第十章 DNA 视觉

第一节 DNA 视觉的动机

记得我在加州路德大学学习计算机视觉的课程时, Dr. Renhart 有一次上课时候说通过计算机卷积计算腐蚀, 通过统计, 我发现所有的噪声过滤都是卷积计算, 比如卷积高斯, 卷积腐蚀, 卷积膨胀等等, 我当时在思考如果有一种非卷积算法来过滤噪, 多好, 于是思路便有了. 后来发现我成功的解决了这个问题. 于是将这些因子进行体系的归纳, 我找到了很多有价值的思路, 其中一点就是 算能优化. 所有的主题 都是围绕着这个词汇.

刚开始写 DNA 视觉的动机, 我没有过多的思索, 认为这是一种高大上的学术, 脑子里对技术的定义一直是 技术沉淀来自于踏实的实践. 现在动机越来越强烈, 我认为是实践积累到了一个需要笔记化的阶段（换句话说就是我的脑子不够用了, 需要工具的辅助）, 于是开始归纳结果思维, 有必要详细描述下.

DNA 的视觉研究, 我有 2 个动机, 一, 能够探索到优化生产力, 生产结构的算法和方向. 二, 满足养疗经[17]的智慧分析组件的优化更新需求.

第二节 DNA 视觉的应用需求

没有需求, 我便给自己设计需求, 养疗经[17]是医学大数据软件, 我先设计一个 DNA 像素颜色观测功能, 然后持续不断的优化它, 测试图片我用一些骨科 X 片, 皮肤病的图片进行分析先, 想到这,就开始研发了. 昨天 20210301: 以后我用年月日标号来描述时间. 和父亲讨论了视觉的需求, 父亲的意思是 3 个字, 对数据的处理要, 快, 广, 准.

我觉得应该多花点实践在非卷积领域进行深入探索. 传统的中医思想 望闻问切 和 理法方药, 讲究的是任何一种技术应用, 应该直接和间接的参与辩证. 所以 DNA 视觉的应用要求应该是能直接参与某类疾病的计算观测和辅助与替代普通的肉眼观测, 或者提供更广泛的辩证对比参考价值.

第三节 DNA 视觉的具体描述

在视觉计算中, 我一开始想到的是肽展公式的酸碱浓度变化, 如果将变嘧啶按酸碱的浓度来变换成 Q 和 D, 那么像素的数字值转换成元基, 就能参与视觉计算, 利用这个原理, 于是我设计了DNA 非卷积腐蚀算法, 代码中的 IDUQ 酸碱变换依据我之前写的肽展公式进行变换.

//函数如下:

```
package OSI. OSU. SI. SD. SU. SQ. ASU. OSU. PSU. MSU. AVQ. ASQ. tin. catalytic. procedure. pde;
@SuppressWarnings("unused")
public class RangePDI{
    public static void main(String[] argv) {
        //System. out. println(new RangePDI(). IOE(16, 20));
    }


//SH生化
//U 3 1
//这个函数我一开始是仅仅按照变嘧顶的酸碱值域来变化观测, 于是设计成仅仅在弱酸弱碱中进行催化反应
//应用: 肽展公式
//作者: 罗瑶光
    //240/4 600//SH特征
    public int[][] IOE(int[][] ps, int VECS) {
        for(int i= 0; i< ps. length; i++) {
            for(int j= 0; j< ps[0]. length; j++) {
                String IDUQ= new RangePDI(). PDS_P_USQ_ECP_I(ps[i][j], 4);
                int[][] OIQ= new int[1][IDUQ. length()];
                for(int k= 0; k< IDUQ. length(); k++) {
                    if(IDUQ. charAt(k)== '2') {
                        if(Math. random()* 100> VECS)
                            { OIQ[0][k]= 3;
                        }else {
                            OIQ[0][k]= 1;
```

```
                    }
                }else {
                    OIQ[0][k]= Integer. valueOf(""+ IDUQ. charAt(k));
                }
            }
            ps[i][j]= new InitonsPDS(). DO_ACP_IDV(OIQ, 4);
        }
    }
    return ps;
}

//DL数字逻辑
//D 1 D
//I I 2
//U 3 U
//Q Q 0
//应用: 肱展公式
//作者: 罗瑶光
//这个算法我按照离散数学的排列 将 IDUQ改成 DIUQ然后进行计算.
    //DL腐蚀 diuq
    public int[][] IPE(int[][] ps, int VECS) {
        for(int i= 0; i< ps. length; i++) {
            for(int j= 0; j< ps[0]. length; j++) {
                String IDUQ= new RangePDI(). PDS_P_USQ_ECP_I(ps[i][j], 4);
                int[][] OIQ= new int[1][IDUQ. length()];
                for(int k= 0; k< IDUQ. length(); k++) {
                    if(IDUQ. charAt(k)== '2') {//g
                        if(Math. random()* 100> VECS)
                            { OIQ[0][k]= 3;
                        }else {
```

```
                                OIQ[0][k]= Integer. valueOf(""+ IDUQ. charAt(k));
                        }
                    }else if(IDUQ. charAt(k)== '3') {//s
                        if(Math. random()* 100< VECS)
                            { OIQ[0][k]= 0;
                        }else {
                                OIQ[0][k]= Integer. valueOf(""+ IDUQ. charAt(k));
                        }
                    }else if(IDUQ. charAt(k)== '1') {//s
                        if(Math. random()* 100< VECS)
                            { OIQ[0][k]= 2;
                        }else {
                                OIQ[0][k]= Integer. valueOf(""+ IDUQ. charAt(k));
                        }
                    }else if(IDUQ. charAt(k)== '0') {//g
                        if(Math. random()* 100> VECS)
                            { OIQ[0][k]= 1;
                        }else {
                                OIQ[0][k]= Integer. valueOf(""+ IDUQ. charAt(k));
                        }
                    }
                }
                ps[i][j]= new InitonsPDS(). DO_ACP_IDV(OIQ, 4);
            }
        }
        return ps;
    }

//DLSH生化计算
//D 1 D
```

//I I 2

//U 3 0

//Q Q 0

//这个算法, 我将生化和语义数字逻辑结合，观察U变嘧啶腐蚀的计算交集.

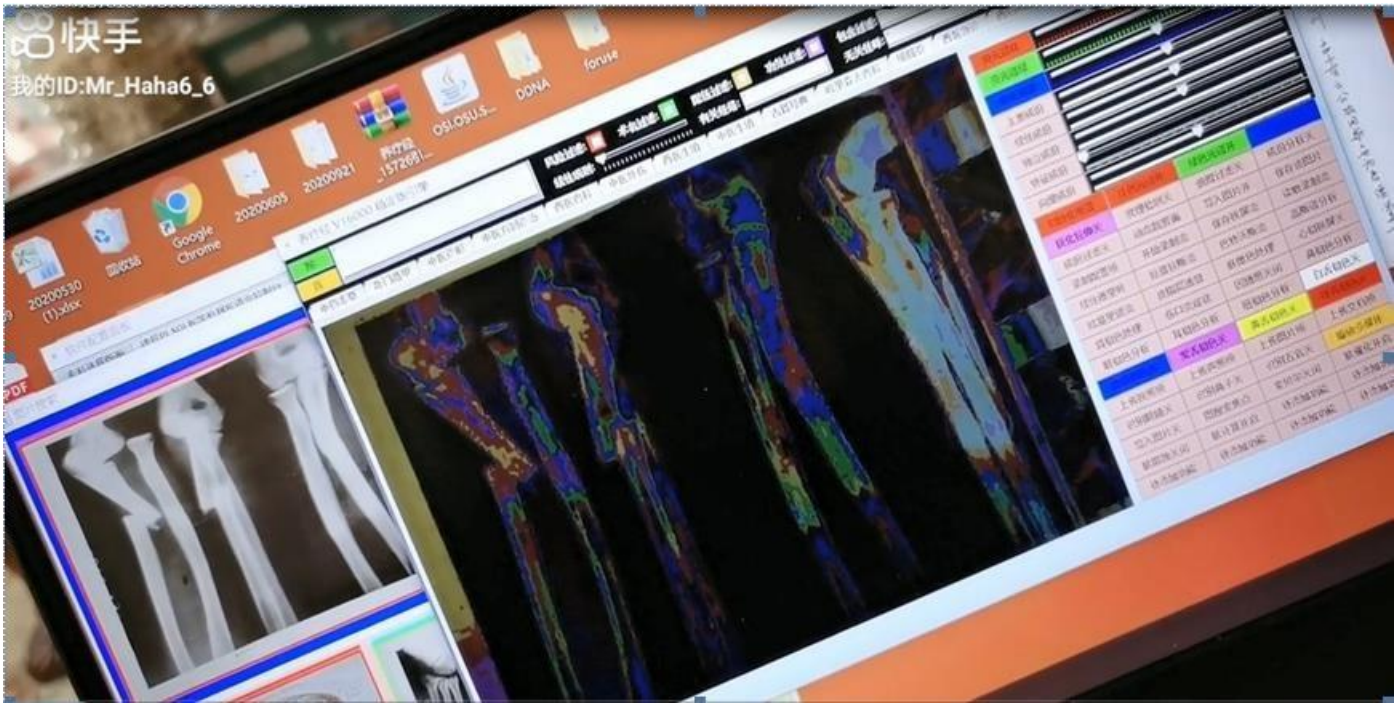
```
public int[][] QPE(int[][] ps, int VECS) {
    for(int i= 0; i< ps. length; i++) {
        for(int j= 0; j< ps[0]. length; j++) {
            String IDUQ= new RangePDI(). PDS_P_USQ_ECP_I(ps[i][j], 4);
            int[][] OIQ= new int[1][IDUQ. length()];
            for(int k= 0; k< IDUQ. length(); k++) {
                if(IDUQ. charAt(k)== '0') {//g D I U Q
                    if(Math. random()* 100> VECS)
                        { OIQ[0][k]= 1;
                    }else {
                        OIQ[0][k]= 0;
                    }
                }else if(IDUQ. charAt(k)== '1') {//s
                    if(Math. random()* 100> VECS)
                        { OIQ[0][k]= 1;
                    }else {
                        OIQ[0][k]= 2;
                    }
                }else if(IDUQ. charAt(k)== '2') {//s
                    if(Math. random()* 100> VECS)
                        { OIQ[0][k]= 3;
                    }else {
                        OIQ[0][k]= 0;
                    }
                }else if(IDUQ. charAt(k)== '3') {//g
```

```
                        if(Math. random()* 100> VECS)
                            { OIQ[0][k]= 3;
                        }else {
                            OIQ[0][k]= 0;
                        }
                    }
                }
                ps[i][j]= new InitonsPDS(). DO_ACP_IDV(OIQ, 4);
            }
        }
        return ps;
    }

    //这段代码用于4进制变换
    public String PDS_P_USQ_ECP(int P_VSQ, int MSP)
        { String ISQ= "";
        while(P_VSQ> 0) { ISQ+=
            P_VSQ/ MSP; P_VSQ%=
            MSP;
        }
        ISQ+= P_VSQ;
        return ISQ;
    }
    //64/4 8/4 2
    public String PDS_P_USQ_ECP_I(int P_VSQ, int MSP)
        { String ISQ= "";
        while(P_VSQ>= MSP) { ISQ+=
            P_VSQ/ MSP; P_VSQ%=
            MSP;
        }
```

```
        ISQ+= P_VSQ;
        return ISQ;
    }
```

第四节 DNA 视觉的应用实现



骨科 X 片分层 DNA 边缘 填充 元基计算应用

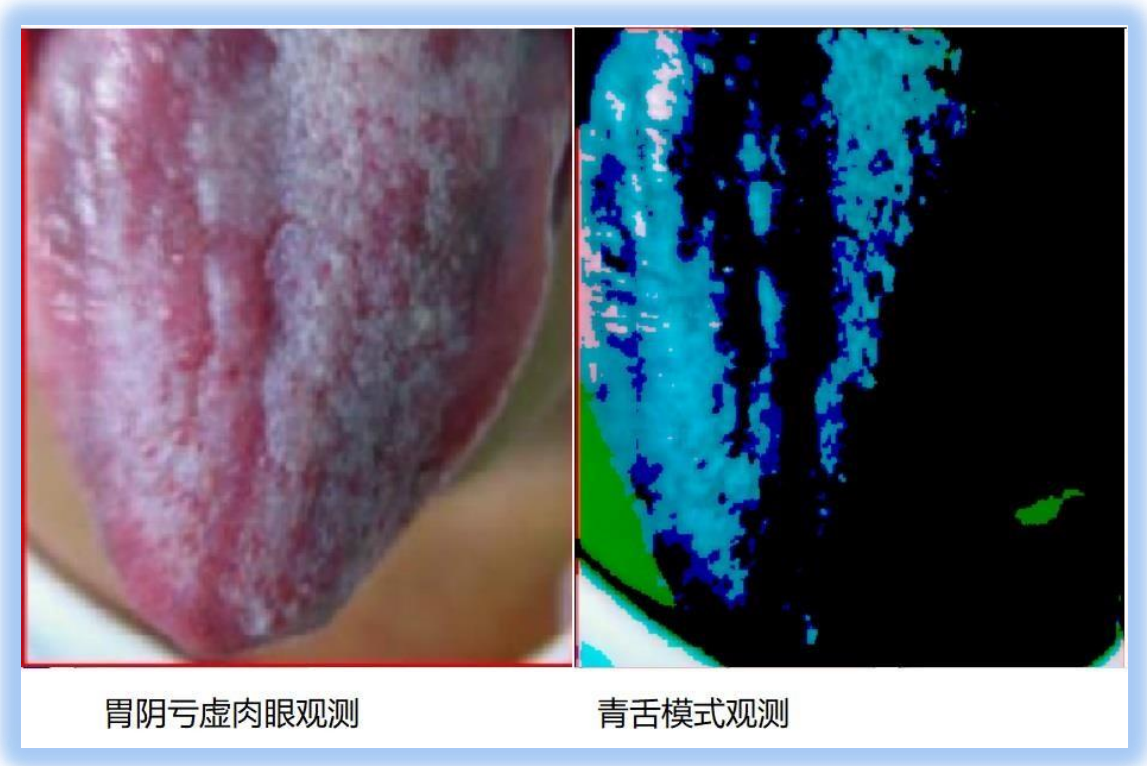
设计这种观测算法一开始, 我在用索贝尔进行边界显示, 我思考了下, 拉普拉斯的凹度和索贝尔的边界 可不可以填充, 于是我又想到了 emboss 算法, 依旧无法分层填充, 感觉分层显影好麻烦, 在这种模糊视觉分析的情况下, DNA 腐蚀视觉算法开始崭露头角了, 如图. 骨科的 X 片仅仅是灰度0~255 区间显色, 而肉眼对色值的辨别远不及计算视觉, DNA 腐蚀算法能比较轻易的将细微的像素差进行反差放大, 方便肉眼观测.

关于上面这张图, 我要感谢Renhart 教授 2011 年的一堂课. 老师在上计算视觉的像素拉伸一堂课,有一次布置了作业在 0 到 255 的像素强度区间 如果低于 30, 肉眼分辨不是很清晰, 于是在这个 0 到30 的像素域来分布了写数据, 希望学生能将这些数据统计和用比例显示出来. 我当时所想的是先将

像素解码, 然后提取像素数据来分类归类, 最后比例放大到 80~255 之间, 显示的很清晰.

我归纳了下这种问题, 个人认为这属于模糊视觉特征识别, 我准备深入, 在 DNA 元基催化与肽计算领域施展下实例研究论证. 另外也标注下, 我能走到今天, 无一不是 每次计算, 笔记, 推导, 编码, 都在医学领域的实践上进行论证.

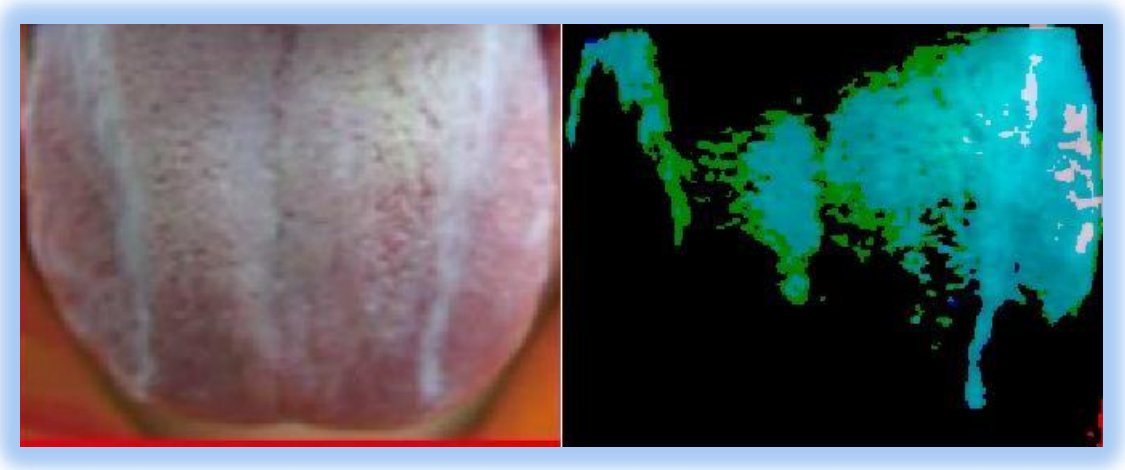
为什么要重点在非卷积视觉计算领域下功夫, 因为非卷积处理图片的速度比卷积快 20~500 倍. 非卷积的矩阵计算是每一个像素仅仅做一次计算, 而卷积计算是做一个 N*N 的矩阵遍历, 然后统计终值, 我产生一个非常严谨的推导论点: 人类大脑的计算结构单元不是耗时巨大的卷积计算模式. 而是一种并行的非卷积拓扑神经网络计算模式. 这种神经网络计算结构能模拟卷积计算来做高级分析而已. 这个论点产生, 我下一步开始论证. 拭目以待.



这里的胃阴亏虚图片是一张示例原图, 之后养疗经[17]处理的便是这些图片的变换和筛选观测. 如上图的右侧青舌观测模式, 便是将图片的像素分离, 然后仅仅筛选一定比值的蓝色和少量其他颜色, 调成了青色模式, 然后观测, 这样观测的结果能知道肾功能的血气比值.

于是按照这个原理, 设计了黄青紫红白五种简要的原色观测模式.

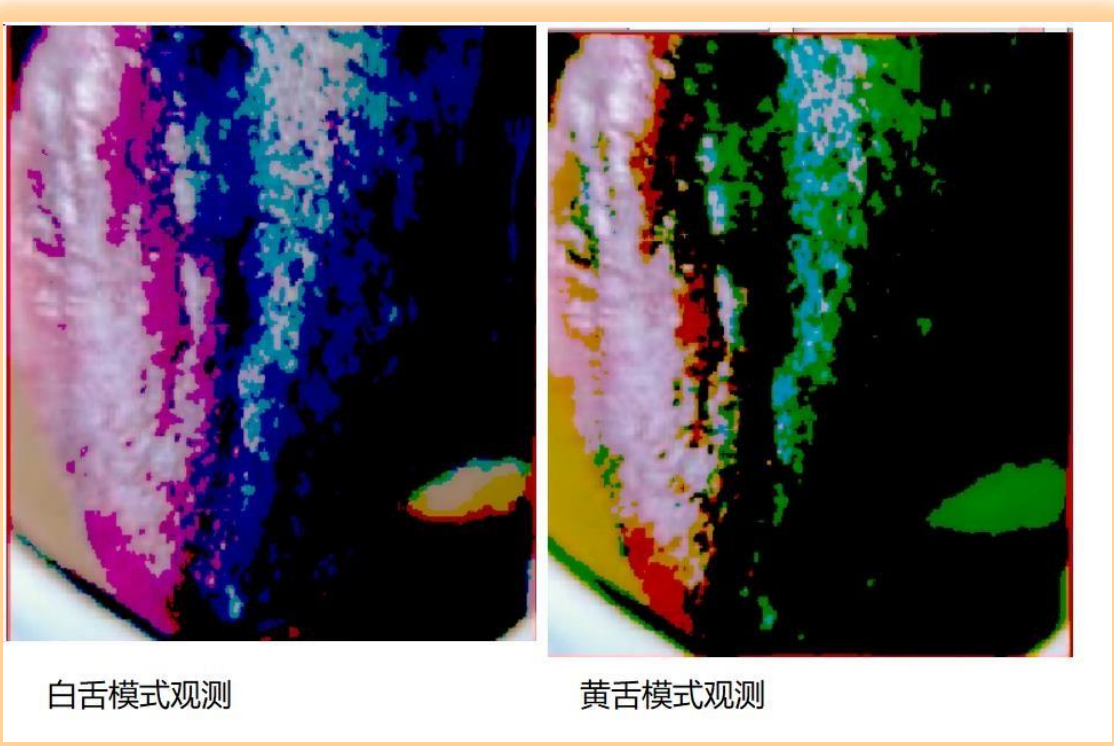
我之后会找个正常的舌头来进行对比



脾胃虚弱肉眼观测　　青舌模式观测

在医学观测中, 胃阴亏虚一种虚火症状, 而脾胃虚弱是一种虚寒症, 如果用常规的肉眼观测都带有白色. 不过是 红白和黄白, 如果不用心仔细去区分, 会产生各种摸棱两可的数据, 混淆思维, DNA 视觉计算在这个领域的应用, 功效点就是迅捷的多途径的对比像素处理差, 为诊治提供多角度的观测对比, 提高辨别能力.

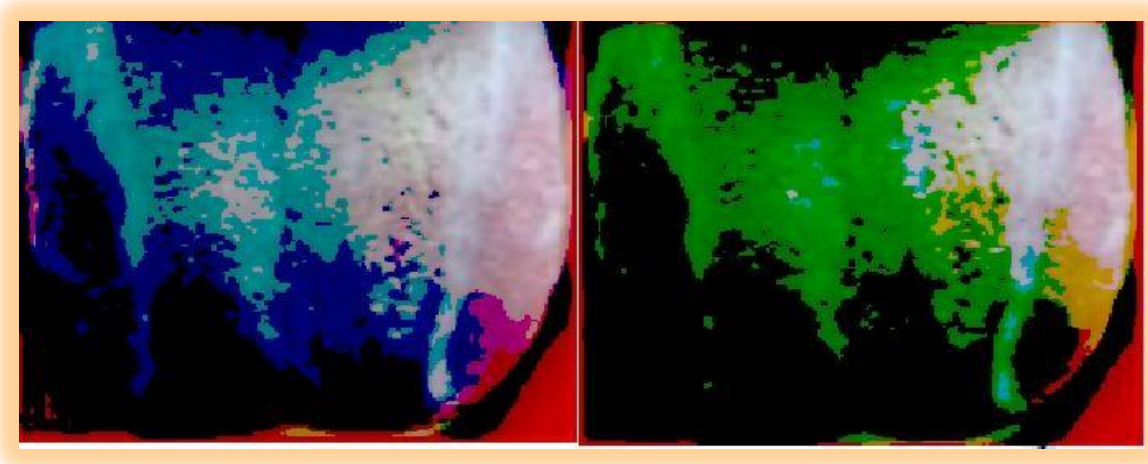
最早做智能相诊的页面时候, 我在思考, 实时卷积处理太耗费运算性能, 如果能用非卷积计算来进行观测处理像素团, 是一种生产技术的更新. 于是开始探索.



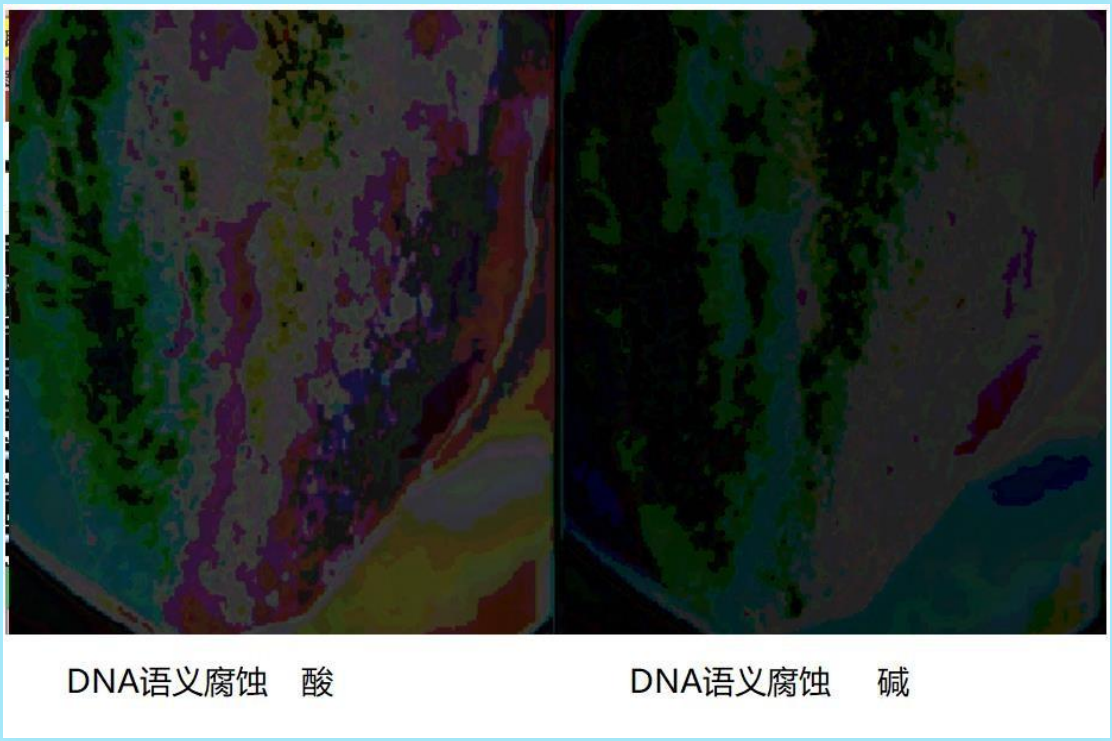
白舌模式观测　　黄舌模式观测

上图胃阴亏虚与下图脾胃虚弱对比

这是比较简单的三原色过滤肉眼辨别模式, 可以看出虚火和虚寒的区别（色泽的浓淡）.



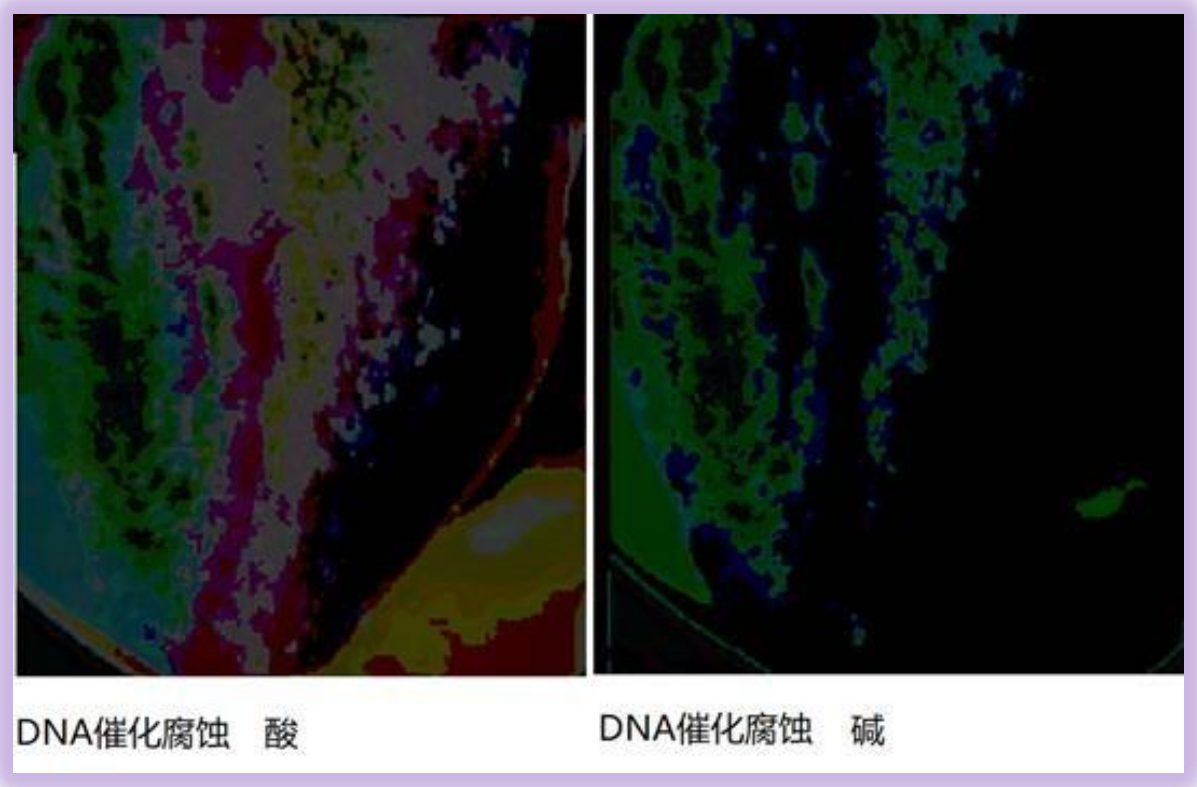
脾胃虚弱白舌模式观测　　脾胃虚弱黄舌模式观测



上图胃阴亏虚与下图脾胃虚弱对比（色泽浓淡与色泽条纹）

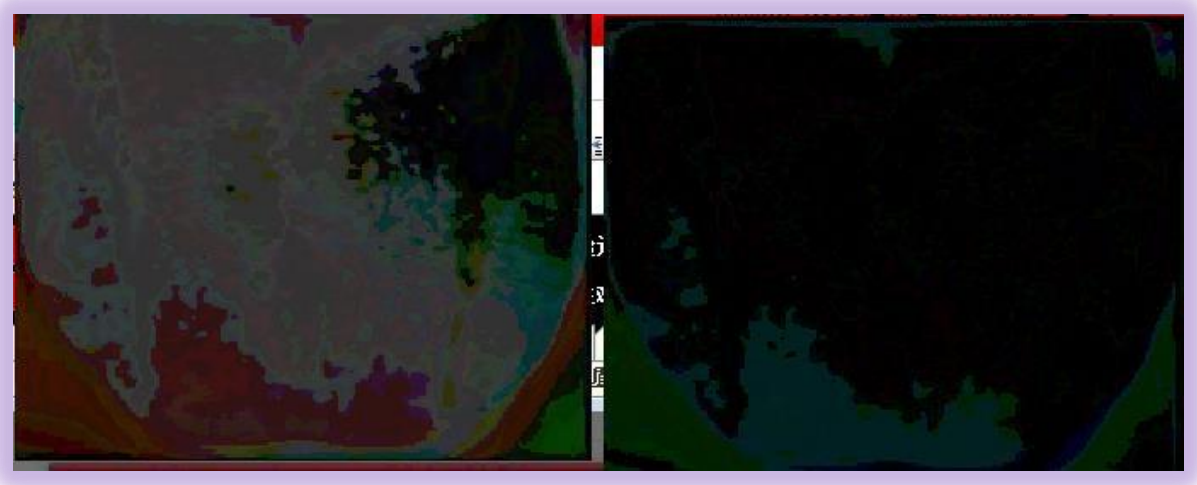
脾胃虚弱 DNA 语义腐蚀 酸　　**脾胃虚弱** DNA 语义腐蚀 碱

模拟元基语义数字逻辑变换, DNA 的语义腐蚀 将像素中的两边极值按精度比例过滤, 我的做法比较简单, 把人类数字进行 4 进制变换, 然后生化模式排列, 然后调用 DNA 元基催化函数计算,



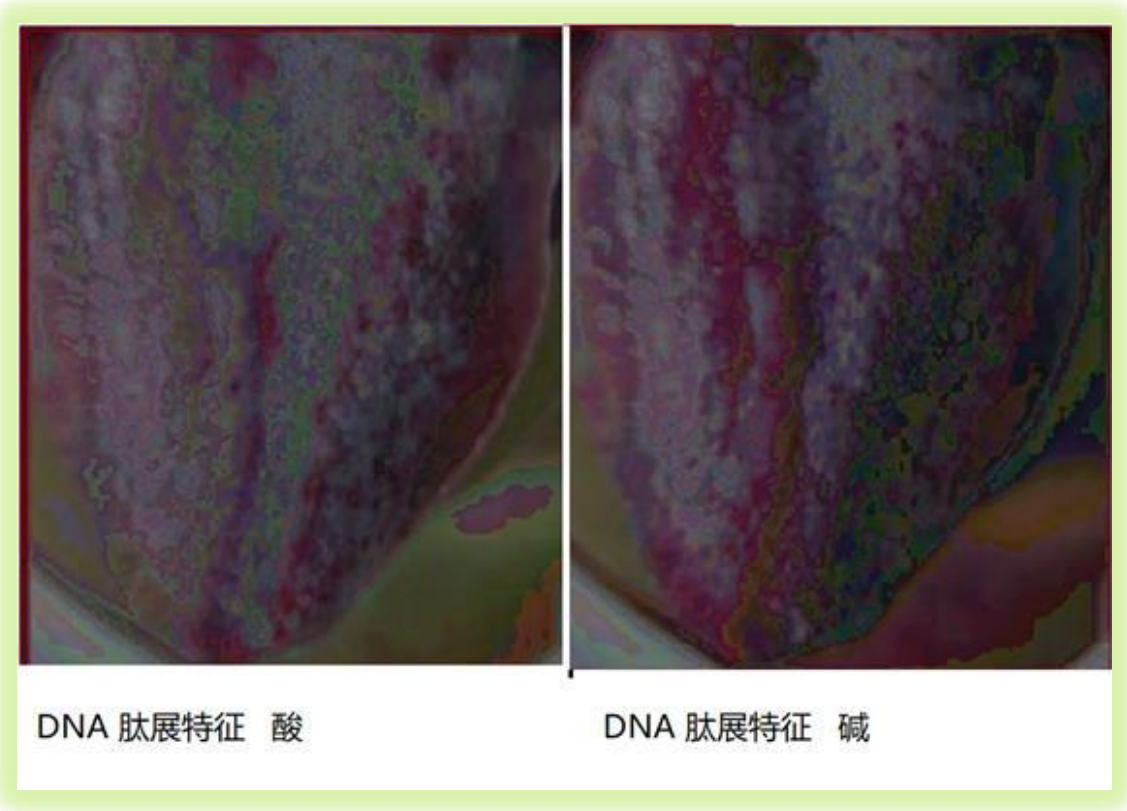
模拟元基强腐蚀肽展变换

上图胃阴亏虚与下图脾胃虚弱对比（色泽浓淡与色泽条纹）



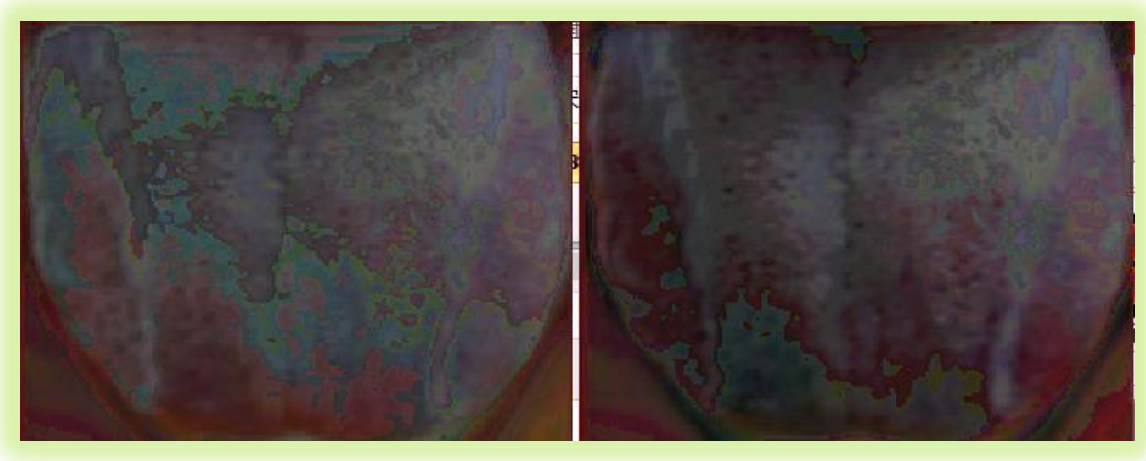
脾胃虚弱 DNA 催化腐蚀 酸

脾胃虚弱 DNA 催化腐蚀 碱

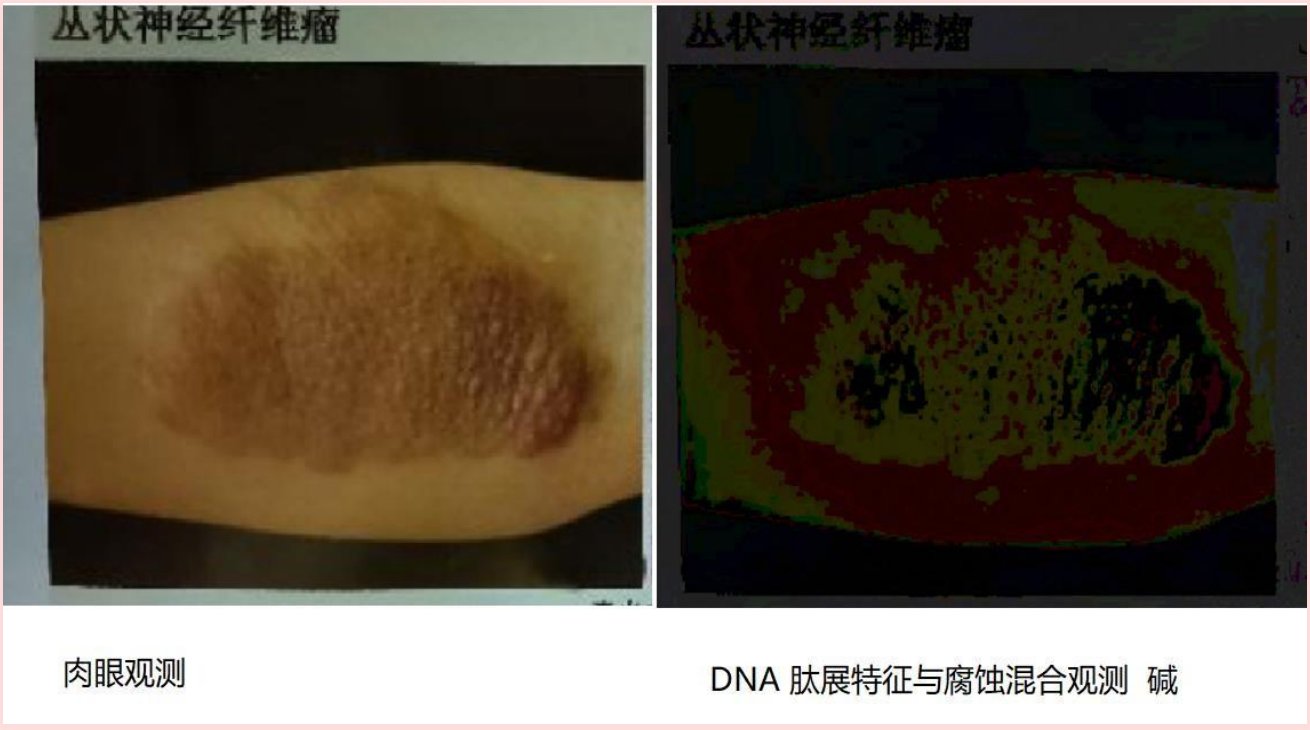


模拟元基弱腐蚀肽展变换

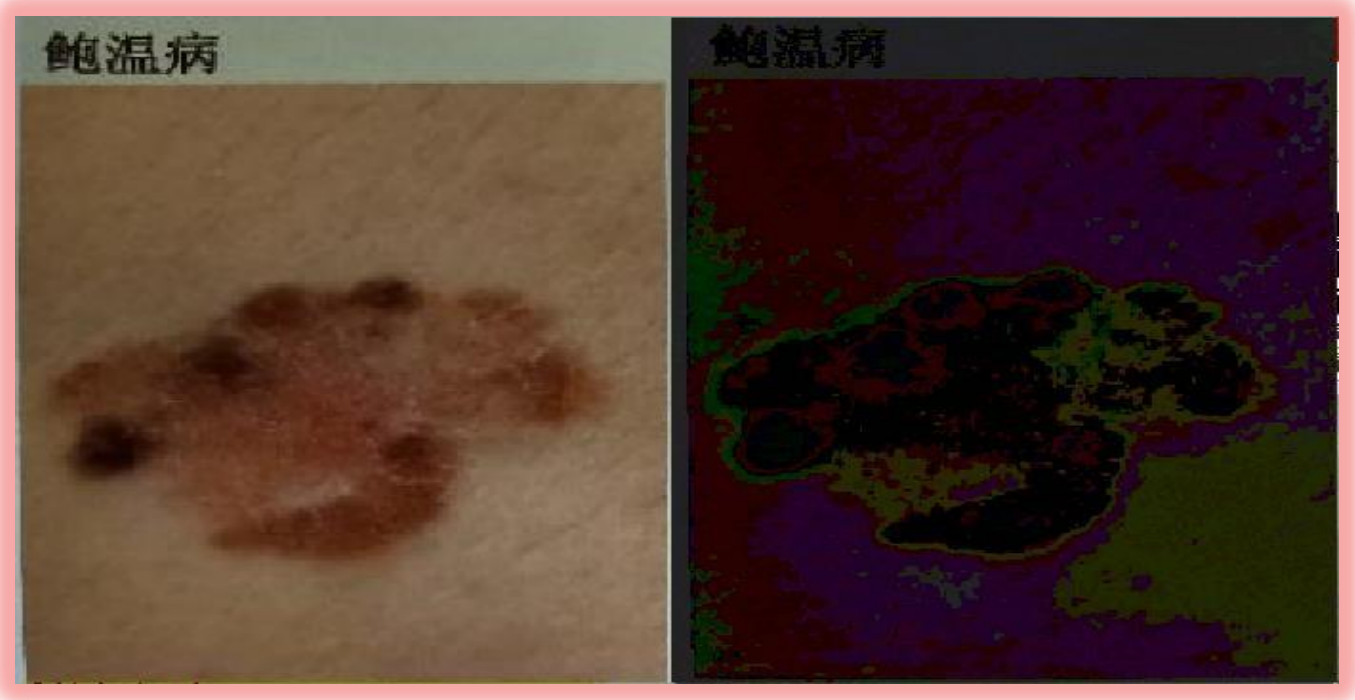
上图胃阴亏虚与下图脾胃虚弱对比（色泽条纹分布）



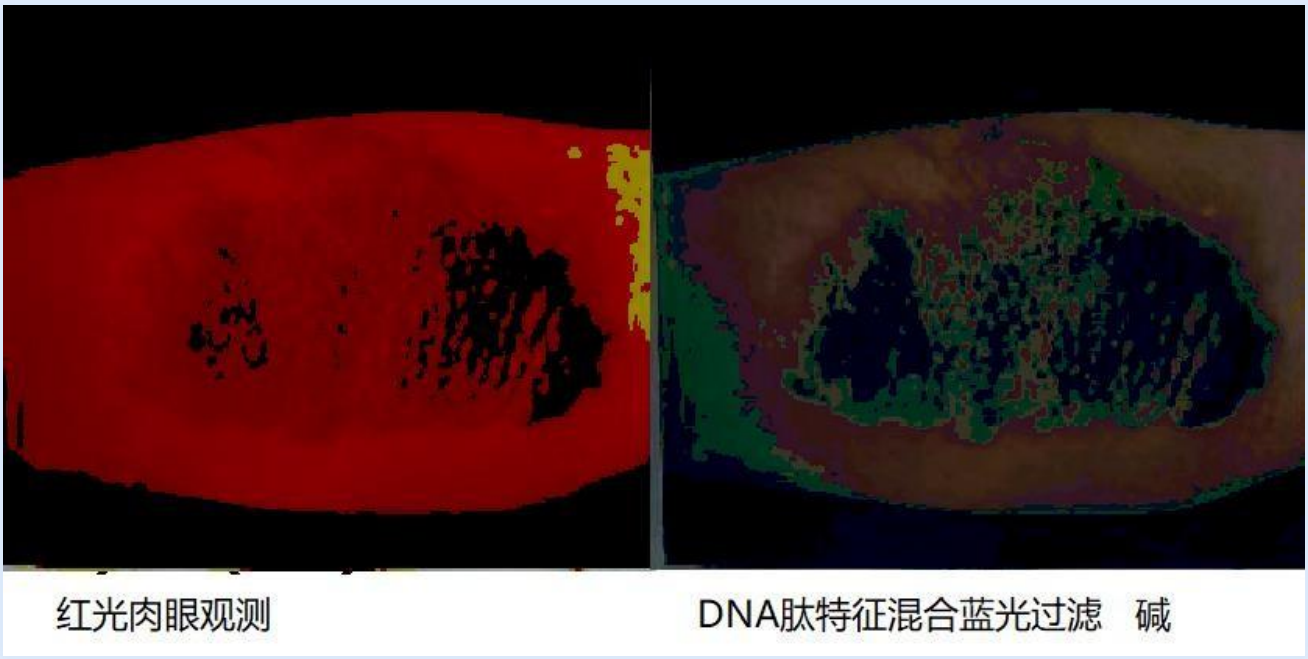
脾胃虚弱 DNA 肽展特征 酸　　**脾胃虚弱** DNA 肽展特征 碱



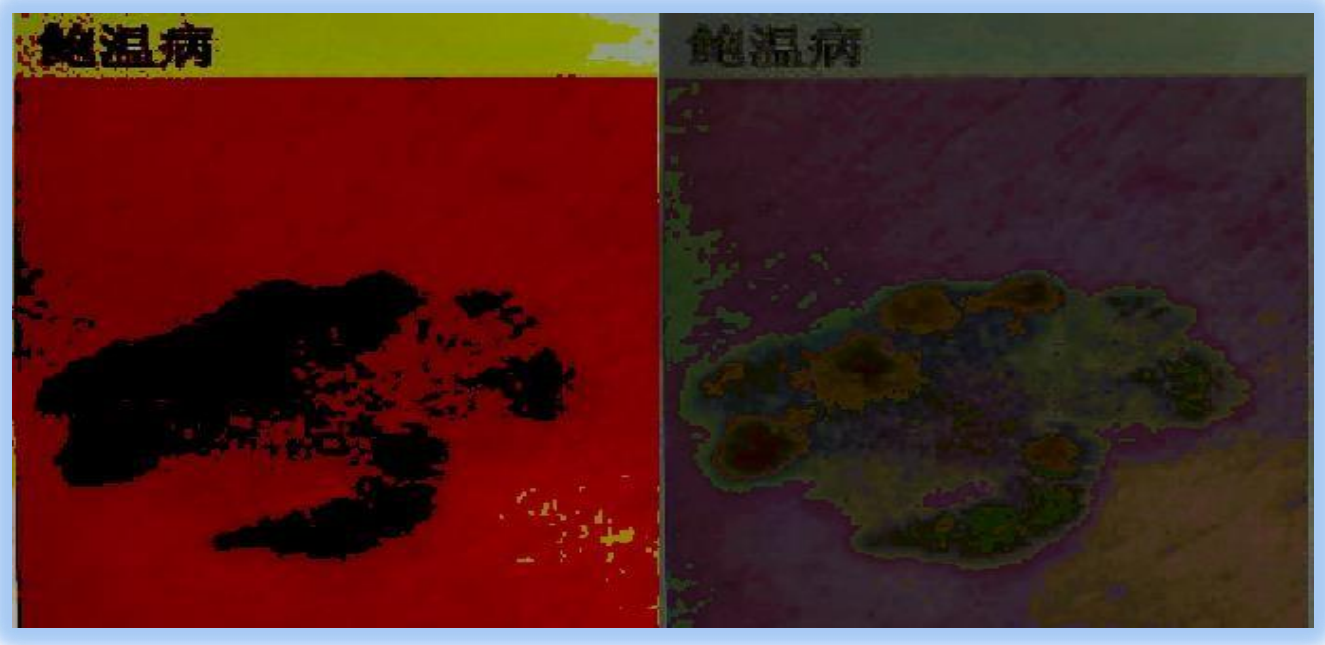
上图丛状神经纤维瘤 与 下图鲍温病对比（边缘分布特征）



肉眼观测 DNA 肽展特征与腐蚀混合观测 碱



上图丛状神经纤维瘤 与 下图鲍温病对比（色素开始分层）



红光肉眼观测　　DNA 肽特征混合蓝光过滤 碱

上面这些图片, 比较清晰的描述了 DNA 视觉的应用面广阔, 特别是在模糊的数据中, 价值体现比较大, 特别是在 模糊轮廓显影, 模糊像素集反差和分层差值填充, 速度快, 质量高, 应用领域广.