Increasing Handwriting Recognition Accuracy Using Morphology

Joonas Lõmps

Abstract—Optical character recognition (OCR) has become more and more common technology, be it automated teller machines (ATM), office scanners or the scanners used at stores. In addition to that the automated recognition of handwritten characters is commonly studied problem in Computer Vision and has several applications in real life. This paper compares the different morphological operations impact on the recognition rates with different features and two commonly used classifiers; Support Vector Machines (SVM) and k-Nearest Neighbor (KNN).

Keywords—Optical Character Recognition, K-Nearest Neighbor, Support Vector Machine, Morphological Processing.

I. Introduction

CHARACTER recognition or handwriting recognition to humans is something that is learned over the years perfected the same way. We are able to understand even poorly written texts. A lot of work has been done for identifying written characters by machines. There are multiple applications that already use character recognition: processing administrative forms, licence plate recognition, voting by post and etc. Moreover the importance of hand written recognition systems is illustrated by the increasing need for it in many fields of applications such as multimedia, electronic libraries, any system using hand written inputs and most importantly digitalizing large number of written documents.

The most common steps for handwriting recognition usually include: preprocessing, feature extraction, recognition and postprocessing. The recognition is usually done by classifying algorithms such as Support Vector Machine [1], K-Nearest Neighbor [2], Hidden Markov Model [3] etc.

My work is conducted on Japanese Hiragana syllabary handwriting and compares the impact of different morphological operations on the accuracy rates of different features and classifiers.

II. Mehtodology

A. Preprocessing

Preprocessing in this paper consists of three main part: normalization, grayscaling and binarization. Reason for it all, is to get the best possible result out of the morphological processing. Normalization includes croping the image as much as possible so that only the character remains on the image and then resizing it to 50x50 pixel. The normalized image of a character is then turned into grayscale, leaving only the pixel intensity information displayed with different shades of gray. Then, it is followed by binarization with a global threshold. End result of the preprocessing is an image with values of 0's for black and background pixels or 255's for white and foreground pixels.

J. Lõmps is with the Institute of Computer Science, University of Tartu, Tartu, 51014 Estonia, e-mail: b12027@ut.ee

B. Morphology process

Mathematical morphology provides a set of different operators for processing digital images based on their shape. The operators are especially useful for the analysis of binary images. When correctly used, the mathematical morphology operators can be used for edge detection, noise removal, image enhancement, image segmentation and etc, all while preserving their shape and eliminating irrelevancies. Mathematical morphology uses sets to represent shapes and the operators use methods defined in the set theory do transform the input image [4].

Our work focuses on the two most basic operations in mathematical morphology: erosion and dilation. Both operations take two inputs. One is the input image, that is to be modified, and the other is the structuring element. The structuring element determines the details how the operator modifies the input image.

Erosion

Erosion is a morphological transformation that combines two sets using the vector subtraction of set elements (Fig. 1). If A and B are sets in Euclidean space E and A is a binary image, then the erosion of the binary image A by the structuring element B is defined by:

$$A \ominus B = \{x \in E \mid x + b \in A, \forall b \in B\} \tag{1}$$

Example of erosion operation:

$$A = \{(0,0),(0,1),(1,0),(1,1),(1,2),(1,3),(2,1),(2,2),$$

$$(2,3),(2,4),(3,2),(3,3),(4,5)\}$$

$$B = \{(0,-1),(-1,0),(0,0),(0,1),(1,0)\}$$

$$A \ominus B = \{(1,1),(2,2),(3,2)\}$$

This is the definition used for erosion operation by [4]. For simplicity the structuring element B can be slid across the binary image A and where B is contained in$A$, its origin point $(0,0)$ is present in $A \ominus B$.

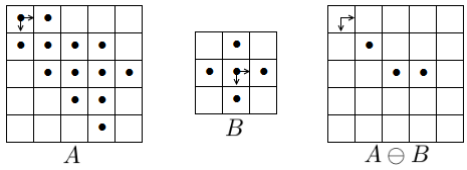

Fig. 1. Illustration of erosion process

Dilate

Dilate is a pseudo-inverse of the erosion. Instead of combining two sets using vector subtraction of set elements it uses their addition (Fig. 2). If A and B are sets in Euclidean space and A is a binary image, then the dilation of the binary image A by the structuring element B is defined by:

$$A \oplus B = \{c \in E \mid c = a + b, \exists a \in A, \exists b \in B\} \quad (2)$$

Example of dilate operation:

$$A = \{(1,1),(1,2),(2,2),(2,3),(3,3)\}$$
$$B = \{(1,-1),(0,0),(0,1)\}$$
$$A \oplus B = \{(2,0),(1,1),(2,1),(3,1),(1,2),(2,2),(3,2),$$
$$(4,2),(1,3),(2,3),(3,4),(2,4),(3,4)\}$$

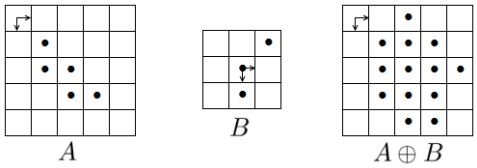

Fig. 2. Illustration of dilate process

This is the definition used for dilate operation by [4]. For simplicity the structuring element B may be slid across the binary image A. Wherever the origin point of B matches a point on A, the structuring element B can be stamped onto image A, creating $A \oplus B$.

C. Feature Extraction

Image representation plays a crucial role in a recognition system. Simple binary image can be fed to a recognizer, but it might not achieve the wanted results. A more compact and characteristic representation of an image is needed for most of the recognition systems to avoid unnecessary complexity and to increase the recognition accuracy rates [5]. Extracting a set of features for each class helps distinguish it from other classes while remaining invariant to characteristic differences within the class [6]. In the following we describe the extracted features used in our approach.

Raw data

The simplest feature, the output of preprocessing without any feature extraction. With our approach it either gives a us a eroded or dilated binary image of a character. Due to using 50x50 images of character it gives us a vector of 2500 data points per character.

Projection Histogram

Projection histogram (PH) is a way of representing a two-dimensional (2-D) image signal into 1-D signal. In short it counts the number of foreground pixels in each column and row in an image and turns the values into a vector with a size of $n + m$ where n and m represent the dimensions of the image [7]. Example of it can be seen on figure 3.

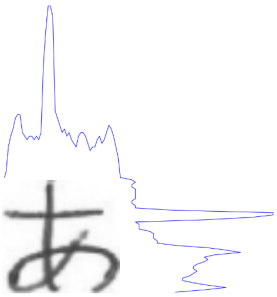

Fig. 3. Example of Projection Histogram

Zoning

Zoning is an feature that divides the input image into a number of predefined sizes and adding up the intensities of foreground pixels in the given zone [8]. We tested two different variation of number of zones and their sizes: 25 zones of 10x10 (zones25) and 100 zones of 5x5 (zones100).
To find the zoning feature vector of an image, all pixel intensities in a given zone need to be added. The values are then transformed into a vector with size of l, where l is the number of zones.
If $f(i,j)$ is a digital image and $I(i,j)$ is the pixel intensity of that image, then the pixel intensities $I(i,j)$ of image in a zone k are calculated by

$$V_k = \sum_{i=1}^{n} \sum_{j=1}^{m} I_k(i,j) \quad (3)$$

where $1 \leq k \leq l$, where l is the number of zones and n and m are the dimensions of the zone.
Equation (3) describes the summation of intensity values of k^{th} zone in the image. The final vector is a list of intensities of every zone.

$$V = [V_k], 1 \leq k \leq l \quad (4)$$

In our approach, it simplifies to counting the foreground pixels due to the use of binary images as input for extracting the features with result that can be seen on figure 4.

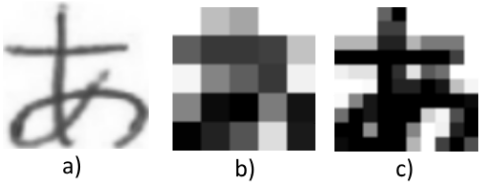

Fig. 4. Example of result of zoning. a) base image, b) zoning $k = 25$, c) zoning $k = 100$

Histogram of Oriented Gradients

Objects appearance and shape can be characterised by the distribution of local intensity gradients or edge directions. It starts by dividing the image into smaller regions called cells, for every cell a local 1-D histogram of gradient direction over the pixels of the cell is compiled. Concatenation of these histograms makes up the descriptor. For improved accuracy, the cell values can be normalized by combining cells into a larger region of the image, called a block, and using its intensity for normalization [9].

D. Classfication

K-Nearest Neighbour

The k-nearest neighbours rule [10] is one of the simplest and well known methods for classification. It classifies examples based on the majority of the k-nearest neighbours found in the training set.
Given a set of n pairs (x_1, θ_2), ..., (x_n, θ_n), where x_i's take values in a metric space X defined a metric d, and θ's take values in the set of possible class indexes $\{1, 2, ..., M\}$.
For any new pair (x, θ), only the value of x is known, and it is wanted to estimate its θ by using the information provided in the set of trained classified points. Then $x_n^{'} \in \{x_1, x_2, ..., x_n\}$ is the nearest neighbour to x if

$$min\ d(x_i, x) = d(x_n^{'}, x) \quad i = 1, 2, ..., n \tag{5}$$

The nearest neighbour rule decides x belongs to the category $\theta_n^{'}$ of its nearest neighbour $x_n^{'}$. In our approach $k = 3$.

Support Vector Machine

Support vector machine is a discriminative classifier defined by a separating hyperplane. Given labeled training data, the algorithm outputs an optimal hyperplane which categorizes new examples. The optimal separating hyperplane maximizes the margin (M) of the training data. Let $\mid \beta_0 + \beta^T x \mid = 1$ represent a hyperplane, where x denotes the training example closes to the hyperplane(also called support vectors). Then the distance between a point x and a hyperplane (β, β_0) is

$$distance = \frac{\mid \beta_0 + \beta^T x \mid}{\parallel \beta \parallel} \tag{6}$$

As the numerator is equal to one and the the distance to the support vector is

$$distance_{support\ vectors} = \frac{\mid \beta_0 + \beta^T x \mid}{\parallel \beta \parallel} = \frac{1}{\parallel \beta \parallel} \tag{7}$$

The margin is then equal to twice the distance of the closest example $M = \frac{2}{||\beta||}$. Maximizing M is equivalent to the problem of minimizing a function $L(\beta)$ subject to some constraints. The constraints model the requirement for the hyperplane to classify all the training example x_i. Formally

$$\min_{\beta, \beta_0} L(\beta) = \frac{1}{2} \parallel \beta \parallel^2 \ \ subject\ to.\ y_i(\beta^T x_i + \beta_0) \geq 1\ \forall i, \tag{8}$$

where y_i represents each of the labels of the training examples. This problem can be solved using Lagrange multipliers to obtain the weight vector β and the bias β_0 of the optimal hyperplane.

III. Results

The results are based on a dataset that the author collected over the summer with the help of Department of Languages of Asian Region. It consist of 12283 images of Hiragana characters. Each character has around 173 samples in 28 different handwriting. The data was collected using a printed template where each of the students wrote 4 to 8 examples of each character. There are 71 different characters in Hiragana syllabary. A big table of results per character can be found in appendix A, because the table is simply too large to fit.
In the authors experiments with the dataset it was found that the most commonly used technique of making the character 1 pixel wide gives the lowest rate of accuracy overall with any of the above mentioned features and classifiers. More precisely, the decrease on average with SVM and any of the mentioned feature was -3.06%, with HOG having the biggest decrease -4.55% and Zone100 with the lowest -1.21%. For KNN the average decrease was -8.63% with the highest being Raw data at -21.9% and lowest Zone25 at -3.61%.
In contrast, dilation of the binarized image provided higher recognition rates than non dilated. For SVM classifier the changes were not that significant and for some features such as Histogram Projection and HOG they decreased the accuracy rate by 1%, but for other features the average increase was around 1.5%. For KNN the increase in recognition accuracy was higher on average, 5.0%. The feature that profited the most from dilation was Raw data with a rise of 11.24%.
The combination of morphology, feature and classifiers that were most accurate were:

1) Dilate - Raw data - SVM with 90.36%
2) Dilate - Zones100 - KNN with 90.22%
3) Dilate - Raw data - KNN with 90.08%
4) Dilate - PH + Zones25 - KNN with 89.98%
5) Dilate - Zones100 - SVM with 89.13%

We can see that all of the top accuracy rates belong to dilated morphology, which leads the author to believe, that it is the way to go.

IV. Conclusion

The paper introduces the reader to the world of computer vision with the emphasis being on the morphology. It introduces image preprocessing, explains the two most commonly used mathematical morphology operations in detail with examples, demonstrates different image representations with the features and then explains classifiers such as KNN and SVM. Based on the results it can be seen that it might be better to dilate images before extracting features as in most of the cases it increases the accuracy rates of classifiers.

Acknowledgment

The author would like to thank Ms Eri Miyano and her students from the Department of Languages of Asian Region for helping in providing the Hiragana dataset used in this research work.

APPENDIX

A. Appendix A

The results table is too large to be added to the document, so it can be accessed by the link:
https://docs.google.com/spreadsheets/d/19aWlHInp02Hubl5ceLvEuxGG2hdI0qYt0mMvyROWB2k/edit?usp=sharing

REFERENCES

[1] D. Singh; M. Aamir Khan; A. Bansal; N. Bansal, "An application of SVM in character recognition with chain code", IEEE Conference on Communication, Control and Intelligent Systems (CCIS) , 2015.

[2] K. B. Baiju, K. Sabeerath, "Online recognition of Malayalam handwritten scripts — A comparison using KNN, MLP and SVM",IEEE International Conference on Advances in Computing, Communication and Informatics (ICACCI), 2016.

[3] H. Choudhury; S. Mandal; S. Devnath; S. R. Mahadeva Prasanna; S. Sundaram, "Combining HMM and SVM based stroke classifiers for online Assamese handwritten character recognition" Annual IEEE India Conference (INDICON), 2015.

[4] Haralick, Robert M., Stanley R. Sternberg, and Xinhua Zhuang. "Image analysis using mathematical morphology." IEEE transactions on pattern analysis and machine intelligence 4, 1987.

[5] N. Arica, F.T. Yarman-Vural, "An overview of character recognition focused on off-line handwriting", IEEE Transactions on Systems, Man, and Cybernetics, 2001.

[6] I. S. Oh, J. S. Lee, and C. Y. Suen, "Analysis of class separation and combination of class-dependent features for handwriting recognition," IEEE Transactions on Pattern Analysis and Machine Intelligence, 1999.

[7] K. Wang, Y. Y. Tang, and C. Y. Suen, "Multi-layer projections for the classification of similar Chinese characters," in Proc. 9th Int. Conf. Pattern Recognition, pp.842-844, 1988.

[8] P. N. Sastry; T.R. V. Lakshmi; N.V. K. Rao; T.V. Rajinikanth; A. Wahab, "Telugu Handwritten Character Recognition using Zoning Features", International Conference on IT Convergence and Security (ICITCS), 2014.

[9] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In IEEE Computer Society Conference on Computer Vision and Pattern Recognition, volume 1, pages 886–893, 2005.

[10] T. Cover and P. Hart. Nearest neighbor pattern classification. In IEEE Transactions in Information Theory, IT-13, pages 21–27, 1967.