


```

        zoom_range=0.2,
        horizontal_flip=True
    )
train_gen = train_datagen.flow_from_directory(
    r'/home/wsuser/work/Finger Dataset/train',
    target_size=(128,128),
    color_mode='grayscale',
    batch_size=32,
    classes=['0','1','2','3','4','5'],
    class_mode='categorical'
)

test_datagen = ImageDataGenerator( rescale=1./255 )
test_gen = test_datagen.flow_from_directory(
    r'/home/wsuser/work/Finger Dataset/test',
    target_size=(128,128),
    color_mode='grayscale',
    batch_size=32,
    classes=['0','1','2','3','4','5'],
    class_mode='categorical'
)

model=Sequential()
model.add(BatchNormalization(input_shape = (128,128,1)))
model.add(Convolution2D(32, (3,3), activation = 'relu', input_shape = (128, 128, 1)))
model.add(MaxPooling2D(pool_size=2))
model.add(Convolution2D(filters=6,kernel_size=4,padding='same',activation='relu'))
model.add(MaxPooling2D(pool_size=2))
model.add(Convolution2D(filters=128,kernel_size=3,padding='same',activation='relu'))
model.add(MaxPooling2D(pool_size=2))
model.add(Convolution2D(filters=128,kernel_size=2,padding='same',activation='relu'))
model.add(MaxPooling2D(pool_size=2))
model.add(Flatten())

```

```

model.add(Dense(units=128,activation = 'relu'))
model.add(Dense(units = 64, activation = 'relu'))
model.add(Dense(units = 32, activation = 'relu'))
model.add(Dense(units = 6, activation = 'softmax'))
model.summary()
model.compile(optimizer='adam', loss = 'categorical_crossentropy',metrics = ['accuracy'])
model.fit_generator(train_gen,
                    epochs=20,
                    steps_per_epoch=18000//32,
                    validation_data=test_gen,
                    verbose = 1,validation_steps=3600//32)
model.save('gesture.h5')
wml_credentials={
    "url":'https://us-south.ml.cloud.ibm.com',
    "apikey":'on6wVLLy-ERS74JlvDrFdJ35GRaHzaCtKxejqR7euwG'
}
client=APIClient(wml_credentials)

```

```

def guid_from_space_name(client,space_name):
    space=client.spaces.get_details()
    return(next(item for item in space['resources'] if
item['entity']['name']==space_name)['metadata']['id'])

```

```

space_uid=guid_from_space_name(client,'Gesture_Deploy')
client.set.default_space(space_uid)

```

```

software_spec_uid=client.software_specifications.get_uid_by_name('tensorflow_rt22.1-py3.9')

```

```

!tar -zcvf gesture_based_tool.tgz gesture.h5

```

```
model_details=client.repository.store_model(model='gesture_based_tool.tgz',meta_props={
    client.repository.ModelMetaNames.NAME:"Gesture Based Tool",
    client.repository.ModelMetaNames.TYPE:"tensorflow_2.7",

    client.repository.ModelMetaNames.SOFTWARE_SPEC_UID:software_spec_uid
    }
)

model_id=client.repository.get_model_id(model_details)
```

Footer

© 2022 GitHub, Inc.

Footer navigation

Terms

Privacy

Security

Status

Docs

Contact GitHub

Pricing

API

Training

Blog

About