

# Chapter 5

## Zone and rule assisted recognition of Meitei Mayek handwritten characters

### 5.1 Introduction

In any HCR system, some errors are inevitable. They can be due to the misrecognition of characters or due to the system not being able to classify certain characters. One of the biggest challenges that faces the recognition systems is the presence of confusing characters. This accounts for low accuracy rate of the system. Present work highlights this problem and proposes a second-stage recognition technique based on zone identification and script-specific orthographic rules to achieve a better recognition accuracy of the concerned script.

Due to the high structural and shape similarity of the confusing characters, it is sometimes extremely difficult even for the human eye to differentiate between them. Similarly, it is challenging for an HCR system to do the same task. This challenge can be dealt with by identifying the confusing characters and then applying certain techniques to distinguish between those characters. The task can be achieved by adopting a second-stage recognition which acts only on the desired characters, i.e., the confusing characters. For this second-stage recognition, there is a need to have features which are more discriminative from the confusing characters point of view. In most works found in literature it is observed that features considered are either the traditional hand-crafted features or features extracted from neural networks. The works lack in the area of incorporating the

peculiarities that exist in a specific script.

In this chapter, a two-stage recognition of Meitei Mayek handwritten characters based on script-specific properties is presented. Our method is based on the finding that most of the characters which are misrecognized in the first-stage recognition are confusing pairs. The two characters in confusing pairs have high inter-class misclassification rates between them. That is, if character classes  $i$  and  $j$  form a confusing character pair, then it means that characters in class  $i$  are misclassified as characters in class  $j$  for a significant number of times and vice-versa. The characters in a confusing pair can be distinguished from one another with the help of 1) zone of the character and 2) certain rules which are script-specific. A word written in Meitei Mayek can be divided into three horizontal zones. These zones can be used to find out the type of character and this forms the basis of the present work. Additionally, we have also used certain orthographic rules that are followed in writing Meitei Mayek to enhance the recognition accuracy in the present study.

## 5.2 Related work

There are several works reported in literature on multi-stage recognition of handwritten characters in other scripts using different approaches. There are works on finding discriminative feature vector for the confusing characters. In one such work, features based on Fisher-ratio is used to distinguish between the similarly-shaped characters of seven different languages [224]. They could achieve a better recognition accuracy on all seven languages. A two-stage approach for recognition of Bengali handwritten characters has been carried out where first stage recognition is carried out with features based on gradient directions and pixel counts [30]. The second stage uses the same feature vector but with a non-uniform grid which helps in identifying the portions of the characters with high shape dissimilarity from those portions with low dissimilarity. An accuracy of 95.84% is achieved with the approach. A two-stage recognition approach is adopted in the work carried out by Bhattacharya et al. [28] where the first recognition stage is a combination of three MLP classifiers. If any of the MLPs in the first stage could not identify a class for a numeral image, then certain estimates of its class conditional probabilities obtained from these classifiers are fed to a second-stage MLP. They reported accuracies of 97.93% to 99.26% on different datasets of Indic scripts. Another work explores multistage recognition based on Genetic Algorithm (GA) and SVM for recognition of handwritten Bangla compound characters [45]. 2.83% improvement

was achieved with the multistage approach compared to the single-stage approach.

The technique proposed in the work of Rahman et al. [162] considers feature vectors in two stages. Five high level features in the first stage divide the characters into seven different classes. The characters in each of these seven classes are then distinguished using low level features in the second stage. They have adopted a combination of multiple classifiers for the final classification and it is found to be more accurate and faster. An improvement of 3.09% in the accuracy is seen with the multistage approach. In another work, a multifeature and multilevel classifier approach is adopted [210]. In their work, for each image of a character, the member of candidate character classes gets reduced at each level leaving only one character class at the fifth level. In the work reported by Alaei et al. [9], the first level considers seven classes instead of ten by grouping similar shaped Arabic numeral classes in a single group. In the second level, the common parts of similar shape classes are eliminated and directional frequencies in contour pixels are computed to identify the numerals. Transition features are also computed for another group to identify numerals in another group. SVM is employed for the classification part. In another work reported by them [10], 32 Persian basic characters are grouped into 8 groups in the first level by employing features from bitmaps based on under sampled bitmaps. Features from chain-code direction frequencies are used for the second stage of the proposed scheme with SVM as classification method.

The approach adopted by Basu et al. [24] divides a Bangla word into three zones viz. upper, middle and lower zones based on the *Matra* hierarchy. A two-pass classification is then carried out for the characters in middle zone. Characters in middle zone are first classified into three categories using a 76-element feature set and an MLP based classifier. For the middle zone characters which are identified as a basic character or part of basic character, they have found out the groups of character classes within which the misclassification rate is high. Then a second pass classification is employed. Features used for this pass are extracted from certain specific region(s) of the entire pattern. Modified shadow and longest run features are then extracted from these regions and fed to MLP. They could achieve a better average test accuracy on Bangla basic characters using the two-stage classification scheme compared to the single-stage classification. They saw an improvement of 2.28% by adopting the two pass approach compared to single-pass approach. Similar method is reported where gradient features are used for first stage recognition to group 90-character classes of Malayalam handwritten characters into 10 groups [91]. In second-stage recognition, specific features such

as presence or absence of loops in different parts of the characters, count of end points, number of loops, etc. are used for each group to discriminate between the classes in the group. Recognition accuracy of 97.92% is achieved using two-stage recognition against 95.98% using single-stage recognition using SVM classifier. In the two-pass approach adopted in the work performed by Das et al. [46] in a soft computing paradigm, coarse and fine recognitions are carried out in first pass and second pass respectively. The first pass uses feature set consisting of Quad tree based Longest Run features. The second pass makes use of the same feature set, the difference being that the image sample is divided at a lesser depth for extraction of features. An improvement in test accuracy is reported for a Bangla handwritten character dataset of 256 pattern classes from 84.66% to 87.26%.

A two-stage recognition framework is proposed in the work reported by Vamvakas et al. [221]. They proposed a new feature extraction method of recursively iterating the character image so that the resulting sub-images have balanced number of foreground pixels. The two-stage recognition is then carried out based on the level of granularity of features extraction method. Classes with high similarity are first merged into groups and for each group of merged classes, granularity features from the level that best distinguishes them are employed to recognize the character images. The system could achieve a significant improvement on recognition accuracy ranging from 2.12% to 6.69% on CEDAR Character Database and CIL database compared to single-stage recognition.

Another work by Sarkhel et al. uses a multi-column multi-scale convolutional neural network (MMCNN) based architecture [188]. The multicolumn network has three columns, each column having three levels and functions independently. The first level of each column comprises of three multi-scale CNN networks. The features from these three multi-scale CNNs are concatenated and fed to an SVM classifier, thus employing three such SVMs for the three columns. A percolation prediction model is employed to decide whether the second and third levels of classification is required or not. In the second and third level of classification, the input image is sub-divided into four and sixteen quadrants respectively. Correspondingly, each quadrant of the divided input image is fed to four and sixteen smaller CNNs in the second and third respectively. A majority voting is performed on the labels assigned by the smaller CNNs in order to identify the final class. A hierarchical approach is performed in the work carried out by Kayumov et al. [94] for the recognition of MNIST dataset. There are two levels of CNNs. The first level CNN picks up two digits with highest activation functions. If the difference between these two values is more than 0.7, then it is


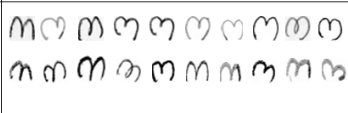
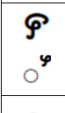
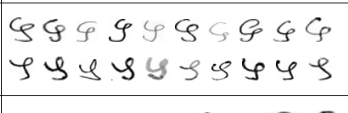



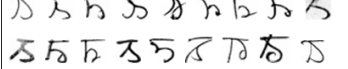
passed through the second level. The second level has 31 CNNs, each of which has been trained on two digits only. From each of the CNNs in both the levels, one winning digit comes out giving out a total of 32 digits. The last block has these 32 characters and a control character which is an output from the first level CNN indicating which CNN to select out of the 31 second level CNNs. The output of last block is the final character. An accuracy of 99.36% is reported. A multilevel multimodal fusion of the visual information and the linguistic semantic information is proposed in the work reported by Xia et al. [235]. They employed a CNN and BiLSTM based *looker* to extract deep features. A DenseNet-based single character recognizer and Google word2vec2 tool are used to generate a set of visual and language embeddings. Then a multimodal fusion network is used to fuse these embeddings on both character level and text-fragment level. The fusion network is finally embedded into an attention-based LSTM to generate the final output. The work is one of the few works we could find that considers the language semantics in addition to visual embeddings as part of recognition. The work described by Wang et al. [225] consists of two stages employing CNN in both the stages. The first stage differentiates inter-group characters and second stage differentiates intra-group characters. They reported a better accuracy rate with the two-stage recognition. A tri-stage feature extraction method is proposed for Malayalam script recognition [222]. The first stage employs geometric feature to group the character classes into groups. Character classes in each group is then differentiated from each other using specific features for each group in the second stage. The third stage is based on the rules that are followed while forming Malayalam words.

A hybrid recognition scheme is explored in the work carried out by Bhattacharya et al. [29]. In the two-stage recognition approach proposed in the work, first stage recognition takes place with shape feature vector and a Hidden Markov Model based classifier. The second stage is performed on eight smaller groups which are identified in the first stage. Wavelet transform features are used with three multilayer perceptron classifiers within each of the eight groups in the second stage. A test accuracy of 90.42% was achieved on a dataset of basic Bangla characters having 50 classes. A multi-stage approach based on HMMs is presented for handwritten Arabic text recognition [5]. The idea is to combine two methods. First method is to have character segments instead of character shapes as HMMs which are called sub-character patterns. This way number of character shapes can be reconstructed using only a few sub-character patterns. Second method is based on the idea that many characters in Arabic share the core shape and only differ based on the numbers and positions of dots and other diacritics. The second

## 5.2. Related work

method follows a multi-stage recognition where the core shapes are separated from diacritics in the first step. The second step involves sharing of similar patterns in the core shapes using the sub-character approach called as sub-core shapes. The training is carried out separately for the sub-core shapes and the diacritics and the results are combined to obtain the final output. They reported comparable results with other state-of-the-art works.

In any script, there are some characters which get misrecognised when fed to the recognizer due to the similarity in their shape and structure. Some of the highly similar-shaped character pairs found in Meitei Mayek is shown in Figure 5-1. In order to recognize such similarly shaped characters, there is a need for additional type of information which are more discriminative in order to distinguish them. In the present work, we seek to explore orthographic rules that exist in Meitei Mayek script by finding out the zonal information which is again a property specific to the script at hand. These rules which are specific to the script, once found out can play a significant role in enhancing the recognition accuracy of a particular script. Moreover, for those scripts which follow a strict set of rules while writing them, the rules can hardly go wrong while incorporating it in the recognition system. Another advantage of the present approach is that the second-stage recognition only takes a comparison function to make decision unlike the previous works where feature extraction and classification takes place in all the stages. This reduces the computational cost in the present work. One drawback, however is that the rules cannot be generalized to multiple scripts.

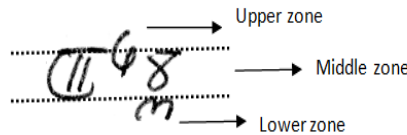
Characters	Handwriting samples
	
	
	
	

**Figure 5-1:** Example samples of highly similar-shaped character pairs

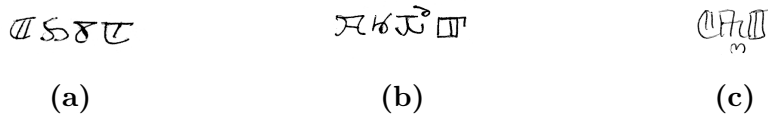
### 5.3 Features of Meitei Mayek

Many scripts have features which are specific to them. Meitei Mayek also has features which are specific to it. The script-specific features can be used in recognition system to help the system achieve a better recognition accuracy. Some features of Meitei Mayek are listed below.

1. **Division into zones:** In Meitei Mayek, a word image can be divided into three zones as shown in Fig. 5-2. The zones are named as upper, middle and lower zones. The characters thus can be divided to be belonging to either one of these zones. There are, however words that have only one or two zones as well. Example words are shown in Fig. 5-3.



**Figure 5-2:** Word consisting of characters in all three zones



**Figure 5-3:** Examples of words consisting of different number of zones. (a) Word with only middle zone (b) Word with upper and middle zones (c) Word with middle and lower zones

2. **Orthographic rules:** An orthography is a set of conventions for writing a language, including norms of spelling, hyphenation, capitalization, word breaks, emphasis, and punctuation<sup>1</sup>. In Meitei Mayek, there are orthographic rules that are followed while writing the language. One of the rules is that the character *ee-lonsum* (𑜇𑜨) is preceded by a vowel when used in a word. Whereas *ee* (𑜇) is either the first character of the word or it is not preceded by a vowel. This rule is used in our work to distinguish between *ee* (𑜇) and *ee-lonsum* (𑜇𑜨). It is described in Section 5.4.4. Some examples of words with *ee* and *ee-lonsum* are shown in Fig. 5-4.

In order to validate this rule, a text corpora written in Meitei Mayek named *The TDIL Hindi-Manipuri Agriculture & Entertainment Text Corpus ILCI-*

<sup>1</sup><https://en.wikipedia.org/wiki/Orthography>

### 5.3. Features of Meitei Mayek

Ee-lonsum	Ee
ꯀꯪꯂꯩ	ꯀꯪ
ꯀꯪꯂꯩ	ꯀꯪ
ꯀꯪꯂꯩꯂꯩ	ꯀꯪ
ꯀꯪ	ꯀꯪꯂꯩ

**Figure 5-4:** Example words consisting of *ee-lonsum* (ꯀꯪ) and *ee* (ꯀꯪ)

$I^2$  is used. The concerned script being in use after several decades, this is the only corpus that we could find till date. As the name suggests, it is a domain-specific text corpus in agriculture & entertainment domain. The corpus has been collected by Jawaharlal Nehru University, New Delhi as part of the Indian Languages Corpora Initiative phase-II (ILCI Phase-II) project, initiated by the MeitY, Govt. of India. It is a parallel corpus collected in Hindi as the source language and has been translated to Meitei Mayek as target language. For the present work, we are only interested in the corpus in target language and hence we have considered the translated text in Meitei Mayek. The original text corpus had some unwanted characters such as (, ), /, -, ., etc. We have cleaned the original text of these characters and have taken the text corpus consisting of only the 55 Meitei Mayek characters. Table 5.1 provides statistics of the two text corpora. And Fig. 5-5 shows the number of occurrence of each character in the cleaned text corpus.

Table 5.1: Statistics of The TDIL Hindi-Manipuri Agriculture & Entertainment Text Corpus ILCI-II

Total number of	Value
characters in the original corpus	1004055
words in the original corpus	196992
lines in the original corpus	14023
characters in the cleaned corpus	994223
words in the cleaned corpus	189901
lines in the cleaned corpus	14023

One important point to be noted here is that the TDIL corpus uses the same character (keystroke) to represent two characters viz. *ee* (ꯀꯪ) and *ee-lonsum* (ꯀꯪ). The character ꯀꯪ is used to represent both the characters. These two characters, have however been assigned two different Unicode values and hence it is an important aspect of the present work to discriminate these

<sup>2</sup>[https://tdil-dc.in/index.php?option=com\\_download&task=showresourceDetails&toolid=1874&lang=en](https://tdil-dc.in/index.php?option=com_download&task=showresourceDetails&toolid=1874&lang=en)



𑜄: 14435	𑜆: 55	𑜇: 62903
𑜅: 44905	𑜇: 33097	𑜈: 18670
𑜆: 27363	𑜈: 46090	𑜉: 10485
𑜇: 50897	𑜉: 7161	𑜊: 35120
𑜈: 22121	𑜊: 43038	𑜋: 13863
𑜉: 45774	𑜋: 64	𑜌: 5666
𑜊: 13389	𑜌: 392	𑜍: 13953
𑜋: 22657	𑜍: 1852	𑜎: 0
𑜌: 13305	𑜎: 26821	𑜏: 9054
𑜍: 4976	𑜏: 9893	𑜐: 1879
𑜎: 15490	𑜐: 24999	𑜑: 1565
𑜏: 4167	𑜑: 3092	𑜒: 1081
𑜐: 11467	𑜒: 22419	𑜓: 577
𑜑: 20416	𑜓: 12387	𑜔: 503
𑜒: 8358	𑜔: 31560	𑜕: 882
𑜓: 5694	𑜕: 20558	𑜖: 455
𑜔: 9577	𑜖: 33222	𑜗: 423
𑜕: 28375	𑜗: 103146	𑜘: 402
𑜖: 32772		𑜙: 758

**Figure 5-5:** Frequency of occurrence of characters in the cleaned corpus

two characters from each other. Moreover, this discrimination plays an important role when tasks such as text-to-speech and machine transliteration are carried out. This is because these two characters are completely different when their usage and meanings are concerned. In order to have an idea of the number of occurrences of the characters, out of the total 26,252 occurrences of these two characters, those which are preceded by a vowel are replaced with 𑜎 and they would be called *ee-lonsum* else they would be considered as *ee*. Following this assumption, there are 5694 occurrences of 𑜎 and 20558 occurrences of 𑜍.

The TDIL corpus has a total of 25696 words containing either *ee-lonsum* (𑜎) or *ee* (𑜍) or both. In order to validate the assumption and the rule implied, we divided the words based on the assumption. Out of these 25696 words, according to the assumption, 21385 words contain *ee-lonsum* (𑜎) only, 4001 contain *ee* (𑜍) only and 310 words have both the characters. Each of these 25696 words has been cross-checked by language experts. It is found that out of the total 21385 words assumed to have *ee-lonsum* (𑜎), 1306 (6.11%) should be *ee* (𑜍) and out of the total 4001 words with the assumed *ee* (𑜍), 80 (1.99%) should be *ee-lonsum* (𑜎). The number of erroneous characters in the words which have both the characters is 0. Therefore, the division of characters based on the assumption has an average accuracy of 94.72% which is a good figure to validate the rule.

- Zones of vowels and non-vowels:** A non-vowel (consonants, half-consonants, numerals) and a vowel in a word differ in terms of zones. The vowels always lie in the upper and lower zones and non-vowels always lie in

## 5.4. Proposed methodology

---

the middle zone. An example is shown in Figure 5-6.

It can be seen from Fig. 5-7 that all the eight vowels that exist in Meitei Mayek script lie either in upper zone or lower zone. All the vowels are written in the form shown in the figure where the dotted circle represents a consonant or a half consonant.

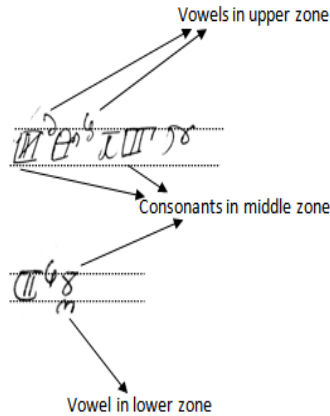


Figure 5-6: Words divided into zones

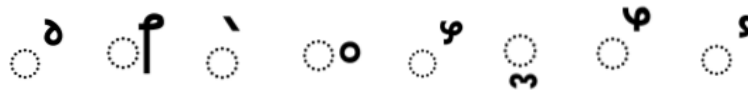
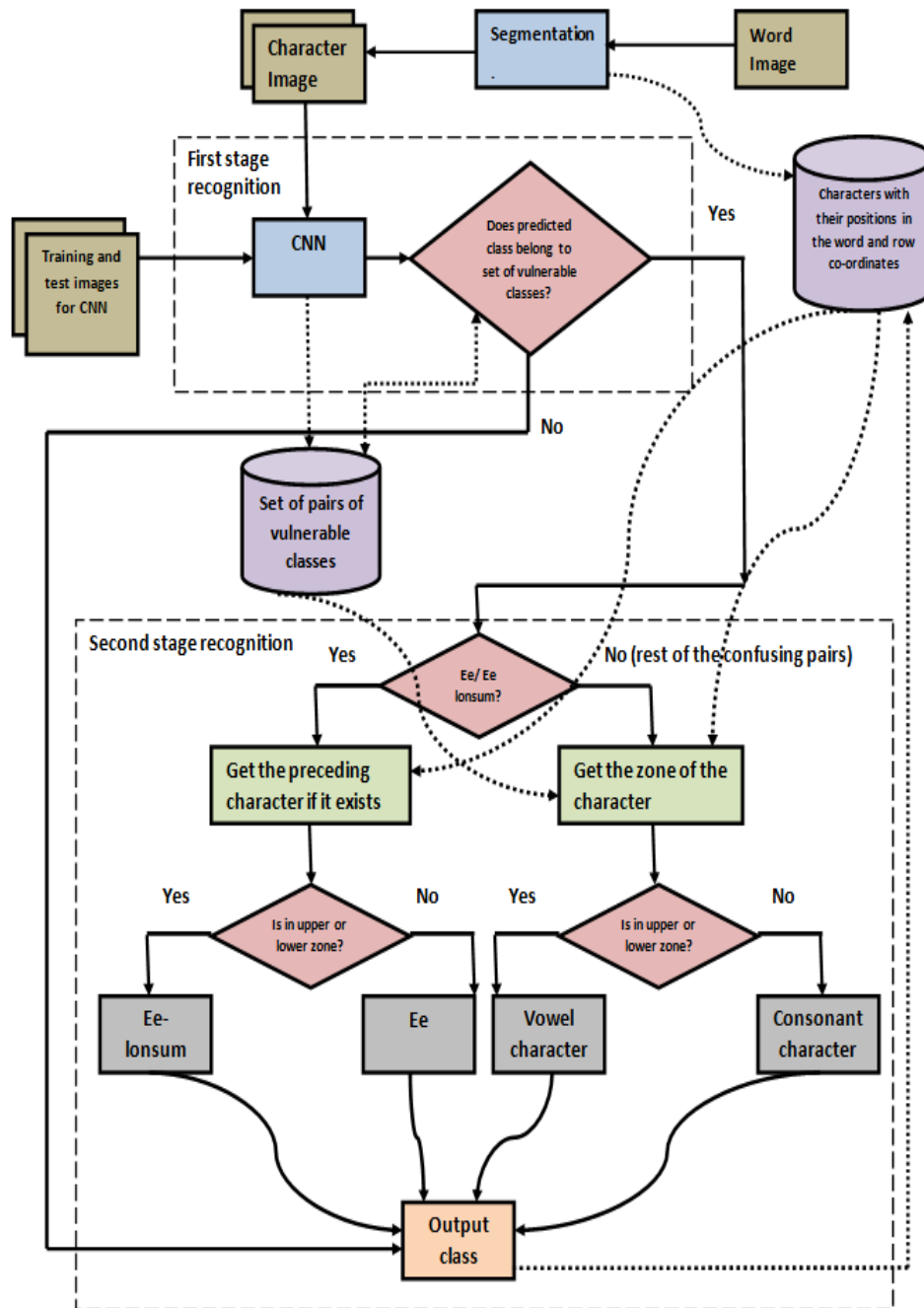


Figure 5-7: Vowels in Meitei Mayek Character Set

## 5.4 Proposed methodology

The overall conceptual framework of the proposed methodology is shown in Figure 5-8. There are two stages involved in the present approach. First-stage recognition is carried out using CNN and second-stage is done using script-specific properties. The CNN in the first stage is trained and tested using TUMMHCD to find out the confusing character pairs. The character classes of the confusing character pairs are named as the vulnerable set in the later sections of the chapter.

Now, the input test image initially in the form of word image is fed to the system. It is then segmented into its constituent characters using the connected component method. During this segmentation process, information regarding the positions and row-coordinates of each character of the word are saved. This information is then utilized in the second stage to find out the type of the misrecognized



**Figure 5-8:** Conceptual framework of the proposed two-stage recognition system. The dotted lines represent interactions of the units with the databases.

## 5.4. Proposed methodology

---

character or the preceding character. This information is then used to identify the character.

The segmented characters are then fed to the first-stage recognition. Here, it predicts the character class of the image using the trained CNN. It also checks if the predicted character class is one of the classes in the vulnerable set. If so, then the input test image is forwarded to the second-stage recognition where script-specific properties-based recognition takes place. Else, the predicted class is the final class of the input image. For the second-stage recognition, the zonal information and certain orthographic rules are used to distinguish between characters in the confusing character pairs. The zonal information of a character is obtained using its row-coordinates that are saved when segmentation was carried out. When an input character image is given to the second stage, it is expected that the character belongs to one of the classes in the vulnerable set. The second-stage recognition is carried out using some conditional statements which help in deciding the final class of an input image. Depending on which confusing pair the predicted character class belongs to, it is forwarded through the second-stage recognition. For example, if the predicted character class is either *ee* or *ee-lonsum*, then the desired confusing character pair is *ee* and *ee-lonsum* pair. In that case, it satisfies the first conditional statement in the second-stage recognition (refer Figure 5-8). Accordingly, the preceding character is retrieved from the saved information and its zone is found out. If the zone of the preceding character is either upper or lower, then the character image in question is an *ee-lonsum*, else it is an *ee*. The details of how each step is carried out are provided in the subsections below.

### 5.4.1 Training CNN

CNN is employed to carry out the first stage recognition as well as to find the vulnerable set. As discussed in the earlier chapter, there are certain pairs of confusing characters in Meitei Mayek which get misrecognized at all times. It has been shown that traditional feature-classifier combinations and various standard CNN models misclassify these characters.

For training and validation purpose, the training set and validation set of TUMMHCD consisting of 65,097 and 7,233 character images respectively are used. The network is executed for a total of 100 epochs with early stopping method with patience set to 20 and validation loss as the criterion to be monitored. Once the best model is found by early stopping method, it is used to test the accuracy on

the test set of 12,794 images. The confusion matrix obtained is used to find the vulnerable set (Figure 5-9).

### 5.4.2 Finding vulnerable set

Vulnerable set is the set of confusing character pairs with the least recognition accuracy. These character pairs are found to have been misrecognized as one another which results in low recognition accuracy of the HCR system. In order for the second stage to be able to identify if a character might have been misrecognized, it has to check whether the recognized character belongs to one of the characters in the vulnerable set. Our approach of finding vulnerable set is based on the information available in the confusion matrix achieved during training of CNN described in Sect. 5.4.1.

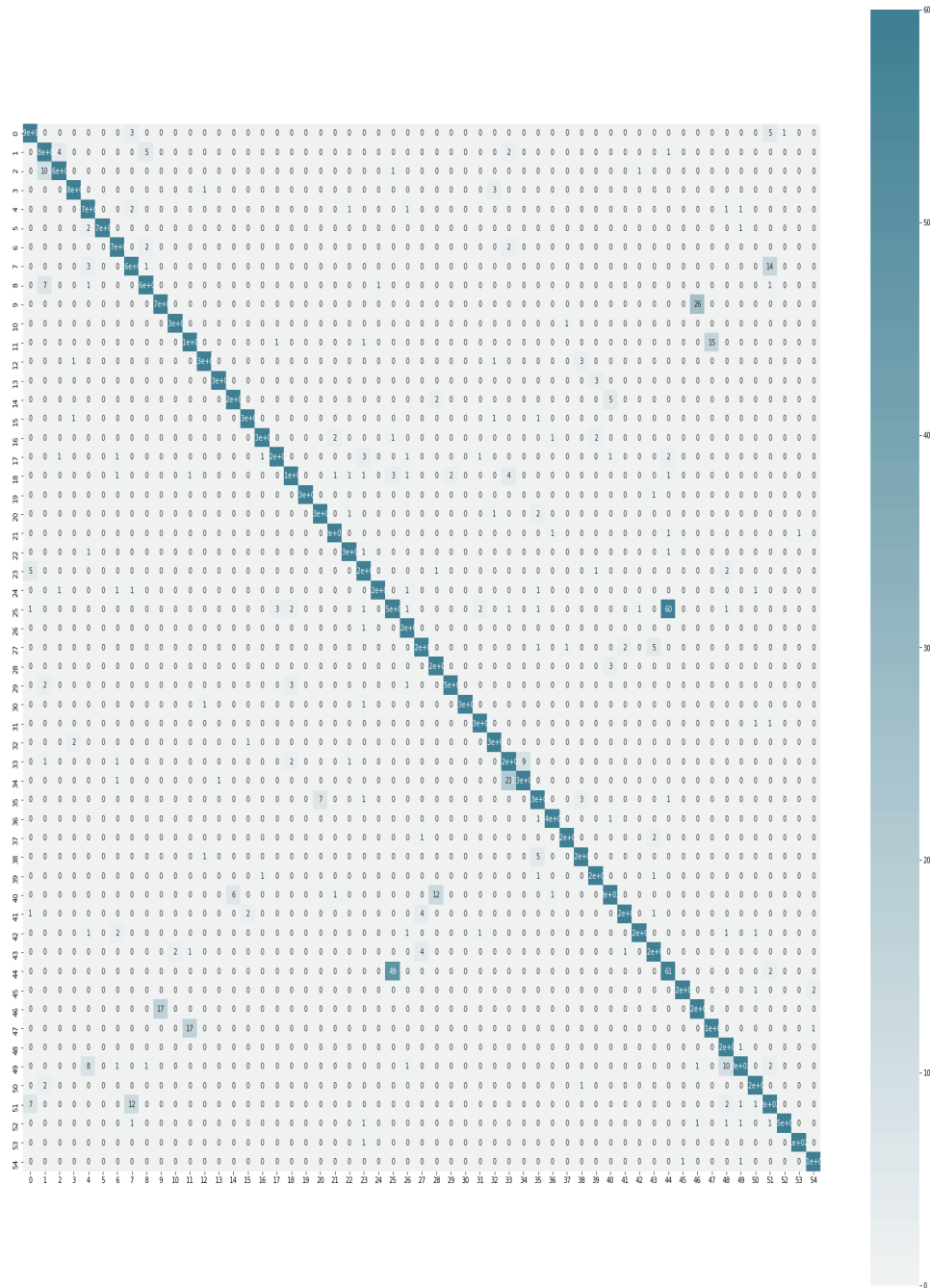
The approach proposed in the present work to find vulnerable set is provided in Algorithm 3. The procedure takes the confusion matrix generated by the previous section as input and gives out the vulnerable set as its output. We have considered two criteria for finding the vulnerable set. First, it calculates the relation between the accuracy of each character class and the average class accuracy of all the classes. For each class  $c_i$  in the dataset, it first finds out the accuracy  $accuracy(i)$ . It then finds out the average accuracy  $avg\_acc$  of all the classes. If the class accuracy of a particular class  $c_i$  is less than the average class accuracy  $avg\_acc$  then class  $c_i$  is shortlisted to be included in the vulnerable set.

For the second criterion, the false negative (FN) of each class is calculated and the relation between FN of a class and the total number of character images in that class is checked. FN of a class is the number of character images which belong to that class but have been misclassified into classes other than itself. For each character class  $c_i$ , FN can be represented by the following Equation 5.1.

$$FN(i) = \sum_{\substack{j=1 \\ i \neq j}}^{55} X(i, j) \quad (5.1)$$

where  $X$  is the confusion matrix and  $i$  and  $j$  are the row and column respectively. It is also important to note that the number of character images in each class is not same. Therefore, in order to generalize the equation for finding the vulnerable set across all classes, we have considered the total number of character images in each class. This is taken into account while trying to find out the

## 5.4. Proposed methodology



**Figure 5-9:** Confusion matrix obtained on the test set of TUMMHCD using CNN to be used for first stage recognition as well as to find vulnerable set. The x-axis indicates predicted class and the y-axis indicates true class.  $cnf[i, j]$  indicates the number of characters in class  $i$  which has been misclassified as class  $j$  (when  $i \neq j$ ), where  $cnf$  is the name of the confusion matrix.

---

**Algorithm 3:** Find vulnerable set

---

```

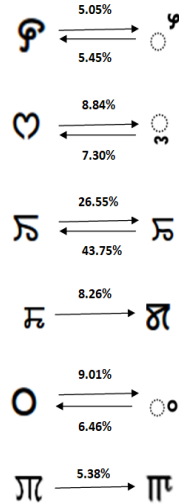
Input: Confusion matrix  $X$ 
Output: Vulnerable set  $V$ 
 $V = \phi$  ▷ no classes in the vulnerable set initially
 $[row, col] = size(X)$ 
 $size(i) = \sum_{j=1}^{col} X(i, j)$  ▷ size(i) - size of the  $i^{th}$  class
/* Find the class accuracy of each class  $c_i$  */
for  $i \leftarrow 0$  to  $row$  do
     $accuracy(i) = \frac{X(i, i)}{size(i)}$  ▷ accuracy of the  $i^{th}$  class
/* Find the average class accuracy of all classes */
 $avg\_acc = \frac{\sum_{i=1}^{row} accuracy(i)}{row}$ 
/* Find the false negative (FN) of each class */
for  $i \leftarrow 1$  to  $row$  do
     $FN(i) = \sum_{j=1}^{col} X(i, j) - X(i, i)$ 
/* Take 5% of the total images in each class to generalize the
   equation as number of instances in each class is different
   and find the vulnerable set */
if  $((accuracy(i) < avg\_acc) \wedge (FN(i)/size(i) > 0.05))$  then
     $V = V \cup i$ 
return  $V$ 

```

---

relation between FN of a particular class and the total number of images in that class. If the number of FN is greater than 5% of the total number of images in that particular class, then that class is shortlisted to be included in the vulnerable set. A value of 5% has been chosen empirically out of the values of 3, 4, 5, 6 and 7%. Moreover, since the overall average recognition accuracy of all the classes is close to 95%, a value of 5% has been chosen. Each character class is checked if it fulfils these two conditions. If it does, then that character class is included in the vulnerable set.

Following these two conditions, the vulnerable set finally has ten character classes with class ids 7, 9, 11, 25, 34, 40, 44, 46, 47 and 51. The corresponding characters in the vulnerable set are  $\mathfrak{P}$ ,  $\mathfrak{O}$ ,  $\mathfrak{C}$ ,  $\mathfrak{S}$ ,  $\mathfrak{K}$ ,  $\mathfrak{N}$ ,  $\mathfrak{D}$ ,  $\mathfrak{O}$ ,  $\mathfrak{L}$  and  $\mathfrak{P}$ . One more observation that could be seen is that out of the ten character classes in vulnerable set, eight are misclassified as pairs, i.e if a character  $i$  belonging to class  $c_i$  is misclassified as character  $j$  belonging to class  $c_j$  then character  $j$  is also misclassified as character  $i$ . The character pairs with highest misclassification rates between them are shown in Figure 5-10. Characters which get misrecognized as pairs viz.  $(\mathfrak{P}, \mathfrak{P})$ ,  $(\mathfrak{O}, \mathfrak{O})$ ,  $(\mathfrak{C}, \mathfrak{L})$  and  $(\mathfrak{S}, \mathfrak{S})$  is the vulnerable set which has been considered for the second-stage recognition.



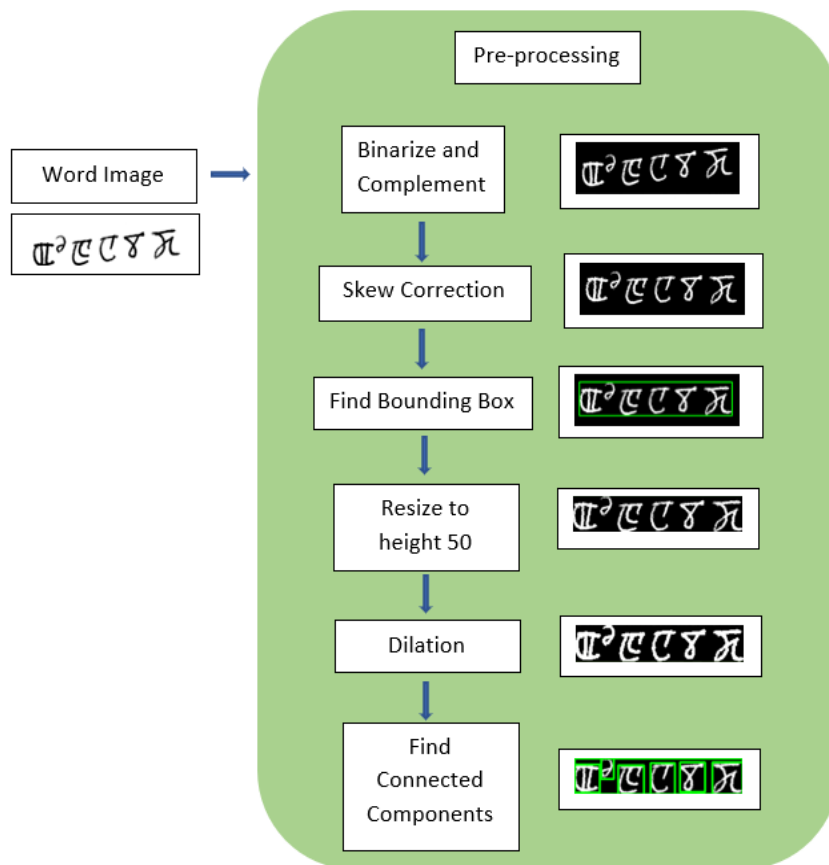
**Figure 5-10:** Character pairs with highest number of misclassification. The arrows from left to right indicate the percentage of character images of the character on the left which has been misclassified as the one on the right and vice-versa.

### 5.4.3 Zone identification

In Meitei Mayek, a word image can be divided into three zones as shown in the Figure 5-2. The zones are named as upper, middle and lower zones. The characters belong to either one of these three zones. There are, however words that have only one or two zones as well. Example words are shown in Figure 5-3. In order to find the zones of the characters, we have proposed a method based on the row coordinates of each character with respect to the row coordinate of the word as a whole. The task of zone identification is carried out simultaneously with the task of segmentation of words into characters. The preprocessing steps adopted for the segmentation of a word into its constituent characters are provided in Figure 5-11. The images on the right of each step are the results obtained after applying each step. The final word image with segmented characters is then fed to the zone identification algorithm to find out the zones of the constituent characters. For preprocessing, the first step is binarization using Otsu’s thresholding method [147]. The next step is skew correction. This step is important since the zones of the characters are calculated by taking the first word as the reference. Therefore, it is desired that the word is free from skew. Skew correction is performed using the projection profile method [158].

Next, the bounding box of the word image is calculated by finding out the first and last white pixels row-wise and column-wise. The word image is traversed row-wise to find the first and last white pixels represented by the red and purple dots in Figure 5-12. The column-wise traversal gives the first and last white





**Figure 5-11:** Preprocessing steps for the segmentation of word into its constituent characters.

pixels as shown by the orange and blue dots. Once the coordinates of these four pixels are found, the coordinates of the two yellow dots are calculated. They represent the top left and bottom right coordinates of the word image. Using these coordinates, the bounding rectangle of the word image represented by the green box is found. The image is then size-normalized to height 50 pixels keeping the aspect ratio intact. Dilation is then carried out to take care of broken characters introduced due to poor binarization. The last step is finding the connected components in the dilated image. The connected components are shown by the green



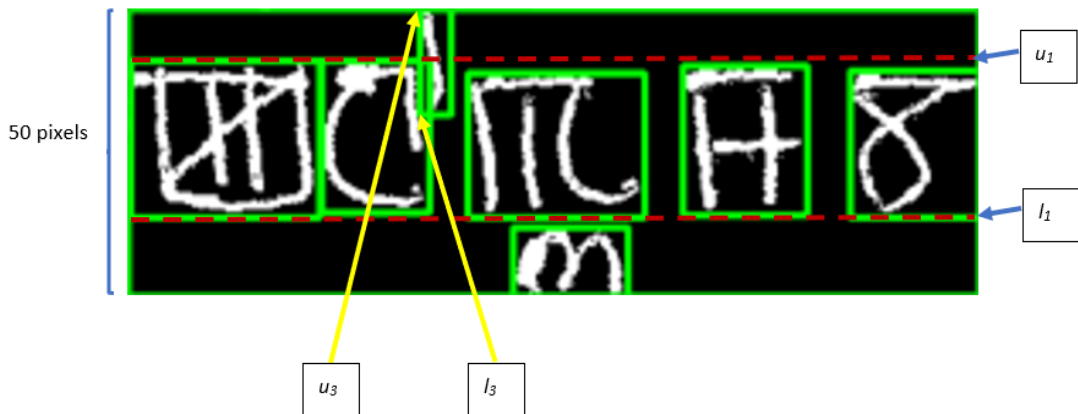
**Figure 5-12:** The row-wise first and last white pixels are depicted by the red and purple dots and those of column-wise are depicted by the orange and blue dots, respectively

#### 5.4. Proposed methodology

bounding boxes in the bottom most image in Figure 5-11. OpenCV's *connected-ComponentsWithStats()* function is used to find the connected components. We discard the connected components whose area is less than or equal to 15 pixels. Then for each of the considered connected component, we use the y coordinates of the uppermost and the lowermost pixels of the bounding box to identify their zones.

The method to find zones based on the y coordinates of the characters is given in Algorithm 4. When a word is written in Meitei Mayek, the first character is always a consonant or a half-consonant or a numeral. This has been statistically found out using the TDIL text corpus where it is seen that the first character in all the 189901 words is one of these three types of characters and all of them lie in the middle zone of a word. Taking the reference of this first character, zones of the rest of the characters are determined. Since there are a lot of variations in natural handwriting, there is a possibility that characters lying in the upper or lower zones tend to cross over to the area of middle zone. This possibility is also taken into account while formulating the algorithm.

In the example given in Figure 5-13, the dotted red lines are the y coordinates of the first character. The  $u_1$  and  $l_1$  represent the uppermost pixel and the lowermost pixel, respectively. These two y coordinates are taken as the references to calculate the zones of the rest of the characters in the word. There are seven characters in the word and therefore there are seven  $u_i$ s and  $l_i$ s, each representing a character. For example,  $u_3$  and  $l_3$  represents the uppermost and lowermost pixels of the third character. For each character  $char_i(i \neq 1)$ , the algorithm checks:



**Figure 5-13:** Zone identification based on the uppermost and lowermost pixel (represented by dotted red lines) of a word image.

---

**Algorithm 4:** Find zones

---

**Input:** Word  $W$

**Output:** A vector  $zone$  containing the zones of the constituent characters,  
 $zone(i)$  is the zone of the  $i^{th}$  character of  $W$

Let  $char_1$  be the first character in  $W$  and let  $u_1$  and  $l_1$  be the y coordinates  
of the uppermost and lowermost pixels of its bounding box respectively

Let  $n =$  number of characters in  $W$

$zone(i) =$  middle ▷ First character is always in the middle zone

**for**  $i \leftarrow 2$  **to**  $n$  **do**

$char_i =$  the  $i^{th}$  character

$u_i =$  y coordinate of the uppermost pixel of  $char_i$

$l_i =$  y coordinate of the lowermost pixel of  $char_i$

**if**  $((u_i - u_1 > 5) \vee (l_i - l_1 > 5)) \wedge (|l_i - u_i| < \frac{3}{4}|l_1 - u_1|)$  **then**

$zone(i) =$  upper

**end**

**else if**  $((u_1 - u_i > 5) \vee (l_1 - l_i > 5)) \wedge (|l_i - u_i| < \frac{3}{4}|l_1 - u_1|)$  **then**

$zone(i) =$  lower

**end**

**else**

$zone(i) =$  middle

**end**

**end**

**return**  $zone$

---

- if the difference between the  $u_1$  and  $u_i$  or the difference between the  $l_1$  and  $l_i$  is greater than 5. It also checks if the height of  $char_i$  is less than  $\frac{3}{4}$  of the height of the first character. If both the conditions are true, then  $char_i$  is in the upper zone.
- Else, if the difference between the  $u_i$  and  $u_1$  or the difference between the  $l_i$  and  $l_1$  is greater than 5 and the height is less than  $\frac{3}{4}$  of the height of the first character, then the character lies in the lower zone.
- Else, the character is in the middle zone.

### 5.4.4 Rules used

The script-specific features of Meitei Mayek (refer Section 5.3) help the system in distinguishing the characters in the confusing character pairs identified earlier. It is described as follows.

1. All confusing pairs except for (ꯃ, ꯃ) are pairs consisting of a non-vowel and a vowel. These pairs of confusing characters can be distinguished by finding the zone of the characters. The fact that vowels always lie either in the upper or lower zone helps the system in identifying the correct class that a particular character belongs to.
2. The confusing pair, *ee* (ꯃ) and *ee-lonsum* (ꯃ) can be distinguished by the orthographic rule which says that *ee-lonsum* (ꯃ) is preceded by a vowel and *ee* (ꯃ) is either the first character of the word or it is not preceded by a vowel. The type of the preceding character is found out using the zonal information. That is, the preceding character is a vowel if the zone is either upper or lower else it is a non-vowel.

## 5.5 Experiments

The experimental setup and results of the proposed methodology are described in the following sections.

### 5.5.1 Dataset details

As described earlier, TUMMHCD is used for training the CNN and for finding the vulnerable set. For the purpose of zone identification and performance evaluation of the proposed approach, a dataset of 100 words is used. The words are written in Meitei Mayek by individuals who know how to write the script naturally. This dataset has been collected by us personally keeping in mind the variations that exist in natural handwriting. For the database creation, some words were randomly chosen from the TDIL corpus. 100 individuals were asked to write these words in a sheet of paper. We have considered 50 individuals and two words from each of these 50 individuals, making a total of 100 words. There are a total of 565 characters. It was observed during the sample collection that most individuals write the characters in a word which are non-touching. This writing constraint

of words with non-touching characters only have been considered for the present work. The word images are scanned at 300 dpi in grayscale format. They are then saved in TIFF image format.

### 5.5.2 Experimental results on zone identification

The zone identification part is of paramount importance in the proposed system. The recognition in the second stage primarily depends on the zone information of the characters. Therefore, we have evaluated the performance of zone identification algorithm proposed in the present work. This has been carried out by comparing the predicted zones against the ground truth values and the results are provided in Table 5.2. The zone identification accuracy of the characters in the vulnerable set is also shown separately in the table.

Table 5.2: Performance results of zone identification algorithm

Number of characters	Correctly identified	Inorrectly identified	Accuracy
565 (total)	533	32	94.34%
101 (vulnerable set)	95	6	94.06%

### 5.5.3 Evaluation metrics

Four evaluation metrics have been employed to evaluate the performance of the proposed system. Since the task at hand has varying number of instances in each class, the weighted average metrics have been considered. They are:

1. Weighted average precision (P): To calculate the weighted average precision, precision for each class is calculated first, and their average is weighted by the number of instances for each class. The equation is:

$$P = \frac{\sum_{i=1}^{55} (P_i * |c_i|)}{\sum_{i=1}^{55} |c_i|} \quad (5.2)$$

where  $P_i$  is the precision for class  $c_i$  and  $|c_i|$  is the number of instances in class  $c_i$ .

2. Weighted average recall (R): To calculate the weighted average recall, recall for each class is calculated first, and their average is weighted by the number of instances for each class. The equation is:

$$R = \frac{\sum_{i=1}^{55} (R_i * |c_i|)}{\sum_{i=1}^{55} |c_i|} \quad (5.3)$$

where  $R_i$  is the recall for class  $c_i$  and  $|c_i|$  is the number of instances in class  $c_i$ .

3. Weighted average F-measure (F): The formula to calculate the weighted average F-measure is given by:

$$F = \frac{\sum_{i=1}^{55} (F_i * |c_i|)}{\sum_{i=1}^{55} |c_i|} \quad (5.4)$$

where  $F_i$  is the F-measure for class  $c_i$  and  $|c_i|$  is the number of instances in class  $c_i$ .

4. Accuracy (A): It is the number of characters correctly predicted divided by the total number of predictions. That is:

$$A = \frac{\#correct}{\#total} \quad (5.5)$$

where  $\#correct$  is the number of correct predictions and  $\#total$  is the total number of predictions

### 5.5.4 Overall experimental results

The output of the system is compared with the ground truth. Performance of the system without the second stage is compared against that of the system with the second stage.

Precision, recall and f-measure along with the total number of instances in each class is given in Table 5.3. The average weighted values and accuracy is given in Figure 5-14. As it can be seen from Table 5.3 that the precision, recall and f-measure values of the character classes 7, 9, 11, 25, 44, 46, 47 and 51 which had been identified as vulnerable show improvement when second stage is adopted. It could be inferred that there are low false positive and low false negative rates from high values of both precision and recall respectively. These values for other character classes remain mostly same as they are not touched by the second stage. An increase in the values of precision, recall and f-measure shows that characters which are identified as vulnerable classes could be correctly recognized when fed to the second stage. This brings about an enhanced system with better

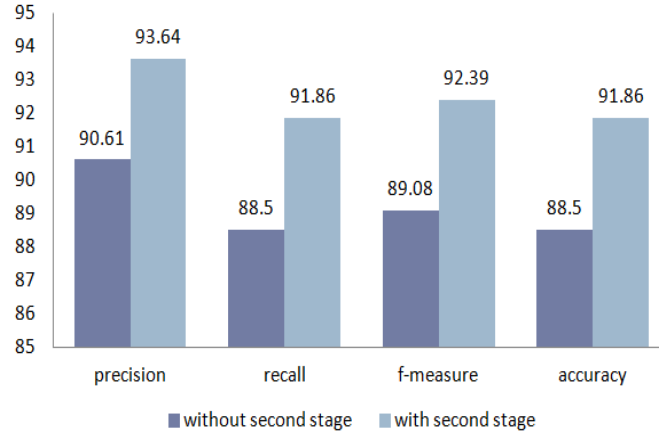
## Chapter 5. Zone and rule assisted recognition of Meitei Mayek handwritten characters

Table 5.3: Total number of instances and values of precision, recall and f-measure of each class obtained with systems with and without the second stage. The highlighted values in bold indicate the improvements.

Character class (Symbol)	Number of instances	Without second stage			With second stage		
		Precision	Recall	F-measure	Precision	Recall	F-measure
0 (ꠄ)	4	0.75	0.75	0.75	0.75	0.75	0.75
1 (ꠅ)	4	0.80	1.00	0.89	0.80	1.00	0.89
2 (꠆)	1	1.00	1.00	1.00	1.00	1.00	1.00
3 (ꠇ)	2	0.33	1.00	0.50	0.33	1.00	0.50
4 (ꠈ)	4	0.75	0.75	0.75	0.75	0.75	0.75
5 (ꠉ)	0	0.00	0.00	0.00	0.00	0.00	0.00
6 (ꠊ)	3	0.75	1.00	0.86	0.75	1.00	0.86
7 (ꠋ)	<b>2</b>	<b>0.33</b>	<b>0.50</b>	<b>0.40</b>	<b>0.40</b>	<b>1.00</b>	<b>0.57</b>
8 (ꠌ)	2	1.00	0.50	0.67	1.00	0.50	0.67
9 (ꠍ)	<b>9</b>	<b>0.73</b>	<b>0.89</b>	<b>0.80</b>	<b>0.89</b>	<b>0.89</b>	<b>0.89</b>
10 (ꠎ)	6	0.86	1.00	0.92	0.86	1.00	0.92
11 (ꠏ)	<b>25</b>	<b>0.85</b>	<b>0.88</b>	<b>0.86</b>	<b>0.96</b>	<b>0.96</b>	<b>0.96</b>
12 (ꠐ)	22	1.00	0.82	0.90	1.00	0.82	0.90
13 (ꠑ)	23	1.00	0.96	0.98	1.00	0.96	0.98
14 (ꠒ)	7	1.00	1.00	1.00	1.00	1.00	1.00
15 (ꠓ)	29	1.00	0.93	0.96	1.00	0.93	0.96
16 (ꠔ)	3	0.67	0.67	0.67	0.67	0.67	0.67
17 (ꠕ)	18	0.94	0.83	0.88	0.94	0.83	0.88
18 (ꠖ)	7	0.71	0.71	0.71	0.71	0.71	0.71
19 (ꠗ)	2	1.00	0.50	0.67	1.00	0.50	0.67
20 (ꠘ)	6	1.00	1.00	1.00	1.00	1.00	1.00
21 (ꠙ)	8	1.00	0.88	0.93	1.00	0.88	0.93
22 (ꠚ)	6	0.83	0.83	0.83	0.83	0.83	0.83
23 (ꠛ)	11	1.00	0.82	0.90	1.00	0.82	0.90
24 (ꠜ)	5	1.00	0.80	0.89	1.00	0.80	0.89
25 (ꠝ)	<b>9</b>	<b>0.29</b>	<b>0.44</b>	<b>0.35</b>	<b>0.75</b>	<b>0.67</b>	<b>0.71</b>
26 (ꠞ)	6	1.00	1.00	1.00	1.00	1.00	1.00
27 (ꠟ)	8	0.88	0.88	0.88	0.88	0.88	0.88
28 (ꠠ)	16	1.00	0.94	0.97	1.00	0.94	0.97
29 (ꠡ)	0	0.00	0.00	0.00	0.00	0.00	0.00
30 (ꠢ)	23	0.96	1.00	0.98	0.96	1.00	0.98
31 (ꠣ)	23	0.96	1.00	0.98	0.96	1.00	0.98
32 (ꠤ)	3	0.67	0.67	0.67	0.67	0.67	0.67
33 (ꠥ)	20	0.94	0.80	0.86	0.94	0.80	0.86
34 (ꠦ)	0	0.00	0.00	0.00	0.00	0.00	0.00
35 (ꠧ)	0	0.00	0.00	0.00	0.00	0.00	0.00
36 (꠨)	0	0.00	0.00	0.00	0.00	0.00	0.00
37 (꠩)	13	1.00	0.92	0.96	1.00	0.92	0.96
38 (꠪)	1	0.50	1.00	0.67	0.50	1.00	0.67
39 (꠫)	8	0.78	0.88	0.82	0.78	0.88	0.82
40 (꠬)	7	0.88	1.00	0.93	0.88	1.00	0.93
41 (꠭)	11	0.91	0.91	0.91	0.91	0.91	0.91
42 (꠮)	6	1.00	1.00	1.00	1.00	1.00	1.00
43 (꠯)	25	0.93	1.00	0.96	0.93	1.00	0.96
44 (꠰)	<b>14</b>	<b>0.62</b>	<b>0.36</b>	<b>0.45</b>	<b>0.86</b>	<b>0.86</b>	<b>0.86</b>
45 (꠱)	30	1.00	0.97	0.98	1.00	0.97	0.98
46 (꠲)	<b>14</b>	<b>0.85</b>	<b>0.79</b>	<b>0.81</b>	<b>0.93</b>	<b>1.00</b>	<b>0.97</b>
47 (꠳)	<b>23</b>	<b>0.86</b>	<b>0.83</b>	<b>0.84</b>	<b>0.96</b>	<b>0.96</b>	<b>0.96</b>
48 (꠴)	54	0.98	0.94	0.96	0.98	0.94	0.96
49 (꠵)	8	0.78	0.88	0.82	0.78	0.88	0.82
50 (꠶)	20	1.00	0.95	0.97	1.00	0.95	0.97
51 (꠷)	<b>7</b>	<b>0.60</b>	<b>0.86</b>	<b>0.71</b>	<b>0.88</b>	<b>1.00</b>	<b>0.98</b>
52 (꠸)	7	0.88	1.00	0.93	0.88	1.00	0.93
53 (꠹)	0	0.00	0.00	0.00	0.00	0.00	0.00
54 (꠺)	0	0.00	0.00	0.00	0.00	0.00	0.00

## 5.5. Experiments

---



**Figure 5-14:** Weighted average precision, recall, f-measure and accuracy values of the system with and without the second stage

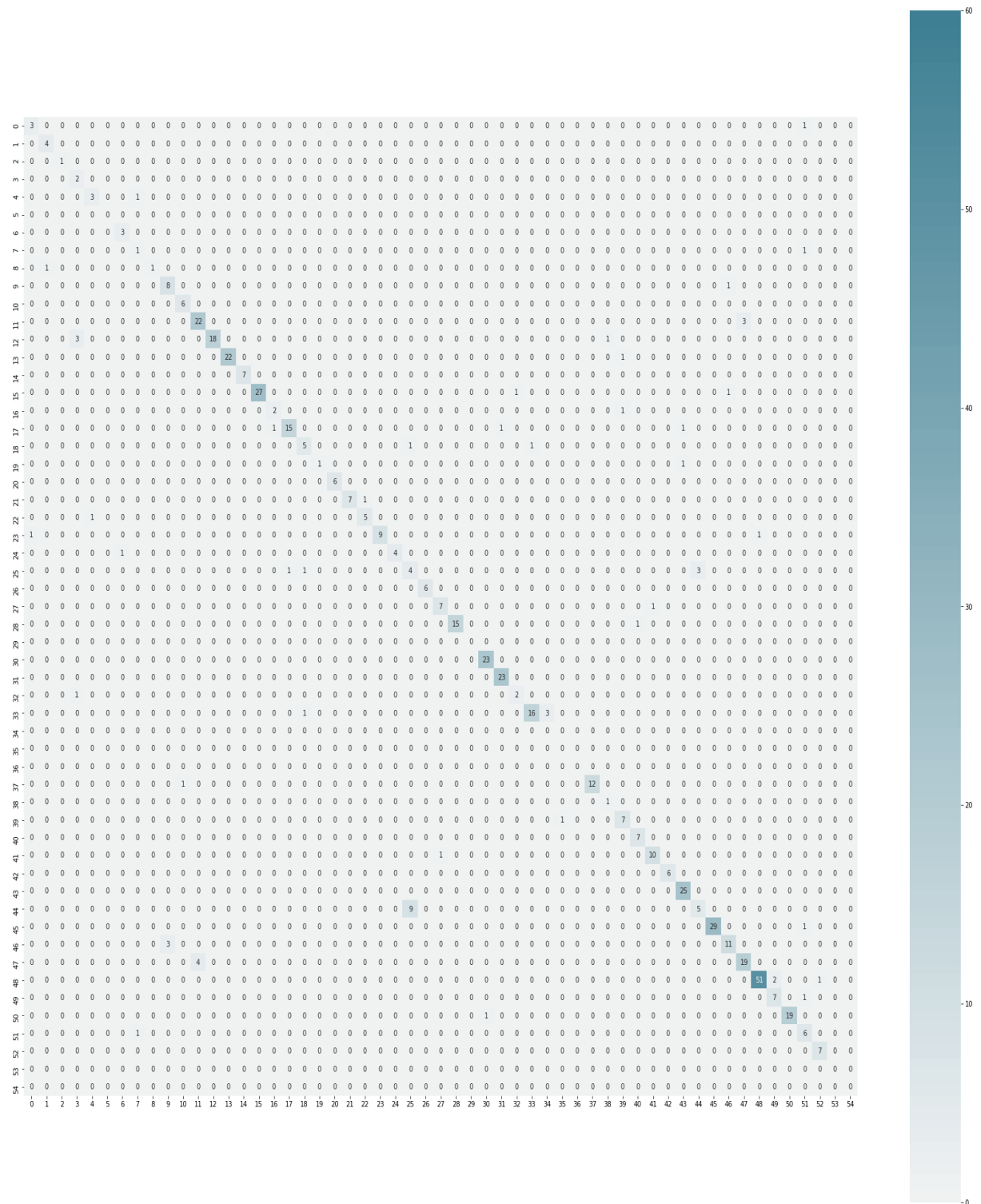
recognition power. Figure 5-14 also shows that the weighted average precision, recall, f-measure and the overall accuracy of the system increase when second stage is incorporated. We could see an improvement of accuracy from 88.495% to 91.858% with the adoption of the second stage. A comparison of the confusion matrix in Figure 5-15 of the system without second stage with the one in Figure 5-16 of the system with second stage reveals the improvement clearly. Most of the non-diagonal elements are zero because for these character classes, recognition accuracy achieved is 100%. Whereas there are some classes with accuracy less than 100% for which non-diagonal elements not equal to zero. For example, for character classes 44 and 47, there are non-diagonal elements with values 9 and 4 respectively. Both diagonal and non-diagonal values are zero for those characters which are not present in the test set.

### 5.5.5 Error analysis

While performing error analysis of the system, it was found that identifying the correct zone for characters is not always successful. This is due to the varying writing styles of different individuals and the crossing over of characters in one zone to another zone. In some cases, this is also because of the irregular size of characters in a word. Although the proposed method could identify most of the character zones correctly, some error in zone identification leads to incorrect identification of characters in the second stage. There are a few cases where second stage also could not recognize the correct class of a character belonging to vulnerable set. This is because of the wrong identification of zone of the characters in zone identification phase. The percentage of such characters whose zones are

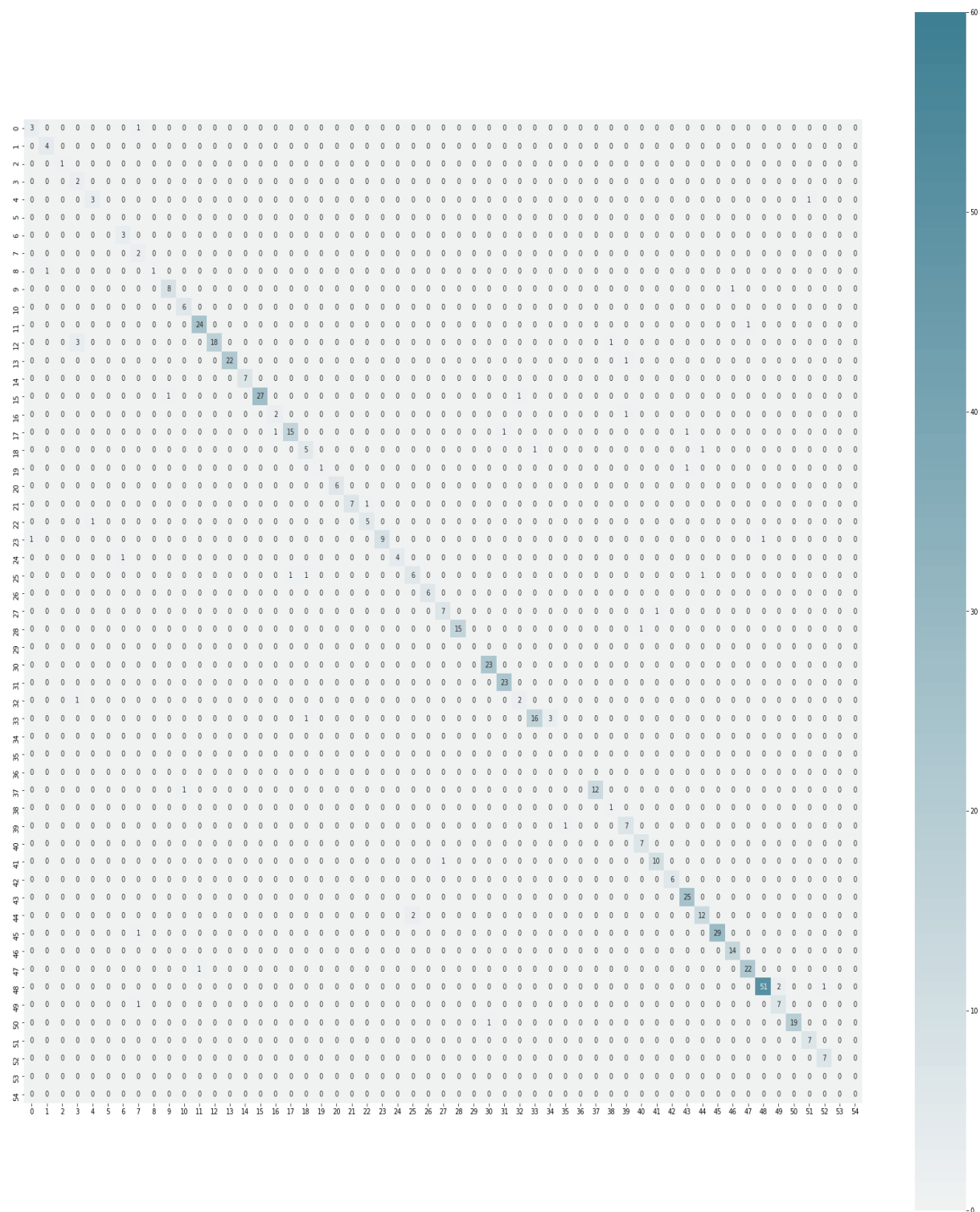


## Chapter 5. Zone and rule assisted recognition of Meitei Mayek handwritten characters



**Figure 5-15:** Confusion matrix obtained with the system without second stage on 100 words database. The x-axis indicates predicted class and the y-axis indicates true class.

## 5.5. Experiments



**Figure 5-16:** Confusion matrix obtained with the system with second stage on 100 words database. The x-axis indicates predicted class and the y-axis indicates true class.

misidentified by the proposed algorithm is 5.66%. There are also certain cases where characters were wrongly classified as belonging to vulnerable set. In those cases again, the second stage could not recognize the correct character class. Out of the total 565 characters, 4 such characters are found in the system. For these four characters, even if the second stage recognition performs correct distinction between the confusing pairs, the final class is still a wrong one. 0.71% of the total number of characters is misclassified as belonging to vulnerable set. This shows that relatively small amount of error is introduced in the zone identification phase and a negligible amount due to misclassification of characters into vulnerable set in the first stage.

### **5.5.6 Comparison with other existing works**

We have performed a comparison of our proposed methodology with other existing methodologies found in literature which adopt more than one stage. Although it is not possible to compare our approach with other works on Meitei Mayek as there are no available work in this particular script. Moreover, there are no word image database available for the same script. We have, however carried out a performance evaluation of our methodology with those found for other scripts. Table 5.4 lists the works with methodologies and other statistics. From the table it is evident that without employing a feature extraction and classification as such in the second stage of recognition, our approach is able to achieve a comparable improvement in the recognition accuracy as those of the state-of-the-art methodologies.

## **5.6 Limitation**

The proposed methodology achieves better accuracy for HCR of Meitei Mayek. This is possible as there are certain script specific properties and orthographic rules which aid in the recognition task. A limitation of the methodology is that those properties cannot be generalized to other scripts. For the proposed methodology to work for other scripts, it is required to look into properties of the concerned scripts which can help achieve a better recognition accuracy. The proposed technique also employs a zone identification technique which sometimes is prone to errors. An error analysis report is given in Section 5.5.5. Also, one constraint considered in this chapter is that words with non-touching characters have been taken for testing.

## 5.7. Conclusion

Table 5.4: Comparison with other methodologies. The last column "Improvement in test accuracy" indicates the improvement achieved by the multi-stage feature extraction/recognition in terms of recognition accuracy as compared to single-stage approach.

Work	Script	No. of classes	Single-stage test accuracy	Multi-stage test accuracy	Improvement in test accuracy
Das et. al.[45]	Bangla complete set	171	76.55	78.93%	2.83%
Wakabayashi et. al.[224]	Similarly shaped numerals and characters of English, Arabic/Persian, Devnagari, Bangla, Kannada, Oriya, Tamil, Telugu	12 confusing pairs	95.29	95.97% (average)	0.68% (average)
Bhattacharya et al.[30]	Bangla basic characters	50	92.78	95.84%	3.06%
John et al.[91]	Malayalam complete set	90	96.43	97.92%	1.49%
Das et al.[46]	Bangla characters	256	84.66	87.26%	2.6%
Vamvakas et al.[221]	CEDAR character database	52	78.42%	85.11%	6.69%
		35	90.7%	94.73%	4.03%
		26	93.78%	95.90%	2.12%
		46	92.53%	95.63%	3.10%
Kayumov et al.[94]	MNIST	10	99.14	99.36%	0.22%
Wang et al.[225]	Chinese (CASIA HWDB1.1 dataset and ICDAR 2013)	368 similar characters	86	89.36%	3.38%
Basu et al.[24]	Bangla basic character	36	78.3	80.58%	2.28%
Present work	Meitei Mayek complete set	55	88.5	91.86%	3.36%

## 5.7 Conclusion

This chapter presents a zone and rule assisted two-stage recognition of Meitei Mayek script. The proposed technique takes care of the confusing character pairs in Meitei Mayek. These character pairs are found to be responsible for low recognition accuracy of Meitei Mayek HCR system. By distinguishing between these character pairs, we could achieve a better recognition accuracy on a dataset consisting of natural handwriting of different individuals. The recognition of characters is assisted by information regarding the zone of character and certain orthographic rules that are followed while writing the script. The proposed system has been evaluated using four evaluation metrics and it has shown significant improvements over the system with single-stage recognition. The second stage recognition does not increase the computational complexity of the overall system as the recognition

## **Chapter 5. Zone and rule assisted recognition of Meitei Mayek handwritten characters**

---

is based on the zone information which is determined by a simple algorithm. Even though the proposed methodology achieves an enhanced accuracy, there are some limitations which have been described in Section 5.6. The next chapter focuses on an approach to overcome some of those limitations.