Normalisation of confidence voting methods applied to a fast handwritten OCR classification

Juan Ramón Rico-Juan[1] and José M. Iñesta[2]

[1] Departamento de Lenguajes y Sistemas Informáticos
Universidad de Alicante, E-03071 Alicante, Spain juanra@dlsi.ua.es [*]
[2] Departamento de Lenguajes y Sistemas Informáticos
Universidad de Alicante, E-03071 Alicante, Spain inesta@dlsi.ua.es

Summary. In this work, a normalisation of the weights utilized for combining classifiers decisions based on similarity Euclidean distance is presented. This normalisation is used by the confidence voting methods to decrease the final error rate in an OCR task. Different features from the characters are extracted. Each set of features is processed by a single classifier and then the decisions of the individual classifiers are combined using weighted votes, using different techniques. The error rates obtained are as good or slightly better than those obtained using a Freeman chain codes as contour representation and the string edit distance as similarity measure, but the complexity and classification time decrease dramatically.

1 Introduction

The combination of classifiers is a strategy widely applied recently in classification tasks. There are many ways to apply a combination of classifiers [1]. Methods based on decision confidences, like those reported in [2], allows to weight the individual classifier decisions in order to obtain a good classification performance.

There are some works which use confidence methods based in a posteriory probabilities (according to the Bayes theory) in classification, as in [3], where several formulas are proposed to estimate this probability, and in [4], where the authors propose a method based on the k-Nearest Neighbour rule.

In this work, we get into the details of the appropriate features, the individual classifiers, and their combination in order to classify isolated handwritten characters.

In the second section, we explain the different features that have been chosen for describing the character image. In the third section, the results ob-

[*] Work supported by the Spanish CICYT under project TIN2006-14932-C02, partially supported by EU ERDF and Generalitat Valenciana I+D+i under project GV06/166

tained when applying different classifiers on a widely used database of isolated handwritten characters are shown. Finally, we present some conclusions and future lines of work.

2 Feature extraction from a binary image

The goal of this feature extraction is to obtain different kinds of features to use with a specific classifier. Three different representations of the same image (figure 1a) have been designed in order to obtain relevant information. The main idea is to summarise the image matrix information in proportional square regions that represent the character shape.

The first step is to locate the character in the image and extract the region of interest (ROI) as the character bounding box. This ROI is where the features described below are extracted from.

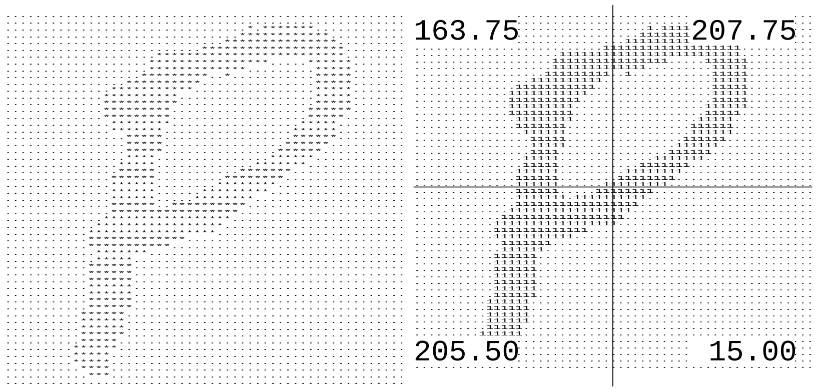
2.1 Foreground features

A classical representation of a bitmap image is a matrix with the black pixels as the smallest portions of the object ($I[x, y] = 1$) and white pixels as the background ($I[x, y] = 0$).

First, morphological filters [5] have been applied to correct gaps and spurious points. Thus, we define square subregions for representing the character ROI. Each subregion is represented by the number of foreground pixels. If a foreground pixel belongs to different regions, each region involved accumulates the proportional value of this pixel, as displayed in an example in figure 1b. The algorithm is detailed below, where $I[I_{XM}, I_{YM}]$ is the bitmap image with $I_{XM} \times I_{YM}$ dimensions and $R[R_{XM}, R_{YM}]$ is the matrix with the proportional summatory of black pixels with $R_{XM} \times R_{YM}$.

```
function computeSummatoryOfRegions($I[I_{XM}, I_{YM}]$, $R[R_{XM}, R_{YM}]$)
  for $x := 1$ to $R_{XM}$ do
    for $y := 1$ to $R_{YM}$ do
      $R[x, y] := 0$
  (startX,startY,endX,endY):=computeBoundingBoxObject($I$)
  for $i$ :=startX to endX do
    for $j$ :=startY to endY do
      if isBlackPixel($I[i, j]$) then
        case $[i, j]$ included in proportion $[x, y]$ :
         $R[x, y] := R[x, y] + 1$
        case $[i, j]$ included in proportion $[x, y]$, $[x, y + 1]$ :
         $R[x, y] := R[x, y]$+proportion(x,y)
         $R[x, y + 1] := R[x, y + 1]$+proportion(x,y+1)
        case $[i, j]$ included in proportion $[x, y]$, $[x + 1, y]$ :
         $R[x, y] := R[x, y]$+proportion(x,y)
```

```
            $R[x+1,y] := R[x,y]$+proportion(x+1,y)
        case $[i,j]$ included in proportion $[x,y]$, $[x,y+1]$, $[x+1,y]$, $[x+1,y+1]$ :
            $R[x,y] := R[x,y]$+proportion(x,y)
            $R[x,y+1] := R[x,y+1]$+proportion(x,y+1)
            $R[x+1,y] := R[x+1,y]$+proportion(x+1,y)
            $R[x+1,y+1] := R[x+1,y+1]$+proportion(x+1,y+1)
end function
```



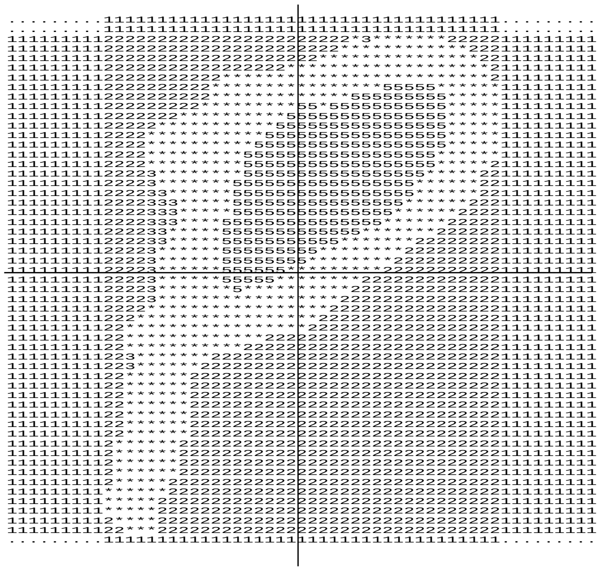
(a) (b)

Fig. 1. a) Original image b) After morphological filters and a 2×2 regions with the proportional summatory of the number of black pixels (object) for each subregion.

2.2 Background features

The algorithm used for extracting the background features is similar to that in [6]. This algorithm is based on four projections (up, down, left, and right) that are plotted for each pixel in the image. When any of these projections touch the foreground object, a counter associated to that pixel is increased in one unit. This way, we can distinguish four different categories of background pixels, according to the value of its counter. In addition, a fifth category is added in order to provide more information: there are two situations that are similar in geometry but totally different from a topological point of view. A background pixel can be surrounded by object pixels and then the projections will touch them and the counter will be 4, but this pixel could belong either to an isolated region or to an actually open region. So, our algorithm assigns a value of 5 to the counter if the pixel lies in an isolated background area.

Therefore, five matrices are extracted as features, one for each counter value. Each cell of these matrices represents the number of pixels with that counter value in the corresponding image subregion, as shown in an example in figure 2.



```
1:
  266.5 257.0
  266.5 257.0
2:
  186.5 274.5
   91.0 475.0
3:
   20.5   4.5
    5.0   0.0
4:
    0.0   0.0
    0.0   0.0
5:
  101.0 174.5
    3.5   0.0
```

Fig. 2. Background features

2.3 Contour features

We extract the object contour encoding the links between each pair of neighbour pixels using 4-chain codes in the way proposed by [7], where only the orientations of the links are taken into account (see figure 3 for an example). We extract four matrices, one for each direction. In a similar way as we do in previous subsections, each cell of these matrices represent the summatory of each region for a direction as we show in an example in the figure 3a.

All these regions are normalised using $p'_{ij} = p_{ij} / \sum p_{ij}$ to obtain better performance where p_{ij} is the cell i, j which represents the proportional summatory of the features in this region.

2.4 Classification schemes

Three kinds of algorithms have been compared based on different features descriptions:

1. Individual classifiers based on the three sets of features described above. The features used are based in the image pixels (image), in the background

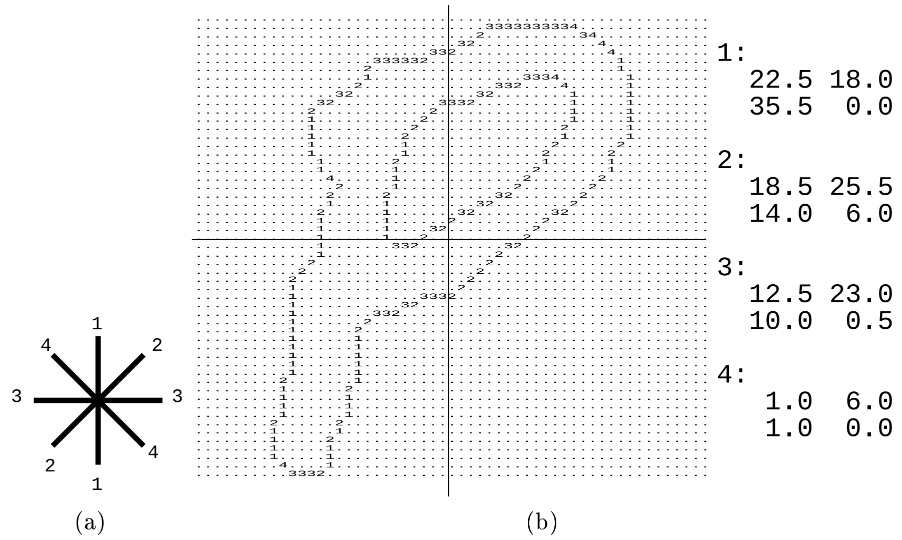


Fig. 3. a) Four 2D directions b) Contour encoded and a 2×2 matrix with the summatory of number of pixels for each direction

information (background) and in the four directions of the countours image (directions).

2. Combination of classifiers based on confidence voting methods [2]:
   - Pandemonium[8]: Every classifier gives one vote with a confidence for every class. The class which receive the highest confidence is the winner.
   - Sum rule: Each class sums the confidence from each classifier. The class that accumulates the highest confidence is the winner.
   - Product rule: It is equivalent to *sum rule*, but the product is used for obtaining the final confidence for each class.

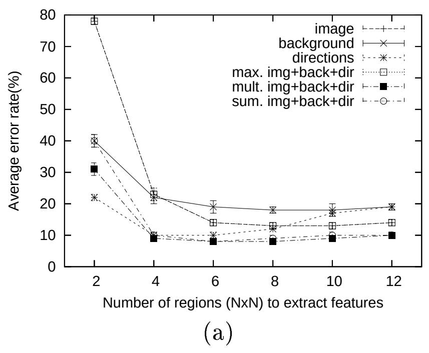
   The confidences of the single classifiers are based on the Euclidean distance to the training set samples. We use a modification of the expression to compute *a posteriory* probability based on [3] and proposed in [4]. The estimation is $\hat{P}(\omega_i|x)$ where ω_i is the i-th class and x is a new example to classify. The new estimation is computed as

$$\hat{P}(\omega_i|x) = \frac{\frac{1}{\varepsilon + \min_{y_i \in \omega_i} \{d(x,y_i)\}}}{\sum_i \frac{1}{\varepsilon + \min_{y_i \in \omega_i} \{d(x,y_i)\}}}$$

   This estimation is based on the nearest neighbour sample. A ε value is introduced that is a positive small value close to 0. It allows to apply this formula without overflow errors.
3. Contour of image as string using 8-Freeman codes as described in [9] (contour-string).

3 Results

A classification task using the NIST SPECIAL DATABASE 3 of the National Institute of Standards and Technology was performed using the different contour descriptions described above to represent the characters. Only the 26 uppercase handwritten characters were used. The increasing-size training samples for the experiments were built by taking 500 writers and selecting the samples randomly. The size of the test is 1300 example in all cases, while the training set has different sizes as 1300, 2600, 3900 and 5200, corresponding to 50, 100, 150 and 200 examples per class.



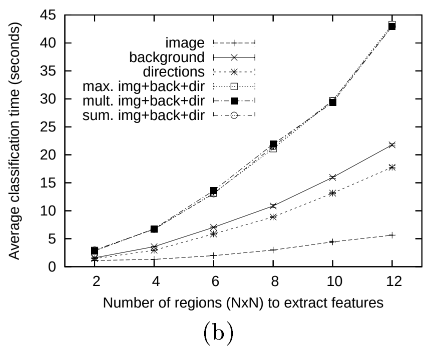
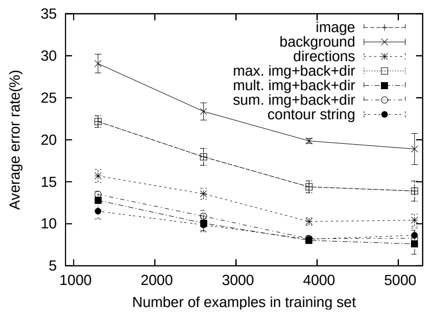


Fig. 4. Preliminary trial tests with a 1300 test examples and 5200 model examples. a) Average classification error rate b) Average classification time

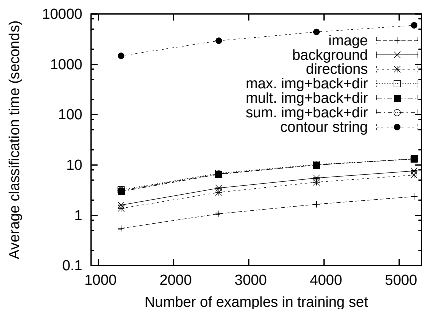
As shown figure 4, in the preliminary trials tested, different numbers of regions were tested. A division of the ROI into 6×6 subregions yielded a lower error rate than the others and the best computation time. Thus, the R_{XM} and R_{YM} parameters for the $computeSummatoryOfRegions$ function was fixed to 6×6 to compute the rest of experiments.

The sensitivity to the classifier ensemble architecture has been tested removing one of the three individual classifiers from the final combination, and the results were always worse than the combination of the three classifiers.

As we can see in the figure 5a, the lowest error rates were obtained by the combination of classifiers using the product rule. Anyway, this result is comparable to that of the combination of classifiers based on the summatory rule and the string countour classifier. In the figure 5b the classification time as a function of the number of samples is shown. Like in the case of the classifier ensemble, the time is much lower than using contour chain codes, because chain codes use a string edit distance with a quadratic complexity and the average length of the strings is of 350 codes per character, while in the proposed approach the classifiers use a, much faster, Euclidean distance.



(a)



(b)

Fig. 5. a) Average classification error rate b) Average classification time

4 Conclusions and future work

A number of ways for shape feature extraction from a binary image have been presented. Each set of features was used for handwritten character recognition by different classifiers based on Euclidean metrics. In order to achieve a better preformance, a weighted voting-based ensemble of those classifiers has been built. A posteriory probability estimation is proposed in order to normalize the confidences provided for each classifier in the voting phase for reducing the final classification error rate.

The classification time using these combination of classifiers was about 500 times faster than string edit distance-based classifiers operating with contour chain codes, obtaining even slightly better error rates.

This combination will be applied to other databases in the future to explore the robustness of the approach and how it performs in terms of both classification time and error rates with other data. In addition, we will explore its capabilities when dealing with other data in the biometric recognition field.

References

1. L. I. Kuncheva. *Combining pattern classifiers : methods and algorithms.* John Wiley & Sons, 2004.
2. Merijn van Erp, Louis Vuurpijl, and Lambert Schomaker. An overview and comparison of voting methods for pattern recognition. In *IWFHR '02: Proceedings of the Eighth International Workshop on Frontiers in Handwriting Recognition (IWFHR'02)*, page 195, Washington, DC, USA, 2002. IEEE Computer Society.
3. M. van Breukelen, D.M.J. Tax R.P.W. Duin, and J.E. den Hartog. Handwritten digit recognition by combined classifiers. *Computacional Linguistics*, 34(4):381–386, 1998.
4. J. Arlandis, J.C. Perez-Cortes, and J. Cano. Rejection strategies and confidence measures for a k-nn classifier in an ocr task. In *16th. International Conference*

on Pattern Recognition ICPR-2002, volume 1, pages 576–579, Québeq, (Canada), 2002. IEEE Computer Society.

5. J. Serra. *Image Analysis and mathematical morphology.* Academic Press, 1982.
6. E. Vellasques, L.S. Oliveira, A.S. Britto Jr., A.L. Koerich, and R. Sabourin. Modeling segmentation cuts using support vector machines. In *Tenth International Workshop on Frontiers in Handwriting Recogntion*, volume 1, pages 41–46, 2006.
7. Hideto Oda, Bilan Zhu, Junko Tokuno, Motoki Onuma, Akihito Kitadai, and Masaki Nakagawa. A compact on-line and off-line combined recognizer. In *Tenth International Workshop on Frontiers in Handwriting Recogntion*, volume 1, pages 133–138, 2006.
8. O. G. Selfridge. Pandemonium: a paradigm for learning in mechanisation of thought processes. In *Proceedings of a Symposium Held at the National Physical Laboratory*, pages 513–526, London, November, 1958. MIT Press.
9. J. R. Rico-Juan and L. Micó. Finding significant points for a handwritten classification task. In A. Campilho and M. Kamel, editors, *International Conference on Image Analysis and Recognition*, number 3211 in Lecture Notes in Computer Science, pages 440–446, Porto, Portugal, jun 2004. Springer.