省赛参赛文档

队伍编号：DIGIX2024TEAM130239

队伍名称：视觉计算与智能认知实验室

Github 链接：https://github.com/superwuu/Behavior-Recognition

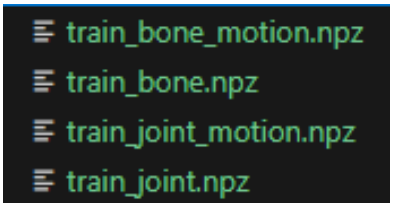
模型权重与推理结果百度网盘链接：https://pan.baidu.com/s/1d2Qv5nMdV1-B2IFo3nXHhg?pwd=inuc

我们队伍的参赛代码基于 MMVRAC 比赛的顶级仓库 https://github.com/liujf69/ICMEW2024-Track10，在此基础上进行提优和创新，结合 TE-GCN 代码仓库 https://github.com/xieyulai/TE-GCN，实现了本次省赛的代码。

比赛相关的代码和权重放置在 github 和百度网盘上，github 上保存比赛使用的仓库代码，分为训练模型部分与结果集成部分。百度网盘上为大文件，包括训练数据、模型权重与推理结果等。

数据说明：

```python
import numpy as np

train_data = np.load('train_joint_motion.npy')
train_label=np.load("train_label.npy")

test_data = np.load('test_joint_motion.npy')
test_label=np.load("test_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_motion.npz', **arrays_dict)
```



按照数据集说明将训练数据转化为多模态数据，我们一共使用了四种组合：bone、joint、bone_motion、joint_motion。而后与验证

集数据进行整合，我们使用 testA 数据集作为验证集，构成 npz 文件作为训练输入。见数据处理文件见 data.ipynb。

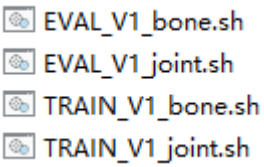
训练推理说明：

依据 ICME2024-Track10 仓库下的 README.md 文件与 tegcn 仓库下 README.md 文件对模型进行训练，自定义了训练方式，修改了模型结构。训练时，与原代码相比，我们额外使用了 random_shift 与 random_move 两种方式。

```
random_choose: False
random_shift: True
random_move: True
random_rot: False
```

在 Track10 仓库下，使用 Mix_former 的 6 个模态模型以及 Mix_GCN 的 12 个模态模型。训练脚本为 Model_inference/Mix_Former 目录下及 Model_inference/Mix_GCN 目录下的 train.sh 脚本，运行文件保存在各自的 output 文件夹下。

在 TE-GCN 仓库下，使用 joint 和 bone 两个模态模型。训练脚本为文件夹下的 TRAIN 系列 sh。



训练完成后得到每个最优的 pt 权重文件，见百度网盘中的 model_weight 文件夹。而后在 Top 仓库下的模型下运行 testB.sh 文件、在 TE-GCN 仓库下运行两个 EVAL 系统的 sh 文件，将 testB 数据集送入各自模型进行推理，得到各自的 pkl 文件，见网盘 testB 文件夹。

同时，收集训练过程的最优验证集结果，即为模型在 testA 推理的 pkl 文件，见网盘 testA 文件夹。

在对 testB 进行推理时，推理脚本为两个目录下的 testB.sh 脚本，结果保存在各自的 output-B 目录下。

而后对这 20 个模型进行集成。

模型集成说明：

集成阶段 1：

得到不同模型对 testA、testB 数据集的推理结果后，需要对它们进行模型集成。我们选择的方式为每个模型的结果赋予一个权重，对所有模型进行加权平均后输出最后的结果。

选择的权重为在 testA 上集成获得最好的结果，获得权重的过程见文件 get_weight.py。运行 run2getW.sh 脚本启动文件。在文件中，我们将每个模型对不同类别的检测结果计算出来，见 get_weights 函数。然后再计算平均值、标准差、以及删去一部分低置信度结果后的平均值，这三个值进行组合作为该模型的权重。此外，我们还使用了随机数的方法查找最优的权重。

```
for file in File:
    r,w,v=get_weights(file,val_txt_file,0.15,1)     # r:去掉低于阈值后的均值    w:所有类的均值      v:标准差
    score=(r ** x) * (w ** y) / (v ** z)
    Rate.append(score)
```

```
while True:
    Rate2=[]
    for it in Rate:
        tmp=random.uniform(0.0, 20000.0)
        Rate2.append(tmp)
    final_score = Cal_Score(File, Rate2, Sample_Num, Numclass)

    Acc = Cal_Acc(final_score, true_label)
```

得到每个模型的权重列表为：

```
Rate=[2866.446560756607, 0.0, 0.0, 0.0, 8062.399562235894, 1887.2742569619966,
      17344.990413912543, 12994.157501998963,
      14716.495604359188, 11192.194873406577, 147.20429556949034, 0.0,
      0.0,0.0, 0.0, 0.0,
      8777.016885449711, 1212.4997536194596, 0.0, 0.0 ]
```

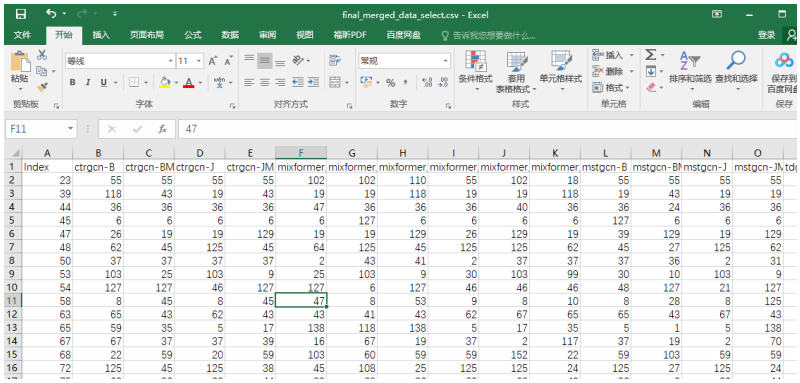
而后运行 run2getRes.sh 脚本，启动 ensemble_eval.py 文件，使用这套权重对 testB 进行推理，得到结果 pred.npy。

集成阶段 2：

此外，在对 testB 数据集进行推理后得到的 pred.npy 文件为数据样本在不同分类下的置信度。我们发现某些样本在多个行为类别下的置信度较为接近，而每个类只选择置信度最大的一个作为其最终结果，故有可能出现识别错误的情况。为此，我们将样本分为了清晰集与模糊集，清晰集中为某一分类置信度远大于其他类的样本，模糊集中为多个分类的置信度相近的样本。

对于模糊集，我们采用投票的方式。我们使用了 20 个模型进行集成，每个集成的模型对这个样本的最终分类都有影响，影响权重为模型权重。对模糊集中所有样本的分类求众数，即为该样本的最终结果。

在此阶段，我们将 output-B 中的各个模型 pkl 权重转化为 npy 文件，见百度网盘 1028res 文件夹，再将它们转化为 csv 文件，即为每个模型对 4599 个样本在每个类别上的置信度，见百度网盘 1028csv 文件夹，每个文件的维度为[4599,155]，通过 npy2csv_20model.py 操作得到动作分类文件。经过手动筛选出包含 935 个样本的模糊集，将每个样本的在测试集的索引 Index，和 20 个模型的动作分类 label 整合到同一个 csv 文件 final_merged_data_select.csv 中，如下：



Index	ctrgcn-B	ctrgcn-BM	ctrgcn-J	ctrgcn-JM	mixformer	mixformer	mixformer	mixformer	mixformer	mixformer	mstgcn-B	mstgcn-BM	mstgcn-J	mstgcn-JM
23	55	55	55	55	102	102	110	55	102	18	55	55	55	55
39	118	43	19	43	19	19	118	19	19	118	19	43	19	19
44	36	36	36	36	47	36	36	36	40	36	36	24	36	36
45	6	6	6	6	6	127	6	6	6	6	127	6	6	6
47	26	19	19	129	19	19	129	26	129	19	39	129	19	129
48	62	45	125	45	64	125	45	125	125	62	45	27	125	62
50	37	37	37	37	2	43	41	2	37	37	37	36	2	31
53	103	25	103	9	25	103	9	30	103	99	30	10	103	9
54	127	127	46	127	127	6	127	46	46	46	48	127	21	127
58	8	45	8	45	47	8	53	9	8	10	8	28	8	125
63	65	43	62	43	43	41	43	62	67	65	65	43	67	43
65	59	35	5	17	138	118	138	5	17	35	5	1	5	138
67	67	37	37	39	16	67	19	37	2	117	37	19	2	70
68	22	59	20	59	103	60	59	59	152	22	59	103	59	59
72	125	45	125	38	45	108	25	125	125	24	125	27	125	24

该文件包含 21 列，第一列是索引，第 1-21 列是通过 20 个模型得到的动作 label；有 935 行，代表 935 个模糊样本。将每一个模糊样本的最终动作结果确定为 20 个模型动作 label 的众数。再找其中一个模型动作为众数（我们选择 final_merged_data_select.csv 每一行中第一个动作 label 为众数的那个模型的置信度）的置信度乘以 20 个模型权重超参数 Rate 的均值,从而得到模糊样本的最终置信度。以上操作详见 vague_set_optimize.py，从而得到最后提交的 pred.npy 结果文件。