Integrating Writing Direction and Handwriting Letter Recognition in Touch-Enabled Devices

Akshay Jayakumar, Ganga S. Babu, Raghu Raman and Prema Nedungadi

Abstract Optical character recognition (OCR) transforms printed text to editable format and digital writing on smart devices. Learning to write programs has made learners trace an alphabet to learn the flow of writing and OCR by itself is less effective as it ignores the directional flow of writing and only focuses on the final image. Our research designed a unique android-based multilingual game-like writing app that enhances the writing experience. A key focus of the research was to compare and identify character recognition algorithms that are effective on low-cost android tablets with limited processing capabilities. We integrate a quadrant-based direction checking system with artificial neural networks and compare it to the existing systems. Our solution has the dual advantage of evaluating the writing direction and significantly increasing the accuracy compared to the existing systems. This program is used as the literacy tool in many villages in rural India.

Keywords Character recognition · Online recognition · Offline recognition · Neural networks · Quadrant-based direction checking · Alphabet rules · Self-organizing map · ANN · OCR

A. Jayakumar (✉) · G.S. Babu · R. Raman · P. Nedungadi
Amrita CREATE, Amrita University, Kollam, Kerala, India
e-mail: akshayjayakumar@am.amrita.edu

G.S. Babu
e-mail: gangasbabu@am.amrita.edu

R. Raman
e-mail: raghu@amrita.edu

P. Nedungadi
e-mail: prema@amrita.edu

S.C. Satapathy et al. (eds.), *Proceedings of the Second International Conference on Computer and Communication Technologies*, Advances in Intelligent Systems and Computing 380, DOI 10.1007/978-81-322-2523-2_38

1 Introduction

The pressing need of dealing with primary children and adult literacy is to offer new methodologies to effectively learn to read and write. In rural India, with limited access to computers, mobile devices can offer motivation, student engagement, and learning outcomes similar to personal computers [1]. Learning to write an alphabet correctly includes reproducing the correct image and adhering to the direction styles followed by a particular language. In our previous work, we presented a case study that showed the effectiveness of the tablet-based writing program in increasing the motivation and significantly reducing the learning time to master the alphabet. This program is being used at a tribal learning center [2].

The majority of existing programs for learning to write evaluate the alphabet based on specific hidden points. A few others use the OCR algorithm to recognize the alphabet drawn on the screen, though there is a need for additional research in Indian languages [3]. Offline character recognition is a technique used to identify a written character where parameters such as strokes, directions, and so on are no longer available. Online character recognition samples the data and attempts to recognize it as the character is written and the order of strokes is readily available [4]. The complexity associated with the recognition varies based on the language, the number of characters/alphabets, and the differences in the alphabet characteristics that makes up each one unique from the others. For example, an alphabet's image might be a subset of another one or it might share similar features. These can impact the performance of the handwriting recognition.

Character recognition typically uses feature-based or template-based methods [5, 6]. The feature-based methods in character recognition include: hidden Markov models [7], support vector machines [8, 9], and artificial neural networks (ANN) [9, 10].

In this paper, we discuss our proposed solution, which is unique in its ability to consider both the directional flow of writing and the shape of the character irrespective of the area of the screen. Our initial version was a great success in terms of motivating the learners and increasing the accuracy of evaluation.

2 Proposed System

Aksharamala provides an interactive alphabet learning and real-time evaluation environment that is designed to make the language learning experience engaging and effective. This application teaches how to trace and write alphabets and makes it fun by showing attractive animations, audio, and further improves its effectiveness as a learning tool with real-time evaluation.

A learner can write with a stylus or use the finger, and our online character recognition interprets and translates the digital data such as finger-strokes or stylus movements into digital text.

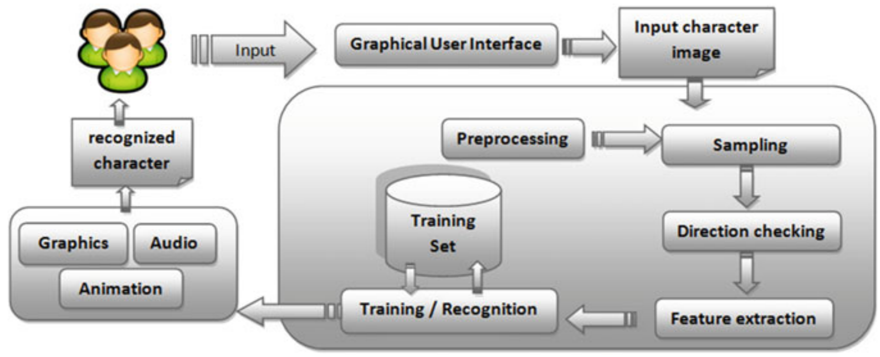

Fig. 1 Architecture of Aksharamala

The proposed system mitigates the weaknesses of other existing writing applications. An existing character recognition algorithm, neural network is used in the proposed system. As direction checking was a primary goal of the proposed system, neural network alone is insufficient as it recognizes alphabets without taking the direction parameters into account. Further research to identify the direction of writing paved the way for the new quadrant-based direction checking system that enabled Aksharamala to facilitate direction checks and significantly improve the recognition accuracy.

The overall functioning of Aksharamala is illustrated in Fig. 1. Prior to writing or tracing the alphabet, a short audio clip and animation aids the learner. Recognition of the drawn alphabet commences when the user has completed the drawing. The letter is extracted as an image and further computational processing such as sampling, preprocessing of the extracted image, direction checking, feature extraction, and recognition is done to identify the alphabet. All these processes are optimized to deliver performance in low processing tablets along with improved accuracy of recognition.

Features of our solution include:

- Traced letter being evaluated in real time to give feedback for the learner.
- Support for multiple languages like Hindi, Malayalam, Tamil, Telugu, Gujarati, and so on.
- Mapping made possible to any of the supported languages with any direction styles, i.e., from left-to-right, right-to-left, top-to-bottom, and so on.

3 Methodology

We discuss our approach for handwritten character recognition, algorithmic inferences, and the working. Our task at hand is to recognize a given unknown alphabet. We first train the system with various handwritings for each alphabet.

The algorithm tries to match the new input to find the most probable candidate so as to uniquely identify the input. Specific features such as direction are also identified in real time, as and when a user draws an alphabet.

3.1 Approach

The major shortcoming of just using ANN is its inability to accurately differentiate between alphabets with similar characteristics and its inability to identify the direction of writing. Integrating quadrant-based direction checking that captures the direction of the writing overcomes the shortcoming associated with ANN.

3.2 Self-organizing Map (Artificial Neural Network)

Several types of analysis, recognition, and interpretation can be associated with handwriting. An algorithm that can recognize a single handwritten character written by any person regardless of its shape, size, and style is required. Here we use Kohonen neural network [11] that is a self-organizing map (SOM). SOM by its property, is able to deduce relationships, and learn and adapt based on the inputs. The approach has been found to be very suitable for handwritten character recognition as it provides fast feature extraction and classification.

The alphabets written on a digital surface using a stylus or finger is cropped out with a fixed ratio, converted, sampled horizontally and vertically, and finally mapped on a 5×7 matrix template [12]. This matrix is then converted into vectors and used as inputs to the two layer self-organizing maps network [12]. The input to a SOM is submitted to the neural network via the input neurons [11]. The input neurons receive floating point numbers that make up the input pattern to the network [11]. Presenting an input pattern to the network will cause a reaction by the output neurons [11]. For a given input, the network is trained using unsupervised learning. On the links between the input layer and output layer a random weight matrix is defined [12]. In the training process of the network, this random weight matrix moves closer and closer to the input [12], when it becomes nearly equal to the input, the training stops. For every input character only one of the output neurons win [12]. In a self-organizing map, the value is produced by only one output neuron which wins and it is either true or false. Therefore, the output from the self-organizing map is usually the index of the neuron that is fired [11].

3.3 Drawbacks of Neural Network

As ANN is highly adaptive, its recognition is tolerant of minor errors and changes in patterns [13]. Figure 2 shows an example of such a case where only the first input is a valid representation of the Malayalam alphabet 'ആ' (Aa).

As ANN chooses the most similar alphabet from the training set, it selects the letter 'ആ' (Aa) for all the input images as seen in Fig. 3, thus accepting error inputs.

Another drawback is that using only ANN ignores the writing direction of the alphabet. Alphabets in some languages are 'left-to-right ' and in some they are 'right-to-left' oriented. To evaluate if a student has learnt to write, the correct writing direction needs to be considered.

3.4 Quadrant-Based Direction Checking

Our quadrant-based direction checking for character recognition complements the functioning ANN. Quadrant-based direction checking involves defining a rule for each alphabet, preprocessing, and mapping the image into a matrix format, defining quadrants, identifying the start and end quadrants, and finally determining the direction and then matching with the rules.

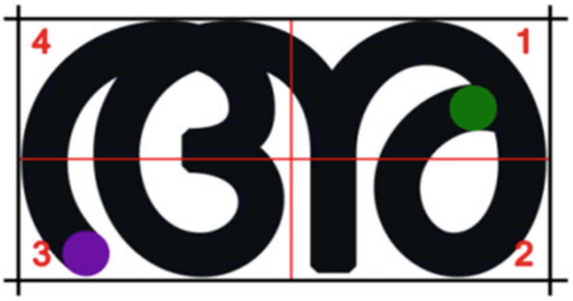

Fig. 2 Quadrant marked alphabet

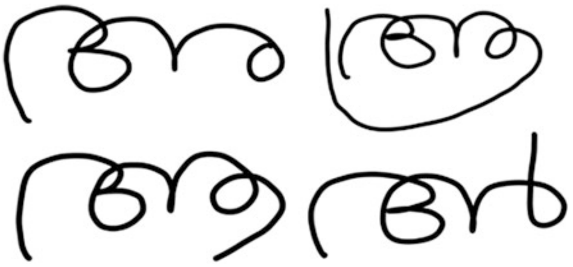
Fig. 3 Input to neural network

Step 1. **Define a rule for each alphabet**. A specific rule, consisting of the features extracted from the training set is defined for each alphabet in a language. The ‘start’ and ‘end’ tag specifies the quadrant where the alphabet would start and end in a regular writing style. The direction tag contains a code that represents the possible writing directions. In some cases, we use the combination of two or more codes to exactly define the direction.

Step 2. **Image Preprocessing**. We use online character recognition to find the set of coordinates for the alphabet written by the user. The character written on the drawing area is then cropped out, converted, sampled, positioned vertically, and mapped on a matrix template.

Step 3. **Define Quadrant**. The output from the previous step is the input to this step. We determine the boundaries and the quadrants for the scanned alphabet, as seen in Fig. 2.

Step 4. **Identify the quadrant in which the start and end coordinates reside**. After defining the quadrants, we determine the quadrants that represent the starting and ending coordinates of the input alphabet. The starting and ending quadrants would vary for different alphabets. Since the same alphabet can be written in different styles, there is a possibility for an alphabet to have more than one quadrant as its starting or ending points. To illustrate this, consider the case of the alphabet ‘ആ’. Alphabet ‘ആ’ can be written in different ways so that the ending coordinates may be in the first or second quadrant, but given any valid instance of the alphabet ‘ആ’, the starting coordinates ought to fall in the third quadrant. Thus, an alphabet can have these values varying depending on the writing style, shape, and stroke width.

Step 5. **Calculating direction and rule matching**. Once the start and end quadrants are calculated, we need to check the direction of the alphabet scanned. Rules derived from the alphabet are compared with the predefined rule to obtain a match. If the input alphabet obeys the rule, it implies that the alphabet is written in the correct direction and it is recognized as valid.

4 Experimentation

Initial tests were conducted with ANN using training data set for languages, mainly Malayalam. The training data set contained only five instances of every alphabet written in various writing styles. Out of the five instances for every alphabet, one was written correctly and the others were written with random variations and shaky hand strokes, this was done with the intention to introduce some imperfections. The input test data to the ANN was extracted from the user written alphabet on the tablet device and it was seen that the ANN-based recognition had an accuracy of 83 %. Our proposed method of the Quadrant-based direction checking along with ANN

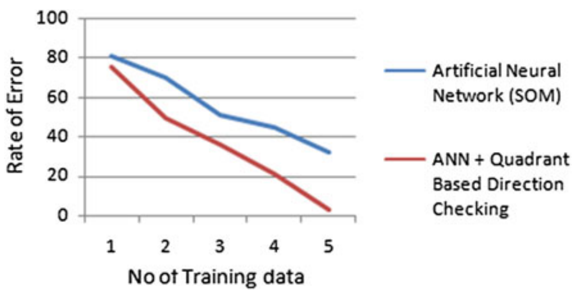

Fig. 4 Error rate produced by the two algorithms

significantly improved the recognition accuracy to 97 %, which was revealed in the test performed on the same input while retaining the previous training set.

The number of training data has a significant effect on the error rate of recognition. For a lower number of training data the error rate was observed to be high and with more training data included, there was a drop in the error rate. For any given case, ANN produced error rates higher than that of ANN combined with direction checking system for the same training data and input (shown in Fig. 4). The results show that the combination of ANN and Quadrant-based direction checking; together with training sets containing more instances of every letter could provide higher accuracy of recognition of alphabets.

5 Conclusion

Technology-enhanced language learning requires an accurate evaluation system with immediate feedback to the learner. A comprehensive evaluation of a learner's writing requires verification and feedback of both the writing direction and the final image. Our proposed integration of ANN and quadrant-based direction checking enhances the previous system in two ways; in its ability to evaluate the writing direction and in improved accuracy of evaluating the final image. Moreover, our method is language independent and has been successfully used in multiple Indian languages such as Malayalam, Tamil, Hindi, Gujarati, and Telugu. While OCR systems typically require numerous samples to learn a character, our method requires fewer training data per character. And most importantly, the system works with the low processing power of android tablets and without the support of higher end servers. The system is deployed in multiple village centers as part of the Amrita rural india tablet-enhanced education (RITE) program.

Acknowledgments This work derives its inspiration and direction from the Chancellor of Amrita University, Sri Mata Amritanandamayi Devi. We are grateful for the support of our colleagues at Amrita CREATE and the staff at Amrita University.

References

1. Nedungadi, P., Raman, R.: A new approach to personalization: integrating e-learning and m-learning. Education Tech. Research Dev. **60**(4), 659–678 (2012)
2. Nedungadi, P., Jayakumar, A., Raman, R.: Low cost tablet enhanced pedagogy for early grade reading: Indian context. In: Humanitarian Technology Conference (R10-HTC), 2014 IEEE Region 10, pp. 35–39. IEEE, Aug 2014
3. Ma, H., Doermann, D.: Adaptive Hindi OCR using generalized Hausdorff image comparison. ACM Trans. Asian Lang. Inf. Process. **2**(3), 193–218 (2003)
4. Cha, S.H., Srihari, S.N.: Writing speed and writing sequence invariant on-line handwriting recognition. Pattern Recogn. **10**, 9789812386533_0020 (2001)
5. Pal, U., Chaudhuri, B.B.: Indian script character recognition: a survey. Pattern Recogn. **37**(9), 1887–1899 (2004)
6. Govindan, V.K., Shivaprasad, A.P.: Character recognition—a review. Pattern Recogn. **23**(7), 671–683 (1990)
7. Shaw, B., Kumar Parui, S., Shridhar, M.: Offline handwritten devanagari word recognition: A holistic approach based on directional chain code feature and HMM. In: International Conference on Information Technology, ICIT'08, pp. 203–208. IEEE (2008)
8. Pal, U., Wakabayashi, T., Kimura, F.: Comparative study of Devnagari handwritten character recognition using different feature and classifiers. In: 10th International Conference on Document Analysis and Recognition, ICDAR'09, pp. 1111–1115. IEEE (2009)
9. Arora, S., Bhattacharjee, D., Nasipuri, M., Basu, D.K., Kundu, M.: Recognition of non-compound handwritten devnagari characters using a combination of mlp and minimum edit distance. arXiv preprint arXiv:1006.5908 (2010)
10. Kompalli, S., Nayak, S., Setlur, S., Govindaraju, V. Challenges in OCR of Dev anagari Documents. In: ICDAR, pp. 327–333. Aug (2005)
11. Heaton, J.: Introduction to Neural Networks with Java. Heaton Research, Inc (2008)
12. Sharma, K.S., Karwankar, A.R., Bhalchandra, A.S.: Devnagari character recognition using self organizing maps. In: 2010 IEEE International Conference on Communication Control and Computing Technologies (ICCCCT). pp. 687–691. IEEE Oct 2010
13. Araokar, S.: Visual character recognition using artificial neural networks.arXiv preprint cs/050501 (2005)