作业1：基于CNN的高频交易噪声过滤与多空信号识别（10%）

数据集：

以Kaggle竞赛数据Jane Street数据为例

数据集链接：<https://aistudio.baidu.com/datasetdetail/300513>

如若数据量过大导致电脑卡顿，可使用前个三部分的partition训练即可，请同学们自行斟酌。

任务目标

核心能力训练：掌握CNN在时序数据处理中的特征提取与降噪方法，理解高频交易数据的特性与挑战；

金融工程思维培养：建立从数据清洗、特征工程到策略回测的完整量化研究流程，初步具备构建金融智能策略系统的能力。

作业内容

1. 数据集与预处理要求

数据来源：Kaggle Jane Street Market Prediction数据集（高频交易匿名数据，含feature\_00至feature\_78市场特征及responder\_6目标变量）；可只选取前三个部分的partition作数据输入

* + 1. 关键预处理步骤：
* 缺省值处理：在缺省值处理阶段，可先统计并打印各特征的缺失比例，辅助判断处理策略。对缺失较严重的特征，视其重要性决定保留或剔除；对少量缺失的特征，则结合数据分布采用中位数或合理推断方式填充，以兼顾数据完整性与模型稳定性特征值选取；（也可使用其他方法）
* 噪声标注：噪声标注通过滑动窗口计算局部标准差，识别波动率异常的数据段。具体来说，当窗口内的波动超过设定阈值（如3倍标准差）时，标记该区间为噪声，以便后续模型重点处理有效信号。
* 时序标准化：对时序数据应用Z-score标准化，消除均值和方差的影响，再结合分位数归一化处理极端值，从而统一不同特征的尺度，提高数据的可比性和模型的鲁棒性。
* 特征重组：通过将数据按固定时间窗口切割，再利用重塑操作将每个窗口内的多维特征转化为二维矩阵，最后增加通道维度，构造符合卷积神经网络输入要求的三维张量。
  + 1. 参考代码框架

*# data\_preprocessing.py*

def load\_data(base\_path, k):

*# TODO: 读取k个数据分区并合并*

    pass

def preprocess\_data(df, target\_col):

*# TODO: 统计缺失，处理缺失，筛选特征列和目标列*

    pass

def mark\_noise(features, window\_size, step, std\_multiplier):

*# TODO: 根据窗口内波动标注噪声标签*

    pass

def create\_image\_data(df, feature\_cols, target\_col, window\_size, step, img\_size):

*# TODO: 标准化，滑窗重塑，生成图像张量及标签*

Pass

def main():

*# 1. 加载数据*

    df = load\_data(base\_path="数据路径", k=？)

*# 2. 数据预处理*

    df, feature\_cols, target\_col = preprocess\_data(df, target\_col='responder\_6')

*# 3. 构造图像数据（包含噪声标签）*

    X\_images, y\_target, y\_noise = create\_image\_data(df, feature\_cols, target\_col,

window\_size=？, step=？, img\_size=(？,？))

1. 模型设计指导
   * 1. 基础架构设计（必做）

1）设计多尺度卷积降噪模块：采用 3×3 和 7×7 卷积核提取不同时间尺度的特征，通过 Concatenate 拼接融合，增强特征表达能力。

2）引入通道注意力机制：使用 SE 模块（Squeeze-and-Excitation） 对通道进行加权，突出有效特征，抑制噪声干扰。

3）构建双输出结构：分别预测目标值和噪声水平，实现主任务和辅助任务的联合建模，提升模型鲁棒性。

参考示例

def build\_multiscale\_cnn(input\_shape):

    # 输入层

    inputs = Input(shape=input\_shape)

    # 多尺度卷积特征提取

    feature\_small = Conv2D(filters=..., kernel\_size=(3,3), activation='relu', padding='same')(inputs)

    feature\_large = Conv2D(filters=..., kernel\_size=(7,7), activation='relu', padding='same')(inputs)

    # 特征拼接融合

    combined\_features = Concatenate()([feature\_small, feature\_large])

    # 注意力机制（SE模块）加权通道特征

    se\_vector = GlobalAveragePooling2D()(combined\_features)

    se\_vector = Dense(units=..., activation='relu')(se\_vector)

    se\_vector = Dense(units=..., activation='sigmoid')(se\_vector)

    se\_vector = Reshape(target\_shape=(1,1, ...))(se\_vector)

    weighted\_features = Multiply()([combined\_features, se\_vector])

    # 进一步卷积提取

    refined\_features = Conv2D(filters=..., kernel\_size=(3,3), activation='relu', padding='same')(weighted\_features)

    pooled\_features = GlobalAveragePooling2D()(refined\_features)

    # 多任务输出设计

    output\_target = Dense(units=1, activation='sigmoid', name='target\_output')(pooled\_features)

    output\_noise = Dense(units=1, activation='sigmoid', name='noise\_output')(pooled\_features)

    # 模型构建

    model = Model(inputs=inputs, outputs=[output\_target, output\_noise])

    return model

1. 实验分析与报告
   * 1. 分析目标

* 使用已训练好的多输出CNN模型，对测试集进行预测并量化评估模型性能。
* 鼓励通过调整模型参数或训练方法，观察指标变化，理解参数对模型效果的影响。
* 结合金融实际场景，分析评估指标对业务的含义。
  + 1. 量化指标：

参考示例：

准确率（Accuracy）精确率（Precision）召回率（Recall）

F1分数（F1 Score）AUC值（ROC AUC） 混淆矩阵（Confusion Matrix）

* + 1. 分析方法：

参考示例：

基于已有模型，尝试调整训练中的关键参数（如学习率、卷积核大小、批大小等）或使用不同的数据增强方法。

重新训练并评估模型，比较各指标的变化。

简单记录参数变化与指标的对应关系。

分析最优模型训练结果。

简述各指标在金融风险管理、交易信号识别等场景的意义。

分析模型性能对实际业务的潜在影响，如误判带来的风险和成本。

作业提交要求（word或pdf）

代码：使用TensorFlow/Kera等；

报告包括：引言、设计思想、流程图（模块介绍）、实现代码、实验结果、结果分析、总结、参考资料

本实验指导书只是提供思路指引，有能力的同学也可以自行定义分析方法并结合实际金融场景进行分析，只要有完整合理的说明和对应的结果截图即可。