

Introduction

Document Image Classification

Symbolic Model and Recognition of Document Images
schedules and form faces. Hence, 20 different form faces (classes) are represented in the database. The images have been automatically derived and synthesized and therefore contain no "real" tax data. Nevertheless, all the images appear to be real hand-printed forms prepared by individuals. There are 900 simulated tax submissions represented in the database averaging 6.22 form faces per submission. All the images are stored in a bit-level black and white raster format.

Algorithm 3.1: Algorithm to obtain a tree-like model of the document images

1. Blur the image and reduce it by one eighth. /* result: 256 grey-level image */
2. Detect the connected components (four connected pixels) of the image.
3. For each connected component of the image:
   - Reduce it to its smallest rectangle.
   - Reduce each rectangle to its central point and assign a value that is proportional to the area of the rectangle to this point.
4. Set $M$ to the maximum but two /* to smooth the value */ and $m$ to the minimum values in the image.
5. Let $f = (M + m)/2$.
6. Consider the following correspondence between intervals and symbols:
   - Interval $[0, f/2]$; symbol 'A'
   - Interval $[f/2, f]$; symbol 'B'
   - Interval $[f, 3f/2]$; symbol 'C'
   - Interval $[3f/2, 2f]$; symbol 'D'
   - Interval $[2f, 3f/2]$, etc.; symbol 'E'
7. Create the root node of the tree.
8. Set node to the root node of the tree.
9. Divide the image in four quadrants.
10. Create a child node to node for each quadrant (from left to right and downwards).
11. For each node (and its corresponding quadrant):
    - If the quadrant contains more than one point, then label the (internal) node with symbol $\sigma$ and go to step 9.
    - If the quadrant has no points, then label the node as a leaf using the symbol $\sigma$.
    - Otherwise, assign a symbol to the node (see step 6).
12. Return the tree obtained.

Quad trees (q-tree) [7] has been broadly used to obtain tree-like representations. These representations have shown to be useful in pattern recognition tasks [10][11]. In order to capture only the more relevant features of the images, the algorithm we propose takes into account a reduced image.

The first step in modelling the layout of the documents was to perform a light blur of the images (we considered 5 x 5 pixels) and a reduction of the image by one eighth. Then, the resulting 256 grey-level images were processed to detect the connected-components. In this process, only four connected pixels were considered. Each connected-component was then substituted by the minimum rectangle that contains the component.

The image of the document was then simplified to an image where all the rectangles were reduced to a pixel in the middle of the rectangle with a weight that is proportional to its area.

In a standard q-tree representation, the root of the tree is associated to the whole image to model and any other internal node with one of the four quadrants of its parent image. The process is repeated recursively until a node represents a one-colour square, then, the node becomes a leaf labeled with the color of the region. The variation of the q-tree representation used in this work ends the recursion when the quadrant does not contain any point. The region is then labeled with the empty leaf. When the region contains only one point, the leaf is marked with a symbol that is proportional to the weight of the point (five different labels were considered). A scheme of the entire process is shown in Algorithm 3.1. An example of the run is shown in Figure 1. Note that no preprocessing of the image (saliency correction, rescaled or similar) was performed.

3.2 Error-Correcting Analysis

The treatment of noisy or distorted samples has always been a problem in pattern recognition tasks, these distortions usually do appear as a side effect to the acquisition, preprocessing or primitive extraction phases.

This work makes use of an error-correcting analysis algorithm that was introduced in [14] and that gives a distance between a tree and a tree automaton.

To calculate the distance between a tree $t$ to a tree automaton $A$, the algorithm explores the tree, and calculates the cost of reducing each subtree of the tree to each state of the automaton. To do this, the algorithm compares the successors of a node with the different transitions of the automaton.

The method visits the tree nodes in postorder. Therefore, when a tree is going to be analyzed, all the distances between its subtrees and the states of the automaton have already been calculated. These distances are stored in a matrix that is indexed by the nodes of the tree and the states of the automaton, thereby avoiding calculations carried out previously (for details see [14]). The whole process is carried out in polynomial time with respect to the size of the tree and the number of states of the automaton. The authors prove that the distance obtained is the minimum one according to the edit operations proposed.

3.3 Tree Language Inference

In our work, we model each class with a tree language. In order to obtain these languages, it is possible to use several algorithms (for instance [5][6][8][13][19]).

In this work, we used a tree language inference algorithm that, in each step of the inference process, considers the automaton of the previous step and a new sample of the training set. Then, the inference algorithm performs an error-correcting analysis on the tree automaton. The modification of the automaton comes determined by the editing operations needed to force the automaton to accept the sample.
The interaction described above was used to represent each case. In order to infer these interactions, we need a representation of the database with attributes to store information.

**Experimentation**

The approach has been used previously in pattern recognition tasks with good results. The approach has been applied to the interaction processes in the model and has achieved good recognition rates. The model has been extended to include more attributes to store information. The approach has been extended to include more attributes to store information. The approach has been extended to include more attributes to store information.
Once the automata were inferred, we used the error-correcting parser on tree automata to obtain the minimum distance of the test sample for each automaton. The document images were classified into the class with a minimum value criterion. This approach has previously been used in other pattern recognition tasks with good performance (e.g., [12][11]).

Our approach needs some samples from each class in order to train the models. Therefore, the four classes with greatest number of available images were selected from the database. These classes were Form 1040 page 1, Form 1040 page 2, Form 1040 Schedules A and B. Example images are shown in Figure 2.

Two different experiments were performed. The models of the first experiment were inferred using 200 samples per class. In the second experiment, we used 300 samples per class. The results of the experiment are shown in Table 1. Note that due to the number of samples available, the number of samples in the test set is not homogeneous.

<table>
<thead>
<tr>
<th>class</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>299</td>
<td>100</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>399</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>17</td>
<td>268</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>277</td>
</tr>
</tbody>
</table>

Table 1. Results of the experimentation. The table on the left shows the confusion matrix when 200 training samples were used. The table on the right shows the confusion matrix when 300 samples were used to infer the automata.

The results show the excellent performance of our approach, an error rate of 1.84% was obtained when 200 samples were considered to infer the automata, and an error rate of 1.18% was obtained when 300 samples were used.

5 Conclusions

In this work, we tackle the classification of hand-filled forms using a syntactic approach. We propose a new procedure for modelling the layout of the forms which obtains a tree-like representation. This representation allows us to use a tree language inference algorithm to obtain a model for each class of the classification task. An error-correcting analysis is carried out to classify the test samples, taking into account the minimum editing-distance.

We used a subset of the NIST Special Database 6 of structured forms and our syntactic approach to carry out the classification. The results of the experiments show the excellent behaviour of the approach: an error rate of 1.18%.

As a future line of work, the preprocessing of the image (for instance, applying methods to correct the slant of the image), should improve the results. Note that the approach does not consider the use of probabilities. Another line of work is the modification of the model to obtain a stochastic one which should also lead to better results.

Finally, the representation process could easily be modified for other pattern recognition tasks and should be explored further. Also, other methods for modelling the physical structure of forms [18] could be used in a syntactic approach to classification to produce good results.

References

2. Structural Recognition of Symbols

We consider and give some perspectives on the problem of recognizing symbols from images. In section (1), we present a system that uses the recognition neural network. In section (2), we present a system that uses the recognition neural network. The two systems are compared in section (3), and the results are used to give some perspectives on the problem of recognizing symbols from images. In section (4), we present a system that uses the recognition neural network. The two systems are compared in section (5), and the results are used to give some perspectives on the problem of recognizing symbols from images. In section (6), we present a system that uses the recognition neural network. The two systems are compared in section (7), and the results are used to give some perspectives on the problem of recognizing symbols from images.
Structural, Syntactic, and Statistical Pattern Recognition

Joint IAPR International Workshops
SSPR 2004 and SPR 2004
Lisbon, Portugal, August 2004, Proceedings