模式识别
基于马氏距离的
遥感影像监督分类

（遥感信息工程学院 2012 级）

班　　级：1201

姓　　名：陈卉君

学　　号：2012302590011

指导教师：马洪超

一、 算法原理

Mahalanobis 距离定义考虑了变量间(样本)相关性的影响,是一种更广义的距离定义。等式中计算了方差与协方差,因此内部变化较大的聚类组将产生内部变化同样较大的类,反之亦然。例如:正确分类的象素可能与其平均值的距离大于属于水体类型的象素值与其平均值的距离, 因为对水体类型来说,一般内部变化不大。马氏距离公式如下:

$D = (X - M_c)T(Cov_c{}^{-1})(X - M_c)$

其中:

D:Mahalanobis 距离

C:某一特定类

X:像素的测量向量

M_c:类型 C 的模板的平均向量

Cov_c:类型 C 的模板中像素的协方差矩阵

T:转置函数

像素将被归入到 D 值最小的类型 C 中。

马氏距离分类方法的主要步骤:

(1)确定需要分类的地区和使用的波段和特征分类数,检查所用各波段或特征分量是否相互已经位置配准;

(2)根据已掌握的典型地区的地面情况,在图像上选择训练区;

(3)计算图像的协方差矩阵,根据选出的各类训练区的图像数据,计算各类均值,确定分类半径;

(4)分类,将训练区以外的图像像元逐个逐类地代入公式,对于每个像元,分几类就计算几次,最后比较所得马氏距离的大小,选择最大值得出类别;

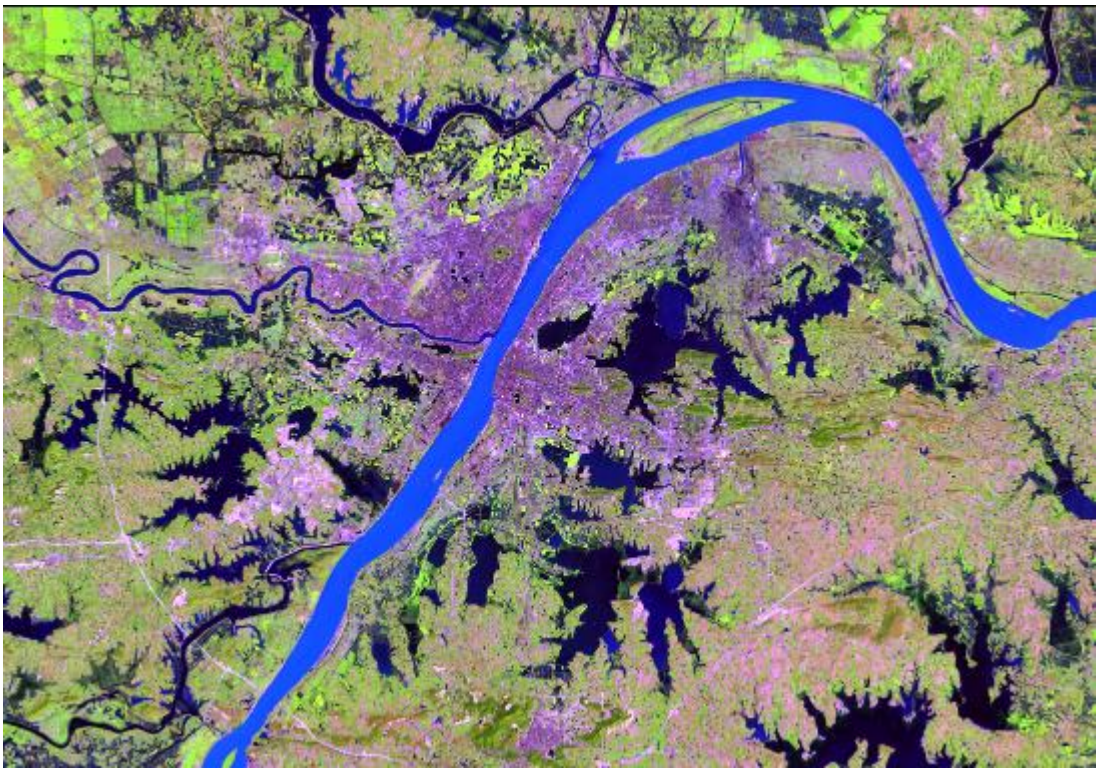
(5)产生分类图,给每一类别规定一个值,如分 10 类,就定每一类分别为 1,2,…,10,分类后的像元值便用类别值代替,最后得到的分类图像就是专题图像。由于最大灰阶值等于类别数,在监视器上显示时需要给各类加上不同的彩色;检验结果,如果分类中错误较多,需要重新选择训练区再作以上各步,直到结果满意为止。

这种方法的优点是,考虑了类型的内部统计指标。缺点是,在协方差矩阵中使用较大的值易于导致对模板(signature)过渡分类,如果在聚类组成训练样本中象素的分布离散程度较高,则协方差矩阵中就会出现大值,计算起来也比较慢;Mahalanobis 距离是参数形式的,意味着每一输入波段的数据必须是正态分布的。

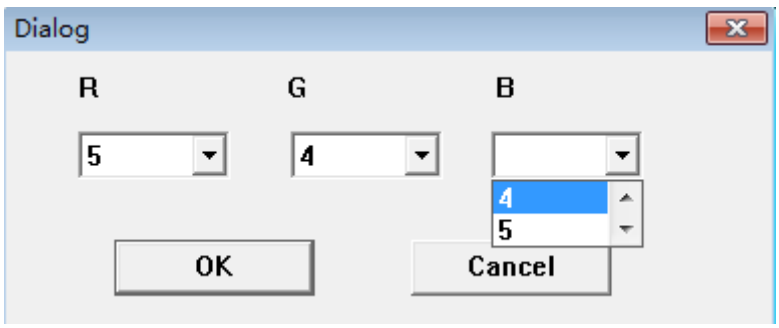
二、 代码实现

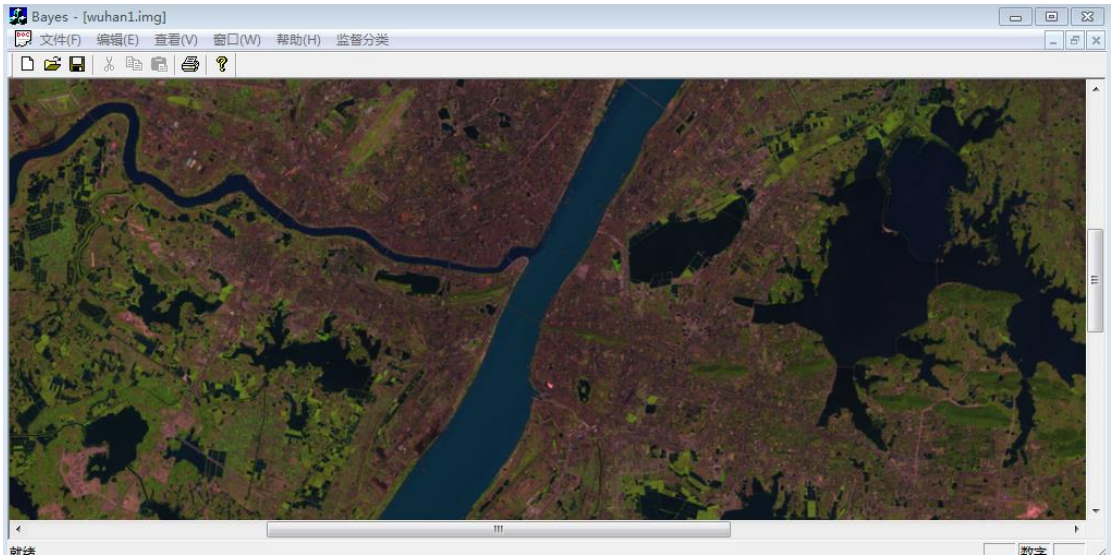
1. 数据的读取

用于分类的影像是在地理空间数据云上下载的武汉市的 TM 影像，使用 ERDAS 软件将 TM 的 1 到 5 波段以及第 7 波段进行合成，合成后影像格式为 img，下图为 543 波段作为 RGB 显示的影像。



为了读取 img 格式的影像，使用了 GDAL(Geospatial Data Abstraction Library)开源库，这是一个可操作各种栅格地理数据格式的库。为了能基于 GDAL 对图像进行显示，还需要使用 CDib 类将影像数据需要作为 RGB 显示的波段存入 BMP 图像来显示。可以选择波段。





使用 ENVI 软件给水体、城镇用地、植被、未利用地四类地物分别选出一定数量的训练样本，用 ENVI 的 Separability Report 功能检查训练样本质量，发现分离系数均大于 1.8（如下图），所以认为样本合格，保存为 txt 文件。

```
Pair Separation (least to most):

grass [Green] 10427 points and bareland [Yellow] 4452 points - 1.83314454
building [Red] 10291 points and bareland [Yellow] 4452 points - 1.97846307
building [Red] 10291 points and grass [Green] 10427 points - 1.98972021
water [Blue] 22303 points and building [Red] 10291 points - 1.99983333
water [Blue] 22303 points and grass [Green] 10427 points - 1.99999993
water [Blue] 22303 points and bareland [Yellow] 4452 points - 2.00000000
```

设置了两个类 AOIPoint 和 Kind，其定义如下：

```
class AOIPoint
{
public:
    DWORD ID; //点编号
    DWORD X;  //X 值
    DWORD Y;  //Y 值
    int kindID;  //所分类别名
    int *bands;  //该点不同波段所对应 DN 值
public:
    AOIPoint();
    virtual ~AOIPoint();
};
```

```
class Kind
{
public:
    int kindID; //编号
    CString name; //名称
    RGBQUAD rgbvalue; //类别标识颜色
    AOIPoint *points; //类别包含点
    DWORD pointnum; //包含点的数量
    int bandnum; //波段数量
public:
    Kind();
    virtual ~Kind();
};
```

利用这两个类，根据 AOI 数据开头提供的类别数量、各类别点的数量以及波段数量动态分配内存，实现训练样本的读取。

2. 算法实现

```
    int nbands = pDoc->nbands; //波段数量
    //kindnum 表示分类的数量
    double **X;  //X 用于存储每个类的各波段均值
    X = create_array(kindnum,nbands);  //为一个动态分配内存

    //求每个类的各波段均值
    long int sum = 0;
    for(int i=0;i<kindnum;i++)
        for(int j=0;j<nbands;j++)
        {
            sum=0;
            for(int k=0;k<pkind[i].pointnum;k++)
                //pkind[i].points[k].bands[j]表示第 i 类的第个 k 点的第 j 波段
DN 值
                sum += pkind[i].points[k].bands[j];
```

```
            X[i][j] = sum/pkind[i].pointnum;
        }

    //C 用来存储每个类的协方差矩阵
    double*** C = (double***)malloc(sizeof(double**)*kindnum);

    //求每个类的协方差矩阵
    for (i=0;i<kindnum;i++)
    {
        DWORD pointnum=pkind[i].pointnum; //第 i 个类的样本点数量
        C[i] = create_array(bandnum,bandnum);
        double **tC = create_array(bandnum,bandnum);
        double** t=create_array(pointnum,bandnum);

        for(int j=0;j<pointnum;j++)
            for(int k=0;k<bandnum;k++)
                t[j][k] = pkind[i].points[j].bands[k];
        cov(t,pointnum,bandnum,tC);  //计算协方差矩阵
        Gauss(tC,C[i],bandnum);  //高斯-约当法求协方差矩阵的逆

        free_array(t,pointnum);
        free_array(tC,bandnum);
    }

    DWORD lHeight = pDoc->lHeight; //遥感影像的高
    DWORD lWidth = pDoc->lWidth;  //遥感影像的宽
    double min = 10000000; int minID; //用于找最小的马氏距离及其对应的类

    //把分类结果存在 Result 矩阵中
    double** Result = create_array(lHeight,lWidth);
    for (DWORD j=0;j<lHeight;j++)
    {
```

```
for (DWORD i=0;i<lWidth;i++)
{
    BYTE *x = (BYTE*)malloc(sizeof(BYTE)*bandnum); //待分类点
    double *d = (double*)malloc(sizeof(double)*kindnum);

    //为待分类点获取数据
    for(DWORD k=0;k<bandnum;k++)
        x[k] = *(pDoc->pData+i+j*lWidth+k*lWidth*lHeight);

    //计算马氏距离公式
    for(k=0;k<kindnum;k++)
    {
        double **a = create_array(bandnum,1);
        for (int q=0;q<bandnum;q++)
            a[q][0] = x[q] - X[k][q];

        double **ta = create_array(1,bandnum);
        zhuanzhi(a,1,bandnum,ta); //矩阵转置
        double **t1 = create_array(1,bandnum);
        xiangcheng(ta,C[k],1,bandnum,bandnum,t1); //矩阵相乘
        double **t2 = create_array(1,1);
        xiangcheng(t1,a,1,1,bandnum,t2);
        d[k] = **t2;

        free_array(ta,1);
        free_array(t1,1);
        free_array(t2,1);
        free_array(a,bandnum);
    }
    //找出该点最小的马氏距离
    for(k=0;k<kindnum;k++)
    {
```

```
                if(d[k]<min)
                {
                    min=d[k];
                    minID=k;
                }
            }
            Result[j][i] = minID;
            min = 1000000;

            delete[] d;
            delete[] x;
        }
    }
```

3. 精度评定

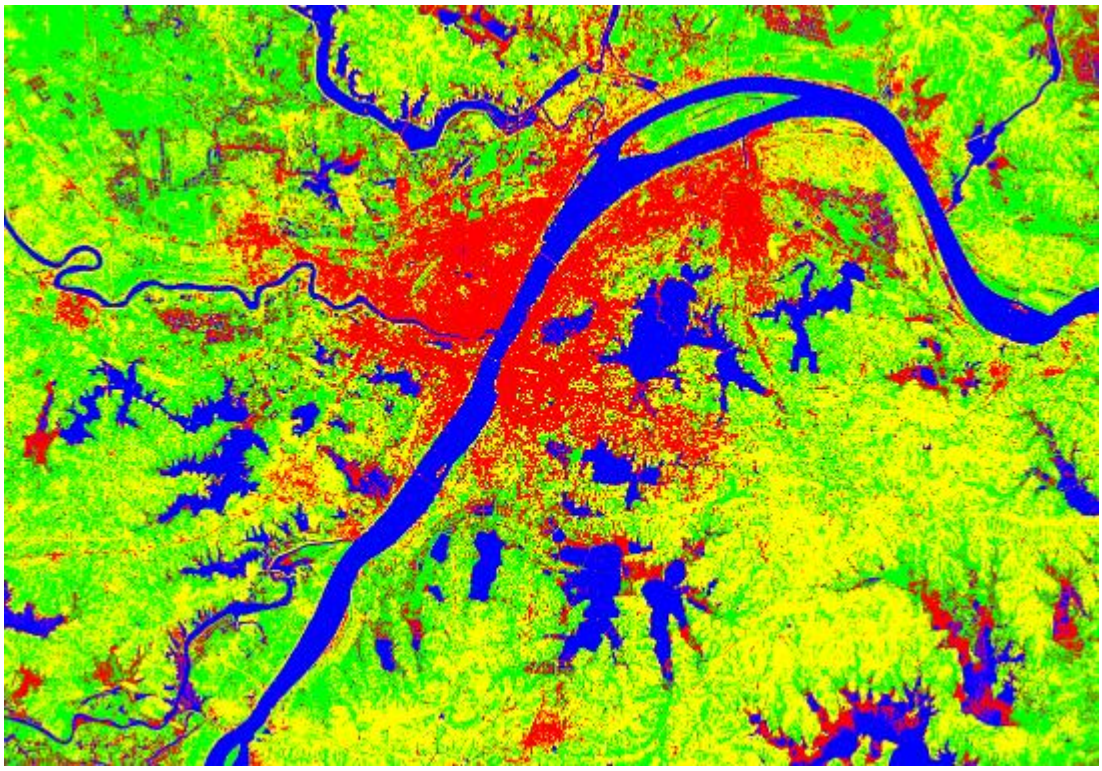
采用混淆矩阵对模板进行分类精度评定，结果如下表，可以看出水体样本选取得最好，而草地和未利用地则相对容易混淆。

	water	building	grass	bare land
water	22292	3	0	8
building	0	10064	25	202
grass	3	39	9488	897
bare land	0	30	195	4227

4. 结果保存

根据 AOI 文件中给的各类的 RGB 值以及上述算法所得的分类结果，生成一个 BMP 文件作为结果，结果如下：

其中红色为城镇用地，蓝色为水体，绿色为植被，黄色为未利用地。



三、心得体会

1. 对类的设计有待加强
2. 对算法的理解是写好程序的基础和关键，对算法理解透了才能更有效率地设计程序
3. 需要了解一下遥感图像处理中常用的开源库，如 GDAL
4. 对于误操作要有相应处理方法和警告。
5. 把程序分成几个功能块，写一步测试一步调试效率较高