Image preprocessing for optical character recognition using neural networks

Author: Miroslav NOHAJ, jookie@szm.sk
Supervisor: Rudolf JAKŠA, jaksa@neuron.tuke.sk

Abstract

Primary task of this master’s thesis is to create a theoretical and practical basis of preprocessing of printed text for optical character recognition using forward-feed neural networks. Demonstration application was created and its parameters were set according to results of realized experiments.

Project definition and task determination

1. Write an introduction about the problematics of optical character recognition of characters and the methods of image preprocessing before optical character recognition.


3. Implement the designed system and simulator of printed text.

4. Realize experiments to determine the settings of the system and to compare the different approaches.

5. Evaluate the realized experiments and their possible practical use.

6. Write a documentation according to the supervisor’s instructions.

Introduction

Almost everyone who is working with computers has to input some text to the computer from the paper. There is not only one way to do that. The smartest
way is to scan the document and let software for optical character recognition (shortened: OCR) transform the scanned image into editable text. The OCR software can use methods like:

- matrix comparation of image with letter examples from library
- feature extraction from image
- recognition of characters using neural networks
- hybrid and combined methods
- other methods

Each method listed above has some advantages and disadvantages, so if you are using OCR software which uses any of those methods, you know what you can expect. Flexibility of the methods listed above varies from one to another, but even the less flexible method’s success can be improved using image preprocessing before the OCR.

The most used methods of image preprocessing before OCR are:

- thresholding based on histogram
- smoothing
- other 2D matrix filters

I won’t describe any of these methods of preprocessing here because they’re pretty much known to the public.
Obr. 2: Image after smoothing (mid) and thresholding (right).

On the picture -1 on the left is what you will get if you will scan the printed text. Note that the scanned image contains levels of gray that were never printed and that were never designed on the screen of the computer. Image like this is not thresholdable - the threshold is useless and the OCR can’t be used.

If you would apply a light smoothing couple times on the image on the picture 1 or a heavy smoothing once, you can get a result like this one: As you can see, the smoothing (in some sort) converted the different levels of gray into less levels of more similiar levels of gray. On the image on the picture 2 in the middle a threshold can be applied with better results than on the image on the left. But as you can see, with threshold you will loose some parts of the text (picture 2 on the right) or you will fail to remove some parts of the background (but the text will not loose anything). So this is the case when the different method of preprocessing must be used.

**Design and implementation**

Most of the standart and well known methods works by determining the new value of the level of shade of the image pixel by multiplying the current level of shade of the pixel and its neighbours by some coefficients. These coefficients are described in a convolution matrix. This matrix can be displayed as a set of input pixels connected with the output pixel with lines with wieghts. (See the picture 3) In artificial inteligence, the most similiar thing to this filter is a forward-feed neural network, because:

- the input can be organized in the mask of $N \times N$
Obr. 3: Schematical drawing of a 2D image filter.

Obr. 4: Schematical drawing of neural network

- the output can be one (or more) pixels
- the signal flows from the input to the output
- the input and output are connected with lines which have weights

So a very similar forward-feed neural network to the 2D filter can look like one shown on the picture 4. I’ve chosen the parameters of used neural network:

- type of neural network: forward-feed full-connection neural network
- learning method: standard error back-propagation
- input layer of $N \times N$ neurons
- hidden layer of $M$ neurons
- output layer of 2 neurons (one for direct level of shade and one for inverted level of shade)
- activation function: \( f(x_i) = \frac{1}{1+e^{-\lambda x_i}} \)

Note that parameters \( M \) and \( N \) were determined by experiments.

The 'Image PreProcessing' application was designed for the purpose of image preprocessing using neural networks, the workflow chart is shown on the picture 5. The application can be done in three different ways:

- fully automatic application
- half-automatic application
- manually controlled application

On the picture 5 are two boxes which are affected by these three different ways. **Determine background and foreground combination** is done automatically in the concept of fully automatic and half-automatic applications and it would require some logic to compare the current foreground and background to known combinations of foreground and background (this could be done by modular neural network). **Create training and testing pattern** would need computer vision and OCR itself to determine the text on the original picture, that’s why it is automatic only in the concept of fully automatic application. Because the main target of this masters thesis was to do research on image preprocessing using neural networks and not to make the best preprocessing application, I decided to use the concept of manually controlled application for my 'Image PreProcessor'. This way user has to input the text he sees on the picture and place the text exactly over the text on the picture and then the application can create the training and testing pattern. You can see a screenshot from the 'Image PreProcessor' on the picture 6.

**Experiments**

Some experiments must have been realised to determine some parameters of the Image PreProcessing system and its neural network. On more scanned documents I have found 25 different combinations of background and
Obr. 5: Workflow chart of Image PreProcessing application.
foreground. I created a 'Printed text simulator' which generates images with text which is texturised with the textures found in the scanned documents. Examples of the generated pictures are on picture 7.

The first experiment was intended to determine which combination of foreground and background on the generated image is the most problematic. As the graph on picture 8 shows that combination number 17 is the most problematic, so it was chosen for the next set of experiments on generated images. The next experiments showed that the neural network must have the input layer of size at least $3 \times 3$ neurons to work sufficiently on the generated image and the hidden layer of size at least 3 neurons.

I was also testing how the initialization of neural network affects the learning of neural network and also the final achieved minimal error of the neural network. I’ve found out that the average minimal error on the 17th combination of foreground and background was 7.82% and that it may vary about 1% because of different initialization.

The other experiment showed that the neural network can process well the image with 15% or less of noise.

I have done also experiments on the scanned image with the scanning
Obr. 7: Example of some foreground and background combinations

Obr. 8: Graph of relation between combination of textures and the minimal error reached on the neural network.
resolution of 200 dpi\textsuperscript{1}. The experiments showed that the best size of input layer of neural network is $16 \times 16$ neurons with this scanned image in this scanning resolution. Also the size of hidden layer was determined and the experiment showed that the hidden layer should have at least 10 neurons.

There were done some experiments with the training pattern generating. If the image preprocessing has to be good enough, the training pattern should have at least 8000 samples and their distribution should be controled.

The best recognition success that was achieved after image preprocessing was 87.9\% of letters correctly recognized. The software used for OCR was ABBYY FineReader 7.0 (trial version).

**Contribution to the research domain**

This master’s thesis does a research of image preprocessing before optical character recognition using neural networks. It is a benefit to the research domain because the image preprocessing using neural networks isn’t adequately examined and it has a lot of potential. I think that the nonlinear adaptive filtering which can be done using neural networks is much more better than linear filtering and it is the future of the image processing and preprocessing. You can find some theoretical and practical basics in this master’s thesis about this topic.

**Conclusion**

As this master’s thesis shows, the neural networks can be used for image preprocessing pretty well with many advantages over normal image preprocessing methods. For more information about mentioned experiments and application please read the full master’s thesis (in slovak language).

\textsuperscript{1}dots per inch
References


[9] Optical character recognition

[10] Mustek, Inc.: Understanding OCR
http://www2.mustek.com/Class/ocrinfo.html

http://www.ccs.neu.edu/home/feneric/charrec.html, 1992
[12] Abdel Belaid: OCR: Print, INRIA, Nancy, Lorraine, France
http://cslu.cse.ogi.edu/HLTsurvey/ch2node5.html CRIN/CNRS

http://www.codeproject.com/script/ann/
ServeHTML.aspx?C=False&id=786&cb=4088493