情報メディアプロジェクトⅠ：演習課題レポート 2

澤 祐里（21–1–037–0801）

2021 年 7 月 24 日

目次

1　課題内容

胃 X 線像 6 枚を用いて、健常胃と異常胃を分類するための方法を提案し、実験結果（実行結果）を示す。更に、その結果から提案手法を考察する。

2　健常・異常症例の特徴

健常胃の画像と異常胃の画像の特徴を以下にまとめる。

- 健常胃は細い管のようなものが確認できる。その管は 2 枚目の画像では、はっきり移っているものの、他の画像ではうっすらとしか写っていない。
- 異常胃には、マスクメロンのような模様が 3 枚全てで確認できる。一方、健常胃では、このような模様は確認できなかった。
- 等高線、判別分析法による 2 値化、エッジ検出、コントラスト調整などの前処理を行ったが、マスクメロン模様が一部消失してしまったため、画像の画素値は画像全体で一定ではなく、マスクメロン模様の画素値は統一されていないことが分かった。また、画像によっては、マスクメロン模様がすべて消えてしまったりしたため、同じクラスの画像においても、画素値は統一されていないことが分かった。

3　提案手法

- 前項の内容から、前処理は行わず、固定の画素値にとらわれずに異常胃を判別する方法を考える。
  → まず、健常胃の特徴は、画像を識別できるほどの特徴ではないと考え、異常胃のマスクメロン模様で判別しようと考えた。マスクメロン模様は異常胃の全ての画像に、びっしりと敷き詰められている。これを、異常胃には同じようなパーツが複数存在していると見立たてると、同じようなパーツが複数存在したら、異常胃であり、そうでなければ健常胃であると判別できると考えた。
    → そこで、私は 3 × 3 の配列を作成し、ある画素の周りを含めた 9 画素をその配列に格納し、中心の画素値と周りの 8 画素を比較する方法を考えた。この方法では具体的な画素値は必要とせず、比較を行うだけであるため、それぞれ画素値が違うマスクメロン模様の特徴をとらえることができると考えた。そのため、これを全画素に対して行い、比較の結果の合計を特徴量として、距離を計算し、ソートすることで異常胃を判別できると考えた。

以下に具体的な手順を示す。

1. 図 1 のように、全画像の読み込みを行い、図 2 のインスタンスを作成。
2. 画素値を入れるための 3 × 3 の配列を作成する。
3. ある座標 (x,y) の画素値と周りの座標 (y-1,x-1)〜(y+1,y+1) の画素値を配列に格納する。
4. 図 3 のようにして、図 4 のような評価関数を呼び出し、中心の画素値と周りの画素値を比較する。
5. (y-1,x-1)〜(y+1,y+1) まで行い、中心の画素値より大きければ、配列 pm[0]〜pm[7] の値を 1 増やして、小さければ、配列 pm[8]〜pm[15] の値を 1 増やす。
6. 上記の処理を端 1 画素を除いた全画素に行うまでループする。
7. 配列 pm[0]〜pm[15] の値をその画像の特徴量として扱い、それぞれの画像と入力画像の特徴量の距離を図 5 のように求める。
8. 距離が近い順に判別結果とする。

```
public static void main(String[] args)
{
    final int normalnum = 3;
    final int abnormalnum = 3;
    identification[] normal = new identification[normalnum];
    identification[] abnormal = new identification[abnormalnum];

    for(int i=0; i<normalnum+abnormalnum; i++){ //全画像データから特徴抽出
        String fileName;
        if(i<normalnum){
            fileName = "last2data/normal/n" + Integer.toString(i+1) + ".bmp";
            GImage img= new GImage(fileName);
            normal[i] = new identification(img);

        }
        else {
            fileName = "last2data/abnormal/ab" + Integer.toString(i+1-normalnum) + ".bmp";
            GImage img= new GImage(fileName);
            abnormal[i - normalnum] = new identification(img);
        }
    }
```

図 1　画像読み込み

```
class identification {

    GImage img;
    final int width;
    final int height;

    identification(GImage img){ //コンストラクタ
        width = img.getWidth();
        height = img.getHeight();
        this.img = img;
    }
```

図 2　idenfication クラス

```
double[][] f = new double[6][16];
int count=16;

for(int i=0;i<3;i++){
    int[] num = normal[i].evaluation();
    count = num.length;
    for(int j=0;j<num.length;j++){
        f[i][j] = num[j];
    }
}
for(int i=0;i<3;i++){
    int[] num = abnormal[i].evaluation();
    for(int j=0;j<num.length;j++){
        f[i+3][j] = num[j];
    }
}
```

図 3　評価関数呼び出し

```
int[] evaluation(){
    int[][] filter = new int[3][3];
    int range = 1;
    int[] pm = new int[16];
    int x_count;
    int y_count;

    for(int y=0;y<height;y++) {
        for(int x=0;x<width;x++) {
            if(y-range >= 0 && y+range < height && x-range >= 0 && x+range < width){
                for(int i=-range;i <= range;i++){
                    for(int j=-range;j <= range;j++){
                        filter[i+range][j+range] = this.img.pixel[y + i][x + j];
                    }
                }
                x_count = 0;
                y_count = 0;
                for(int i=0;i<8;i++){
                    if(i == 3 || i == 5){
                        y_count++;
                        x_count = 0;
                    }

                    if(filter[y_count][x_count] > filter[1][1]){
                        pm[i]++;
                    }
                    else if(filter[y_count][x_count] < filter[1][1]){
                        pm[i + 8]++;
                    }
                    if(i == 3){
                        x_count += 2;
                    }
                    else{
                        x_count++;
                    }
                }
            }
        }
    }
    return pm;
}
```

図 4　評価関数

```
double[] dist = new double[6];
double total = 0;
int select = 2;

for(int i=0;i<normalnum+abnormalnum;i++){
    total = 0;
    for(int j=0;j<count;j++){
        total += (f[select-1][j] - f[i][j]) * (f[select-1][j] - f[i][j]);
    }
    dist[i] = Math.sqrt(total);
}
double[] rank = new double[dist.length];
for(int i=0;i<rank.length;i++){
    rank[i] = dist[i];
}
Arrays.sort(dist);
int[] result = new int[normalnum+abnormalnum];

for(int i=0;i<normalnum+abnormalnum;i++){
    for(int j=0;j<normalnum+abnormalnum;j++){
        if(dist[i] == rank[j]){
            result[i] = j+1;
        }
    }
}
for(int i=0;i<result.length;i++){
    if(result[i] <= 3){
        System.out.println("normal:"+result[i]);
    }
    else{
        System.out.println("abnormal:"+result[i]);
    }
    System.out.println("距離:"+dist[i]);
}
```

図 5　距離計算とソート

4　実験結果

実行結果は以下の表 1 から表 6 のようになった。実行結果を見ると、「abnormal:3」と「normal:1」が足を引っ張っていることが分かる。この 2 枚の画像以外においては、識別ができている。

表 1　normal:1 の計算結果

順位	図形の種類	検索キーとの距離
検索キー	normal:1	0
1	abnormal:1	57064.65
2	abnormal:2	67249.68
3	normal:2	127601.82
4	abnormal:3	185618.27
5	normal:3	195216.89

表 2　normal:2 の計算結果

順位	図形の種類	検索キーとの距離
検索キー	normal:2	0
1	abnormal:3	60024.76
2	normal:3	69234.29
3	normal:1	127601.82
4	abnormal:1	180487.69
5	abnormal:2	192763.47

表 3　normal:3 の計算結果

順位	図形の種類	検索キーとの距離
検索キー	normal:3	0
1	abnormal:3	14820.37
2	normal:2	69234.29
3	normal:1	195216.89
4	abnormal:1	249269.56
5	abnormal:2	261287.28

表 4　abnormal:1 の計算結果

順位	図形の種類	検索キーとの距離
検索キー	abnormal:1	0
1	abnormal:2	14531.47
2	normal:1	57064.65
3	normal:2	180487.69
4	abnormal:3	239523.59
5	normal:3	249269.56

表 5　abnormal:2 の計算結果

順位	図形の種類	検索キーとの距離
検索キー	abnormal:2	0
1	abnormal:1	14531.47
2	normal:1	67249.68
3	normal:2	192763.47
4	abnormal:3	251402.24
5	normal:3	261287.28

表 6　abnormal:3 の計算結果

順位	図形の種類	検索キーとの距離
検索キー	abnormal:3	0
1	normal:3	14820.37
2	normal:2	60024.76
3	normal:1	185618.27
4	abnormal:1	239523.59
5	abnormal:2	251402.24

5 考察

- 実験結果より、この方法においては、異常胃の画像「abnormal:3」の検索結果の上位 3 枚の画像が健常胃であることから、健常胃に近い特徴であることが分かった。しかし、他の異常胃の画像の「abnormal:1」と「abnormal:2」の距離は「14531.47」と互いに近い距離に存在していることから、「abnormal:3」は、他の異常胃画像に比べて、異常胃の中でも写り方が異なっていることなどが考えられる。
- 実験結果より、この方法においては、健常胃の画像「normal:1」の検索上位 2 枚の画像が異常胃であることから、異常胃に近い特徴であることが分かった。今回の方法では、異常胃に存在するマスクメロン模様の特徴を同じパーツが複数存在すると捉え、どれだけ同じ特徴が存在するかで識別しようとしている。しかし、この特徴は画素値の大小関係の比較だけであるため、例え画素値が大きく異なったとしても、大小関係さえ満たしていれば、同じ特徴としてみなされる。これにより、「normal:1」は、健常胃にも関わらず、異常胃と大小関係の構造が似ていたため、異常胃と識別されたと考えられる。

6　むすびと感想

今回の実験では、胃 X 線像 6 枚を用いて、異常胃に存在するマスクメロン模様の特徴を同じパーツが複数存在すると捉え、どれだけ同じ特徴が存在するかで、健常胃と異常胃の分類を行った。その結果、同じ異常胃の画像でも、写り方などの原因により、特徴の抽出が難しい画像が存在することが分かった。また、上記ことから、前処理の重要性も理解することができた。この課題の感想：自分で 1 から問題を解決することの難しさに気づけて、面白かった。

参考文献

[1] https://www.info.kindai.ac.jp/MedProI/