# 机械手势识别

本项目一共提供了两套手势识别数据集，一套是基于网上的偏卡通的手势数据集，另外一套是自己采集的机械手手势数据。第一套数据集存放在./Hand/hand1文件夹下，第二套数据集存放在./Hand/hand2文件夹下。

## 数据处理

数据处理包含数据集主要包含数据集划分和自定义数据加载Dataset类。

第一套数据集划分使用./Hand/data/data\_split.py代码，第二套数据集划分采用./Hand/data/data\_split\_hand2.py代码。执行这个两个代码中的一个，将会在同级目录下生成train.txt和test.txt两个文本文件。

数据加载类分别写在./Hand/model\_hand\_1/dataset.py和./Hand/model\_hand\_2/dataset.py可以按照自己的需求来更改数据加载方式。

当前的图像训练都是将图像放缩到100 x 100的大小，可以自定义调整，调整方式比较简单。

## 模型构建

目前两套数据集共用一个模型，也就是./Hand/model\_hand\_1/model.py和./Hand/model\_hand\_2/model.py是一样的。为了方便更高分辨率图像的训练和预测，在./Hand/model\_hand\_2/model.py中提供了**ResNet50**的模型，可以便于使用更加深层模型。如果将图像放缩到224 x 224大小的话，可以使用该程序下面的预训练模型权重进行训练，能更快加速模型的收敛速度。

预训练模型调用加载如下所示：

resModel = ResModel(classes=6)  
new\_state = {}  
weight\_state = torch.load('./pretrained/resnet50-19c8e357.pth')  
for key, value in weight\_state.items():  
 if 'fc' not in key:  
 new\_state['model.' + key] = value  
 else:  
 new\_state['model.' + key] = resModel.state\_dict()['model.' + key]  
resModel.load\_state\_dict(new\_state)

## 模型训练

模型训练的话可以直接执行./Hand/model\_hand\_1/train.py或./Hand/model\_hand\_2/train.py文件，这两个训练文件分别是训练第一套数据集或第二套数据集，但一定要注意的是./Hand/data/train.txt和./Hand/data/train.txt是否和数据集对应上，如果没对应上则可能出现数据集无法正确加载错误。

两个train.py文件主要需要修改的超参数包含如下：

train\_transform = transforms.Compose([  
 transforms.RandomRotation((-30, 30)),  
 transforms.Resize((100, 100)),  
 transforms.RandomHorizontalFlip(),  
 transforms.CenterCrop((100, 100)),  
 transforms.ToTensor()  
])  
test\_transformer = transforms.Compose([  
 transforms.Resize((100, 100)),  
 transforms.CenterCrop((100, 100)),  
 transforms.ToTensor()  
])  
  
batch\_size = 8  
learning\_rate = 1e-3  
epochs = 100

数据增强的旋转方式、图像放缩的尺度变化、图像剪裁的尺度、batch\_size超参数的调整、learning\_rate超参数的调整、epochs超参数的调整。

## 软件使用

在这个项目中，我们也制作了一个可视化的软件，便于项目成果的演示，主要包含这几个文件，具体如下下图1所示：

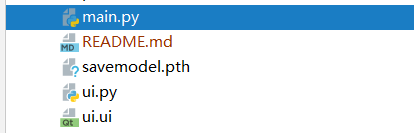


图1 软件包含的代码文件

主要需要保留的有三个文件：main.py、ui.py和savemodel.pth三个文件，后续可以打包成exe可执行程序，便于在其他电脑上进行使用。

软件界面如下所示：



图2 软件操作界面

软件支持两种数据选择方式，包含摄像头捕捉图像和电脑磁盘读取图像两种方式。选择/捕捉到对应的图像后将会在界面的图像框中进行显示。显示后便可以点击**开始预测按钮**，则软件开始加载savemodel.pth进行图像预测，并将模型预测的结果显示在软件的右上角方框中。

如果需要改变软件的窗口显示结果，则按照以下教程进行安装**pyqt5**，具体见：<https://blog.csdn.net/qq_45041871/article/details/113775749>。

如需打包程序成**exe**文件，只需在**terminal**跳转到**main.py**对应的根目录，然后执行命令**pyinstaller -F main.py**，等程序执行结束，将在在**main.py**的同级目录生成一个**dist**文件夹，文件夹中存在一个**main.exe**文件，然后将**savemodel.pth**存放在该**dist**文件夹下就可以执行**main.exe**文件。

# EMG识别

EMG手势识别也提供两份数据集，一份是公开的SIA\_delsys\_16\_movements\_data数据，另外一份是自己使用一个肌肉传感器采集的数据。两个数据集也存在不一样的数据处理方式，需要以一一对应，否则将会先问题。

## 数据处理

对于公开的SIA\_delsys\_16\_movements\_data数据，我们采用./EMG/data/data.py进行数据处理，处理后将会得到image\_feature.h5和time\_feature.h5文件。同样采用./EMG/data/process\_sku\_data.py进行数据处理，处理后将会得到image\_feature.h5。这是模型的输入文件，必须严格控制其输入格式。

对于数据加载Dataset类，我们在./EMG/data/dataset.py中进行实现，可以根据自己的需求进行实现。

采样数据我们一般放在./EMG/data/sku\_data文件夹存放数据。

## 模型训练

模型要训练的话可以直接执行./EMG/data/train.py文件，需要注意的是模型加载的数据需要修改、模型预测种类的数量需要修改，可以按照自己的需求进行修改。

我们这里针对开放的数据集提供了四种模型，均存放在./EMG/model文件夹目录下，分别是一维卷积（TextCNN）conv1D.py、二维卷积conv2D.py、随机森林randomForest.py、支持向量机svm.py。其中一维卷积conv1D.py可能需要更改某些参数，来满足你的模型需求。对于在自己采集的数据集，则仅有conv1D.py模型可以支持使用，目前只有一个数据采样器，如果后面会有扩展的话，可以支持多种方式实现。

重要！！！train.py中load\_image\_feature函数中**/100**需要和process\_sku\_data.py中的**>100**对应上。predict.py和inference.py中的sample\_data函数也是一样的，需要对应上。

## 模型推理

模型推理主要参考predict.py和inference.py文件。主要需要修改**arduino**板子的端口和波动率，模型加载权重路径。

# 训练曲线查看

打开tensorboard需要先跳转到log对应的根目录下，然后执行 tensorboard --logdir=log命令即可。

# 卷积计算公式

stride(步长)、input\_shape(输入图像大小)、padding(填充)、kernel\_size(卷积核大小)、output\_shape(输出维度)

eg. input\_shape=(6,6)、kernel\_size=(3,3)、padding=(1,1)、stride=(2,2)

结果：output\_shape=(4,4)