# Introduction:

The data set of this project focusing on Colon Pathology samples, originally from MedMNIST, released on Scientific Data in 2023. The data set including the pictures of different tissues, including adipose, background, debris, lymphocytes, mucus, smooth muscle, normal colon mucosa, cancer-associated stroma, colorectal adenocarcinoma epithelium. The data set is provided as training data for model training, contributing to the establishment of advance auto cancer identification system.

This project uses three different technics, including \_\_\_\_\_\_\_\_\_\_, MLP and CNN, in training auto picture classification models. Aim to compare the difference of these technics, their advantages, disadvantages and features. Finally we are going to analysis their strength and suitability in this scenario.

Comparing these three different models, from 无参模型 to 深度卷积, can help us to understand the evolution of AI technics and application as well. Especially in medical fields, which model can better classify images is a attention attracting issue. Which will also influence the future progress of medical industry.

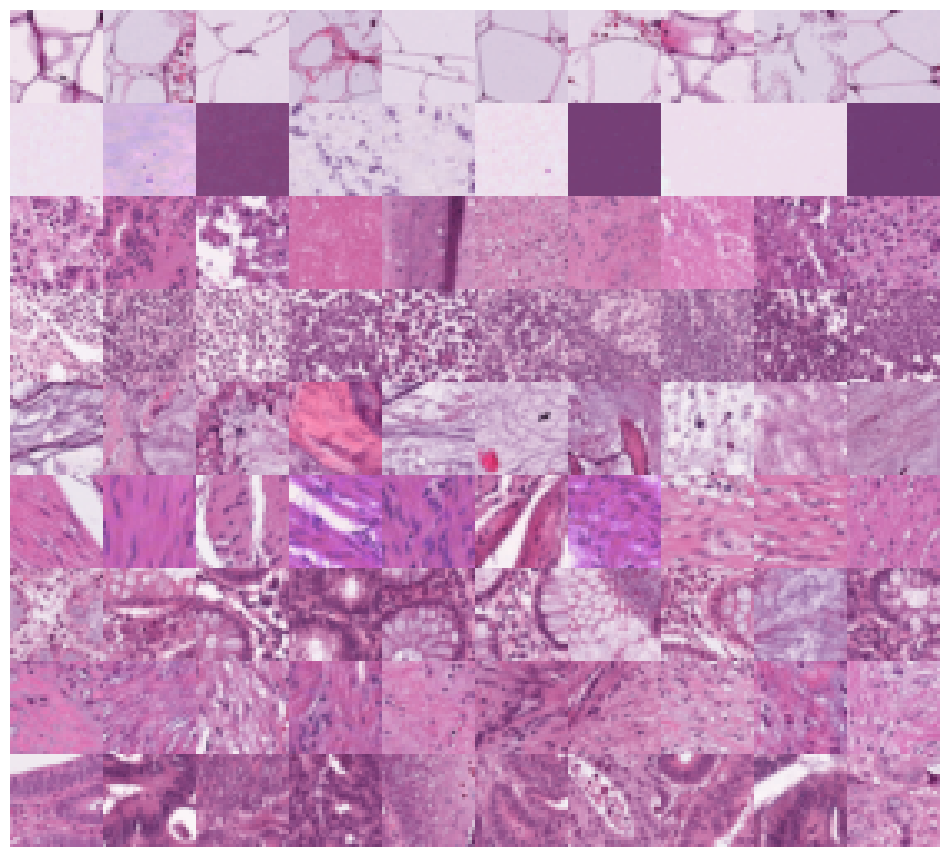
Project includes following sections: data preprocessing, model design, hyperparameter tuning and comparing.

# Data exploration:

原始样本包含training 89996,validation 10004, test 7180, 由学校提供的版本仅包含training 32000条和test 8000条。

所有的样本被分为9种类别，class0-class8.

**Data Modality:** Colon Pathology，



图为试输出的数据图片，从上到下分别为Label0-label8, label0 为adipose, label1为background,label2 为debris, label3 为lymphocytes， label4为mucus, label 5为 smooth muscle， label6为normal colon mucosa，label7 为 cancer-associated stroma，label8为 colorectal adenocarcinoma epithelium。

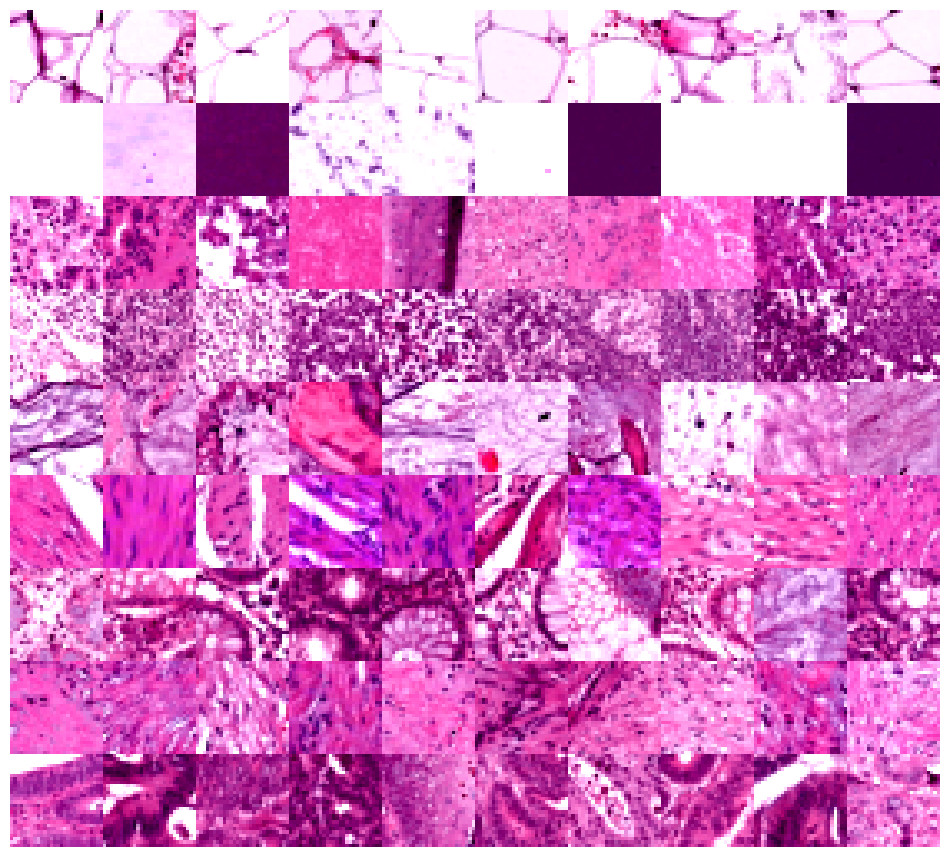
通过观察输出图片了解到，background类别内存在显著的颜色差异，可能造成捕捉特征的困难; smooth muscle 类别有些样本的颜色特别的鲜艳; normal colon mucosa、cancer-associated stroma和colorectal adenocarcinoma epithelium类别之间，有些样本无法通过人眼的观察分辨出其区别，这些样本的特征并不明显，可能对模型的训练造成阻碍。

# Preprocessing

1. 首先考虑到样本图片的主要色调差异不大，考虑通过强化对比度来强化其特征。

操作：线性拉伸 (X−127.5)\*2+127.5，将灰度动态范围放大到 [0,255]

预期效果：让暗区更暗、亮区更亮，增强不同组织间的整体灰度/颜色差异.

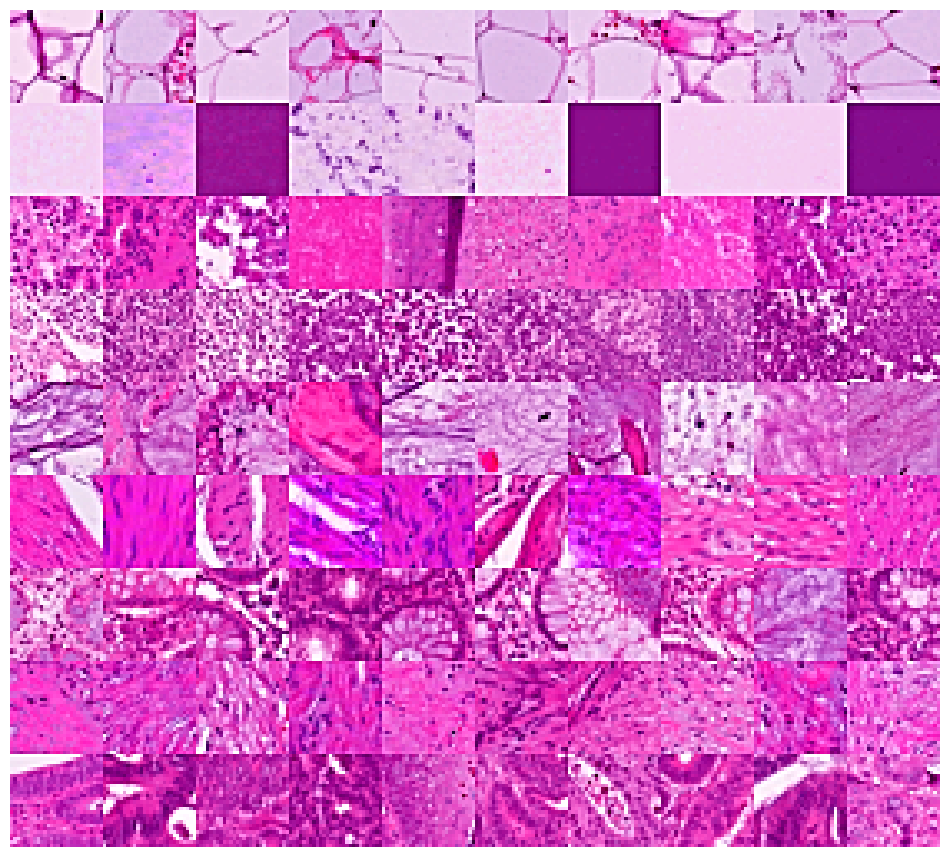


可以看到，在放大对比度之后，样本在视觉上显示出了更加显著的特征。

下面这张是锐化的样本图，锐化之后纹理的分界变得更加的清晰，可以帮助模型捕捉特征。

操作：单独做 unsharp\_mask，突出图像中的高频边缘和纹理。

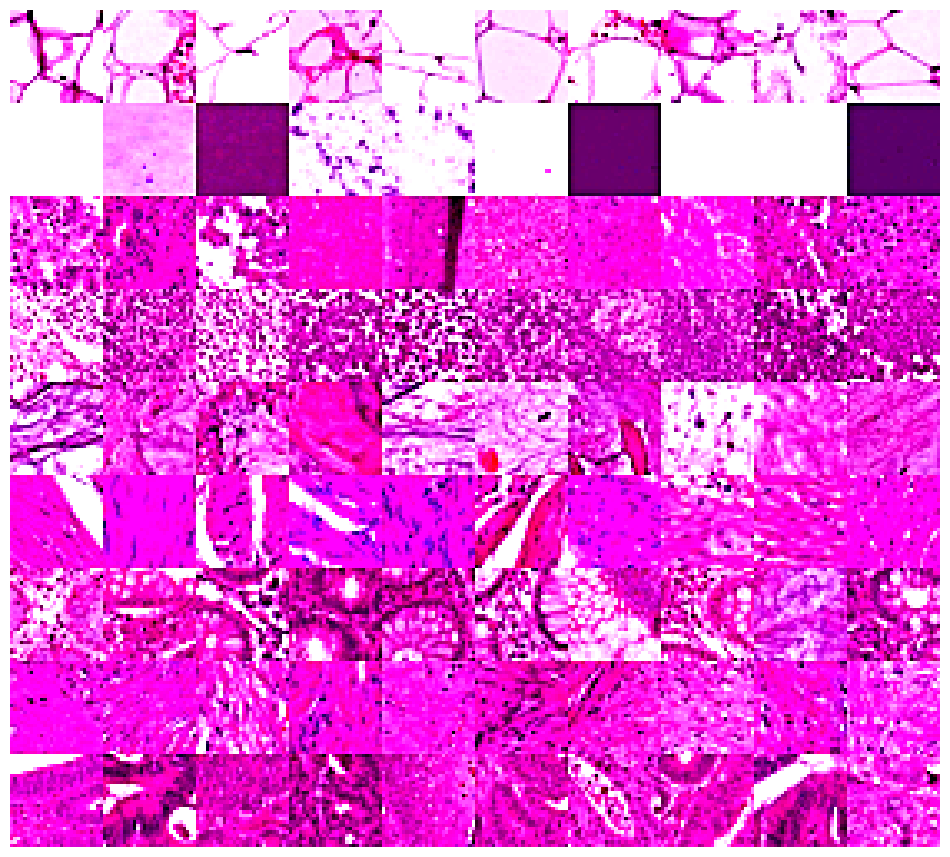
预期效果：让细胞边界、组织间隔等高频特征更加清晰，帮助网络更快捕获边缘模式。



下面这张是先锐化在增强了对比度的组织图，

串联操作：先 unsharp\_mask 再放大对比度

预期效果：在突出高频边缘的同时，将这些边缘进一步提亮；



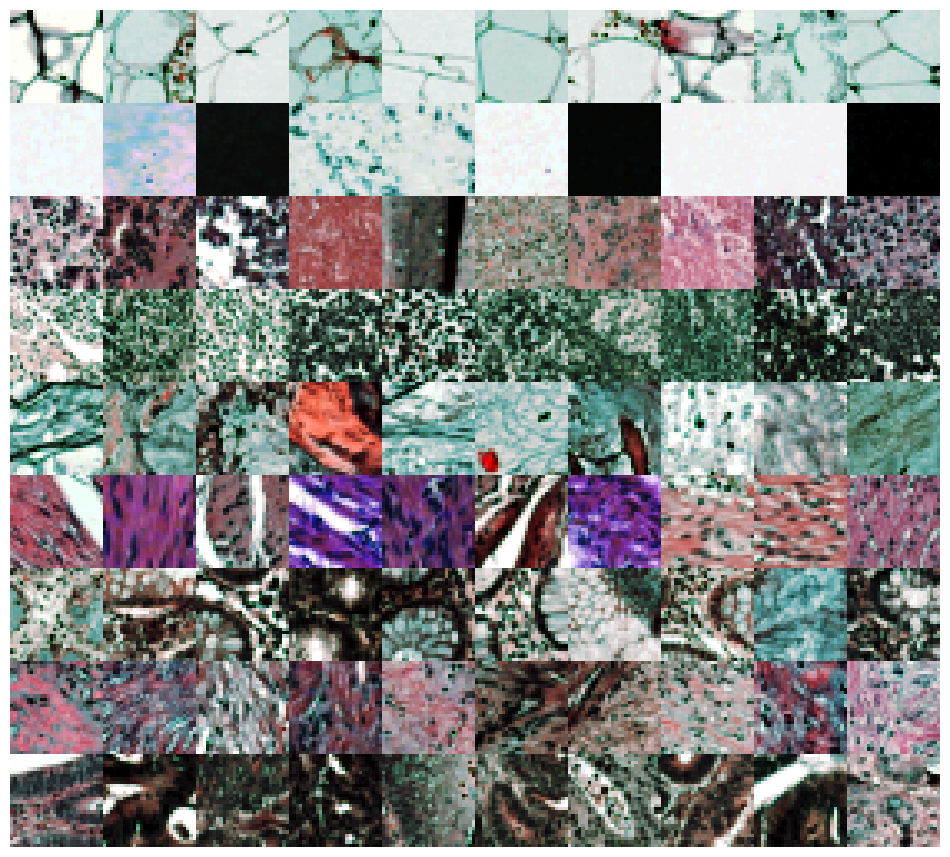
下图为自适应对比度+局部拉伸

操作：

ImageOps.autocontrast(cutoff=1%) 去掉极端亮暗 1% 像素后拉伸动态范围；

ImageOps.equalize 做全局直方图均衡，改善暗区与亮区细节。

预期效果：在保留整体色彩平衡的同时，提高全局各区域的对比度，不会过度放大噪声。

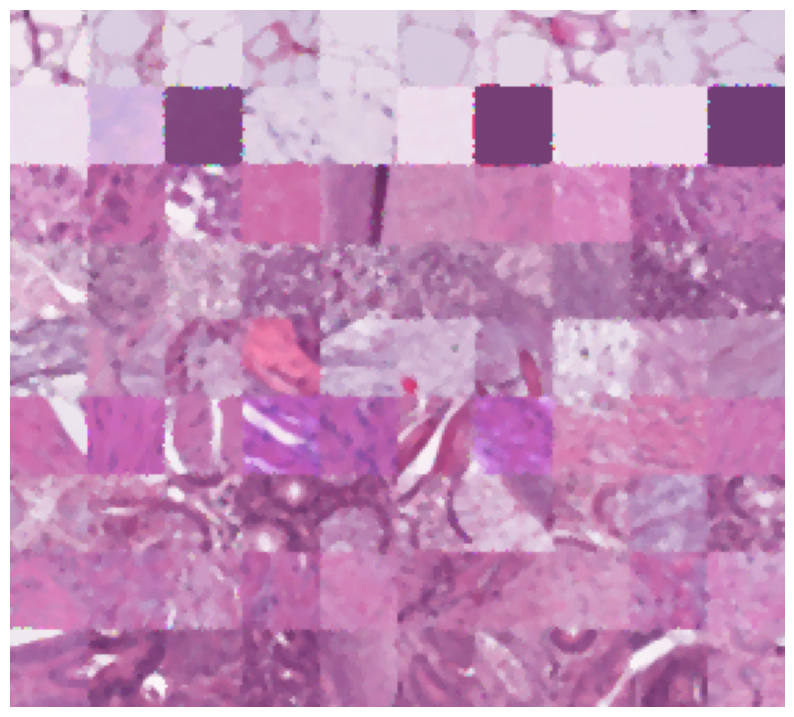


小窗口去噪（Median+Mode Filter）

操作：MedianFilter(size=3) 去除椒盐噪声 → ModeFilter(size=3) 平滑离群像素

预期效果：保留主要边缘与纹理的同时，剔除掉随机分布的噪点和压缩伪影，提升信噪比。

强化关键结构,模型能更专注于“有意义的组织学模式”而不是噪声；



我们将使用上述预处理过的数据集用于模型的训练，比较不同的预处理方法会对训练的结果产生什么样的影响。

# Method

## Theory

### Algorithm of choice from first six weeks of course

### MLP

Basicly, Multilayer Perceptron(MLP) is a kind of fully-connected neural network, also a kind of non-linear model.

Comparing to a linear model( f(x) = x ⋅ W), a non-linear adds activation methods, or non-linear methods (f(x) = g(x ⋅ W1) ⋅ W2), which allows a non-linear model to learn more features from training data.

There are several activation methods, the most used methods are

Sigmoid

(0,1)

Hyperbolic Tangent(tanh)

(-1,1)

Rectified Linear Unit (ReLU)

These methods play different rules in the model.

Back to the formula of non-linear function, , columns in the weight metric(W1 and W2 here) are called cells, and activation methods(g(x) here) are called layers. According to the definition of MLP, there must be multiple layers, which means at least two, so the formula can be like this and also, there are biases .

The training method of MLP is called backpropagation, we calculation the influence of each layer through the classic chain rule in derivative of composite function.

Each factor in this formula represents the influence of the corresponding layer. This process is also called gradient descent.

In each training step, we calculate the loss based on the label value and the output of model and update weights in each layer base on the loss. There is also a factor decides how much we should update, which is learning rate. Obviously, the bigger the learning rate is, the further we would update the weights.

## Strengths and weaknesses

## CNN

## Architecture and hyperparameters

### MLP

 **tf.keras.layers.RandomFlip("horizontal", seed=42)**

* **用途**：在训练时以 50% 概率对输入图像做左右翻转（水平翻转）。
* **mode="horizontal"**：指定只做“水平”翻转（左右镜像），可选项还有 "vertical"（上下翻转）和 "horizontal\_and\_vertical"（两者皆可）。 [Keras](https://keras.io/api/layers/preprocessing_layers/image_augmentation/random_flip/?utm_source=chatgpt.com)
* **seed=42**：为内部随机过程指定种子，以保证每次运行时“哪些样本被翻转”是可复现的。

 **tf.keras.layers.RandomRotation(0.1, seed=42)**

* **用途**：在训练时随机旋转图像一个角度，让模型对仿射旋转更鲁棒。
* **factor=0.1**：当是单个浮点数时，旋转角度在 −0.1×2π,+0.1×2π-0.1 × 2π, +0.1 × 2π−0.1×2π,+0.1×2π（即约 ±36°）范围内均匀采样；也可传元组如 (−0.2, 0.3) 来指定非对称的上下限。
* **seed=42**：固定旋转角度采样的随机种子，使每次训练时旋转角度的随机序列一致。

 **tf.keras.layers.RandomZoom(0.1, seed=42)**

* **用途**：在训练时随机进行图像的缩放（放大或缩小），提高对尺度变化的鲁棒性。
* **height\_factor=0.1**（width\_factor 默认为相同值）：当是单个浮点数时，垂直（或水平）缩放比例在 1−0.1,1+0.11−0.1, 1+0.11−0.1,1+0.1（即 90%–110%）范围内均匀采样；也可传元组来分别设定上下限。 [Keras](https://keras.io/api/layers/preprocessing_layers/image_augmentation/random_zoom/?utm_source=chatgpt.com)
* **seed=42**：固定缩放比例采样的随机种子，使每次训练时缩放操作可复现。

**层级与参数详解**

1. **Flatten(input\_shape=(28,28,3))**
   * 作用：将输入的三维张量（28×28×3）“摊平”为一维向量（长度=28×28×3=2352），以供后续全连接层处理。
   * 参数：input\_shape 指定每个样本的形状（不含 batch 维度）。  
     [Keras](https://keras.io/api/layers/reshaping_layers/flatten/?utm_source=chatgpt.com)
2. **第一隐藏层 Dense(500, activation="relu")**
   * units=500：该层包含 500 个神经元（每个神经元都有自己与上一层 2352 个输入的权重向量及一个偏置）。
   * activation="relu"：ReLU 激活函数，max⁡(0,x)\max(0,x)max(0,x)，有助于加速收敛并缓解梯度消失。
   * 运算：输出 = ReLU( 输入·权重 + 偏置 )。
3. **Dropout(0.3)**
   * rate=0.3：每个训练步骤随机“断开”本层 30% 的神经元输出，将其置为 0，其余 70% 会按 1/(1−0.3)1/(1-0.3)1/(1−0.3) 比例缩放，保持总激活和不变。
   * 目的：防止过拟合，让网络学到更鲁棒的特征。
4. **第二隐藏层 Dense(300, activation="relu") + Dropout(0.2)**
   * 同第一隐藏层，但这里 rate=0.2，保留更多神经元（80%），作为一种正则化手段。
   * 保持与前层相同规模（300 个神经元），有利于网络学习更深的非线性组合。
5. **第三隐藏层 Dense(300, activation="tanh") + Dropout(0.2)**
   * activation="tanh"：双曲正切函数，输出范围 [−1,1][-1,1][−1,1]，在某些任务上能提供对称激活特性。
   * 通常 tanh 收敛比 ReLU 慢一些，这里可视为 “特征变换层”，为输出层提供多样化表示。
   * 同样用 Dropout(0.2) 做正则化。
6. **输出层 Dense(9, activation="softmax")**
   * units=9：9 个神经元，对应 9 类别的打分。
   * activation="softmax"：对打分做指数归一化，输出各类别的预测概率，和为 1。

**compile 配置**

* **optimizer=Adam(learning\_rate=1e-3)**
  + Adam 优化器：结合了动量（第一矩矩估计）和自适应学习率（第二矩估计），在大多数任务上收敛稳定而高效。
  + learning\_rate=0.001：初始步长，若过大易振荡，若过小收敛慢；1e-3 是常见默认值。
* **loss="sparse\_categorical\_crossentropy"**
  + 用于整数标签（0–8）的多分类交叉熵损失。
  + 内部会先把网络输出概率与 one-hot 编码标签做交叉熵计算，再求均值。
* **metrics=["accuracy"]**
  + 在训练/验证/测试过程中，会额外计算并报告分类准确率。

**这种设计的优点**

1. **结构简单易调**：Sequential + Dense + Dropout，适合初步对比不同预处理或超参下 MLP 表现。
2. **ReLU + tanh 组合**：前两层用 ReLU 快速提取稀疏高维特征，第三层 tanh 提供对称非线性，有助于平滑边界。
3. **多级 Dropout**：0.3 → 0.2 → 0.2 的正则化，可逐层防止过拟合，同时保留足够容量学习复杂模式。
4. **Adam 优化器**：适合多参数网络且对超参数不太敏感，收敛快。

调用

**数据归一化**

**作用**：将像素值从原本的 0–255 缩放到 0–1 区间，

 **好处**：

1. 加快神经网络训练收敛速度（避免梯度爆炸／消失）；
2. 让所有输入特征落在相同量级，提升优化器效率。

**模型训练（fit）**

* **validation\_split=0.1**
  + 自动从训练集中拿出最后 10% 作为验证集，用于监控过拟合与早停。
* **epochs=20**
  + 最多完整遍历训练集 20 次；
  + EarlyStopping 触发时会提前终止。
* **batch\_size=64**
  + 每次梯度更新用 64 张图，权衡训练速度与内存占用。
* **EarlyStopping 回调**
  + **监控指标**：val\_loss（验证集损失），而非训练损失；
  + **patience=5**：若 5 个 Epoch 内验证损失都没变好，就认为模型已收敛／开始过拟合，停止训练；
  + **restore\_best\_weights=True**：训练结束后自动回滚到验证集上表现最好的那次权重。
* **优点**：
  + 节省不必要的训练时间；
  + 有效避免过拟合，提升最终模型在新数据上的泛化能力；
  + 自动“找”到最优 Epoch，无需手动对比多个模型文件。

**绘制损失曲线**

* 作用：把每个 Epoch 的训练损失和验证损失画在一张图上，
* 优点：
  + 可视化训练过程，直观判断是否收敛、是否过拟合（验证损失开始上升时即过拟合）；
  + 有助于调参，例如增大或减小 learning rate、调整网络结构。

**测试集评估**

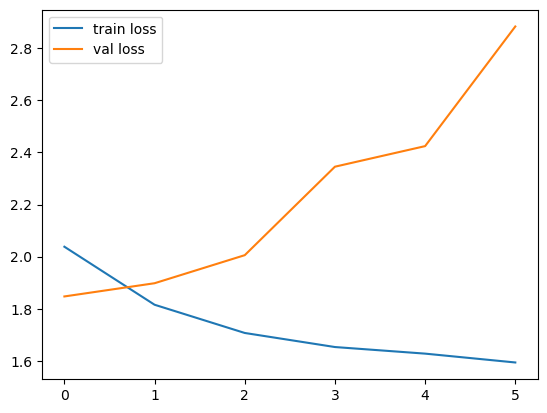
**** evaluate：在完全独立的测试集上计算损失和指标（accuracy）。

 优点：

* 给出模型在未见过数据上的真实表现；
* 与训练/验证阶段区分，防止信息泄漏。

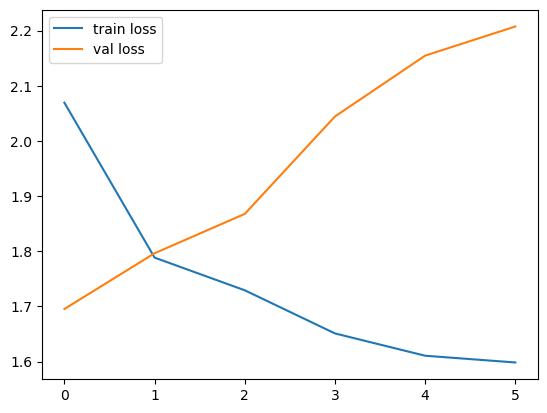
#### 模型设计观察结果

* 1. 单纯的增加层数并不能直接带来准确性的提升。
  2. 训练过程中发现模型对训练数据的过度拟合依旧是主要问题，在初始的模型中，已经通过设置dropout比例，设置早停来防止过度拟合，但是问题依然严重。



尝试在通过在训练数据中认为的制造噪音来提高模型的泛化能力。

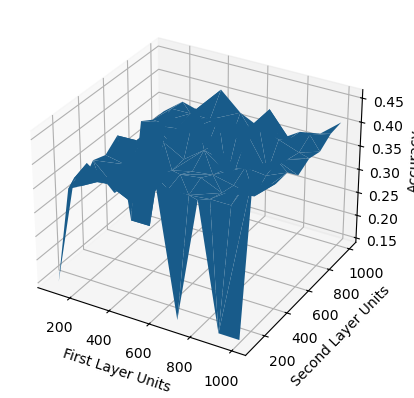
在人为的添加翻转，旋转，缩放的噪音数据进行数据增强后，模型的泛化能力得到显著的提高，随着与训练数据的拟合，验证数据的损失上升程度得到了控制。



#### 调整模型深度和神经节数量对训练的影响：

模型的训练已对所有随机因素设置了种子，确保相同的模型每次fit的获得相同的结果

下述实验基于未经过预处理的原始训练数据进行。



图片描述了一二层神经节数量和测试数据集精确度之间的关系，因为算力原因只探索了两层神经网络。从上图中可以看到，神经节的数量和准确性之间并不存在明显的相关关系精确度最高的组合是500 100，测试集精确度为0.3826当然这和随机种子也有关系。

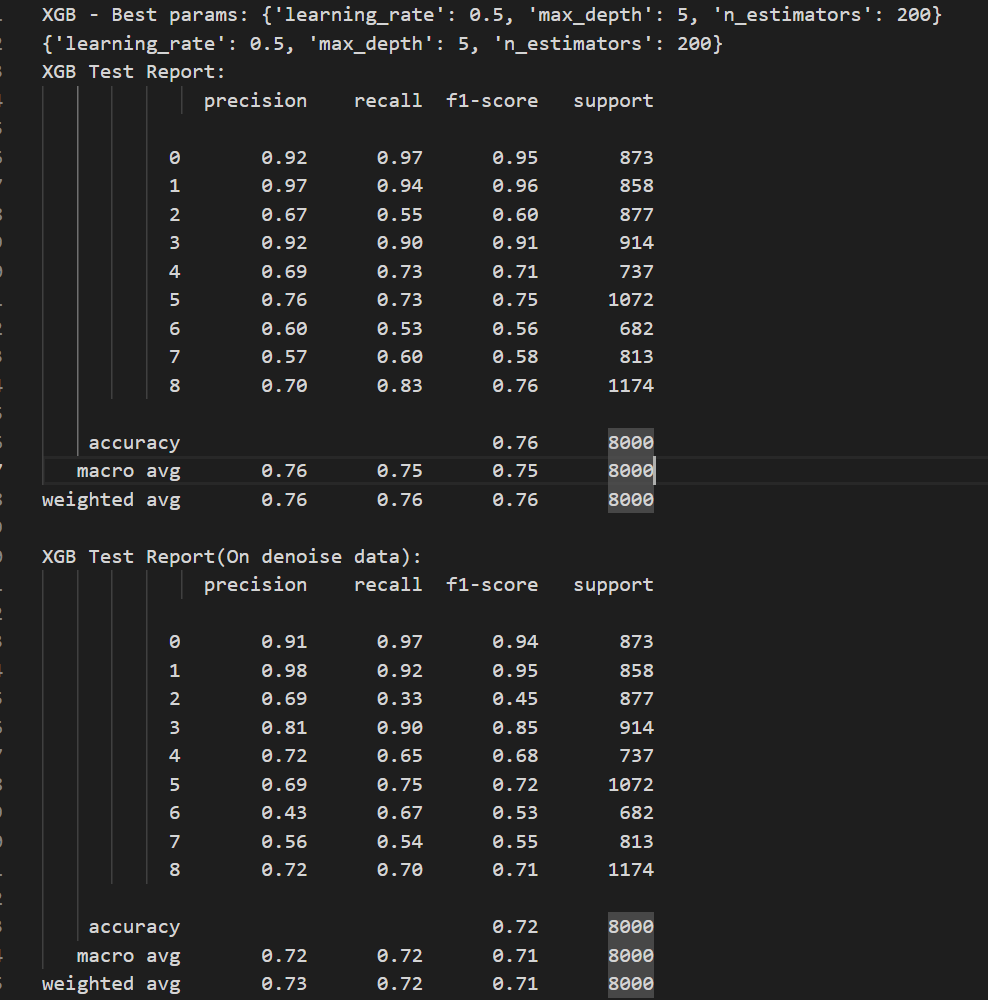
下面取500 100 来测试其他训练数据集。

Denoise数据集精确度为0.4159

Sharp&&contrast 数据集精确度为0.2351

再次说明了，强化样本的特征并不能提高模型的准确度，反而因为特征纹理过于明显，导致模型在训练中学习到了过多的独有特征，导致overfit，反倒是降噪的效果更好。

# Results and discussion



## MLP

选取的Hyperparameters为：1.第一层神经网络的activation method. 2. 图片的随机翻转模式. 3. Learning-rate.

Activation method 的 tuning range 是 ['relu', 'sigmoid', 'tanh'].

Flip\_mode的tuning range 是['horizontal', 'vertical', 'horizontal\_and\_vertical']

Learning-rate的tuning range 是 [0.1, 0.01, 0.001]

选择这三个超参来进行调参，主要是出于训练时间的考虑。在之前已经通过锁定种子的方式对不同的cell数量的训练结果进行对比，但若是在调参中考虑cell数量或是layer数量就会导致参数组合过多，训练时间长到无法接受。

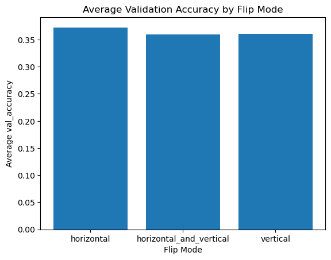
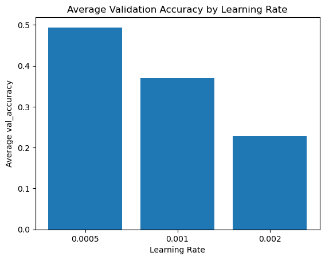
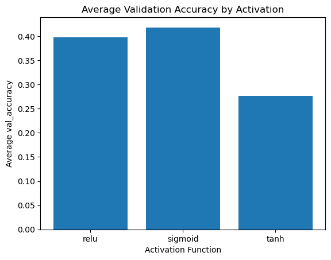
总共3\*3\*3=27种情况，因此设置max\_trails=27.

每组参数训练三次取平均值。

调参的数据集为denoise数据集，因为该数据集在之前的锁定种子的模型训练中取得了最好的表现。

|  | **activation** | **learning\_rate** | **flip\_mode** | **val\_accuracy** |
| --- | --- | --- | --- | --- |
| 11 | relu | 0.0005 | horizontal | 0.451250 |
| 6 | relu | 0.0005 | horizontal\_and\_vertical | 0.484896 |
| 10 | relu | 0.0005 | vertical | 0.473542 |
| 13 | relu | 0.0010 | horizontal | 0.419792 |
| 14 | relu | 0.0010 | horizontal\_and\_vertical | 0.414271 |
| 12 | relu | 0.0010 | vertical | 0.435312 |
| 15 | relu | 0.0020 | horizontal | 0.388854 |
| 16 | relu | 0.0020 | horizontal\_and\_vertical | 0.304375 |
| 22 | relu | 0.0020 | vertical | 0.206979 |
| 1 | sigmoid | 0.0005 | horizontal | 0.531771 |
| 2 | sigmoid | 0.0005 | horizontal\_and\_vertical | 0.526250 |
| 0 | sigmoid | 0.0005 | vertical | 0.535208 |
| 8 | sigmoid | 0.0010 | horizontal | 0.477396 |
| 3 | sigmoid | 0.0010 | horizontal\_and\_vertical | 0.490208 |
| 5 | sigmoid | 0.0010 | vertical | 0.485313 |
| 21 | sigmoid | 0.0020 | horizontal | 0.219062 |
| 18 | sigmoid | 0.0020 | horizontal\_and\_vertical | 0.240000 |
| 17 | sigmoid | 0.0020 | vertical | 0.258333 |
| 4 | tanh | 0.0005 | horizontal | 0.486042 |
| 7 | tanh | 0.0005 | horizontal\_and\_vertical | 0.482187 |
| 9 | tanh | 0.0005 | vertical | 0.475313 |
| 19 | tanh | 0.0010 | horizontal | 0.234271 |
| 25 | tanh | 0.0010 | horizontal\_and\_vertical | 0.145625 |
| 20 | tanh | 0.0010 | vertical | 0.232500 |
| 23 | tanh | 0.0020 | horizontal | 0.145625 |
| 26 | tanh | 0.0020 | horizontal\_and\_vertical | 0.145625 |
| 24 | tanh | 0.0020 | vertical | 0.145625 |

上表为27种参数组合输出的结果，其中表现最好的是,sigmoid 0.0005,vertical组合，accuracy为0.535208。



Column charts above show the average accuracy of each hyperparameter, as we can see the activation methods and learning rate brings significant influence to the model performance, while flip mode has alm激活函数（Activation Method）作为超参的影响

非线性表达能力：不同的激活函数如 ReLU、Leaky ReLU、tanh、sigmoid 等，对网络引入的非线性程度不同。

ReLUmax(0,x)）在正区间梯度恒定，收敛快且抗梯度消失，但对“全负”区域零梯度敏感。

tanh（双曲正切）输出在 [−1,1] 之间，能够居中分布，有助于梯度流动，但在两端仍会饱和。

sigmoid输出在 (0,1) 之间，早期常用于二分类，但对大输入易饱和、梯度消失明显。

梯度传播与收敛速度：

选择 ReLU 比 sigmoid/tanh 更能保持梯度大小，加速收敛；

而 tanh 在某些小网络或归一化输入下能促进更稳定的更新。

对过拟合与泛化的影响：

带有平滑曲线（tanh、sigmoid）的激活容易产生更平缓的决策边界，有时能改善泛化；

ReLU 在稀疏性（大量神经元输出为 0）上有天然优势，可视为一种正则化效果，但也可能需要配合 Dropout/BatchNorm 避免“死亡 ReLU”。ost no influence.

学习率（Learning Rate）作为超参的影响

步长控制：学习率 决定了每次梯度下降更新权重时“走多大步”：

过大：训练时参数会在最优点附近来回震荡甚至发散，无法收敛；

过小：每步更新都很谨慎，收敛极慢，容易陷入局部极值。

收敛速度与稳定性：

合理的学习率能让损失函数快速下降且平稳，避免出现曲线震荡或长时间停滞；

CNN

