## 代码部署

将<https://github.com/Winnie-Qi/Surname-Classification-with-RNNs/blob/main/Surname-Classification-with-RNNs.py> 移植至本地环境执行。数据集surnames\_with\_splits.csv在学习通-资料-数据。

以下两种部署方式可选：

1、保留Surname-Classification-with-RNNs.py文件，将运行结果截图放至README.md中。

2、新建Surname-Classification-with-RNNs.ipynb文件（需合理划分代码单元格），移植Surname-Classification-with-RNNs.py的内容，提交时结果应显示在单元格下方。无需README截图。

## 3、数据预处理验证

# 在SurnameVectorizer初始化后添加

print("Char Vocab特殊标记索引：",

f"<BEGIN>:{vectorizer.char\_vocab.begin\_seq\_index}",

f"<END>:{vectorizer.char\_vocab.end\_seq\_index}")

# 打印样本向量化结果

sample\_str = "Zhang"

vec, length = vectorizer.vectorize(sample\_str)

print(f"样本'{sample\_str}'的向量化结果:\n{vec}\n有效长度:{length}")

## 4、模型结构验证

输出模型各层级参数维度：

# 打印模型各层参数维度

print("\n模型结构：")

for name, param in classifier.named\_parameters():

print(f"{name.ljust(20)} | 维度：{tuple(param.size())}")

预期输出示例：

emb.weight | 维度：(字符表大小, 100)

rnn.rnn\_cell.weight\_ih | 维度：(64, 100)

rnn.rnn\_cell.weight\_hh | 维度：(64, 64)

fc1.weight | 维度：(64, 64)

fc2.weight | 维度：(国家数, 64)

观察点：

* 嵌入层维度是否正确（num\_embeddings应为字符表大小）
* RNN输入/隐藏层维度是否匹配
* 全连接层维度是否合理

## 5、RNN序列处理验证

# 在forward方法中插入调试信息

print(f"输入序列形状：{x\_in.shape} -> 嵌入后：{x\_embedded.shape}")

print(f"RNN输出形状：{y\_out.shape} -> 聚合后：{y\_out[:, -1, :].shape if x\_lengths is None else '动态索引'}")

观察点：

* 理解序列长度如何影响column\_gather操作
* 验证时间步维度是否被正确处理

## 6、模型性能评估

训练50个epoch后，输出模型在**测试集**上的损失值和分类准确率指标。

## 7、推理能力验证

在模型推理验证（Inference）阶段，输出“McMahan”，“Nakamoto”，“Wan”，“Cho”这四个姓的国籍分类预测结果及其对应的概率置信度。

推理函数predict\_nationality已给出，可进行调用。

## 8、成果提交

保存文件或截图，等待与下周内容一起上传。