2020 International Conference on Computer Science and Software Engineering (CSASE2020)
University of Duhok, Duhok, Kurdistan Region – Iraq

Font Recognition of English Letters Using Normalized Central Moments Features

Aveen Jalal Mohammed
Computer Science Department
University of Duhok
Duhok, Iraq
avin.kovli@gmail.com

Hasan S. M. Al-Khaffaf
Computer Science Department
University of Duhok
Duhok, Iraq
hasan.salim@uod.ac

***Abstract*— Optical font recognition is an important process applied before or after optical character recognition. This paper presents a system for recognizing English fonts of character images. Feature selection plays a major role in recognizing the font; hence, we used normalized central moments (NCM) as the feature of choice in this study. What differentiates this study from others is the attempt to use another popular feature (distance profile features) used by other researchers and compare the results of the two. The support vector machine (SVM) method is used in training and testing. A system is developed that extracts the two features and trains two SVM models. Simulation results based on a dataset of 27,620 images belonging to three English fonts show that the proposed system can achieve an overall 94.9% correct recognition rate based on normalized central moments, while the system can achieve an overall 94.82% correct recognition rate when using distance profile features.**

Keywords*— *English font recognition, normalized central moments, distance profile, support vector machines.

I. Introduction

Font recognition is an important step in document image analysis that can either be applied as a preprocessing step prior to optical character recognition (OCR) or as a postprocessing operation to reveal the font used during the creation of the document to create a portable document file (PDF) that is faithful to the original paper document. The Roman script has many fonts each with unique width, height, weight, size, slope, and design [1]. Fig. 1 shows a sample of many fonts currently used.

What differentiates this paper from others is the attempt to use two feature extraction methods and perform a comparison of the recognition rate of the two features based on the support vector machine classification method.

The proposed system has three steps: preprocessing, feature extraction and classification. In the first step, the image is resized, and converted into binary and noise removed. In the second step, two kinds of features are extracted: distance profile features (DP) and normalized central moment features (NCM). The feature values are then normalized. The final step is classification where SVM is applied. Fig. 2 shows a diagram of the three phases of the proposed system: training, validation, and testing. In the feature extraction step, either DP or NCM features are used based on the experiment.

II. Related Works

Ramanathan et al. [2] presented a method for font recognition based support vector machines (SVMs) and Gabor filter features. The method showed 93.54% accuracy in the English language.
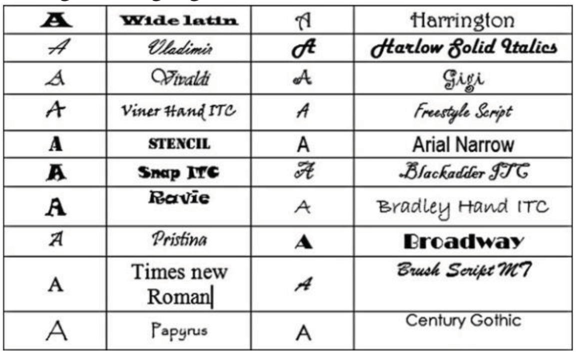
Fig. 1. Different English Language Fonts. Adapted from Doermann and Tombre [1].

Another study was performed by Jaiem et al. on 10 Arabic fonts [3]. The authors used a steerable pyramid in feature extraction and for recognition the authors used k-nearest-neighbors and a back-propagation artificial neural network (BPANN). Overall, BPANN shows better recognition accuracy than k-nearest neighbors.

The system developed by Bharath and Rani [4] has three steps; in the first step, the system reads a character image and preprocesses it. In the second step, distance profile features are computed. In total, 74 features per character image are generated. In the last step, a support vector machine (SVM) and *k*-nearest neighbors (KNN) are used for classification. Five different font styles are tested, and the system achieves approximately 80% average accuracy for SVM and 75% for KNN.

In 2017, Jaiem et al. presented a font recognition system based on steerable pyramids called (AFR/SP) Arabic font recognition steerable pyramids [5]. The system uses three levels of text entity analysis: word, line and text block. In feature extraction, steerable pyramids and two statistical variables (standard deviation and mean) are used. This study used two databases, APTID/MF and APTI with different resolutions. In the classification phase, the BPANN is used. The experimental results for high-resolution text block samples show high recognition rates of approximately 99% and 93% for low-resolution; for text line and word levels in

978-1-7281-5249-3/20/$31.00 ©2020 IEEE

low-resolution, the recognition rates are 90.67% and 78.98% respectively.

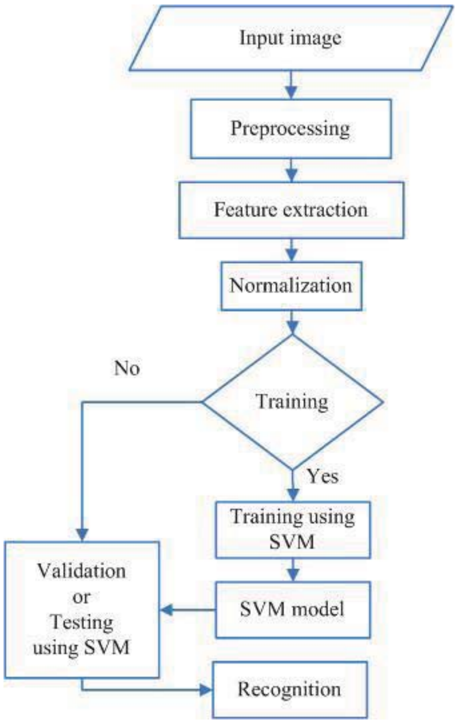

Fig. 2. Diagram of the Proposed System.

Tensmeyer et al. presented a system that classifies font at two levels: text lines and text pages by using the convolutional neural networks (CNNs) framework [6]. At the line level, the King Fahd University Arabic Font Database (KAFD) was used to recognize 40 Arabic fonts, while at the page level, two datasets were used: the KAFD and Latin Medieval Manuscripts (CLaMM) databases, which classify 12 English scripts. The author split the dataset into three sets, namely: training, validation and testing sets; which helps in choosing the most accurate model. For comparison, the authors used two CNN architectures: the AlexNet architecture, which has five convolution layers, and the state-of-the-art ResNet-50. In the training, two models for each the ResNet and AlexNet were created from the KAFD database on the line level and page level, and in the CLaMM database, one model was created for each of architecture. In the validation, the ResNet architecture had better performance for both datasets. In the testing, the recognition rate at the line level was better than that at the page level of 98.8%, while the page level obtained 86.6%.

In Senobari and Khosravi [7], for recognizing ten Farsi fonts, the authors combined two feature extraction algorithms: Sobel and Roberts features (SRF) and wavelet transform. To recognize a font of text block images, the gradient features were extracted using Sobel filter. A feature vector for Roberts's operator was also extracted. The results for recognizing font using the MLP classifier for each of the Sobel and Roberts classifiers were 89.41% and 88.12%, respectively. Then, these two feature vectors were merged to create the SRF and the recognition accuracy was much better with the best results among the other filters. Therefore, SRF is combined with the symlet wavelet transform which is called SRW, and the recognition rate was 95.56%. To reduce the number of features, the principal component analysis (PCA) algorithm was used. In their work, the authors implemented an 8-channel Gabor filter to compare the results of SRW, which was approximately 93.14%. Different wavelet filters were used such as (Haar, Db1, and symlet). By using the MLP classifier, their results were compared. The symlet filter obtained the Gabor algorithm. The accuracy for the Gabor method was 86.36% using the same classifier.

Recently, Al-Khaffaf and Musa presented a system for font recognition [8]. The system was based on eigenfaces and was designed to recognize English fonts. Simulation results on three fonts show a recognition rate of 97% based on 6,144 sample images.

III. Feature Extraction

A. Distance Profile Features (DP)

The DP feature is a histogram feature that contains left distance, right distance, main diagonal and secondary diagonal profile features. The left distance profile feature is computed for each row, by counting the number of background pixels until the first foreground pixel from left to right [5]. The right distance is computed as the left distance but from right to left. For the main diagonal two features are extracted: the top left corner and bottom left corner, while for the secondary diagonal, the top right corner and bottom right corner are counted. Since the image size is 51×51, we have a total of 106 features: 51 right features, 51 left features, and 4 diagonal features.

B. Normalized Central Moments (NCM)

Normalized central moments are the base of invariant moments or geometrical moments or seven moments [9]. Invariant moments were presented by Hu [10]. They help recognize shapes at different rotations, scales, and positions [1].

In the case of image processing, the image is represented in a two-dimensional discrete function where f(x, y) is the pixel intensity, so we use (m_{pq}) to represent the image's moment. The summation of p and q is the order of the moment, and they are both positive integer numbers [10, 11]. To reach these seven moments, you need to calculate from order zero to two. Each of these orders has a meaning, where (m_{00}) of order zero acts as an area of the object when the image is binary; otherwise, it is a mass of the image [12]. The center of gravity of the image can be expressed by (m_{10} and m_{01}), which are first order [13], and (m_{10}/m_{00}, m_{01}/m_{00}) is the centroid of the image, which can also be represented by $\bar{x}$ and $\bar{y}$, respectively [9], as in "(2)" . For the distribution of mass with respect to the coordinate axes of the image, second-order moments, (m_{20}, m_{11}, and m_{02}) are used [1] and are also called moments of inertia. "(1)" is used for computing these moments.

$$m_{pq} = \sum_x \sum_y x^p y^q f(x,y) \quad (1)$$

$$\bar{x} = \frac{m_{10}}{m_{00}} \quad \text{and} \quad \bar{y} = \frac{m_{01}}{m_{00}} \quad (2)$$

As mentioned before, the centroid of the image can be found by using moments of first order and zero order. This will help to have the central moments that are invariant when the position of the object is changed [10, 14].

Equation 3 is used to calculate central moments of order zero until order three:

$$\mu_{pq} = \sum_p \sum_q (x-\bar{x})^p (y-\bar{y})^q f(x,y) \quad (3)$$

Then we compute γ by the following equation:

$$\gamma = 1 + \left(\frac{p+q}{2}\right) \quad (4)$$

After finding γ, the NCMs are calculated by dividing the central moments by the zero order of the central moment, which is powered to γ, by the equations below:

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}{}^{\gamma}} \quad (5)$$

$$\mu_{00} = m_{00} \quad (6)$$

A feature vector of size 10, which consists of (η_{00}, η_{01}, η_{02}, η_{03}, η_{10}, η_{11}, η_{12}, η_{20}, η_{21}, η_{30}), is obtained.

In this paper, NCM was used instead of invariant moments because after testing the proposed system on these two features, the results of normalized central moments were much better than invariant moments.

IV. Support Vector Machines (SVM)

An SVM is a supervised machine learning algorithm that can be used in classification and regression. The SVM was originally a binary (2-class) classification method developed in 1995 by Cortes and Vapnik [15]. An SVM can classify both linear and nonlinear data based on finding the best hyper-plane that has the maximum margin between support points.

Support vector machines classify linearly separable data by finding the best d-dimensional hyper plane that separates all the training data of one class from those of other classes. Let us suppose we have this training set:

$$T = \{(\bar{x}_1, y_1), \cdots, (\bar{x}_m, y_m)\},$$

Each input data point has d features ($\bar{x}_i \in R^d$) with a class label that has one of two values ($y_i \in \{-1, 1\}$) [16]. The support vector machine can be written as:

$$\bar{w}.\bar{x}_i + b \geq +1 \text{ when } y_i = +1 \quad (7)$$

$$\bar{w}.\bar{x}_i + b \leq -1 \text{ when } y_i = -1 \quad (8)$$

where the vector $\bar{w}$ is perpendicular to the hyper plane $\bar{w}.\bar{x} + b = 0$, and b is a constant [17]. We are able to combine, " (7) " and "(8)" into one equation as follows:

$$y_i(\bar{w}.\bar{x}_i + b) \geq 1 \ \forall i \quad (9)$$

Now, the best hyper-plane that has a maximum margin between the two classes must be found among all available hyper-planes. For finding the margin, the distance of a data point on the hyper plane in H1 to H2 is calculated by subtracting the data point on H1 from H2, as in Fig. 3. To make the data point perpendicular to the decision boundary it will be multiplied by $\bar{w}/\|\bar{w}\|$ [16]. In addition, $\frac{2}{\|\bar{w}\|}$ will be obtained, and for mathematical convenience, it will be converted to $1/2\,\|\bar{w}\|$. Therefore, $\|w\|$ needs to be minimized. The Lagrangian method is used to solve the problem, where α_i are the positive Lagrange multipliers [18]:

$$L(w,b,\alpha) = \frac{1}{2}\,||\bar{w}||^2 - \sum_{i=1}^{m} \alpha_i [\, y_i(\bar{w}.\bar{x}_i + b) - 1\,] \quad (10)$$

From $\frac{\partial L}{\partial \bar{w}} = 0$ and $\frac{\partial L}{\partial b} = 0$ we have:

$$\bar{w} = \sum_{i=1}^{m} \alpha_i y_i \bar{x}_i \quad (11)$$

$$\sum_{i=1}^{m} \alpha_i y_i = 0 \quad (12)$$

Substituting "(11)" and "(12)" into "(10)", the dual problem of "(10)" is obtained:

$$L(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \bar{x}_i . \bar{x}_j \quad (13)$$

The training points that have $\boldsymbol{\alpha}$ greater than zero are called support vectors. This formulation of the SVM is called the hard margin SVM. The following equation is used to predict the class of new entered data:

$$y(\bar{x}_i) = \text{sign}\left(\sum_{j=1}^{S} \alpha_j y_j (\bar{x}_i . \bar{x}_j) + b\right) \quad (14)$$

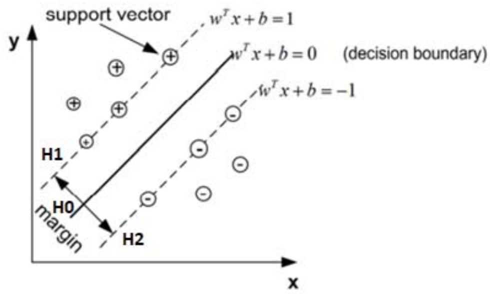

Fig. 3. SVM Classification. Taken from Katiyar et al. [19].

When data are nonlinearly separable, there are two ways of solving this issue: a soft margin or feature transformations (kernels).

In soft margin, the aim is not to have 100% correct classifications, but to make as few mistakes as possible. To handle such a problem; the constraints of the optimization problem are modified by adding positive slack variables (ζ):

$$y_i(\boldsymbol{\bar{w}.\bar{x}_i} + b) \geq 1 - \zeta_i \quad (15)$$

To avoid selecting a very large value of ζ for every training point, $\sum_{i=1}^{m} \zeta_i$ will be added into the objective function [20]:

$$\min_{w,b,\zeta} \quad \frac{1}{2}||\bar{w}||^2 + C\sum_{i=1}^{m} \zeta_i \quad (16)$$

$$\text{s.t } y_i(\bar{w}.\bar{x}_i + b) \geq 1 - \zeta_i \ \forall I \text{ and } \zeta_i \geq 0, \ \forall i$$

The width of the soft margin can be controlled by a corresponding penalty parameter C [21]. A high C tries to classify every training data pint correctly; on the other hand, a low value of C ensures a smooth decision surface. Using ζ and C in the objective function (16) means that we are trying to not only minimize $||\bar{w}||^2$ but also minimize $\sum_{i=1}^{m}\zeta_i$. The Wolfe dual problem will be solved under a slightly different constraint:

$$\max_{\alpha} \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j \bar{x}_i.\bar{x}_j \quad (17)$$

$$\text{s.t } 0 \le \alpha_i \le C, \forall \text{i and } \sum_{i=1}^{m}\alpha_i y_i = 0$$

In feature transform or kernel, the basic idea is to transform the data from a nonlinear space into a linear space by mapping input vectors x ∈ R^n into vectors Φ(x) [22]:

$$\bar{\boldsymbol{x}} \in R^n \rightarrow \Phi(\bar{\boldsymbol{x}}) = [\Phi(\bar{\boldsymbol{x}}_1), \Phi(\bar{\boldsymbol{x}}_2) \ldots \Phi(\bar{\boldsymbol{x}}_n)] \in R^f$$

Let us return to the Wolfe dual Lagrangian function (13); the value of a training example x is not used; only the dot product$(\bar{\boldsymbol{x}}_i.\bar{\boldsymbol{x}}_j)$ between two training examples is used. Therefore, this product is replaced with the scalar product $K(\bar{\boldsymbol{x}}_i,\bar{\boldsymbol{x}}_j)$ of the respective kernel functions and it will return the same results as in linear space. The kernel function computes the inner product between two projected vectors [23]:

$$K(\bar{\boldsymbol{x}}_i,\bar{\boldsymbol{x}}_j) = \{\Phi(\bar{\boldsymbol{x}}_i), \Phi(\bar{\boldsymbol{x}}_j)\}$$

$$L(\alpha)=\sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j K(\bar{x}_i,\bar{x}_j) \quad (18)$$

The soft-margin dual problem can be rewritten:

$$\max_{\alpha} \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j K(\bar{x}_i,\bar{x}_j)$$

$$\text{s.t } 0 \le \alpha_i \le C,\ \forall \text{i and } \sum_{i=1}^{m}\alpha_i y_i = 0$$

The hypothesis function for predicting the class of the test image is as follows:

$$y(\bar{x}_i) = \text{sign}\left(\sum_{j=1}^{S}\alpha_j y_j K(\bar{x}_i,\bar{x}_j) + b\right) \quad (19)$$

The RBF kernel depends only on the radial distance which is the Euclidean distance ($||\bar{x}_i - \bar{x}_j||$) [24]:

$$K(\bar{x}_i,\bar{x}_j) = exp^{\frac{-||\bar{x}_i - \bar{x}_j||^2}{2\sigma^2}}$$

The Gamma parameter is the distance that a single training example can reach and it is always σ>0, with low values meaning that the system can reach 'far' data, and high values meaning that the 'close' data can be reached.

V. Experimental Results

A system is developed using the Python programming language on a computer equipped with a Core i7 5500U processor, 2.40 GHz and 8 GB RAM. The system has three phases: training, validation and testing. A dataset of 27,620 sample character glyph images was obtained from Al-Khaffaf et al. [25]. TABLE I shows the distribution of data among the three phases of the experiment.

TABLE I. Distribution of Sample images among the Three Phases of this Work.

Stage	#Samples	Percentage
Training	17,677	64%
Validation	4,419	16%
Testing	5,524	20%
Total	27,620	100%

The dataset includes three English fonts: Comic Sans MS (Comic), DejaVu Sans Condensed (DejaVu), and Times New Roman (Times). Fig. 4 shows the sample character images used in this work. The distance profile and normalized central moment features for 17,677 character images were extracted in the training. The feature vectors of size 106 and 10 values were obtained for DP and NCM respectively, and then normalized and stored in a comma separated values (CSV) file separately for each feature type DP and NCM.

To predict the font class of the test images, the feature vector of the training character images and their classes were fed to the SVM classifier.

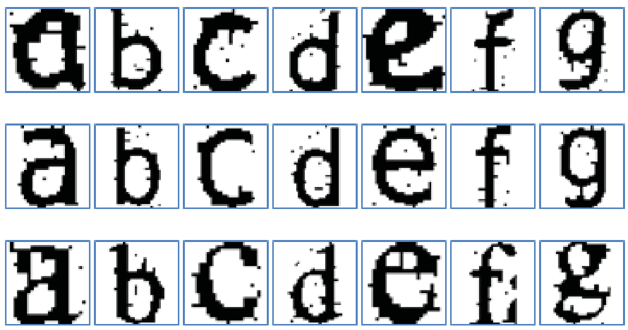

Fig. 4. Sample Letters used in our Experiment (a, b, c, d, e, f and g). Top: Comic, Middle: DejaVu, Bottom: Times.

In the validation phase, for each of the feature extraction methods, seventy-five experiments for various trials, depending on the values of the C and gamma parameters of the SVM with an RBF kernel, were carried out to reveal a pair of C and gamma values that were most suitable for recognition. In total, 4,419 sample images were used in the validation phase. For DP, the gamma and C pair of (1/10, 7) was shown to give highest recognition accuracy of 96.51%. For NCM, the gamma and C pair of (1/26, 35) yielded the highest recognition accuracy of 96.4%. Fig. 5 shows the recognition rate of the validation phase of seventy-five experiments using NCM features.

After the pair of values, gamma and the C parameter was revealed in the validation phase, a test on 5,524 images was performed depending on those values. The recognition rate was 94.9% when the pair was (1/26, 35), considering NCM features. This was a slight improvement over the recognition rate of 94.82% using DP features. TABLE II shows a comparison of many methods of English letter font recognition.

TABLE II. COMPARISON OF ANY METHODS.

Method/Work	Features	#Samples	Accuracy
Bharath and Rani [4]	DP	780	80%
Al-Khaffaf and Musa [8]	Eigenfaces	6,144	97%
Proposed	DP	27,620	94.82%
Proposed	NCM	27,620	94.9%

VI. DISCUSSION

Considering DP features, our recognition rate of 94.82% is better than the 80% that was reported by Bharath and Rani [4]. This could be due to two reasons. (i) We used a higher resolution images of 51×51 compared with 35×35 used by Bharath and Rani, which means that more features were used to train the SVM algorithm. (ii) We performed normalization of features that resulted in higher recognition rates compared with their work, which lacked data normalization. Additionally, we performed a more rigorous experiment involving 27,620 samples compared with only 780 samples used in the Bharath and Rani study. However, they studied 10 fonts compared to only 3 in our case. Using only 3 fonts is a limitation of our study.

Al-Khaffaf and Musa had a higher recognition accuracy of 97% compared to our results. However, they used a smaller dataset, which, according to our own experiments, yielded better results than when using a larger dataset. Al-Khaffaf and Musa also used principal component analysis (PCA) to reduce the number of features in training to a minimum, while our work used all calculated features.

VII. CONCLUSIONS

In this paper, a system for recognizing English language fonts was developed. The system uses NCM features and SVM as a classification method. Experimental results show that the system can recognize fonts with 94.9% accuracy considering NCM features compared to 94.82% when using DP features. Our results are significant for two reasons. (i) We used a large dataset compared to other previous studies. (ii) The SVM parameters were decided based on a validation phase, meaning rigorous experiments were used to find proper parameter values rather than rules of thumb. While the improvement is marginal, however, we showed that recognizing fonts based on any of these features is (i) feasible with a high recognition rate and (ii) using these two features yields a similar recognition rate. In future work, we plan to use other types of features, such as zone based features, in addition to the two features used in this study. Another direction to be inspected in the future is using PCA to reduce the set of features to only representative features that have a major effect on recognition.

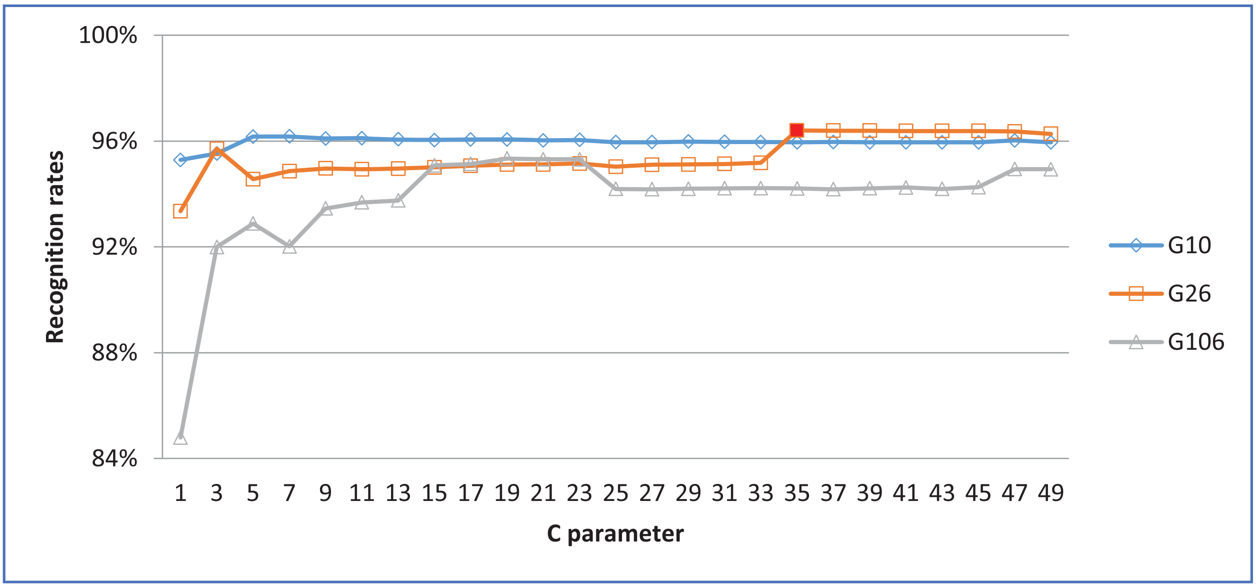

Fig. 5. Recognition Rate for the Validation Phase of NCM Features.

REFERENCES

[1] D. Doermann and K. Tombre, Handbook of Document Image Processing and Recognition, Springer-Verlag London, 1st Edition, 2014.

[2] R. Ramanathan, K.P. Soman, L. Thaneshwaran, V. Viknesh, T. Arunkumar, and P. Yuvaraj, "A novel technique for english font recognition using support vector machines." In 2009 International Conference on Advances in Recent Technologies in Communication and Computing, Kottayam, Kerala, India, pp. 766-769. IEEE, 2009.

[3] F. K. Jaiem, S. Kanoun, and V. Eglin, "Arabic font recognition based on a texture analysis." 2014 14th International Conference on Frontiers in Handwriting Recognition. IEEE, 2014.

[4] V. Bharath and N. Shobha Rani, "A font style classification system for English OCR." In 2017 International Conference on Intelligent Computing and Control (I2C2), Coimbatore, India, pp. 1-5. IEEE, 2017.

[5] F.K. Jaiem, F. Slimane, and M. Kherallah, "Arabic font recognition system applied to different text entity level analysis." 2017 International Conference on Smart, Monitored and Controlled Cities (SM2C). IEEE, 2017.

[6] C. Tensmeyer, D. Saunders, and T. Martinez, "Convolutional neural networks for font classification." 2017 14th IAPR international conference on document analysis and recognition (ICDAR). vol. 1, IEEE, 2017.

[7] E.M. Senobari and H. Khosravi, "Farsi font recognition based on combination of wavelet transform and sobel-robert operator features." 2012 2nd International eConference on Computer and Knowledge Engineering (ICCKE), IEEE, 2012.

[8] H. Al-Khaffaf and N. Musa, Optical english font recognition in document images using eigenfaces. Revista Innovaciencia, vol. 6, no. 1, p. 1-11, 2018.

[9] P. Romulus, Y. Maraden, P.D. Purnamasari, and A.A.P. Ratna, "An analysis of optical character recognition implementation for ancient Batak characters using K-nearest neighbors principle." In 2015 International Conference on Quality in Research (QiR), Lombok, Indonesia, pp. 47-50, IEEE, 2015.

[10] M.K. Hu, "Visual pattern recognition by moment invariants." IRE transactions on information theory, vol. 8, no. 2,pp. 179-187, 1962.

[11] L. Keyes and A. Winstanley, "Using moment invariants for classifying shapes on large-scale maps." Computers, Environment and Urban Systems, vol. 25, no. 1,pp. 119-130, 2001.

[12] A. Soumya and G.H. Kumar, "Classification of Ancient Epigraphs into different periods using Random Forests." In 2014 Fifth International Conference on Signal and Image Processing, Bangalore, India, South Korea, pp. 171-178. IEEE, 2014.

[13] H. Althobaiti and C. Lu, "Arabic Handwritten Characters Recognition Using Support Vector Machine, Normalized Central Moments, and Local Binary Patterns. "In Proceedings of the International Conference on Image Processing, Computer Vision, and Pattern Recognition (IPCV), pp. 121-127. ACM, 2018.

[14] N.V. Rao, G.R.M. Babu, and B.R. Babu, "A radial basis function neural network to recognize handwritten numerals with normalized moment features from skeletons." In 2013 International Conference on Recent Trends in Information Technology (ICRTIT), Chennai, India, pp. 68-72. IEEE, 2013.

[15] C. Cortes and V. Vapnik, "Support-vector networks." Machine learning, vol. 20, no. 3, pp. 273-297, 1995.

[16] Sh. Zhang and H. Qiao, "Face recognition with support vector machine." In IEEE International Conference on Robotics, Intelligent Systems and Signal Processing, 2003. Proceedings. 2003, vol. 2, Changsha, Hunan, China, pp. 726-730. IEEE, 2003.

[17] D. Boswell, "Introduction to support vector machines." Departement of Computer Science and Engineering University of California San Diego, 2002.

[18] A. Kowalczyk, "Support vector machines succinctly." Syncfusion Inc, 1st Edition, 2017.

[19] G. Katiyar, A. Katiyar, and Sh. Mehfuz, "Off-Line Handwritten Character Recognition System Using Support Vector Machine." American Journal of Neural Networks and Applications, vol. 3, no. 2, pp. 22-28, 2017.

[20] N. Deng, Y. Tian, and Ch. Zhang, Support vector machines: optimization based theory, algorithms, and extensions. Chapman and Hall/CRC, 2012.

[21] V. Kecman, "Support vector machines–an introduction." In Support vector machines: theory and applications, pp. 1-47. Springer, Berlin, Heidelberg, 2005.

[22] Y. Xu, S. Zomer, and R.G. Brereton, "Support vector machines: a recent method for classification in chemometrics." Critical Reviews in Analytical Chemistry, vol. 36, no. 3-4, pp. 177-188, 2006.

[23] B. Scholkopf, K. Sung, C.J.C. Burges, F. Girosi, P. Niyogi, T. Poggio, and V. Vapnik, "Comparing support vector machines with Gaussian kernels to radial basis function classifiers." IEEE transactions on Signal Processing, vol. 45, no. 11, pp. 2758-2765, 1997.

[24] C.M. Bishop, Pattern recognition and machine learning. Springer-Verlag New York, 1st Edition, 2006.

[25] H. Al-Khaffaf, F. Shafait, M. Cutter, and T. Breuel, “On the performance of decapod’s digital font reconstruction.” In 21st International Conference on Pattern Recognition (ICPR), 2012, pp. 649–652, Tsukuba, Japan, IEEE, 2012.