システム創成 プロジェクトI
画像認識 演習（第4回）

システム創成情報工学科

演習担当：尾下 真樹、齊藤 剛史、
徳永 旭将、宮野 英次

プロジェクトⅠ 日程(1)

- 1週目 画像認識(1)
  - 3限目 講義(特徴量を使った識別)(佐藤)
  - 4限目 講義(演習説明)(尾下or宮野or齊藤or徳永)
  - 5限目 演習
- 2週目 画像認識(2)
  - 3限目 講義(演習説明)(尾下or宮野or齊藤or徳永)
  - 4～5限目 演習
- 3週目 画像認識(3)
  - 3～5限目 演習
- 計画書提出(3週目5限目まで)

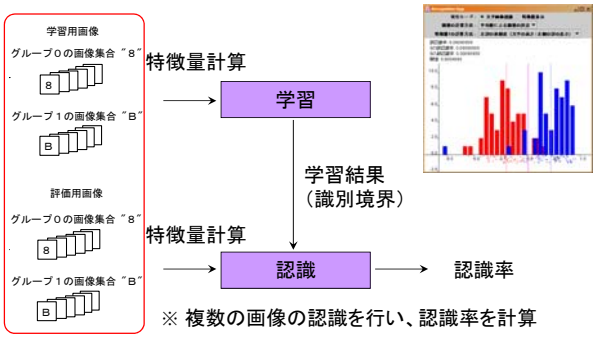
プロジェクトⅠ 日程(2)

- プログラム提出(4週目の前日まで)
- 課題画像収集作業を4週目3限目までに終える。
  スキャン作業は4週目5限目までに終える。
- 4週目 識別精度
  - 3限目 講義(識別精度)(本田)
  - 4限目 講義(演習説明)(尾下or宮野or齊藤or徳永)
  - 5限目 演習
- 5週目 自由演習
  - 3～5限目 演習
- 6週目 プレゼンテーション
  - 3～4限目 プレゼンテーション

識別精度評価 演習

- これまでのプログラムでは、学習用と評価用に同じ画像データを使用していた
  - 識別精度を評価する上では望ましくない
    - 学習に使ったデータを正しく認識できるのは当たり前、未知のデータを正しく認識できるかが重要
- 今回の演習では、学習用と評価用に使用するデータを変更するプログラムを追加し、識別精度を確認する
  - 10-Fold Cross Validation法
  - (Bootstrap法は、作成・実験しなくとも良い)

画像認識プログラムの概要(復習)



識別精度評価 演習

- 学習用・評価用に、それぞれ大量のデータを用意することができればベストだが、それには手間がかかるので、なるべく少ないデータで精度を正しく評価できる方法を用いる
- Cross Validation法
  - 用意したデータの一部(少数)を評価用に、残り(多数)を学習用に分けて利用
- Bootstrap法
  - 評価用データを学習用データからランダムに選択

注意

- 識別精度と、識別精度の精度を区別して考える必要がある

1. 識別精度
   - 自分の作成したプログラムで画像を正しく識別できたか？（＝誤認識率）
2. 識別精度の精度
   - 自分が求めた「1. 識別精度」は、正確か？
   - 正確な識別精度を求めるためには、適切な学習用・評価用データを使って計算する必要がある

演習内容・手順

- 識別精度評価の方法を追加する
  - 変更するのは RecognitionApp.java のみ
  - 資料の12章にもとづいてプログラムを変更

1. 学習用画像と評価用画像の配列を追加し、学習と評価に異なる画像を使えるようにする
2. 全画像を学習・評価に使用する方法を追加
3. Cross Validation法を追加
4. 実験

変更前のプログラムの確認（1）

- 学習・評価に共通の画像配列を使用

```
// サンプル画像
protected BufferedImage   sample_images0[];
protected BufferedImage   sample_images1[];
```

  - sample_images0　グループ0の画像配列
  - sample_images1　グループ1の画像配列

変更前のプログラムの確認（2）

- 学習・評価処理

```
// サンプル画像を使った文字画像認識のテスト
public void   recognitionTest()
{
        // 全てのサンプル画像を使って学習
        recognizer.train( sample_images0, sample_images1 );

        // 全てのサンプル画像を使って誤認識率を計算
        ....
        for ( int i=0; i<sample_images0.length; i++ )
        {
                char_no = recognizer.recognizeCharacter( sample_images0[ i ] );
                ....

        }
        for ( int i=0; i<sample_images1.length; i++ )
        {
                char_no = recognizer.recognizeCharacter( sample_images1[ i ] );
                ....

        }
        ....
}
```

学習用と評価用の画像を区別（1）

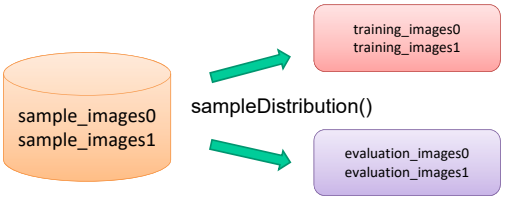
- 学習用と評価用の画像配列を定義

```
// 学習用画像
protected BufferedImage   training_images0[];
protected BufferedImage   training_images1[];

// 評価用画像
protected BufferedImage   evaluation_images0[];
protected BufferedImage   evaluation_images1[];
```

  - 学習用画像　training_images0, training_images1
  - 評価用　evaluation_images0, evaluation_images1

学習用と評価用の画像を区別（2）

- 読み込んだ画像を、学習用と評価用の画像配列に振り分け
  - sampleDistribution() メソッド内で、使用する方法に応じて、振り分け処理を実行



学習用と評価用の画像を区別（3）

- 学習・評価に各画像配列を使用するよう変更

```
public void  recognitionTest()
{
        // 学習・評価に用いるサンプル画像の決定
        sampleDistribution();

        // 学習用画像を使って学習
        recognizer.train( training_images0, training_images1 );

        // 評価用画像を使って認識率を計算
        ・・・
        for ( int i=0; i<evaluation_images0.length; i++ )
        {
                char_no = recognizer.recognizeCharacter( evaluation_images0[ i ] );
                ・・・
        }
        for ( int i=0; i<evaluation_images1.length; i++ )
        {
                char_no = recognizer.recognizeCharacter( evaluation_images1[ i ] );
                ・・・
        }
        ・・・
}
```

演習内容・手順

- 識別精度評価の方法を追加する
  - 変更するのは RecognitionApp.java のみ
  - 資料の12章にもとづいてプログラムを変更

1.学習用画像と評価用画像の配列を追加し、学習と評価に異なる画像を使えるようにする

2.全画像を学習・評価に使用する方法を追加

3.Cross Validation法を追加

4.実験

精度評価方法の切替

- どの方法を用いるかを示す変数を追加
  - 列挙型 DistributionMethod を定義

```
// 学習用・評価用画像の決定方法の設定
enum   DistributionMethod
{
        USE_ALL_SAMPLES,    全画像を学習・評価に使用
        CROSS_VALIDATION,   Cross Validation法
        BOOTSTRAP           Bootstrap法
};

// 学習用・評価用画像の決定方法
protected DistributionMethod   distribution_method =
                        DistributionMethod.CROSS_VALIDATION;
```

全画像を学習・評価に使用する方法

- sampleDistribution()メソッドでの処理

```
// 全てのサンプル画像を学習と評価に使用
if ( distribution_method == DistributionMethod.USE_ALL_SAMPLES )
{
        // 全てのサンプル画像を学習用画像の配列にコピー
        training_images0 = new BufferedImage[ sample_images0.length ];
        for ( int i=0; i<sample_images0.length; i++ )
                training_images0[ i ] = sample_images0[ i ];
        training_images1 = new BufferedImage[ sample_images1.length ];
        for ( int i=0; i<sample_images1.length; i++ )
                training_images1[ i ] = sample_images1[ i ];

        // 全てのサンプル画像を評価用画像の配列にコピー
        evaluation_images0 = new BufferedImage[ sample_images0.length ];
        for ( int i=0; i<sample_images0.length; i++ )
                evaluation_images0[ i ] = sample_images0[ i ];
        evaluation_images1 = new BufferedImage[ sample_images1.length ];
        for ( int i=0; i<sample_images1.length; i++ )
                evaluation_images1[ i ] = sample_images1[ i ];
}
```

Cross Validation法（1）

- Cross Validation のための変数を追加

```
// Cross Validation 法を用いるときのグループ数
protected int   cv_number_of_folds = 4;

// Cross Validation 法を用いるとき、何番目のグループを評価に使用するか設定
protected int   cv_evaluation_fold = 0;
```

読み込んだ画像全体

cv_number_of_folds 個のグループに分ける

cv_evaluation_fold 番目のグループを評価に、
他のグループを学習に用いる

Cross Validation法（1）

- Cross Validation のための変数を追加
  - 以下は 4-fold の図で説明（実験は10-foldで行う）

```
// Cross Validation 法を用いるときのグループ数
protected int   cv_number_of_folds = 4;    ※ 実験では 10 とする

// Cross Validation 法を用いるとき、何番目のグループを評価に使用するか設定
protected int   cv_evaluation_fold = 0;
```

読み込んだ画像全体

cv_number_of_folds 個のグループに分ける

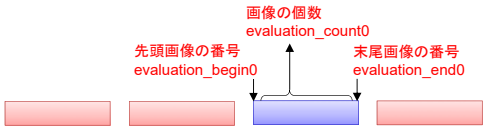
cv_evaluation_fold 番目のグループを評価に、
他のグループを学習に用いる

Cross Validation法（2）

- 学習用と評価用の画像配列に振り分け
  - sampleDistribution()メソッドに処理を追加

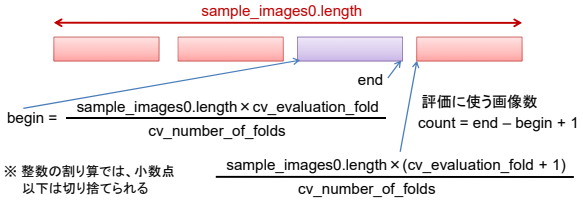
```
// Cross Validation 法を使用
if ( distribution_method == DistributionMethod.CROSS_VALIDATION )
{

}
```

  - 評価に使う画像の範囲を示す変数を定義・計算



Cross Validation法（3）

- 評価に使う画像の範囲を示す変数を計算
  - evaluation_begin0, evaluation_end0, evaluation_count0
  - 全画像数（sample_image0.length）、cv_number_of_folds、cv_evaluation_fold にもとづいて計算

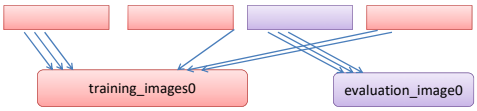


Cross Validation法（4）

- 学習用と評価用の画像配列に振り分け
  - 各画像を格納する配列を初期化

```
training_images0  = new BufferedImage[必要なサイズ];
evaluation_images0 = new BufferedImage[必要なサイズ];
```

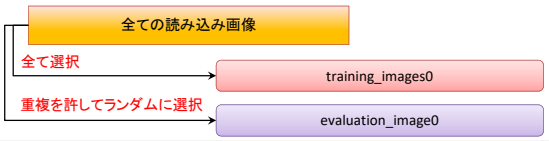
  - 求めた変数にもとづいて、振り分け処理



- グループ0・1の両方で同様の処理を実行

参考：Bootstrap法

- 学習用画像には、全ての画像を使用
- 評価用画像には、全ての画像から、重複を許してランダムに選択
  - 全画像と同じ個数分、評価画像をランダムに選択
    - 乱数は java.lang.Math.random() で取得可能（0.0以上～1.0未満の実数を返す）



実験・考察

- 今回作成した各方法で実験を行い、認識評価の結果がどのように変化するかを確認・考察
  - 全画像を学習・評価に使用する方法
    （dm = DistributionMethod.USE_ALL_SAMPLES）
  - 10-Fold Cross Validation法
    （dm = DistributionMethod.CROSS_VALIDATION、
    cv_number_of_folds = 10）
    - cv_evalution_fold = 0 の場合
    - cv_evalution_fold = 1 の場合
    - ・・・
    - cv_evalution_fold = 9 の場合
    - 10回分の平均・標準偏差も求める