**染色体深度学习报告**

**项目目标：**

求：需要一个算法，进行分类或者检测。算法输入测试数据(暂未提供)，输出测试数据分类或检测结果，要求①测试数据中分类或检测出所有dic数据且假阳性率低于40%，②ace(f)的假阳性率<30%。要求①的重要性高于要求②

**数据集描述：**

数据集包含六种类别的染色体图像，包括dic, ace, normal, min, r, tri。其中，dic和ace类别对于实验目的尤为重要。数据集已被分为训练集、验证集和测试集。

**实验环境：**

Python, Anaconda, Visual Code, TensorFlow

**分割数据集:**

在给出的染色体数据集中有6个子数据集。我将其分成60%作为训练集，百分之20%作为验证集，剩下20%作为训练集使用。

分割后的训练集和验证集的数量：

Found 28795 images belonging to 6 classes.

Found 9599 images belonging to 6 classes.

**选择模型：**

在选择模型中，我选择的是CNN神经网络模型，因为它在处理复杂的图形上面具备良好的性能。

**建立虚拟环境**：

由于tensorflow2.0对python以及各个包的版本都有要求。我这里使用Anaconda创造的环境配置如下：tensorflow:2.04, python：3.819, numpy:1.19.5。

**网络架构：**

模型采用了序贯模型结构：

因为数据集中的图像都为灰色图像，所以在输入层中图像只有一个颜色通道。

三个卷积层，每层后接Batch Normalization和LeakyReLU激活函数，用以提取图像特征。

每个卷积层后跟一个最大池化层，以减少参数数量和计算量。

一个全连接层，用以学习非线性组合。

Dropout层，用于减少过拟合。

输出层为一个具有softmax激活的全连接层，用于输出六个类别的概率。

初步采用32-64卷积核，以及64神经元，降低训练成本和时间并防止过拟合。

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Layer (type) Output Shape Param #

=================================================================

conv2d (Conv2D) (None, 148, 148, 32) 320

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

batch\_normalization (BatchNo (None, 148, 148, 32) 128

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

leaky\_re\_lu (LeakyReLU) (None, 148, 148, 32) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

max\_pooling2d (MaxPooling2D) (None, 74, 74, 32) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

conv2d\_1 (Conv2D) (None, 72, 72, 64) 18496

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

leaky\_re\_lu\_1 (LeakyReLU) (None, 72, 72, 64) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

max\_pooling2d\_1 (MaxPooling2 (None, 36, 36, 64) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

conv2d\_2 (Conv2D) (None, 34, 34, 64) 36928

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

leaky\_re\_lu\_2 (LeakyReLU) (None, 34, 34, 64) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

max\_pooling2d\_2 (MaxPooling2 (None, 17, 17, 64) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

flatten (Flatten) (None, 18496) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

dense (Dense) (None, 128) 2367616

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

leaky\_re\_lu\_3 (LeakyReLU) (None, 128) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

dropout (Dropout) (None, 128) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

dense\_1 (Dense) (None, 6) 774

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

activation (Activation) (None, 6) 0

=================================================================model.

**损失函数和优化器**：

损失函数：使用定制的Focal Loss，以便更好地处理类别不平衡问题，关注那些难以分类的样本。

优化器：RMSprop

**训练过程：**

使用ImageDataGenerator进行图像的实时数据增强，包括旋转、平移、缩放和水平翻转。

使用混合数据生成器，以不同权重从多数类和少数类生成器中获取数据，以处理类不平衡。

**类权重**：  
为了解决数据不平衡的问题，特别增大了少数类（min, r, tri）的类权重，使模型在训练过程中更多地关注这些少数类。

**训练分析：**

loss: 5.4395 - accuracy: 0.4567- precision: 0.23631 - recall: 0.0098 - auc: 0.64382 - val\_loss: 6.4323 - val\_accuracy: 0.5240 - val\_precision: 0.3743 - val\_recall: 0.0042 - val\_auc: 0.5461

loss: 0.6387 - accuracy: 0.7016 - precision: 0.7016 - recall: 0.7016 - auc: 0.8218 - val\_loss: 0.6423 - val\_accuracy: 0.7141 - val\_precision: 0.7141 - val\_recall: 0.7141 - val\_auc: 0.8290

训练过程显示模型在训练集上的性能逐渐提升，但在验证集上的表现并未表现出明显的进步。最终的测试结果显示，尽管精度较高，但召回率较低，尤其是对于关键类别dic和ace。数据集中除了’dic’和‘ace’，其余四个数据集都非常小。准确率不适合作为主要评估指标。

**结论：**

1. 目前模型在处理不平衡数据集方面仍有改进空间。可能需要进一步调整Focal Loss参数或尝试其他策略来处理类不平衡。
2. 该模型训练时间过长，可以尝试一些调整去降低训练成本和时长。