更加高效与优化的方式，需要选择合适的聚类模型、监督信号以及更加优秀的训练技巧。

在筛选得到 1500 个模糊集样本后，我们对其中的每个样本进行模型投票操作，即以众数的方式选择最终的预测结果。我们将 20 个模型对模糊集中每个样本的分类结果进行汇总，剔除掉分类准确率较低的模型，选择剩余模型中对其预测最多的分类最后该模型的预

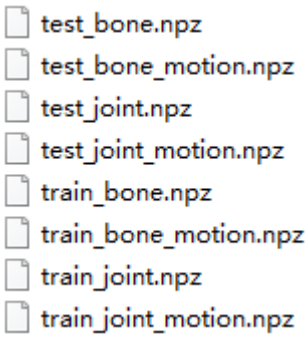
测结果。将模糊集与清晰集相结合，得到最终的输出预测结果。

二、使用说明

2.1 数据说明

我们按照数据说明将训练集、验证集和测试集数据都进行了模态的转换，而后将训练集数据与验证集数据进行绑定，生成符合模型训练所需的 npz 文件，转换代码见文件 data.ipynb。

```python
import numpy as np

train_data = np.load('data/train_joint_bone.npy')
train_label=np.load("data/train_label.npy")

test_data = np.load('data/val_joint_bone.npy')
test_label=np.load("data/val_label.npy")

print(train_data.shape)
print(train_label.shape)

print(test_data.shape)
print(test_label.shape)

arrays_dict = {
    'x_train': train_data,
    'y_train': train_label,
    'x_test': test_data,
    'y_test': test_label,
}

np.savez('train/train_joint_bone.npz', **arrays_dict)
```



转换完成后，得到的训练数据文件与测试数据文件见网盘下 data 文件夹。

2.2 训练与推理说明

比赛使用了两个开源仓库，分别为 ICME2024-Track10 仓库与 TE-GCN 仓库。在两个仓库下，我们修改了数据预处理代码，新增了四种数据增强方式。

在 Track10 仓库下，使用 Mix_former 的 6 个模态模型与 4 种数据增强方式的组合，共训练了 22 个模型，训练脚本为 Mix_former

目录下的 train.sh 文件；使用 Mix_GCN 的 12 个模态模型与 4 种数据增强方式的组合，共训练 33 个模型，训练脚本同样为其目录下 train.sh 文件。

在 TE-GCN 仓库下，使用两个模态模型与 4 种数据增强方式的组合，共训练 6 个模型，训练脚本为 script-train 文件下的文件。

训练完毕后得到各自模型的 pt 权重文件，所有的权重文件见网盘下 train-pt 文件夹。

在推理阶段，对于 Track10 仓库的两个系列模型，运行各自文件夹下的 testval.sh 与 test.sh 文件，分别得到 val 数据集与 test 数据集的推理结果；对于 TE-GCN 仓库，运行 script-test 文件夹下的文件，同样得到 val 数据集和 test 数据集上的推理结果。

推理完毕完得到的 val 数据集上推理结果见网盘下 pkl-val 文件夹，test 数据集上推理结果见网盘下 pkl-test 文件夹。

2.3 模型集成说明

模型集成的三个阶段代码见 ensemble 文件夹，stage0,1,2 分别对应集成三个阶段的集成代码。

在 stage0 文件夹下，我们完成了在 val 数据集与 test 数据集上对每个模型内部的不同数据增强方式间的初步集成，得到每个模型的 pkl 文件。处理前后的 pkl 文件分别见文件夹 stage0/orig 文件夹与 stage0/inter 文件夹。

在 stage1 文件夹下，getBestParameter 子文件夹对 val 数据集搜索出合适的模型集成参数，运行 inter_run2getBestParameter.sh 脚本

启动文件。getFinalPredict 子文件夹对 test 数据集进行推理，运行 inter_run2getFinalPredict.sh 文件启动文件，得到 pred.npy 文件。

在 stage2 文件下，所有的代码集成到 allinOne.py 文件夹下，按照步骤依次进行即可得到最终的预测文件。

```
if __name__ == "__main__":
    # 1.模型的推理npl转npy再转csv
    model_pkl2npy2csv()

    # 2.集成的推理结果文件获得类别csv
    pred_npy2csv()

    # 3.将集成的预测结果与每个模型的预测结果相结合
    merge_predWithModel()

    # 4.选择模糊集,输入集成的预测结果,返回索引列表
    get_vagueSet_index()

    # 5.从合并的csv中找到模糊集,保存成单独的csv
    get_vagueSet_csv(idx_res=get_vagueSet_index())

    # 6.取众数,完成最后的输出
    Rate=[]
    get_finalPred(Rate)

    # 7.在val上验证准确性
    get_acc()
```

在这一阶段，首先将模型集成的 pkl 文件转成各自的 csv 文件，该文件为模型对 test 数据集每个样本的分类，见网盘下 ensemble/csv 文件夹。而后将 stage1 推理的 pred.npy 同样转成 csv 文件，见 ensemble/pred.npy 文件。接着将两者的 csv 文件相互结合，得到 ensemble/mergedAllData.csv 文件。

在 get_vagueSet_index 函数下进行模糊集的筛选，而后将模糊集列表索引输入 get_vagueSet_csv 函数得到模糊集的 csv 文件。最后运行 get_finalPred 函数得到最终的预测输出结果。