Kannada Character Recognition Using Multi-Class SVM Method

[1]Kusumika Krori Dutta, [2]Sunny A.S., [3]Anushua Banerjee, [3]Divya Rashi B, [3]Chandan R, [3]Deepak Vaprani
kusumika@msrit.edu , sas.bellary@gmail.com anushuabanerjee01@gmail.com divyarashi1824a@gmail.com , chandan261199@gmail.com , deepakvaprani03@gmail.com
[1]Assistant Professor , Dept. of Electrical and Electronics Engineering, M.S. Ramaiah Institute of Technology, Bangalore, India
[2]Engineer/Scientist II, Electric Power Research Institute, North Carolina, USA.
[3]Student, Dept. of Electrical and Electronics Engineering, M.S. Ramaiah Institute of Technology, Bangalore, India

***Abstract*— Character recognition also truncated to OCR (Optical Character Recognition), is the translation of images: handwritten, typewritten (either mechanically or electronically), or even just a simple text printed into a machine-editable text. Among them, one of the most relevant types is handwritten character recognition. Each handwritten content is composed of symbols, alphabets, etc. that are very syllabic with a distinct font style. Character recognition is achieved through segmentation, feature extraction, and classification, by using any one the machine learning methods thereby making tremendous advancements. This paper keenly focuses on the recognition of handwritten Kannada alphabets. Different machine learning classification strategies have known to be applied to achieve this recognition. Although, in this paper, we mostly center around the procedures dependency on the solution provided by SVM classifiers using Python. For simplicity, we focus on only four Kannada alphabets in this paper.**

Keywords— Handwritten Kannada alphabets, binary SVM Multi-class SVM, Python

I. Introduction

The field of machine learning provides a wide scope of techniques that actively contribute to solving various real-world practical problems,[8]. One of the important subsets of machine learning is the method of learning traditionally in two ways based on the training data namely: unsupervised and supervised. The algorithms based on this supervised learning takes in the data as the inputs that are labeled and predict the new output data points accordingly. Among the algorithms, this paper proposes the use of the Support Vector Machine (SVM) algorithm as the respective supervised machine learning model that is designed hierarchically. The calculation and process proceeded by choosing a reasonable multi-class SVM technique in a way to deal with the acknowledgment of transcribed Kannada letters, [4].

The foundation of SVM is the statistical learning theory [1]. As a result, it now assumes a significant role in the learning-based process and is utilized by data analysts, mathematicians, and researchers, additionally by architects as well. Support Vector Machines have met with numerous accomplishments in various learning undertakings, [5]. The principle distinction among this algorithm and other regular classifiers like K-means is that it applies to risk and loss minimization results whereas the others are mainly experimental reduction of risks. Therefore, we picked on a few of the estimations and examined their presentation with respect to the training time, testing time, and desired accuracy as the key parameters that needed to be achieved [1].

Kannada is one of the standard languages spoken in the Southern part of India and is spoken broadly in the state of Karnataka. The Kannada content is syllabic. It has 34 consonants and 15 vowels. An in-depth exploration review was finished with respect to the works and techniques proposed,[2], for the Kannada letters, for it to be acknowledged to make this paper more reasonable in regards to the utilization of the said SVM approach,[6]. There have been some astonishing works that have been done in the past with respect to the recognition of Kannada letters. Kannada, as a language is very well articulated and as it may, is verifiably consisting of more than the forty-nine characters in the letters altogether[3]. Because many of the letters could merge in outlining the other composite letters that are part of the language, ideally, the character picture formed in this language dictionary is relatable to a monosyllable character,[7].

The basic starting step for any kind of recognition is creating a dataset. Our customized dataset was made by writing all of the Kannada letters in 10 different font styles, each letter. Data augmentation was practiced on this dataset to increase the diversity of the samples. This could help us train some for our proposed model, while others test the performance to achieve the highest possible accuracy. Hence this paper presents an insight into the working algorithm used, with proven experimental data.

In this paper, the Multi-Class Support Vector Machine Classifier is implemented to recognize Handwritten Kannada alphabets using Python [9]. The paper is divided into the following sections, the first section is the introduction, followed by the method and methodology of implementation. The results obtained are discussed and concluded in the latter sections.

II. THEORETICAL FOUNDATION FOR SVM

One of the most well-known machine learning algorithms in supervised learning is Support Vector Machine (SVM). It is applicable for both classification and regression problems. SVM aims to find a hyperplane that is used to distinctly classify data points and hence separate classes. SVM can accommodate N number of features and requires less computation power. Let x_i and y_i represent labelled training data and their outputs respectively[4]. A suitable decision function must fit the training dataset satisfactorily, avoiding overfitting and under fitting. Maximized margin is an important consideration when deciding the hyperplane that classifies data points. It provides additional stability to future predictions. SVM is an example of a non-probabilistic binary classifier that can be used with modification as a multiclass classifier. In its form as a multiclass form, it has a one-against-rest approach, whereas in its primary form it follows one-against-one approach.

The estimation of rankings of the test vector is achieved through the use of the prediction function during the testing time in SVM. Data obtained is then mapped to a space of transformation where the features become linear. The space obtained from a “Mapping Function” Φ(x), and is called “Features Space”. A kernel is used to map non-linear input space into a linear feature space. This is done as a calculation of mapping function which is considered difficult and sometimes even to be impossible.

Amara [1] instead of locating the mapping function, it uses a kernel feature K, which represents the scalar product. This is shown inside the formula:

$$\Phi(x_i)\Phi(x_j) \quad = \quad k(x_i,\, x_j) \tag{1}$$

Some of the typical kernel functions are given below:

1. Linear:

It is one of the simplest kernels. It is denoted as follows:

$$\mathrm{k}(x, \mathrm{y}) = \mathrm{x}^{\mathrm{T}}\mathrm{y} + \mathrm{c} \tag{2}$$

2. Polynomial:

This is a non-stationary kernel and is well suited to normalized dataset. It is denoted as follows:

$$\mathrm{k}(x, y) = (\alpha x^T y + c)^d \tag{3}$$

3. Gaussian Kernel:

This is a radial basis function kernel. It is denoted as:

$$\mathrm{k}(x, y) \ = e^{-\left(\frac{||x-y||^2}{2\sigma^2}\right)} \tag{4}$$

It can also be implemented as

$$k(x, y) = e^{-\gamma||x-y||^2} \tag{5}$$

4. Hyperbolic Tangent Kernel:

It is also referred to as a Sigmoid Kernel and as a multilayer Perceptron Kernel. It is very popular in SVM as it originates from Neural Network theory. It is denoted as follows:

$$\mathrm{k(x, y)} = \tanh(\alpha x^T y + c) \tag{6}$$

where α, γ, T, and d are parameters. Some other kernel functions are Exponential Kernel, Laplacian Kernel, Log Kernel, Spline Kernel, etc.

III. MULTICLASS SVM BASED ON BINARY CLASSIFICATION

SVMs were initially designed and implemented to solve binary classification problems. Hence, to enable multi-class problems, these have to be sub-divided into a series of binary classification problems. This can follow either the one-against-one approach or the one-against-all approach. The reduction of multiclass problems into various multiple binary problems can be considered as one approach. Binary classification SVMs with a decision technique to solve the multi-class pattern recognition issues, [3]. All the SVMs are independently trained. In the following sections, One-Against-One and One-Against-All methods are introduced. Let us assume a training dataset (x_i, y_i) consisting of N data samples which belong to M classes. Here, Φ is the mapping function and $c_i \in \{1,2, ..., M\}$ is the class of x_i.

A. *One Against All*

Since this is an early extension of the binary SVM, we consider the One-Against-All (OAA) method. Here, M binary SVM models based on the classes are constructed. Each SVM model is built to differentiate its own class from the rest (M-1) classes. The last SVM is trained with all the dataset samples from that class with those labels and all the other classes’ examples with negative labels. The favored class is denoted as y_i and is defined as shown below:

$$y_i = +1 \text{ if } c_i = k \tag{7}$$

$$y_i = -1 \text{ if } c_i \neq k \tag{8}$$

The most appropriate hyperplane is then built to distinguish and separate (N/M) positive examples (y_i = +1) from N (M − 1)/M negative examples (y_i = −1). The class whose discriminant feature has the maximum value is then assigned to a new class based on the below-shown formula:

$$y_i = argmax_{x_i}, \ldots M((w_i)^T\phi(x) + b_i) \tag{9}$$

Each test data-sample is then classified through the “winner-takes-all” decision technique. Therefore, the total number of SVMs utilized in this technique is M, which is the number of classes.

B. *One Against One*

In the One-Against-One method, the Classifier is trained to distinguish each class (k, m). Thus, the total number of SVMs in this method is M (M −1)/2. For a training example x_i, the class y_i is defined as given below:

$$y_i = +1 \text{ if } c_i = k \tag{10}$$

$$y_i = -1 \text{ if } c_i = m \tag{11}$$

There are various strategies and different techniques to classify a new pattern. This method uses the following equation to calculate the correct class of a new pattern:

$$sin((w^{km})^T\phi(x) + b^{km}) \tag{12}$$

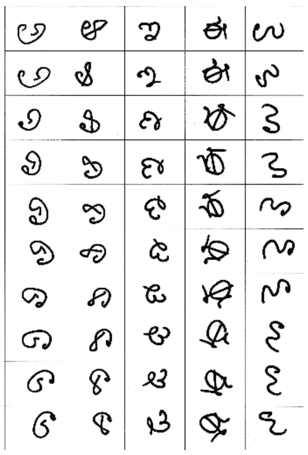
If the value of the choice function in equation 11 for a pair of classes (k, m), is negative, then m wins a vote. Otherwise, the class k wins a vote. The classes are then assigned with test samples based on the number of votes they gather.

IV. METHODOLOGY

This section discusses the methodology implemented in this paper. It discusses topics from the acquisition of data to the working of the model.

A. *Data Acquisition*

The samples for our dataset were obtained through our classmates each possessing different handwriting and variations in their writing, [10]. The letters were written on paper in the form of a grid which was then scanned to form a digital image as shown in Figure. 1 (a). The image was then cropped accordingly to isolate each letter. For our dataset, we have used the following letters as shown in Figure. 1 (b).


(a)

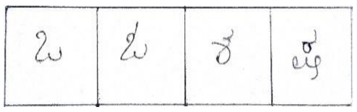

(b)

Figure 1: a) scanned imag , b) resized alphabets

B. *Pre-processing of data*

Pre-processing is a vital step in character recognition due to the efficacy it brings to the process. The raw image data needs to be processed so that it can be used for training and training purposes. Before going further, pre-processing is done to standardize the data to reduce complexity and to increase the algorithm efficiency by improving the readability of the data.

1. Greyscale and image inversion: The images were read in greyscale to reduce the image matrix from a three-dimensional one in RGB to a two-dimensional matrix in which each element ranges from 0 to 255. The original images had white backgrounds meaning that the maximum number of pixels read closer to the 255 value. Hence the images were inverted to make the outline of the character white and the background black as shown in Figure. 2. to reduce complexity.

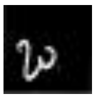


Figure 2. Inverted Character Image

2. Size reduction: Each image varied in their size and hence need to be brought to a standard size. In our paper, we keep the standard size to 28x28 i.e., 28 pixels in width, and 28 pixels in length. This standardization makes it easy to train the model with the data.

3. Normalization and feature scaling: The obtained matrix from the above steps will be 28x28 greyscale matrix. To further reduce the range of the matrix we normalize the values to be either 0 or 1. This is done by thresholding any values below 127 to be 0 and any value above it to be 1. Since most classifiers are based on distance, this reduced the stress on computation for the classifier.

4. Conversion to CSV: The matrices were then converted to a 1D array with 784 elements in each array. The converted arrays were then written into a Comma-Separated Values (CSV) file for better readability by the model.

C. *Feature Extraction*

Feature extraction is one of the main parts of a character recognition algorithm. Features are used to classify a given object into its respective class hence the ideal selection and extraction of features is important to boost the accuracy and efficiency of the model. In this paper, we deal with 784 features as each image is 28x28 in size. The need to reduce the features was not relevant in this paper as a reduction in the features using Principle Component Analysis (PCA) and other such feature reduction techniques did not yield a drastic increase in efficiency in our model. Hence, we have used just the Support Vector Classifier in our paper.

D. *Classifier and Classification*

Classification is the main step for character recognition. The characters are classified into various character classes based on their features. Many classification techniques such as K Nearest Neighbor (KNN), Random forests, Neural networks, etc., can be used for such classification. In this paper, we discuss the use of SVM means for classification.

The used SVM classifier has multiple parameters to be set to obtain optimal results. As discussed earlier, we have used an RBF kernel for our purpose as we are dealing with multiclass classification and required a higher dimension hyperplane for the purpose. The regularization parameter C, which helps in maximizing the distance between classes and minimizing the margin for misclassification [5]. The selection of a value for this parameter is mostly based on trial and error. We found that the value of 2 is optimum for our model. Another parameter of importance to us is the decision function type parameter which tells the model to use one against all or one against one method for classification. We have used one against one for our purpose.

The steps involved in our model are:

1. Loading the data: The training and testing data with their labels and features in the form of CSV file in loaded on the model.
2. Separation of labels and features: The data is separated into targets which are the labels corresponding to each character and features.
3. Training and Testing split: The data, both features, and target, are then split into training data and test data.
4. Training the SVM Classifier: The appropriate training data and the corresponding label associated with the training data along with other parameters i.e., kernel type, regularization parameter, decision function type, etc., are inputted to the classifier which then outputs a classification model, which is further used to predict the class of the test input during the testing phase of the model.

5. Testing of the model: Once trained, the model then has to be tested to verify the legitimacy of it. The features of the test data are loaded and the model outputs the predicted label for each case.

6. Accuracy: The predicted labels are then compared with the actual labels and the accuracy of the model is calculated depending on the number of labels it has predicted correctly.

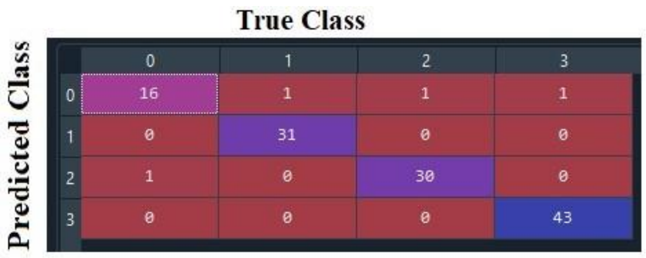
V. RESULT AND DISCUSSION

On the application of Support Vector Machine algorithm to our data set of handwritten Kannada letters, in different ratios of training data to test data, the following results were obtained; For a Training data to Test data ratio of 90:10, the efficiency of the algorithm on pre-processed test data was found to be 96.77%. For Training data to Test data ratio of 80:20, the efficiency of the algorithm on preprocessed test data was found to be 89.1566%. For a Training to Test data ratio of 85:15, the efficiency of the algorithm on preprocessed test data was found to be 95.69%. These results can be seen visually in the confusion matrices in the following Figure 3, Figure 4, and Figure 5.

A confusion matrix is a visual representation of the performance of a classification algorithm based on the known true values of the data set. It is a table where the columns of the matrix represent the true classes whereas the rows of the matrix represent the predicted classes by the classifier. All the predictions that were correct are shown along with the principle diagonal of the matrix while the misclassifications are represented along the lower and upper half of the diagonal.

The accuracy calculation using a 4x4 confusion matrix for a particular class can be done by simply summing up the True Positives (TP) and True Negatives (TN) of the class and dividing it by the sum of True positives, True negatives, False Positives (FP) and False Negatives (FN), i.e.,

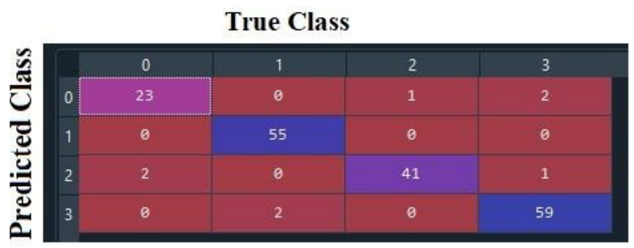
$$Accuracy = (TP + TN)/(TP + TN + FP + FN) \quad (13)$$


True Class

Predicted Class

	0	1	2	3
0	16	1	1	1
1	0	31	0	0
2	1	0	30	0
3	0	0	0	43

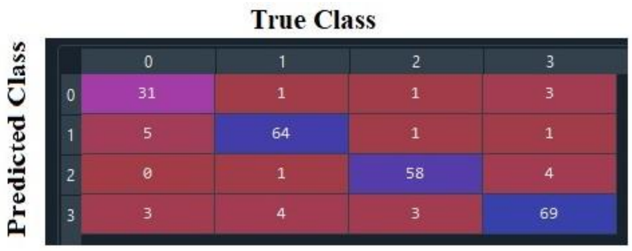
Figure 3: Confusion matrix for 99-10 split


True Class

Predicted Class

	0	1	2	3
0	23	0	1	2
1	0	55	0	0
2	2	0	41	1
3	0	2	0	59

Figure 4 Confusion matrix for 85-15 split


True Class

Predicted Class

	0	1	2	3
0	31	1	1	3
1	5	64	1	1
2	0	1	58	4
3	3	4	3	69

Figure 5 Confusion matrix for 80-20 split

In the above figure, Fig. 3, the accuracy of the zeroth class can be done by summing the (0, 0) element and the sum of the remaining elements apart from the ones in the zeroth column and zeroth row i.e., (1, 1),(1, 2),(1, 3),(2, 1),(2, 2),(2, 3),(3, 1),(3, 2) and (3, 3).
TP = 16
TN = 31 + 0 + 0 + 0 + 30 + 0 + 0 + 0 + 43 = 104
False Positives is the sum of the remaining elements of the zeroth row and False Negatives is the sum of the remaining elements of the zeroth column.
FP = 1 + 1 + 1 = 3
FN = 0 + 1 + 0 = 1
Therefore, the accuracy of the zeroth column element is
Accuracy = (104+16)/(104+16+3+1) = 0.967
This follows similarly to the other classes as well. During our testing, we found that the other methods such as the one against one and one against all yielded the same results. We have opted to use one against one in our model as mentioned before.

VI. CONCLUSION

This paper implements a handwritten Kannada character recognition machine with the use of the SVM classifier. There are several algorithms based on OCR techniques for individual recognition. The first step of the proposed algorithm is the pre-processing stage. The pre-processing stage consists of the operations to provide a clean individual image, that can be used directly and correctly by the feature extraction level. The image is then converted to a 28x28 black and white image for further processing. The larger size of the image can give irrelevant information and will occupy more space. Hence the efficiency of the algorithm is compromised and the variety of features and the computing time increases. The algorithm uses the SVM multiclass One-vs-One technique which is more appropriate for Kannada character recognition is based on two-fold characterization that gives accurate results. This work utilizes SVM classifiers for the usage of pixel values as a measure to recognize the characters. SVM gives good results when computing with a small sample size because it has the upper hand in nonlinear and high-dimensional pattern recognition and has better approximation ability and generalization ability.

REFERENCES

[1] Swapnil Sharma, Anumol Sasi and Alice N. Cheeran, “A SVM based character recognition system”, 2nd IEEE International Conference on Recent Trends in ElectronicsInformation & Communication Technology (RTEICT), India, 2017.

[2] Marwa Amara, Khaled Ghedira, Kamel Zidi and Salah Zidi, “A comparative study of multi-class support vector machine methods for Arabic characters recognition”, ACS 12th International Conference of Computer Systems and Applications (AICCSA), Morocco, 2015.

[3] Rajni Kumari Sah and K Indira, “Online Kannada Character Recognition Using SVM Classifier”, IEEE International Conference on Computa- tional Intelligence and Computing Research (ICCIC), India, 2017.

[4] S. R. Kunte and R. D. Sudhaker Samuel, “Online character recognition for handwritten Kannada Characters using wavelet features and neural classifier”, IETE Jounal of Research, Vol. 46, 2000 - Issue 5 : New Tools in Image Processing.

[5] Prasad, M. Mahadeva, M. Sukumar, and A. G. Ramakrishnan, “Divide and conquer technique in online handwritten Kannada character recog- nition”, Proceedings of the international workshop on multilingual OCR. ACM, 2009

[6] Shwetha D. and Ramya S., “Comaprison of smoothing techniques and recognition methods for online kannada character recognition system”, 978-14-4799-6393-5/14 IEEE ICAETR, 2014.

[7] Kunwar, Rituraj, K. Shashikiran, and A. G. Ramakrishnan, “Online handwritten Kannada word recognizer with unrestricted vocabulary”, Frontiers in Handwriting Recognition (ICFHR) International Conference on.IEEE, 2010.

[8] Mr. Brijain R Patel and Mr. Kushik K Rana, “A Survey on Decision Tree Algorithm For Classification”, International Journal of Engineering Development and Research, Volume 2, Issue 1, 2014, pp. 1-5.

[9] Pedregosa et al., “Scikit-learn: Machine Learning in Python”, JMLR 12, pp. 2825-2830, 2011.

[10] Kusumika Krori Dutta, Aniruddh Herle, Lochan Appanna, Tushar A, Tejaswini K, “Classification of Kannada Hand written Alphabets using Multi-Class Support Vector Machine with Convolution Neural Networks”, 2nd International Conference on Innovative Computing and Cutting-edge Technologies,(ICICCT2020), Bangkok, Thailand, 11-12 September, 2020.