CHAPTER 2
SOFT SETS BASED INSTANCE SELECTION IN
CHARACTER RECOGNITION

2.1 INTRODUCTION

Knowledge discovery and data mining (KDD) is growing rapidly to meet the challenge of managing ever-growing data due to the extensive use of computers and ease of data collection. Data explosion issue is addressed by algorithm scale-up and/or data reduction.

Most data sets will contain a certain amount of redundancy which motivates the need for data reduction. Instance selection is a way of data reduction. It pertains to methods or algorithms that select or search for a representative portion of data that can fulfill a KDD task as if the whole data is used. It is directly related to data reduction and becomes increasingly important because of processing efficiency and/or storage efficiency. The ideal outcome of instance selection is a model independent, minimum sample of data that can accomplish tasks with little or no performance deterioration i.e. for a given data mining algorithm M, its performance P on a sample 's' of selected instances and on the whole data 'w' is roughly P(M_s) ≈ P(M_w). (Liu & Motoda 2002)
The motivation for instance selection are:

- Data are not purely collected for one application; but an application is about one aspect of a domain.
- There are missing data, redundant data and errors during collecting and recording. It is therefore paramount to clean data to remove irrelevant ones as well as noise and/or redundant data.
- Data can be too overwhelming to handle. An algorithm is limited by its capability to handle data in terms of sizes, types and formats; hence the algorithm cannot be effectively carried without data reduction.

Benefits of Instance selection are:

- It is a way of reduction and hence makes the prediction more clear and visible.
- It is possible and feasible to attempt it now with advanced statistics and accumulated experience.
- It can result in many advantages like less computational overhead and time.
- High quality data will lead to high quality results with reduced costs for computation.

The central point of instance selection is approximation. The task at hand is to achieve as good end results as possible by approximating the whole data with the selected instances. There are many ways of achieving approximation of the original data through instance selection. The methods of
instance selection are categorized in terms of sampling, classification, clustering, and instance labeling. Instance labeling is the method of instance selection explored in this chapter.

Offline character recognition is a PR problem that refers to the classification problem of identifying handwritten characters. As the style and method of writing varies from person to person, the difficulty of identifying the correct character increases. The performance of a character recognition system depends on the selection of features, instances and the classifier. In this chapter, a new algorithm for instance selection is proposed to characterize the instances as good / bad. The key parameters that characterize the instances are obtained using reduct soft sets and an interval estimate is defined for the key parameters for labeling the instances through statistical approach. The labeled instances are then used in the training phase of an MLP and testing is performed to investigate whether soft sets based instance selection contributes towards increased classification accuracy.

2.2 RELATED WORK

Liu & Motoda (2002) specified that feature selection, extraction, construction, and instance selection are ways of data reduction. Instance selection reduces data and enables a data mining algorithm to function and work effectively with huge data. The task is to achieve as good mining results as possible by removing noisy and irrelevant data in the process.

Jankowski & Grochowski (2004) list the reasons for reducing the training set to a smaller one as follows: to reduce the noise in original dataset as some learning algorithms may be noise-fragile, to reduce the amount of computation for lazy learning algorithms, to use vector selection with new prototype selection algorithms. Based on the application-type, the algorithms for selection of instances are divided into noise filters, condensation
algorithms and proto-type searching algorithms. Moreover, such algorithms work in different ways either incremental, decremental or a mixture of both. They also compared the algorithms on the datasets from the UCI machine learning repository and presented the results (Grochowski & Jankowski 2004).

Olvera-Lopez et al (2010) have outlined that instance selection is a process of selecting useful instances in a training set to be provided to a supervised classification algorithm and ignoring the non useful ones thereby reducing runtimes in the classification and / or training stages of a classifier. They review the instance selection methods based on two groups: Wrapper methods where the selection criterion is based on the accuracy obtained by a classifier and Filter methods where the selection criterion uses a selection function which is not based on a classifier. The following procedure is followed in Wrapper based methods: Let T be a training set and \( S \subseteq T \). Each instance in T is classified using S as training set; if an instance ‘p’ is misclassified, then ‘p’ is included in S to ensure that new instances similar to ‘p’ will be classified correctly. Due to this criterion, noisy instances are retained.

Chaudhuri & Bhattacharya (2000) demonstrate that an intelligent selection of training patterns can considerably improve the speed and efficiency of the training of the multi-layer perceptron (MLP) network. They propose a technique, which at first cleverly picks up samples near the decision boundary without actually knowing the position of decision boundary. A training set, generated using this idea, results in quick and better convergence of the training algorithm.
2.3 PROPOSED METHODOLOGY

The character recognition system is tested on two different data sets. One is a subset of five vowels of a south Indian language, Tamil, where each character image is divided into sections and the number of black pixels within each section is used to calculate the normalized distance and angle measures of the image to be used as features. The features are extracted for all the images and there are five classes equivalent to the five vowels used. Second is a subset of ten capital letters from English letter database, but the features of statistical moments and edge counts are extracted from the image and available in the data base. There are ten classes equivalent to the ten alphabets used. A multi-layer perceptron (MLP) is used as the classifier.

For each character, the data is divided into training and testing test. The training set is further divided into reference and to_label set. The characterization algorithm uses the reference set to compute feature mean values, partition into good and bad sets and produce parameterized reducts for each of them. The labeling algorithm labels the instances of the to_label set based on interval estimate and the parameterized reducts obtained above. The labeled instances are used in the training phase of the MLP to study if there is any impact in increase of accuracy in the testing phase. Hence the following methodology is proposed:

- Specific features from the character images are used in feature extraction,
- Reduct Soft sets based characterization algorithm is applied to label the instances and
- A Multi-layer Perceptron (MLP) is used for verification, where the labeled instances in different proportion are used in the training phase.
The raw character image undergoes preprocessing, feature extraction, instance selection and then verification stages.

2.3.1 Data Sets Used

The proposed method is tested on two different data sets, a smaller subset of hand printed characters of Tamil and a bigger subset of English capital letters.

Tamil is a classical South Indian language spoken by a segment of the population in Singapore, Malaysia, and Sri Lanka apart from India. The Tamil alphabet consists of 247 letters (consonant, vowels and consonant vowel combinations). Each letter is represented either by a separate symbol or as a combination of discrete symbols. Samples of each of these letters form a separate class. Only vowels are shown in Figure 2.1. The experiments are conducted on a subset of Tamil vowels shown in Figure 2.2 whereas Figure 2.3 and Figure 2.4 show sample set of images of the vowel ‘aa’ and ‘uu’.

<table>
<thead>
<tr>
<th>Tamil</th>
<th>English</th>
</tr>
</thead>
<tbody>
<tr>
<td>ஆ</td>
<td>a</td>
</tr>
<tr>
<td>ஐ</td>
<td>aa</td>
</tr>
<tr>
<td>ྡ</td>
<td>e</td>
</tr>
<tr>
<td>ྨ</td>
<td>ee</td>
</tr>
<tr>
<td>ஐ</td>
<td>o</td>
</tr>
<tr>
<td>ற</td>
<td>oo</td>
</tr>
<tr>
<td>ஆ</td>
<td>au</td>
</tr>
<tr>
<td>க</td>
<td>ak</td>
</tr>
</tbody>
</table>

Figure 2.1 Tamil Vowels
Figure 2.2 Vowels Considered

Figure 2.3 Sample Images of ‘aa’

Figure 2.4 Sample Images of ‘uu’
Ten alphabets A to J of English have been taken. Each character image has 16 primitive numerical attributes (statistical moments and edge counts) and these are used as features.

2.3.2 Preprocessing

The preprocessing stage involves skeletonization, boundary detection and size normalization for the input Tamil character images and they have been scaled and converted to a standard size of 128 x 128.

No preprocessing is required for English alphabets as the data are directly stored in the dataset.

2.3.3 Feature Extraction

The feature extraction stage uses normalized distance and angle measures from the Tamil character image for achieving high recognition rates as suggested by Chanda & Majumder (2004). The image is divided into 16 sections and the portions lying in each section is used for extraction of features. The normalized distance and angle measures are calculated for all the sections. Let \( n_s \) be the number of black pixels present in a section \( s \), with \( s=1, 2, 3 \ldots 16 \). For each section, the normalized vector distance is the sum of distances of all black pixels in a section divided by the number of black pixels present in that section. The normalized distance and angle are calculated as

\[
d_s = \frac{1}{n_s} \sum_{i=1}^{n_s} \sqrt{(x_m - x_i)^2 + (y_m - y_i)^2} \tag{2.1}
\]

and

\[
a_s = \frac{1}{n_s} \sum_{i=1}^{n_s} \tan^{-1} \left( \frac{y_m - y_i}{x_m - x_i} \right) \tag{2.2}
\]
where \((x_i, y_i)\) are the co-ordinates of a pixel in a section and \((x_m, y_m)\) are the co-ordinates of the center of the character image. This gives one normalized vector distance for each section and totally sixteen values for each character. Thus 32 features (distance and angle measures) for every character image are extracted.

Sixteen features are available from the database for the English alphabets and are directly used.

The performance of the classifier depends on preprocessing, feature extraction, feature subset selection and classification algorithm. This chapter focuses on the instance selection methodology.

2.3.4 Instance Selection

The proposed method focuses on instance selection through soft sets for a character recognition application. The method proposes to label instances as good or bad to be suitably used as training set in a classification algorithm. A two-pronged approach is followed: Instances without noise which are classified correctly are utilized; Moreover, those noisy instances which also affect the classification accuracy of the algorithm are also retained. They are extracted from the full training set on a reduct soft sets based approach. The validation of the proposed instance selection methodology is done on a MLP using backpropagation learning. The instances are characterized as good or bad and the influence of selected instances based on such characterization over the generalization capability of the MLP is studied.

Soft sets

several operations and their properties on soft sets. Moreover, Maji et al (2002) developed fuzzy soft sets and characterized fuzzy soft sets by introducing some basic operations. Chen et al (2005) proposed a new definition of parameter reduction for soft sets and have used it to improve the application of soft sets in a decision making problem. Also Roy & Maji (2007) developed a fuzzy soft set theoretic approach to decision making problems.

The basic concepts regarding soft sets used in instance selection are defined as follows:

**Definition 2.1 : Soft Set**

Let U be an initial universal set and let E be a set of parameters. A pair (F,E) is called a soft set (over U) if and only if F is a mapping of E into the set of all subsets of the set U.

The soft set is a parameterized family of subsets of the set U. Every set $F(\epsilon)$, for $\epsilon \in E$, from this family may be considered as the set of $\epsilon$ — *elements* of the soft set (F, E), or as the set of $\epsilon$ — *approximate* elements of the soft set.

**Definition 2.2 : Knowledge Representation System**

A knowledge representation system can be formulated as a pair $S = (U,A)$ where A is nonempty finite set of primitive attributes.

Every primitive attribute of A is a total function $a : U \rightarrow V_a$, where $V_a$ is the set of values of a, called the range of a. Attribute, feature and parameter are synonymously used.
Definition 2.3: Indiscernibility Relation

With every subset of attributes, $B \subseteq A$, we associate a binary relation $IND(B)$, called an indiscernibility relation, defined by $IND(B) = \{(x, y) \in U \times U : \forall a \in B, a(x) = a(y)\}$. Obviously, $IND(B)$ is an equivalence relation and $IND(B) = \bigcap_{a \in B} IND(a)$.

Definition 2.4: Dispensable, Indispensable and reduct

Let $R$ be a family of equivalence relations and let $A \in R$. $A$ is dispensable in $R$ if $IND(R) = IND(R - \{A\})$; otherwise $A$ is indispensable in $R$. The family $R$ is independent, if each $A \in R$ is indispensable in $R$; otherwise $R$ is dependent. If $R$ is independent and $P \subseteq R$, then $P$ is also independent. $Q \subseteq P$ is a reduct of $P$, if $Q$ is independent and $IND(Q) = IND(P)$, that is to say that $Q$ is the minimal subset of $P$ that keeps the classification ability.

Definition 2.5: Optimal Choice object

The choice value of an object $h_i \in U$ is $c_i = \sum h_{ij}$, where $h_{ij}$ are the entries in the table of the reduct soft set. Find $k$, for which $c_k = \max c_i$. Then $h_k$ is the optimal choice object.

Definition 2.6: Attribute reduct and Parameterized reduct

A reduct $R$ of $E$, the set of attributes, is an attribute reduct of $E$, if the optimal choice object changes for $R$ from that of $E$. If the optimal choice object does not change, then it is a parameterized reduct of $E$.

Illustrative Example

Let $U = \{h_1, h_2, h_3, h_4, h_5, h_6\}$ be the universe of houses and $E = \{e_1, e_2, e_3, e_4, e_5, e_6, e_7\}$ be a set of parameters where
Suppose $F(e_1) = \{h_1, h_4, h_5\}$

$F(e_2) = \{h_2, h_6\}$

$F(e_3) = \{h_1, h_2, h_3, h_4, h_5, h_6\}$

$F(e_4) = \{h_1, h_2, h_4, h_6\}$

$F(e_5) = \{h_1, h_3, h_6\}$

$F(e_6) = \{h_2, h_5\}$

$F(e_7) = \{h_2, h_3\}$

Soft set $(F,E)$ is a collection of approximations and tabulated in Table 2.1

**Table 2.1 Soft Set Representation of Houses**

<table>
<thead>
<tr>
<th></th>
<th>U</th>
<th>e1</th>
<th>e2</th>
<th>e3</th>
<th>e4</th>
<th>e5</th>
<th>e6</th>
<th>e7</th>
<th>Max(E)</th>
</tr>
</thead>
<tbody>
<tr>
<td>h1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>h2</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>h3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>h4</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>h5</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>h6</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>4</td>
</tr>
</tbody>
</table>

$R_E = \text{Indiscernibility relation induced by E;}$ The optimal choice object is $h_2$ for soft set $(F,E)$.

Partition induced by $R_E = \{\{h_1\}, \{h_2\}, \{h_3\}, \{h_4\}, \{h_5\}, \{h_6\}\}$

Partition induced by $\{e_1, e_2, e_4, e_5, e_6, e_7\} = \{\{h_1\}, \{h_2\}, \{h_3\}, \{h_4\}, \{h_5\}, \{h_6\}\}$;

Partition induced by $\{e_1, e_4, e_5\} = \{\{h_1\}, \{h_2\}, \{h_3\}, \{h_4\}, \{h_5\}, \{h_6\}\}$;

Partition induced by $\{e_2, e_4, e_5\} = \{\{h_1\}, \{h_2\}, \{h_3\}, \{h_4\}, \{h_5\}, \{h_6\}\}$;
Partition induced by \( \{e4, e5\} = \{\{h1, h6\}, \{h2, h4\}, \{h3\}, \{h5\}\}; \)

Partition is invariant for \( \{e1, e2, e4, e5, e6, e7\}, \{e1, e4, e5\} \) and \( \{e2, e4, e5\} \) to that of \( R_E \). Hence, \( \{e1, e2, e4, e5, e6, e7\}, \{e1, e4, e5\} \) and \( \{e2, e4, e5\} \) are reducts of \( (F, E) \). The optimal choice object is \( h_1 \) for \( \{e1, e4, e5\} \) and \( h_6 \) for \( \{e2, e4, e5\} \). Since the optimal choice object changes, they are attribute reducts and not parameterized reducts. But the optimal choice object does not change for \( \{e1, e2, e4, e5, e6, e7\} \), hence it is a parameterized reduct. \( \{e4, e5\} \) is not a reduct since the partition varies.

**Reduct Soft Sets based instance selection**

A subset of instances of a character of the training set is taken as the reference set. The following methodology is proposed for soft sets based instance selection:

- a feature matrix for all instances in the reference set (ref_set) is constructed.

- two matrices: good_inst, bad_inst along with the key parameters are obtained by applying instance characterization algorithm (Algorithm 1) on ref_set.

- a feature matrix for all instances in to_label set is constructed.

- instance labeling algorithm (Algorithm 2) is applied to all instances in the to_label set

Algorithm 1 that characterizes the instances is specified in Figure 2.5 and Algorithm 2 that labels the instances is specified in Figure 2.6.
Algorithm 1: Instance Characterization in a training set

**Input:** Feature matrix \([F]\) of a particular character

**Output:** \(\text{ref\_means}(M)\), Set of good \((G)\) and bad instances \((B)\)

1: \(n \leftarrow \) number of instances in \(F\);
2: \(m \leftarrow \) number of features in \(F\);
3: **for** each feature in the feature matrix \([F]\)
   4: \(M_j \leftarrow \frac{\sum F_{ij}}{n};\)
   5: soft matrix \(S_{ij} = \begin{cases} 1, & \text{if } F_{ij} \geq M_j \\ 0, & \text{otherwise} \end{cases}\)
6: **end for**
7: \(t \leftarrow \lfloor m/2 \rfloor\)
8: **for** each instance in \(S\)
   9: \(N_i \leftarrow \sum_{j=1}^{m} S_{ij}\)
10: **if** \(N_i > t\),
   11: insert into \(G\); // it is a good instance
   12: **else**
   13: insert into \(B\); // it is a bad instance
14: **end if**
15: **end for**

**Figure 2.5 Instance Characterization Algorithm**

While executing the labeling algorithm, the following exceptional cases may arise:

- If step 12 labels the instance as good but \(C < p/2\), then the instance is labeled as bad.
- If step 12 labels the instance as bad, but \(C > p/2\), then the instance is labeled as good.
The confidence interval can be decided depending on the nature of accuracy required by the application.

**Algorithm 2: Instance Labeling Algorithm**

**Input:** feature matrix of to_label set [L], ref_means(M)

**Output:** label Vector for instances in L

1: Find a reduct soft set (G,Q) for G
2: \( p \leftarrow \) number of parameters of Q
3: For each parameter in Q obtain the confidence interval \( (\mu_i \pm 2\sigma_i) \), \( i=1,2,\ldots,p \)
4: Find a reduct soft set (B,R) for B
5: \( s \leftarrow \) number of parameters of R
6: For each parameter in R obtain the confidence interval \( (\mu_i \pm 2\sigma_i) \), \( i=1,2,\ldots,s \)
7: \( m \leftarrow \) number of features in L
8: \( t \leftarrow \lfloor m/2 \rfloor \)
9: for each instance in L
10: compute \( [S_{ij}] \) where \( S_{ij} = \begin{cases} 1, & \text{if } R_{ij} \geq M_j \\ 0, & \text{otherwise} \end{cases} \)
11: \( N_i \leftarrow \sum_{j=1}^{m} S_{ij} \)
12: If \( N_i > t \) // looks like a good instance
13: Identify the ‘p’ key parameters
14: \( C \leftarrow \) count of parameters in Q that lies within confidence interval
15: If $C > p/2$  // yes
16:     label $\leftarrow G$
17: else  // no
18:     label $\leftarrow B$
19: end if
20: else
21:     Identify the ‘s” key parameters
22:     $C \leftarrow$ count of parameters in $R$ that lies within confidence interval
23: If $C > s/2$
24:     label $\leftarrow B$
25: else
26:     label $\leftarrow G$
27: end if
28: end if
29: end for

Figure 2.6 Instance Labeling Algorithm

2.3.5 Verification

Guozhong (1996) has proved that input noise is found to be effective in improving the generalization performance for classification problems.

Garcia et al (2006) identified that when the data to an ANN are imbalanced, it damages the performance of the classifier. They filtered the
atypical or noisy patterns of the majority-class keeping all samples of the minority-class and showed an enhancement in the classification accuracy.

The proposed methodology uses a MLP for classification of the Tamil vowels. Instances are selected based on reduct soft sets and input noise has been added to improve the generalization performance of the classifier.

2.4 EXPERIMENTAL RESULTS AND DISCUSSION

The data of the Tamil vowels are taken from the HP labs website (HP Labs 2009). The data set contains approximately 300 isolated samples, each of 156 Tamil “characters” written by native Tamil writers including school children, university graduates and adults from the states of Karnataka and Tamil Nadu, India. The data was collected using HP Tablet PCs and in the standard UNIPEN format. This is a linguistic resource for online Tamil handwritten character recognition. An offline version of the data is available in the form of bi-level TIFF images, generated from the online data using simple piecewise linear interpolation with a constant thickening factor applied.

The data for the English alphabets A to J are taken from UCI machine learning repository (Bache & Lichman 2013). The character images were based on 20 different fonts and each letter within these 20 fonts were randomly distorted to produce a file of 20,000 unique stimuli. Each stimulus was converted into 16 primitive numerical attributes (statistical moments and edge counts) which were then scaled to fit into a range of integer values from 0 through 15. There are around 20000 instances for the 26 alphabets of which 2000 instances each from A to J was used.

The computational experiments are conducted to investigate whether the reduct soft sets based instance selection contributes towards
increasing classification accuracy. As more parameters lead to computational complexity, parameter reduction becomes essential. Hence reduct soft sets have been used in instance selection. Further, there are no techniques available to characterize the instances.

2.4.1 Data Preparation

The algorithm selects a subset of instances as a training set. In order to study the effect of the proposed algorithm on instance selection, three different data sub sets are grouped and subjected to the classifier.

For the Tamil vowels, data set A consists of 250 instances, of which 150 are used for training. Similarly, data set B consists of 200 instances, of which 100 are used for training and data set C consists of 100 instances of which 50 are used for training. Test sets of data set A consists of the 150 training instances, the 100 non-trained instances, and 250 instances taking both together. Test sets of data set B consists of the 100 training instances, the 100 non-trained instances, and 200 instances taking both together. Test sets of data set C consists of the 50 training instances, the 50 non-trained instances, and 100 instances taking both together.

Similarly, for the English alphabets, data set A1 consists of 2000 instances, of which 1200 are used for training. Data set B1 consists of 1600 instances, of which 800 are used for training and data set C1 consists of 1000 instances, of which 500 are used for training. Test sets of data set A1 consists of the 1200 training instances, the 800 non-trained instances, and 2000 instances taking both together. Test sets of data set B1 consists of the 800 training instances, the 800 non-trained instances, and 1600 instances taking both together. Test sets of data set C1 consists of the 500 training instances, the 500 non-trained instances, and 1000 instances taking both together.
2.4.2 Performance Analysis

A subset of training set is used as the reference set (ref_set) and the remaining instances are used as the to_label set. The algorithms proposed characterize the instances as good and bad and identify the key parameters for the respective category. By identifying the nature of the instances it is not necessary to check all the parameters. The algorithms are written in MATLAB running on Intel Core 2 Quad CPU @ 2.66 GHz PC with 2GB of RAM.

Tamil Vowels Data Set

The architecture of the MLP for Tamil vowels consists of 32 input nodes, 25 hidden nodes and 5 output nodes where the bit is set on the corresponding position of the recognized alphabet. The values of the parameters chosen are 0.95 for momentum constant and 0.6 for learning rate. So, the MLP was implemented without noise to input patterns and with noise added to the training patterns and the results tabulated.

The instances are subjected to characterization and labeling algorithms. Table 2.2 shows the nature of the instances obtained to the Tamil vowels for a subset of 50 instances. Three test sets are considered in order to understand the applicability of the proposed algorithm under different situations. The training of the network has been continued till the system error on the training set becomes less than 0.001. After each training set trains the MLP, the test patterns under the three sets are subjected to classification for 10 times and the average misclassification error is computed. This allows us to obtain a reliable estimate of the accuracy.

Table 2.3 specifies the accuracy of the MLP for data set A. Figure 2.7 depicts the accuracy of dataset A with noise.
Table 2.2 Characterization of the instances

<table>
<thead>
<tr>
<th>Vowel</th>
<th>Good</th>
<th>Bad</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>aa</td>
<td>34</td>
<td>16</td>
<td>50</td>
</tr>
<tr>
<td>ee</td>
<td>34</td>
<td>16</td>
<td>50</td>
</tr>
<tr>
<td>u</td>
<td>24</td>
<td>26</td>
<td>50</td>
</tr>
<tr>
<td>uu</td>
<td>33</td>
<td>17</td>
<td>50</td>
</tr>
<tr>
<td>ak</td>
<td>29</td>
<td>21</td>
<td>50</td>
</tr>
</tbody>
</table>

Table 2.3 Accuracy for Data Set A

<table>
<thead>
<tr>
<th>Test Set A</th>
<th>with random training instances</th>
<th>with Soft Set Good : Bad (60 : 40) %</th>
<th>with Soft Set Good : Bad (50 : 50) %</th>
<th>with Soft Set Good : Bad (40 : 60) %</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of Instances</td>
<td>W/o noise</td>
<td>With noise</td>
<td>W/o noise</td>
<td>With noise</td>
</tr>
<tr>
<td>150(trg)</td>
<td>93</td>
<td>99</td>
<td>94</td>
<td>99</td>
</tr>
<tr>
<td>100</td>
<td>84</td>
<td>88</td>
<td>84</td>
<td>88</td>
</tr>
<tr>
<td>250</td>
<td>90</td>
<td>95</td>
<td>90</td>
<td>95</td>
</tr>
</tbody>
</table>

Similarly, Table 2.4 specifies the accuracy of the Neural Network algorithm for data set B and Figure 2.8, its corresponding graph.
Figure 2.7 Accuracy of Data Set A (with noise)

Table 2.4 Accuracy for Data Set B

<table>
<thead>
<tr>
<th>Test Set B</th>
<th>with random training instances</th>
<th>with Soft Set Good : Bad (60 : 40) %</th>
<th>with Soft Set Good : Bad (50 : 50) %</th>
<th>with Soft Set Good : Bad (40 : 60) %</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>No. of Instances</td>
<td>W / o noise</td>
<td>W / o noise</td>
<td>W / o noise</td>
</tr>
<tr>
<td>100(trg)</td>
<td>93</td>
<td>94</td>
<td>92</td>
<td>98</td>
</tr>
<tr>
<td>100</td>
<td>79</td>
<td>80</td>
<td>79</td>
<td>81</td>
</tr>
<tr>
<td>200</td>
<td>88</td>
<td>87</td>
<td>86</td>
<td>89</td>
</tr>
</tbody>
</table>

Table 2.5 specifies the accuracy of the Neural Network algorithm for data set C and Figure 2.9 its corresponding graph.
Figure 2.8 Accuracy of Data Set B (with noise)

Table 2.5 Accuracy for Data Set C

<table>
<thead>
<tr>
<th>Test Set C</th>
<th>with random training instances</th>
<th>with Soft Set Good : Bad (60 : 40) %</th>
<th>with Soft Set Good : Bad (50 : 50) %</th>
<th>with Soft Set Good : Bad (40 : 60) %</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of Instances</td>
<td>W / o noise</td>
<td>With noise</td>
<td>W / o noise</td>
<td>With noise</td>
</tr>
<tr>
<td>50(trg)</td>
<td>85</td>
<td>99</td>
<td>84</td>
<td>94</td>
</tr>
<tr>
<td>50</td>
<td>73</td>
<td>81</td>
<td>61</td>
<td>68</td>
</tr>
<tr>
<td>100</td>
<td>77</td>
<td>88</td>
<td>73</td>
<td>81</td>
</tr>
</tbody>
</table>

In all the three cases, the testing was carried out on the trained instances alone, test instances alone and a mixture of trained and test
instances together. Such a method of testing allows one to discuss on the final results with more clarity.

Figure 2.9 Accuracy of Data Set C (with noise)

The accuracy measures in \((p, q, r)\) represent ‘p’ as the accuracy with 60% good and 40% bad instances, ‘q’ as the accuracy with 50% good and 50% bad instances, and ‘r’ as the accuracy with 40% good and 60% bad instances. The accuracy of the MLP for the data set A (250 instances) using soft sets is comparable (95, 94, 93) against (95) obtained when the instances are selected in a random manner. But for the data set B (200 instances), the accuracy (89, 91, 93) exceeds the accuracy (89) obtained when the instances are selected at random. And for the data set C (100 instances), the accuracy (81, 81, 88) is again comparable with the accuracy (88) obtained with random instance selection.

It can be seen that when the training set is reduced from 250 to 200, the proposed algorithm shows an appreciable increase in accuracy, but it goes down when it is further reduced to 100 instances. As the convergence of the
MLP depends on the size of the training set, the speed is greatly improved by this approach compared to the conventional approach where the training patterns are chosen at random.

The threshold value of ‘t’ used in Algorithm 1 and Algorithm 2 was modified from (0.5*m) to (0.4*m), to study its impact. As more number of instances is labeled as good, required number of bad instances is not available with the limited data set and hence experiments could not be conducted. The case is the reverse when ‘t’ is modified to (0.6*m)

**English Letter Data Set**

The architecture of the MLP for English alphabets is 16 input nodes, 25 hidden nodes and 10 output nodes where the bit is set on the corresponding position of the recognized alphabet. The values of the parameters, 0.95 for momentum and 0.6 for learning rate remain the same. As above, the MLP was implemented without noise to input patterns and with noise added to the training patterns and the results tabulated. Tables 2.6, 2.7 and 2.8 show the accuracy for data set A1, B1 and C1 with t=0.5*m.

**Table 2.6 Accuracy for Data Set A1 with t=0.5*m**

<table>
<thead>
<tr>
<th>Test Set A1</th>
<th>with random training instances</th>
<th>with Soft Set Good : Bad (60:40) %</th>
<th>with Soft Set Good : Bad (50:50) %</th>
<th>with Soft Set Good : Bad (40:60) %</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of Instances</td>
<td>W / o noise</td>
<td>With noise</td>
<td>W / o noise</td>
<td>With noise</td>
</tr>
<tr>
<td>1200(trg)</td>
<td>86</td>
<td>93</td>
<td>88</td>
<td>93</td>
</tr>
<tr>
<td>800</td>
<td>83</td>
<td>85</td>
<td>86</td>
<td>87</td>
</tr>
<tr>
<td>2000</td>
<td>85</td>
<td>90</td>
<td>88</td>
<td>91</td>
</tr>
</tbody>
</table>
Table 2.7 Accuracy for Data Set B1 with t=0.5*m

<table>
<thead>
<tr>
<th>Test Set B</th>
<th>with random training instances</th>
<th>with Soft Set Good : Bad (60 : 40) %</th>
<th>with Soft Set Good : Bad (50 : 50) %</th>
<th>with Soft Set Good : Bad (40 : 60) %</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of Instances</td>
<td>W / o noise</td>
<td>With noise</td>
<td>W / o noise</td>
<td>With noise</td>
</tr>
<tr>
<td>800(trg)</td>
<td>89</td>
<td>96</td>
<td>87</td>
<td>97</td>
</tr>
<tr>
<td>800</td>
<td>79</td>
<td>82</td>
<td>76</td>
<td>80</td>
</tr>
<tr>
<td>1600</td>
<td>86</td>
<td>88</td>
<td>82</td>
<td>88</td>
</tr>
</tbody>
</table>

Table 2.8 Accuracy for Data Set C1 with t=0.5*m

<table>
<thead>
<tr>
<th>Test Set C1</th>
<th>with random training instances</th>
<th>with Soft Set Good : Bad (60 : 40) %</th>
<th>with Soft Set Good : Bad (50 : 50) %</th>
<th>with Soft Set Good : Bad (40 : 60) %</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of Instances</td>
<td>W / o noise</td>
<td>With noise</td>
<td>W / o noise</td>
<td>With noise</td>
</tr>
<tr>
<td>500(trg)</td>
<td>88</td>
<td>93</td>
<td>83</td>
<td>96</td>
</tr>
<tr>
<td>500</td>
<td>76</td>
<td>79</td>
<td>62</td>
<td>66</td>
</tr>
<tr>
<td>1000</td>
<td>82</td>
<td>86</td>
<td>72</td>
<td>82</td>
</tr>
</tbody>
</table>

The accuracy of the MLP with t=0.5*m, for the data set A1 (2000 instances) using soft sets (91, 91, 90) is comparable against (90) obtained when the instances are selected in a random manner. Similarly, for the data set B1 (1600 instances), the accuracy (88, 86, 86) is comparable against (88) obtained when the instances are selected at random. But for the data set C1
(1000 instances), the accuracy (82, 74, 82) is a little less compared to (88) obtained with random instance selection.

Table 2.9 Accuracy for Data Set B1 with t=0.4*m

<table>
<thead>
<tr>
<th>Test Set B1</th>
<th>with random training instances</th>
<th>with Soft Set Good : Bad (60 : 40) %</th>
<th>with Soft Set Good : Bad (50 : 50) %</th>
<th>with Soft Set Good : Bad (40 : 60) %</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of Instances</td>
<td>W / o noise</td>
<td>With noise</td>
<td>W / o noise</td>
<td>With noise</td>
</tr>
<tr>
<td>800(trg)</td>
<td>89</td>
<td>96</td>
<td>89</td>
<td>98</td>
</tr>
<tr>
<td>800</td>
<td>79</td>
<td>82</td>
<td>76</td>
<td>79</td>
</tr>
<tr>
<td>1600</td>
<td>86</td>
<td>88</td>
<td>84</td>
<td>87</td>
</tr>
</tbody>
</table>

Table 2.10 Accuracy for Data Set C1 with t=0.4*m

<table>
<thead>
<tr>
<th>Test Set C1</th>
<th>with random training instances</th>
<th>with Soft Set Good : Bad (60 : 40) %</th>
<th>with Soft Set Good : Bad (50 : 50) %</th>
<th>with Soft Set Good : Bad (40 : 60) %</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of Instances</td>
<td>W / o noise</td>
<td>With noise</td>
<td>W / o noise</td>
<td>With noise</td>
</tr>
<tr>
<td>500(trg)</td>
<td>88</td>
<td>93</td>
<td>92</td>
<td>98</td>
</tr>
<tr>
<td>500</td>
<td>76</td>
<td>79</td>
<td>66</td>
<td>70</td>
</tr>
<tr>
<td>1000</td>
<td>82</td>
<td>86</td>
<td>81</td>
<td>82</td>
</tr>
</tbody>
</table>

As more instances are labeled as good with t=0.4*m, required bad instances are not available for validation. The accuracy of the MLP with t=0.4*m, for the data set B1 (1600 instances), (87, 87, 86) is comparable against (88) obtained when the instances are selected at random. But for the
data set C1 (1000 instances), the accuracy (82, 83, 80) is a little less compared to (88) obtained with random instance selection.

Table 2.11 Accuracy for Data Set B1 with \( t=0.6*m \)

<table>
<thead>
<tr>
<th>Test Set B1</th>
<th>with random training instances</th>
<th>with Soft Set Good : Bad (60 : 40) %</th>
<th>with Soft Set Good : Bad (50 : 50) %</th>
<th>with Soft Set Good : Bad (40 : 60) %</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of Instances</td>
<td>W / o noise With noise</td>
<td>W / o noise With noise</td>
<td>W / o noise With noise</td>
<td>W / o noise With noise</td>
</tr>
<tr>
<td>800(trg)</td>
<td>89</td>
<td>96</td>
<td>91</td>
<td>95</td>
</tr>
<tr>
<td>800</td>
<td>79</td>
<td>82</td>
<td>80</td>
<td>85</td>
</tr>
<tr>
<td>1600</td>
<td>86</td>
<td>88</td>
<td>87</td>
<td>88</td>
</tr>
</tbody>
</table>

With \( t \) set as 0.6*m, more number of instances are labeled as bad, and hence required good instances are not available for validation. Experiments could be conducted only for data set B1. The accuracy of the MLP for the data set B1 (1600 instances), (88, 87, 88) is comparable against (88) obtained when the instances are selected at random.

The results show that the proposed algorithm is more effective if the training set consists of a relatively more percentage of bad instances. The identification of nature of instances is found to be an integral part of recognition to enhance the accuracy.

The proposed algorithm allows one to identify the type of instances and also provides the freedom to decide on the percentage of good instances and percentage of bad instances to be used in a training phase so as to arrive at a good recognition rate in the testing phase in a PR system. The algorithms are tested on a subset of Tamil vowels and a subset of English alphabets and
found that the results are encouraging. The application is generic in nature as the instances are character images and it is applicable for any recognition problem.

2.5 CONCLUSION

In supervised learning, instance selection is an important step where the superfluous instances are removed from the data set. In this chapter, a new algorithm for identification of the nature of instances for hand-printed character recognition system has been presented. The key parameters that characterize the instances using reduct soft sets with interval estimates have been obtained. This emphasizes the fact that parameter reduction method in soft sets with an interval is very useful for identifying the nature of instances in a PR system.

The proposed method is experimented on two data sets, a subset of Tamil vowels and a subset of English letters, using MLP as a classifier and the results are encouraging. It has been verified that the reduced training set so obtained has good performance in accuracy. The reduced training set also has the additional advantages of reduced runtimes in the learning and classification stages. Instance selection finally leads to effective and efficient data mining. It is to be noted that the proposed algorithms are simple and easy to verify. As the characters used are images, it can be applied on any PR problem.