Edit distance for ordered vector sets. A case of study

Juan Ramón Rico-Juan and José M. Iñesta[⋆]

Departamento de Lenguajes y Sistemas Informáticos
Universidad de Alicante, E-03071 Alicante, Spain,
{juanra, inesta}@dlsi.ua.es

Topics: Dominant Points, Pattern Recognition, Structural Pattern Recognition

Abstract. Digital contours in a binary image can be described as an ordered vector set. In this paper an extension of the string edit distance is defined for its computation between a pair of ordered sets of vectors. This way, the differences between shapes can be computed in terms of editing costs. In order to achieve efficency a dominant point detection algorithm should be applied, removing redundant data before coding shapes into vectors. This edit distance can be used in nearest neighbour classification tasks. The advantages of this method applied to isolated handwritten character classification are shown, compared to similar methods based on string or tree representations of the binary image.

1 Introduction

The description of an object contour in a binary image as a string [1] using Freeman codes [2] or using a tree representation structure [3,1] is widely used in pattern recognition. For using these structures in a recognition task, the edit distance is often used as a measure of the differences between two instances. Both, string edit distances [4] and tree edit distances [5] are used, depending on the data structures utilised for representing the problem data. In this paper, in order to obtain a representation of the object contour from a binary image, an ordered vector set is extracted, and an edit distance measure is defined between pairs of instances of this representation. This measure is an extension of the string edit distance, adding two new rules and changing vectors by symbols.

Freeman chain codes keep very fine details of the shapes since they code the relations between every pair of adjacent pixels of the contours. To avoid computation time and in order to remove irrelevant details, a dominant point detection algorithm is needed. The goal is to reduce the features that represent a binary image in order to remove redundant data to compute the distance faster, keeping the final classification time low and good error rates.

[⋆] Work supported by the Spanish CICYT under project TIC2003-08496-CO4 and Generalitat Valenciana I+D+i under project GV06/166

The remainder of this paper consists of four sections. In section 2, two different representations of the same binary image are extracted. In section 3, a new distance based in ordered vector set is defined. In section 4, the results of experiments in a classification task, applying string and ordered vector set edit distances are presented. Finally in section 5, the conclusions and future word are presented.

2 Feature extraction from a binary image

The goal of the ordered vector set is to describe the contour of an object using the least possible number of elements. The classical representation of a contour in a binary image links the contour pixels with their neighbors using 0 to 7 (see Fig. 1) codes which represent a discrete number of 2D directions. This way, a string that represents the contour is obtained (Fig. 2 top-right).

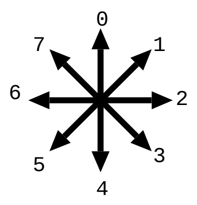


Fig. 1. Freeman 2D code

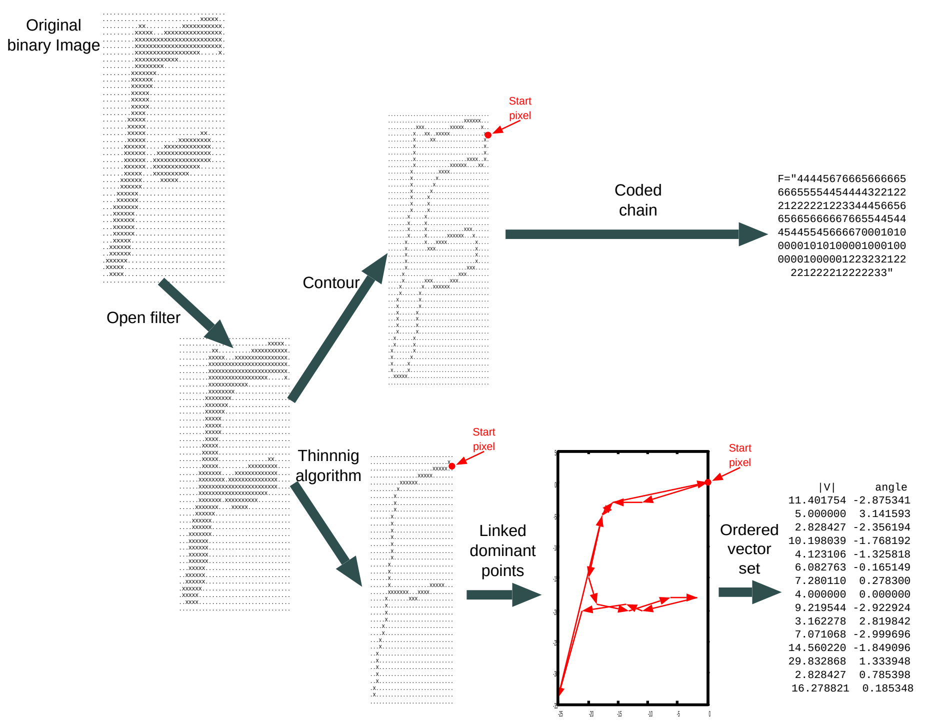
This kind of feature extraction assumes that all linked pixels are of equal importance. If we select the most representative points of the contour and link all these points, a compact representation of 2D figures is obtained, with less features than using Freeman codes.

The idea is to select a set of dominant points in a contour [6,7], link those points following the contour of the figure using 2D vectors, and then use these ordered vector set to represent the image (Fig. 2 bottom-right).

In a particular application of handwritten character recognition, it is recommended to apply some filter operations to original image before extracting and coding the contours [8] including an opening filter [9] and a thinning algorithm [10] in order to remove noise and redundant information.

3 Ordered vector set edit distance

The string edit distance definition [4] is based on three edit operations: insertion, deletion, and substitution. Let Σ the alphabet, $A, B \in \Sigma^*$ two finite strings of characters, and Λ is a null character. $A\langle i\rangle$ is the ith character of the string A; $A\langle i : j\rangle$ is the substring form the ith to jth characters of A, both inclusive.



```
F="444456766656666665
666555544544443221222
122222212233444456656
656656666676655445444
544554566666700010100
000101010000100010000
001000000122323212222
  1222221222222233"
```

```
     |V|      angle
11.401754 -2.875341
 5.000000  3.141593
 2.828427 -2.356194
10.198039 -1.768192
 4.123106 -1.325818
 6.082763 -0.165149
 7.280110  0.278300
 4.000000  0.000000
 9.219544 -2.922924
 3.162278  2.819842
 7.071068 -2.999696
14.560220 -1.849096
29.832868  1.333948
 2.828427  0.785398
 16.278821  0.185348
```

Fig. 2. General scheme. From the binary image, morphological filters are applied to correct gaps and spurious points. Thus, contour and skeleton are obtained. From the first, the chain code is obtained and from the second, the ordered vector set is extracted using a dominant point selection algorithm.

An edit operation is a pair $(a,b) \in (\Sigma \cup \{\Lambda\})^2 : (a,b) \neq (\Lambda,\Lambda)$. So, the basic edit operations are substitution $a \to b$, insertion $\Lambda \to b$ and deletion $a \to \Lambda$. If a generic cost function is associated to each operation $\gamma_s(a \to b)$, the cost of the sequence of edit operations that transforms a finite string A in B is defined as

$$d_s(A,B) = \min \begin{cases} \gamma_s(\Lambda \to B\langle 1\rangle) + d_s(A, B\langle 2:|B|\rangle) & |B| \geq 1 \\ \gamma_s(A\langle 1\rangle \to \Lambda) + d_s(A\langle 2:|A|\rangle, B) & |A| \geq 1 \\ \gamma_s(A\langle 1\rangle \to B\langle 1\rangle) + d_s(A\langle 2:|A|\rangle, B\langle 2:|B|\rangle) & |A| \geq 1 \wedge |B| \geq 1 \\ 0 & |A| = 0 \wedge |B| = 0 \end{cases}$$

The similar idea of an ordered string is extended to an ordered vector set. Let $V,W \in (\mathbb{R}\times[0,2\pi])^*$ a finite set of vectors and Λ is a null vector. $V\langle i\rangle$ is the vector ith in the set V, $V_N\langle i\rangle$ is the norm and $V_\alpha\langle i\rangle$ is the angle of the ith vector; $V\langle i:j\rangle$ is the subset from ith to jth component vectors of V, both included.

Now, an edit operation is a pair $(v,w) \in (\mathbb{R}\times[0,2\pi])\,,\,(v,w) \neq (\Lambda,\Lambda) : (v,w^*) \cup (v^*,w)$. So, the basic edit operations are substitution (1 to 1) $v \to w$, substitution (1 to N) called fragmentation $v \to w^+$, substitution (N to 1) called consolidation $v^+ \to w$, insertion $\Lambda \to w$ and deletion $v \to \Lambda$. Here, we have considered the case that one vector could be replaced by N, or vice versa.

When using dominant points, it is usual that a small change in the contour generates a new dominant point, so when comparing two prototypes 1 vector in the first prototype can be similar to N continuous vectors from the second prototype.

The cost of sequence of edit operations that transforms a finite ordered vector set V into W, if we establish a cost function $\gamma_v(v^*,w^*)$, is defined as

$$d_v(V,W) = \min \begin{cases} \gamma_v(\Lambda \to W\langle 1\rangle) + d_v(V, W\langle 2:|W|\rangle) & |W| \geq 1 \\ \gamma_v(V\langle 1\rangle \to \Lambda) + d_v(V\langle 2:|V|\rangle, W) & |V| \geq 1 \\ \gamma_v(V\langle 1\rangle \to W\langle 1\rangle) + d_v(V\langle 2:|A|\rangle, W\langle 2:|B|\rangle) & |V| \geq 1 \wedge |W| \geq 1 \\ \underset{j\in[2,|W|]}{\gamma_v(V\langle 1\rangle \to W\langle 1:j\rangle) + d_v(V\langle 2:|V|\rangle, B\langle j+1:|W|\rangle)} & |W| > 2 \\ \underset{i\in[2,|V|]}{\gamma_v(V\langle 1:i\rangle \to W\langle 1\rangle) + d_v(V\langle j+1:|V|\rangle, B\langle 2:|W|\rangle)} & |V| > 2 \\ 0 & |V| = 0 \wedge |W| = 0 \end{cases}$$

In a similar way to the efficient (dynamic programming technique) algorithm proposed in [4] for computing the string edit distance, it can be extended to compute the ordered vector set edit distance in the following way:

```
1. Function vectorEditDistance($V$,$W$)
2.    $D[0,0] := 0$;
3.    for $i := 1$ to $|V|$ do $D[i,0] := D[i-1,0] + \gamma_v(V\langle i\rangle \to \Lambda)$;
4.    for $j := 1$ to $|W|$ do $D[0,j] := D[0,j-1] + \gamma_v(\Lambda \to W\langle j\rangle)$;
5.    for $i := 1$ to $|V|$ do
```

```
    for $j := 1$ to $|W|$ do
      $m_1 := D[i-1, j-1] + \gamma_v \left( V \langle i \rangle \rightarrow W \langle j \rangle \right)$;
      $m_2 := D[i-1, j] + \gamma_v \left( V \langle i \rangle \rightarrow \Lambda \right)$;
      $m_3 := D[i, j-1] + \gamma_v \left( \Lambda \rightarrow W \langle j \rangle \right)$;
      $m := \infty$;
      for $k := 1$ to $|V|$ do
        if $(i-k) \geq 0$ then
            $m := \min \{ m,\, D[i-k, j-1] + \gamma_v \left( V \langle i-k : i \rangle \rightarrow W \langle j \rangle \right) \}$;
      endfor
      for $k := 1$ to $|W|$ do
        if $(j-k) \geq 0$ then
            $m := \min \{ m,\, D[i-1, j-k] + \gamma_v \left( V \langle i \rangle \rightarrow W \langle j-k : j \rangle \right) \}$;
      endfor
      $D[i,j] := \min(m, m_1, m_2, m_3)$;
    endfor
  endfor
return $D[i,j]$
```

The complexity of the string edit distance algorithm is proportional to the length of both strings, $\mathcal{O}(|A|\,|B|)$. In the case of the *vectorEditDistance*, it has three nested loops and the complexity is $\mathcal{O}(|V|\,|W| \max\{|V|\,|W|\}\,\mathcal{O}(\gamma_v))$, but if we consider that a vector can be replaced by a fixed constant number of vectors and the function γ_v defined bellow, the complexity is reduced to $\mathcal{O}(|V|\,|W|)$. Thus, the cost is similar to that of the string edit distance.

To compute the difference between one vector and a set of N vectors, used in *vectorEditDistance*, the following function is utilised:

```
Function $\gamma_v \left( V \langle k \rangle \rightarrow W \langle i : j \rangle \right)$
  float $auxN := 0$, $aunAng := 0$, $r := 0$, $rSubs := 0$, $rLeft := 0$
  $auxN := V_N \langle k \rangle$   //Norm single vector
  $auxAng := V_\alpha \langle k \rangle$   //Angle single vector
  for $l := i$ to $j$ do
    if $auxN \geq 0$ then //Left norm single vector
      $rSubs := rSubs + auxN * \mathrm{closest}(auxAng, W_\alpha \langle l \rangle)$
      $auxAng := W_\alpha \langle l \rangle$
    endif
    $auxN := auxN - W_N \langle l \rangle$
  endfor
  if $auxN \geq 0$ then //Left norm single vector
    $rLeft := auxN * \mathrm{kInsertion}$
  else //Norms $W$ vectors $> V$
    $rLeft := -auxN * \mathrm{kDeletion}$
  endif
return $rSubs + rLeft$
```

where closest($angle1, angle2$) returns the smallest angle between both parameters, resulting a value in $[0, \pi]$. The kInsertion = kDeletion = $\pi/2$ is the maximum possible difference between two angles.

The functions $\gamma_v\left(V\langle i.j\rangle \rightarrow W\langle k\rangle\right)$ and $\gamma_v\left(V\langle i\rangle \rightarrow W\langle j\rangle\right)$ are similar. In the first case, the parameters change the order and in the second case, both parameters are unitary vectors.

The insertion and deletion functions are defined as $\gamma_v\left(\Lambda \rightarrow W\langle j\rangle\right) = |W\langle j\rangle| *$ kInsertion and $\gamma_v\left(V\langle i\rangle \rightarrow \Lambda\right) = |V\langle j\rangle| *$ kDeletion.

4 Experiments

Three algorithms have been compared based on different contour descriptions:

1. Classical Freeman chain code extracted from the object contour in the binary image. Any point reduction method is applied.
2. The ordered vector set extracted from the dominant points computed by the algorithm described in [7], that will be referred as non collinear dominant points (NCDP).
3. The new structure based in the ordered vector set extracted from dominant points described in [6]. In this article, $1-$curvature and $k-$curvature algorithms are defined in order to select dominant points using these measures. The authors showed that the obtained dominant points were similar for both curvature measures, so we utilised the faster one: $1-$curvature.

In the preliminary trials tested, the algorithm $1-$curvature obtained lower error rates than NCDP. Thus, the k parameter in the *vectorEditDistance* function was tuned when applied to $1-$curvature. The k parameter is the maximum number of continuous vectors that was set to $k = 1$.

A classification task using the NIST SPECIAL DATABASE 3 of the National Institute of Standards and Technology was performed using the different contour descriptions enumerated 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 nearest neighbour (NN) technique was used for perform classification.

Figure 3 shows the comparison between the error rate in the vector classification task evaluated for different sizes, k (*vectorEditDistance*). This experiment shows that the error rate decreases linearly when the k grows to a limit. If k grows the number of computations increases as well the classification time. In this case, we found the lowest error rate with the lowest k, so the optimal parameter value was $k = 3$.

The figure 4 shows the classification error rate and the time used in the classification of 50 examples per class as a function of different training set.

In all cases the use of Freeman chain codes generates a lower error rate (less than 9%) in recognition than using ordered vector sets, although the classification time is much higher. Thus, the ordered vector set description based on dominant

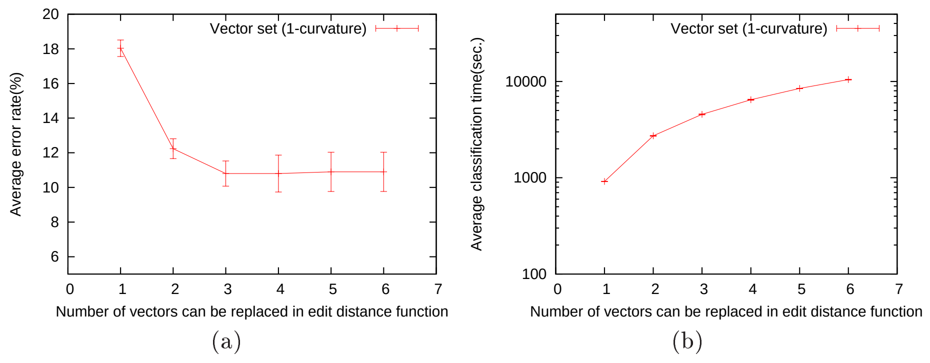


Fig. 3. Results for NN classification of characters obtained with ordered vector set ($1-$curvature), different training set (200 examples per class) and test set (50 samples per class and 26 character classes) as a function of different number of vectors that can be replaced in a substitution operation in the vector edit distance: (a) average error rate $\pm$ standard deviation; (b) average classification time.

points $1-$curvature [6] is a good trade-off choice. It obtains also a low error rate (less than 11%) and it is 10 times faster than using the Freeman chain codes.

5 Conclusions and future work

The computation of the edit distance between ordered vector sets that represent the contour of an object in a binary image (based on dominant point computation using 1-curvature) is one order of magnitude faster than using Freeman chain codes, and it has just a slightly higher error rate when using it for recognition. The edit distance defined in this paper to compare ordered vector sets has similar complexity than that of string edit distance. Since the size of the ordered vector set is significatively lower than that of strings for representing the same object, the time needed for computing the distance needed for classification is much lower.

As it can be seen in the results section the error rate using ordered vector set based on dominant points is similar to that of using the Freeman chain code.

As future work we planned to use some special labels for each vector to describe the curved shape of the original image in order to obtain a better description of the binary image contour and decrease the error rate in this classification task. Another possible line of future work is to apply algorithms such as [11] in order to optimise the cost functions for the ordered vector set edit distance.

References

1. Rico-Juan, J.R., Micó, L.: Comparison of AESA and LAESA search algorithms using string and tree edit distances. Pattern Recognition Letters **24(9)** (2003)

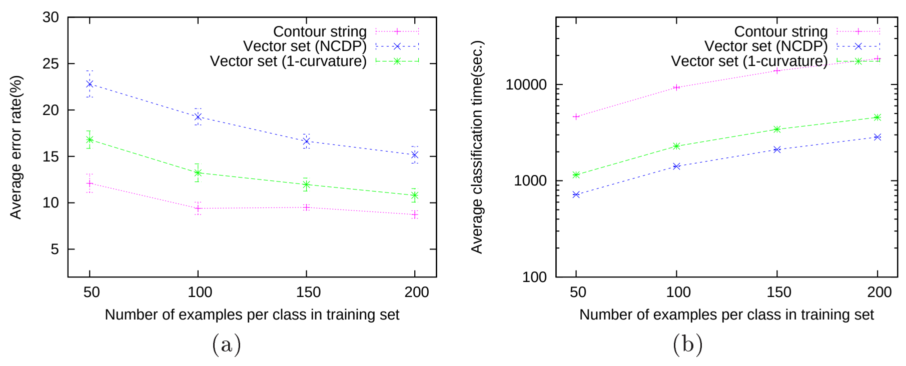


Fig. 4. Results for NN classification of characters obtained with different contour representations as a function of different training example sizes: (a) average error rate ± standard deviation; (b) average classification time.

1427–1436

2. Freeman, H.: On the encoding of arbitrary geometric configurations. IRE Transactions on Electronic Computer **10** (1961) 260–268
3. Rico-Juan, J.R., Micó, L.: Some results about the use of tree/string edit distances in a nearest neighbour classification task. In Goos, G., Hartmanis, J., van Leeuwen, J., eds.: Pattern Recognition and Image Analysis. Number 2652 in Lecture Notes in Computer Science, Puerto Andratx, Mallorca, Spain, Springer (2003) 821–828
4. Wagner, R.A., Fischer, M.J.: The string-to-string correction problem. J. ACM **21** (1974) 168–173
5. Zhang, K., Shasha, D.: Simple fast algorithms for the editing distance between trees and related problems. SIAM Journal of Computing **18** (1989) 1245–1262
6. Teh, C.H., Chin, R.T.: On the detection of dominant points on digital curves. IEEE Trans. Pattern Anal. Mach. Intell. **11** (1989) 859–872
7. Iñesta, J.M., Buendía, M., Sarti, M.A.: Reliable polygonal approximations of imaged read objects though dominant point detection. Pattern Recognition **31** (1998) 685–697
8. Rico-Juan, J.R., Calera-Rubio, J.: Evaluation of handwritten character recognizers using tree-edit-distance and fast nearest neighbour search. In Iñesta, J.M., Micó, L., eds.: Pattern Recognition in Information Systems, Alicante (Spain), ICEIS PRESS (2002) 326–335
9. Serra, J.: Image Analysis and mathematical morphology. Academic Press (1982)
10. Carrasco, R.C., Forcada, M.L.: A note on the Nagendraprasad-Wang-Gupta thinning algorithm. Pattern Recognition Letters **16** (1995) 539–541
11. Ristad, E., Yianilos, P.: Learning string-edit distance. IEEE Transactions on Pattern Analysis and Machine Intelligence **20** (1998) 522–532